

Athman Bouguettaya
Manfred Hauswirth
Ling Liu (Eds.)

LNCS 6997

Web Information System Engineering – WISE 2011

12th International Conference
Sydney, Australia, October 2011
Proceedings

 Springer

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

Alfred Kobsa

University of California, Irvine, CA, 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

TU Dortmund University, Germany

Madhu Sudan

Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbruecken, Germany

Athman Bouguettaya Manfred Hauswirth
Ling Liu (Eds.)

Web Information System Engineering – WISE 2011

12th International Conference
Sydney, Australia, October 13-14, 2011
Proceedings

Volume Editors

Athman Bouguettaya

RMIT University, GPO Box 2476, Melbourne, VIC 3001, Australia

E-mail: athman.bouguettaya@rmit.edu.au

Manfred Hauswirth

National University of Ireland, Digital Enterprise Research Institute (DERI)

IDA Business Park, Lower Dangan, Galway, Ireland

E-mail: manfred.hauswirth@deri.org

Ling Liu

Georgia Institute of Technology, College of Computing

266 Ferst Drive, Atlanta, GA 30332-0765, USA

E-mail: lingliu@cc.gatech.edu

ISSN 0302-9743

e-ISSN 1611-3349

ISBN 978-3-642-24433-9

e-ISBN 978-3-642-24434-6

DOI 10.1007/978-3-642-24434-6

Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011937345

CR Subject Classification (1998): H.4-5, H.3.3-5, J.1, D.2, H.5, C.2

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

© Springer-Verlag Berlin Heidelberg 2011

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.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

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

Preface

Welcome to the proceedings of WISE 2011, the 12th International Conference on Web Information Systems Engineering. This year, WISE returned to Australia for the first time since 2004. So far, WISE has been held in Hong Kong (2000), Kyoto, Japan (2001), Singapore (2002), Rome, Italy (2003), Brisbane, Australia (2004), New York, USA (2005), Wuhan, China (2006), Nancy, France (2007), Auckland, New Zealand (2008), Poznan, Poland (2009), and Hong Kong (2010). WISE provides an international forum for researchers and engineers to present the latest research in Web technologies and Web systems engineering. The papers contained in these proceedings address challenging issues in software services, Web application engineering and modelling, Web search, social networks, Web semantics, and information retrieval and extraction. This year, WISE received 96 submissions from countries all over the world, including Australia, Austria, Brazil, Canada, China, France, Germany, Hong Kong, Greece, Iran, Ireland, Japan, Malaysia, The Netherlands, Norway, Spain, Sweden, Taiwan, Tunisia, UK, and the USA.

After a thorough reviewing process, 17 papers were selected for presentation as full papers. The acceptance rate is 18%. In addition, 11 papers were selected for presentation as short papers. This year's WISE also included 7 demo papers.

No conference is possible without the efforts of a large number of people and WISE this year was no exception. We wish to thank everyone involved, including those who worked diligently behind the scenes and without formal recognition. We would like to thank the members of the WISE 2011 Organization Committee: Yanchun Zhang as WISE Society Liaison, Michael Sheng as Workshop Chair, Armin Haller as Publicity Chair and Web Master, Helen Paik as Publication Chair, Boualem Benatallah and Claude Godart as Demo Chairs, Fabio Casati and Schahram Dustdar as Tutorial Chairs, and Ee Peng Lim and Xiofang Zhou as Panel Chairs.

We would especially like to thank the Program Committee members and reviewers for a rigorous and robust reviewing process. We are also grateful to CSIRO and the International WISE Society for generously supporting our conference.

October 2011

Dimitrios Georgakopoulos
Athman Bouguettaya
Manfred Hauswirth
Ling Liu

Organization

General Chair

Dimitrios Georgakopoulos CSIRO, ICT Centre, Australia

PC Co-chairs

Athman Bouguettaya RMIT University, Australia
Manfred Hauswirth DERI, Galway, Ireland
Ling Liu Georgia Tech, GA, USA

Workshops Chair

Michael Quan Z. Sheng University of Adelaide, Australia

Panel Co-chairs

Ee Peng Lim Singapore Management University, Singapore
Xiofang Zhou University of Queensland, Australia

Tutorial Co-chairs

Fabio Casati University of Trento, Italy
Schahram Dustdar TU Vienna, Austria

Demo Co-chairs

Boualem Benatallah University of New South Wales, Australia
Claude Godart University of Lorraine, France

Publication Chair

Hye-Young Helen Paik University of New South Wales, Australia

Web Directions Liaison

John Allsopp Westciv, Australia

Publicity Chair and Webmaster

Armin Haller CSIRO, ICT Centre, Australia

WISE Society Liaison

Yanchun Zhang

Victoria University, Australia

Program Committee

Karl Aberer	EPFL, Switzerland
Soon Ae Chun	City University of New York, USA
Marco Aiello	University of Groningen, The Netherlands
Hyerim Bae	Pusan National University, South Korea
Amitabha Bagchi	IIT Delhi, India
Wolf-Tilo Balke	University of Hanover, Germany
Bhuvan Bamba	Oracle, USA
Luciano Baresi	Politecnico di Milano, Italy
Srikanta Bedathur	MPI Saarbruecken, Germany
Ladjel Bellatreche	ENSMA-Poitiers, France
Salima Benbernou	University of Paris-Descartes, France
Elisa Bertino	Purdue University, USA
Sami Bhiri	DERI, Ireland
Sourav Bhowmick	Nanyang Technological University, Singapore
Walter Binder	University of Lugano, Switzerland
Brian Blake	University of Notre Dame, USA
Roi Blanco	Yahoo! Research, Spain
Shawn Bowers	UC Davis, USA
David Buttler	Lawrence Livermore National Laboratory, USA
Rajkumar Buyya	University of Melbourne, Australia
Ying Cai	Iowa State University, USA
Fabio Casati	University of Trento, Italy
Sven Casteleyn	Universidad Politécnica de Valencia, Spain
Wojciech Cellary	Poznan University of Economics, Poland
Francois Charoy	Nancy University, France
Keke Chen	Wright State University, USA
Lei Chen	HKUST, Hong Kong
Yueguo Chen	Renmin University of China, China
Vassilis Christophides	University of Crete, Greece
Yondohn Chung	Korea University, South Korea
Oscar Corcho	Universidad Politécnica de Madrid, Spain
Philippe Cudre-Mauroux	MIT, USA
Bin Cui	Beijing University, China
Theodore Dalamagas	National Technical Univ. of Athens, Greece
Flavio De Paoli	Università di Milano Bicocca, Italy
Alex Delis	University of Athens, Greece
Gill Dobbie	University of Auckland, New Zealand
Xiaoyong Du	Renmin University of China, China
Schahram Dustdar	Vienna University of Technology, Austria

Rik Eshuis	Technical Univ. of Eindhoven, The Netherlands
Elena Ferrari	University of Insubria at Como, Italy
Joaoeduardo Ferreira	University of São Paulo, Brazil
Claude Goddart	University of Nancy, France
Armin Haller	CSIRO, Australia
Michael Hausenblas	DERI, Ireland
Katja Hose	MPI, Germany
Utku Irmak	Yahoo! Labs, USA
Yoshiharu Ishikawa	Nagoya University, Japan
Xu Jianliang	Hong Kong Baptist University, Hong Kong
James Joshi	University of Pittsburgh, USA
Dimka Karastoyanova	University of Stuttgart, Germany
Panos Karras	NUS, Singapore
Jinho Kim	Kangwon National University, South Korea
Markus Kirchberg	HP Labs, Singapore
Birgitta Koenig-Riess	University of Jena, Germany
Manolis Koubarakis	University of Athens, Greece
Shonali Krishnaswamy	Monash University, Australia
Hady Lauw	A*Star, Singapore
Sebastian Link	Victoria University of Wellington, New Zealand
Chengfei Liu	Swinburne University of Technology, Australia
Wenyin Liu	City University of Hong Kong, Hong Kong
Xumin Liu	Rochester Institute of Technology, USA
Shiyong Lu	Wayne State University, USA
Zaki Malik	Wayne State University, USA
Brahim Medjahed	University of Michigan, USA
Weiyi Meng	Binghamton University, USA
Sebastian Michel	University of Saarland, Germany
Miyuki Nakano	University of Tokyo, Japan
Surya Nepal	CSIRO, Australia
Anne Ngu	Texas State University, USA
Nikos Ntarmos	University of Ioannina, Greece
Mourad Ouzzani	Purdue University, USA
Claus Pahl	Dublin City University, Ireland
Evaggelia Pitoura	University of Ioannina, Greece
Dimitris Plexousakis	University of Crete, Greece
Florian Rosenberg	IBM TJ Watson, USA
Duncandubugras Ruiz	PUCRS, Brazil
Casper Ryan	RMIT, Australia
Sherif Sakr	NICTA, Australia
Kai-Uwe Sattler	TU Illmenau, Germany
Ralf Schenkel	University of Saarbruecken, Germany
Sangeetha Seshadri	IBM Almaden Research Center, USA
Jun Shen	University of Wollongong, Australia
Michael Sheng	University of Adelaide, Australia

Wanita Sherchan	CSIRO, Australia
Lidan Shou	Zhejiang University, China
Steffen Staab	University of Koblenz-Landau, Germany
Torsten Suel	Polytechnic Institute of NYU, USA
Zahir Tari	RMIT University, Australia
Samir Tata	Télécom SudParis, France
Michiaki Tatsubori	IBM Tokyo Research Laboratory, Japan
Kerry Taylor	CSIRO, Australia
Peter Triantafyllou	University of Patras, Greece
Kunal Verma	Accenture Technology Labs, USA
Wei Wang	University of New South Wales, Australia
Chiemi Watanabe	Ochanomizu University, Japan
Ingmar Weber	Yahoo! Research, Spain
Raymond Wong	University of New South Wales, Australia
Josiane Xavier Parreira	DERI, Ireland
Lai Xu	Bournemouth University, UK
Jefferey Xu Yu	Chinese University of Hong Kong, Hong Kong
Jian Yang	Macquarie University, Australia
Jianwei Yin	Zhejiang University, China
Qi Yu	Rochester Institute of Technology, USA
Pingpeng Yuan	Huazhong Univ. of Science and Technology, China
Xiaofang Zhou	University of Queensland, Australia
Lei Zou	Beijing University, China

Additional Reviewers

Abdelghani, Bakhtouchi	Giannopoulos, Giorgos
Alhosban, Amal	Hashmi, Khayyam
Aly, Ahmed	Hua, Wen
Baez, Marcos	Huynh, Tan Dat
Baryannis, George	Jain, Prateek
Bikakis, Nikos	Jiang, Yu
Bulanov, Pavel	Jin, Lei
Castillo, Carlos	Jung, Harim
Comerio, Marco	Khazalah, Fayez
Dahman, Karim	Kotzinos, Dimitris
Degeler, Viktoriya	Leitner, Philipp
Derguech, Wassim	Li, Jianxin
Dimopoulos, Konstantinos	Li, Zhixu
Ebaid, Amr	Lin, Can
Franke, Jörn	Lofi, Christoph
Gaaloul, Walid	Masoumzadeh, Amirreza
Gao, Shen	Mirylenka, Daniil
Geng, Ke	Narendula, Rammohan
Georgievski, Ilche	Ntarmos, Nikos

Ouziri, Mourad
Palmonari, Matteo
Park, Jun Pyo
Patlakas, Ioannis
Paternier, Achille
Ramanath, Maya
Rezig, El Kindi
Rodriguez, Carlos
Roy Chowdhury, Soudip
Rykowski, Jarogniew
Sathe, Saket
Seufert, Stephan
Soror, Sahri
Sung, Min Kyoung

Tai, Hideki
Takabi, Hassan
Taluđer, Nilothpal
Tamura, Takayuki
Van Woensel, William
Wang, Mingxue
Wetzstein, Branimir
Winck, Ana T.
Yahya, Bernardo Nugroho
Yang, Zhenglu
Yang, Ziwei
Zeginis, Chrysostomos
Zhou, Rui
Zhu, Jia

Table of Contents

Full Papers

Finding Homoglyphs - A Step towards Detecting Unicode-Based Visual Spoofing Attacks	1
<i>Narges Roshanbin and James Miller</i>	
Holistic Approaches to Identifying the Sentiment of Blogs Using Opinion Words	15
<i>Xiuzhen Zhang and Yun Zhou</i>	
Characterizing Web Syndication Behavior and Content	29
<i>Zeinab Hmedeh, Nelly Vouzoukidou, Nicolas Travers, Vassilis Christophides, Cedric du Mouza, and Michel Scholl</i>	
Software-as-a-Service in Small and Medium Enterprises: An Empirical Attitude Assessment	43
<i>Till Haselmann and Gottfried Vossen</i>	
Trust-Based Probabilistic Query Answering	57
<i>Achille Fokoue, Mudhakar Srivatsa, and Robert Young</i>	
Top-K Possible Shortest Path Query over a Large Uncertain Graph	72
<i>Lei Zou, Peng Peng, and Dongyan Zhao</i>	
Training a Named Entity Recognizer on the Web	87
<i>David Urbansky, James A. Thom, Daniel Schuster, and Alexander Schill</i>	
Exploiting Available Memory and Disk for Scalable Instant Overview Search	101
<i>Pavlos Fafalios and Yannis Tzitzikas</i>	
Unsupervised User-Generated Content Extraction by Dependency Relationships	116
<i>Jingwei Zhang, Yuming Lin, Xueqing Gong, Weining Qian, and Aoying Zhou</i>	
Sentiment Analysis with a Multilingual Pipeline	129
<i>Daniella Bal, Malissa Bal, Arthur van Bunningen, Alexander Hogenboom, Frederik Hogenboom, and Flavius Frasinca</i>	
Achieving Multi-tenanted Business Processes in SaaS Applications	143
<i>Malinda Kapuruge, Alan Colman, and Jun Han</i>	

TOAST: A Topic-Oriented Tag-Based Recommender System	158
<i>Guandong Xu, Yanhui Gu, Yanchun Zhang, Zhenglu Yang, and Masaru Kitsuregawa</i>	
Prefix-Based Node Numbering for Temporal XML	172
<i>Curtis E. Dyreson and Kalyan G. Mekala</i>	
Transparent Mobile Querying of Online RDF Sources Using Semantic Indexing and Caching	185
<i>William Van Woensel, Sven Casteleyn, Elien Paret, and Olga De Troyer</i>	
An Automation Support for Creating Configurable Process Models	199
<i>Wassim Derguech and Sami Bhiri</i>	
Graph-Based Named Entity Linking with Wikipedia	213
<i>Ben Hachey, Will Radford, and James R. Curran</i>	
Prediction of Age, Sentiment, and Connectivity from Social Media Text	227
<i>Thin Nguyen, Dinh Phung, Brett Adams, and Svetha Venkatesh</i>	

Short Papers

Word Sense Disambiguation for Automatic Taxonomy Construction from Text-Based Web Corpora	241
<i>Jeroen de Knijff, Kevin Meijer, Flavius Frasinca, and Frederik Hogenboom</i>	
A Composite Self-organisation Mechanism in an Agent Network	249
<i>Dayong Ye, Minjie Zhang, and Quan Bai</i>	
From Interface Mockups to Web Application Models	257
<i>José Matías Rivero, Gustavo Rossi, Julián Grigera, Esteban Robles Luna, and Antonio Navarro</i>	
Automatic Web Information Extraction Based on Rules	265
<i>Fanghui Hu, Tong Ruan, Zhiqing Shao, and Jun Ding</i>	
An Artifact-Centric View-Based Approach to Modeling Inter-organizational Business Processes	273
<i>Sira Yongchareon, Chengfei Liu, and Xiaohui Zhao</i>	
Enhance Web Pages Genre Identification Using Neighboring Pages	282
<i>Jia Zhu, Xiaofang Zhou, and Gabriel Fung</i>	
Stepwise and Asynchronous Runtime Optimization of Web Service Compositions	290
<i>Philipp Leitner, Waldemar Hummer, Benjamin Satzger, and Schahram Dustdar</i>	

Supporting Sharing of Browsing Information and Search Results in Mobile Collaborative Searches	298
<i>Daisuke Kotani, Satoshi Nakamura, and Katsumi Tanaka</i>	
GTrust: An Innovated Trust Model for Group Services Selection in Web-Based Service-Oriented Environments	306
<i>Xing Su, Minjie Zhang, Yi Mu, and Quan Bai</i>	
Trust as a Service: A Framework for Trust Management in Cloud Environments	314
<i>Talal H. Noor and Quan Z. Sheng</i>	
Quality Driven Web Services Replication Using Directed Acyclic Graph Coding	322
<i>An Liu, Qing Li, and Liusheng Huang</i>	
Demo Papers	
OntoAssist: Leveraging Crowds for Ontology-Based Search	330
<i>Winston H. Lin and Joseph Davis</i>	
MashSheet: Mashups in Your Spreadsheet	332
<i>Dat Dac Hoang, Hye-Young Paik, and Wei Dong</i>	
SOAC Engine: A System to Manage Composite Web Service Authorization	334
<i>Haiyang Sun, Weiliang Zhao, Jian Yang, and Guizhi Shi</i>	
A Change Analysis Tool for Service-Based Business Processes	336
<i>Yi Wang, Jian Yang, and Weiliang Zhao</i>	
A Novel Approach for Interacting with Linked Open Data	338
<i>Armin Haller and Tudor Groza</i>	
Cloud Broker: Helping You Buy Better	341
<i>Mohan Baruwal Chhetri, Quoc Bao Vo, Ryszard Kowalczyk, and Cam Lan Do</i>	
FACTUS: Faceted Twitter User Search Using Twitter Lists	343
<i>Takahiro Komamizu, Yuto Yamaguchi, Toshiyuki Amagasa, and Hiroyuki Kitagawa</i>	
Author Index	345

Finding Homoglyphs - A Step towards Detecting Unicode-Based Visual Spoofing Attacks

Narges Roshanbin and James Miller

Department of Electrical and Computer Engineering, University of Alberta
roshanbi@ualberta.ca, jm@ece.ualberta.ca

Abstract. Visual spoofing has become a serious web security problem. The dramatic growth of using Unicode characters on the web has introduced new types of visual attacks. The main source of these attacks is the existence of many similar glyphs (characters) in the Unicode space which can be utilized by attackers to confuse users. Therefore, detecting visually similar characters is a very important issue in web security. In this paper, we explore an approach to defining the visual similarity between Unicode glyphs. The results of the experiments show that the proposed method can effectively detect the “amount” of similarity between a pair of Unicode glyphs.

Keywords: Phishing, Visual spoofing attacks, Web security, Unicode attacks, Kolmogorov Complexity, Normalized Compression Distance.

1 Introduction

Phishing is an important web security problem that affects an increasing number of users. A phishing scam utilizes visually or semantically similar links or web pages to spoof unwary users to expose their secure information such as passwords, bank accounts, credit card numbers, etc.

With the internationalization of the Internet, the utilization of Unicode characters in webpages has become widespread. Moreover, the introduction of Internationalized Resource Identifiers (IRI) and Internationalized Domain Names (IDN) enables users to use their own languages in domain names. Unicode which is a universal character encoding standard, has the potential capacity of more than one million characters including a huge set of letters, punctuation marks, mathematical symbols, technical symbols, arrows and diacritics covering all of the written languages in existence. Hence, there are many similar characters in the Unicode space; using Unicode characters in domain names, webpages and emails introduces new types of visual spoofing attacks commonly referred to as *Unicode attacks*. Unicode-based visual spoofing attacks are considered a significant web security problem as using Unicode text in web documents becomes more popular. In a Unicode attack, characters within a legitimate string are replaced with Unicode characters which are either visually or semantically similar. One kind of Unicode attack that refers only to visually similar web links is the homoglyph attack. An example is replacing the Latin ‘a’ (U+0061) with the Cyrillic ‘a’ (U+0430) in “paypal.com”.

Unicode attacks can be viewed as having two differing implementations: ‘*text as text*’ (character), and ‘*text as images*’ (glyphs). Dealing with *text* has the advantage that browsers have access to the codes of characters. For example, by processing “top.com” as a character string (0074 **03BF** 0070 002E 0063 006F 006D), it is easily detectable that it is a mixed-script spoofing in which the Latin character ‘o’ is replaced by the visually-similar Greek character ‘ο’. Therefore, it is easier for browsers to detect similar characters and prevent Unicode attacks when they have access to the *text*.

The focus of this paper is on *images of text* or glyphs, which is believed to represent a more complex problem. For example, Unicode glyphs in the bodies of web pages cause significant visual confusion leading unwary users to take inadvisable options. One example is shown in Fig 1. Here, the image of the character ‘o’ in “.com” is substituted for the image of the character ‘ό’ (Greek Small Letter Omicron with Tonos). The phisher to avoid automated analysis of the URL simply presents as an image tag. Considering such examples, there is a necessity to be able to detect visual similarity between images of Unicode characters.

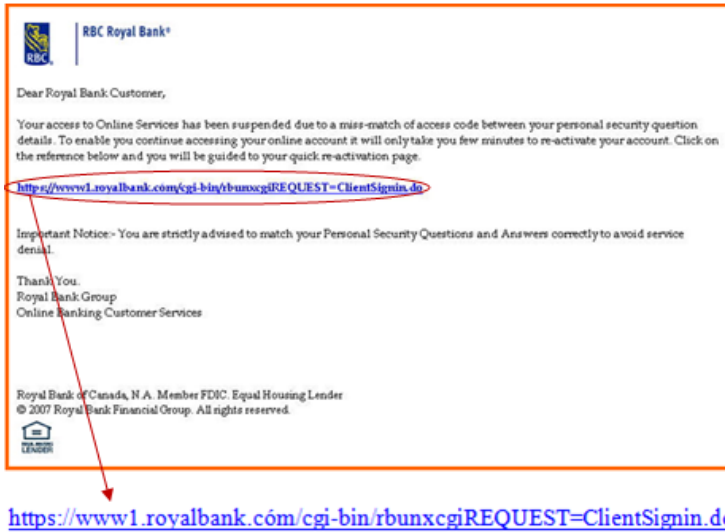


Fig. 1. Confusion resulted from the similarity between Unicode glyphs

The remainder of this paper is as follows: Different types of Unicode attacks are introduced in Section 2. Section 3 discusses homoglyphs in Unicode and the issues when using homoglyphs in web pages. Section 4 describes some existing defenses against Unicode attacks. In section 5, we discuss our proposed method. The experiment we designed to test our approach is described in section 6. In section 7, we provide the results of our experiments and finally, we conclude this paper and discuss our future work in Section 8.

The main contributions of this paper are: (1) realizing a working definition of homoglyphs; (2) producing an actualization for this definition; and (3) producing a tentative list of homoglyphs existing in the Unicode space.

2 Different Types of Unicode-Based Spoofing Attacks

Unicode-based spoofing attacks include:

Mixed-script Spoofing: A character from a different script replaces a character in the string. (Fig 2)

top.com	0074 006F 0070 002E 0063 006F 006D	Latin small o
top.com	0074 03BF 0070 002E 0063 006F 006D	Greek omicron

Fig. 2. Example of Mixed-script spoofing

Whole-script Spoofing: An entire domain name in one script (e.g. Latin) is replaced by a visually similar domain name in another script (e.g. Cyrillic). For example, Cyrillic ‘scope.com’ is used to spoof Latin ‘scope.com’.

Single-Script Spoofing: Similar characters within one script are replaced; example:

so̅s.com	0073 006F 0337 0073 002E 0063 006F 006D	o + combining slash
so̅s.com	0073 00F8 0073 002E 0063 006F 006D	

Fig. 3. Single-script spoofing

Combining Mark Order Spoofing: Some Unicode characters can be expressed as combinations of diacritical marks. For example, Å can be expressed as <U+0041> and <U+030A>. In languages where canonical ordering is not performed, the reordering of characters (<0041>+<030A> vs. <030A>+<0041>) can cause spoofing.

Inadequate Rendering Support: Fonts or text-rendering systems with inadequate support for displaying glyphs cause this kind of visual spoofing attack; e.g. some fonts visually display the number ‘1’ and a lowercase ‘l’ as the same.

Bidirectional Text Spoofing: When characters belonging to right-to-left writing systems such as Hebrew or Arabic are mixed with characters of left-to-right scripts, a bidirectional text is formed. It may lead to visual spoofing attack. Fig 4 presents an example in which the insertion of a left-to-right ‘a’ between two Arabic components changes their display order.

http://سلاّم دائّم.com
http://سلاّم.a دائّم.com

Fig. 4. Bidirectional Text Spoofing

Syntax Spoofing: Syntax characters of URLs such as ‘/’, ‘.’ or ‘?’ are substituted with a visually similar Unicode character. For example, in “http://macchiato.com/x.bad.com”, if the last regular ASCII SLASH (U+002F) is replaced with FRACTION

SLASH (U+2044), the domain appears to be ‘macchiato.com’ but is in fact ‘bad.com’.

Numeric Spoofing: This kind of visual spoofing attack is a result of using digits of different scripts. For example, digits in Bengali are {০ ১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯} while in Oriya, they are {୦ ୧ ୨ ୩ ୪ ୫ ୬ ୭ ୮ ୯}. Consider for example string ‘୪୨’ which is Bengali digit ‘4’ followed by Oriya digit ‘2’ and its numeric value is ‘42’. However, the string is visually confusable with ‘89’.

3 Homoglyph

A glyph is a particular image that represents a character or part of a character. Hence, characters differ from glyphs. A character is the smallest component of written language that has semantic value [1]. Two different characters can change the meaning of a word. For example, capital ‘B’ and lowercase ‘b’ are two characters as they are different in Latin orthography (‘Bill’s meaning differs from ‘bill’s). However, italic ‘b’ and bold ‘b’ are not considered two different characters. There are different glyphs with various shapes to display a specific character depending on script, usage or context. The rendering system is responsible for choosing the glyph. For example, Latin lower case ‘a’ is shown by a variety of glyphs in Fig 5.

A homoglyph is two or more glyphs that cannot be differentiated by instant visual inspection. The concept is defined between images (glyphs), not characters (codes). Depending on the rendering system, the typeface and the font size that are used, two glyphs that are ordinarily distinguishable may be very similar. Fig 6 provides some examples of homoglyphs either within a script or between scripts.

Character	Sample glyphs
Latin small letter A(ASCII code 65)	À Á Â Ã Ä Å

Fig. 5. Various glyphs displaying character ‘a’

Single script	ආ	භ	U+0DB6 (Sinhala letter Alpaprana Bayanna) and U+0D9B (Sinhala letter Mahaaprana Kayanna)
	ก	ท	U+0E05 (Thai character Kho Khon) and U+0E15 (Thai character To Tao)
Complex script	ಉ	ಊ	U+0C8E (Kannada letter E) and U+0DB0 (Sinhala letter Mahaaprana Dayanna)
	৭	٩	U+09ED (Bengali digit Seven) and U+0669 (Arabic-Indic digit Nine)

Fig. 6. Examples of Homoglyphs in Unicode

3.1 Homographs

If two different “strings” are represented by the same sequence of glyphs, they are called homographs [2]. In other words, homographs are a group of characters that share the same written form but different meanings. For example, the three sequences

of characters shown in Table 1 are homographs. In row 2, the Latin lowercase letter ‘o’ is substituted for the Greek lowercase letter ‘o’. While the letter ‘o’ in the first row (U+006F) and letter ‘ö’ in the third row (U+00F6) may not seem very similar, when they are located within a sequence of characters, they become more confusable (‘.com’ and ‘.cöm’).

When two visually similar Unicode glyphs come in words, in between other glyphs, it is more difficult to differentiate between them. For example, consider two Sinhala words ‘ප්ලවිතලය’ and ‘ප්ලවිඵලය’. The former is a valid word meaning ‘Consequence’. The latter is a non-word resulting from substituting the fourth character ‘ඵ’ (U+0D91) for its visually similar character ‘ඵ’ (U+0DB5).

Table 1. An example of homographs

.com	002E 0063 006F 006D (Latin small letter ‘o’)
.com	002E 0063 038F 006D (Greek small letter ‘o’)
.cöm	002E 0063 00F6 006D (Latin small letter ‘o’ with Diaeresis)

The problem becomes much more difficult by considering the fact that there are different rules for combining glyphs to make words in different scripts. In English, a sequence of characters, including consonants and vowels, located beside each other creates a word. Characters in a Latin word are not connected and are written in a left-to-right direction. However, the concatenation of glyphs to make words is not so simple in other languages. Some scripts are written from right to left or in a vertical direction. In cursive languages, letters get their appropriate contextual glyph form and then connect to each other to create a word. Fig 7 shows some examples.

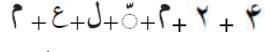

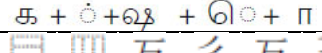



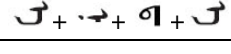

	→		(a)
	→		(b)
 2FF3 2FF2 4E02 5F61 4E02 5F50 76BF	→		(c)
	→		(d)

Fig. 7. Examples of concatenation rules in different scripts

Example (a) represents the Arabic word ‘teacher’. Since Arabic is a cursive script, it connects glyphs in a word to each other. This language is bidirectional. While letters are written right-to-left, the direction for numbers is left-to-right. Example (b) displays vowel reordering in a Tamil word. Although it is written from left-to-right, a left-side dependent vowel causes reordering. In (c), the formation of an un-encoded CJK ideograph is described. In this sequence, U+2FF2 and U+2FF3 show the relative positioning of the operands running from left to right and from top to bottom. Finally, (d) represents a Mongolian word meaning ‘here’ which is written vertically from top to bottom.

3.2 Using Glyphs in Web Pages

In a webpage, different images, animated pictures, video clips, various text effects, font variations, colors and backgrounds distract users from a single character. It makes distinguishing between two similar Unicode glyphs very difficult and increases the possibility of phishing attacks. Some issues of using Unicode glyphs in webpages are exemplified in Fig 8. Column (a) represents how a complex background can cause difficulty in differentiating two similar Sinhala characters ශ් and ශ්. Making a distinction between small ‘L’ and capital ‘i’ is difficult in column (b) because of insufficient contrast between foreground and background colors. Column (c) shows a ‘dot’ which is replaced by a ‘hyphenation point’. The Two resulting strings are generally distinguishable; but in a webpage with many distractions they are not. Column (d) displays the effect of surrounding text on differentiating between two visually confusable glyphs. In this example, the first ‘L’ in both words is replaced by capital ‘i’. It is easily distinguishable in ‘gloBal’; but it is not in ‘flexible’. Column (e) represents the effect of character spacing.


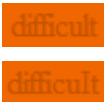

		foo.com	flexible	flexible	table 1
		foo.com	gloBal	flexible	table 1
(a)	(b)	(c)	(d)	(e)	(f)

Fig. 8. Unicode glyphs in web pages

Using some fonts makes the distinction between homoglyphs more challenging. Column (f) displays one string using two different fonts. In the first one, small letter ‘L’ and digit ‘1’ are different; however, in the second one, they are confusable. Small font or display size as experienced on smartphones, adds further levels of confusion.

The result of all of these complications is a situation so complex that a complete technical answer maybe infeasible. In this paper, we seek to explore the root of these issues, the similarity or dissimilarity between a pair of homoglyphs. However, as illustrated “scaling” up from a working definition of a homoglyph to a working definition of a homograph and finally a context-invariant (or dependent) working definition of a homograph remains elusive at this point.

4 Current Defenses against Homograph Attacks

There are some techniques to mitigate the effectiveness of phishing attacks. For example, Liu et al [3] propose a region-based approach based on the visual comparison of the Document Object Model. Another example is the method proposed by Fu et al that performs a visual similarity assessment using the EMD algorithm [4]. Another group of anti-phishing strategies concentrates on toolbars of or extensions to web browsers. Using Black/white lists in browsers is an example of such methods. The disadvantage of this list is that many phishing websites are not reported.

Reputation scoring, reported by some webpages such as WOT and iTrustPage or the anti-phishing community, is another anti-phishing technique.

There are some strategies to defend against homoglyph attacks. A widely adopted strategy is Punycode which is proposed by the Unicode Consortium [1]. It transforms non-ASCII characters of a domain name uniquely and reversibly to ASCII characters [5]. For example, Punycode maps Cyrillic ‘a’ to the string “xn—80a”. The drawback of Punycode is that its output is not human-readable. Another strategy is coloring Unicode strings [1]. This method, by assigning different colors to various languages/scripts, helps users detect mixed-script strings. A similar approach is to highlight numbers or ASCII characters [6]. Fu et al have proposed another approach to detect IDN homoglyph attacks. In their method, they make a Unicode character similarity list based on the amount of visual/semantic similarity between each pair of characters; then use regular expressions to detect phishing patterns [7]. In order to measure character similarity, they use a simple pixel-overlapping method.

Since the most available defensive techniques are based on text processing performed by browser extensions or plug-ins, the shortage of methods to distinguish visual similarity between Unicode glyphs, which are located in both domain names and the contents of webpages, is concerning.

5 Proposed Method

There are a lot of similar glyphs in Unicode space. However, the amount of similarity between two glyphs varies. Our main problem is the definition of similarity and dissimilarity between glyphs. If someone asks: “are these two images similar?”, there is no absolute answer for this question. Instead, a spectrum can show the degree of similarity between two images (Fig 9). The concept of *similarity* is highly affected by human perception. What one person calls similar might be dissimilar from another person’s viewpoint. Therefore, specifying a boundary between similar and dissimilar is very difficult and changes from person to person.

Two images are considered completely dissimilar, if they are random with respect to each other; in other words, if we cannot describe one given the other [8]. Hence, we can define two glyphs as *anti-homoglyphs* (completely dissimilar) if they are random with regard to each other. This is the left-most point in the spectrum of Fig 9. In contrast, a *homoglyph* cannot be defined as a specific point in this spectrum. We can define it as a threshold. Glyphs having a degree of similarity more than that threshold are considered homoglyphs. We use Kolmogorov Complexity theory to measure randomness [9].

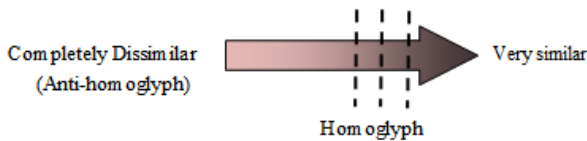


Fig. 9. Positions of Homoglyph and Anti-homoglyph in the similarity spectrum

The Kolmogorov complexity of an object x , denoted by $K_U(x)$, is defined to be the length of the shortest program that can produce that object on a universal Turing machine. Kolmogorov Complexity theory provides a “viewpoint” on the random content within an object [9]. Kolmogorov defines a string random; if there is no program that can produce it with a length shorter than its own length. The conditional Kolmogorov complexity of a string x given a string y , denoted by $K(x|y)$, is defined as the length of a shortest program that prints x when y is executed.

The information distance between x and y can be defined as:

$$E(x, y) = \max \{K(x|y), K(y|x)\} \quad (1)$$

$E(x, y)$ is the maximum length of the shortest program that prints x from y and y from x [8]. This distance needs to be normalized because in expressing similarity, we are more interested in relative distance. For example, two strings of 10000 bits differ by 100 bits are considered relatively more similar than two strings of 100 bits that differ by 100 bits. The Normalized Information Distance (NID) between strings x and y is defined as [8]:

$$d(x, y) = \frac{\max\{K(x|y^*), K(y|x^*)\}}{\max\{K(x), K(y)\}} \quad (2)$$

Although NID is a valid distance metric, it is based on the non-computable Kolmogorov complexity function $K()$; therefore, it also does not have a practical implementation. Replacing $K()$ with a practical function makes NID computable. NCD, Normalized Compression Distance, can be viewed as the result of substituting $K()$ for a practical data compressor $C()$ [10]:

$$\text{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}} \quad (3)$$

In this definition, $C()$ gives the compressed length of a string using an arbitrary compression algorithm. xy means the concatenation of strings x and y .

We have used NCD as a similarity metric, which approximates the non-computable Kolmogorov complexity, to measure the degree of similarity between glyphs. According to Kolmogorov theory, two homoglyphs must be highly compressed using the information from each other. However, two “anti-homoglyphs”, are random with respect to each other and hence cannot be compressed using information from the other glyph.

5.1 Challenges of Realizing NCD for Glyphs

The first step to calculate NCD is to produce images of Unicode characters. We need to decide about the font face, font size and the dimensions of images. In an ideal situation, the same font face should render all the glyphs. However, it is impossible to use a single font for rendering all the characters since there is no specific font covering the whole Unicode space. Therefore, the least possible number of fonts is utilized for this purpose.

In order to make two glyphs comparable, we fixed the font size and dimensions of the images. While arbitrary, the dimensions were set to 1000 to ensure that the compressors had sufficient information to achieve high compression rates where applicable. In addition, Unicode includes many complicated characters, especially in non-Latin scripts. The image representing such complex characters should have sufficient resolution to show all their details.

The selection of a compressor, which is appropriate for our data, is another issue. The most important parameter in NCD estimation is to choose an optimized compression algorithm. The ideal compressor for our purpose is one that provides not only a high compression ratio, but also a reasonable speed as we are dealing with a large number of images. Having images (glyphs) as our inputs, we can use either a 1 or 2-Dimensional (lossy or lossless) compressors. While 2-D compression algorithms treat glyphs as rectangular images, 1-D methods consider them as byte-streams. Dealing with an image as a string destroys the spatial relationships between pixels. A 1-D compressor at first performs an image linearization step to recast two-dimensional images to 1-dimensional space. Another important step in the measurement of NCD is concatenating two images. A 2-D compressor must decide the direction of concatenation, and how to concatenate when the images are of different sizes; a 1-D compressor just concatenates two strings.

Cilibrasi [8] argues “the NCD minorizes all similarity metrics based on features that are captured by the reference compressor involved”. Different references use various compressors such as bzip2, gzip, PPMZ and JPEG2000 to measure NCD [10]. [10] discusses that although 1-D compression algorithms destroy the spatial relationships in the image, they produce better results than 2-D algorithms. Because of this uncertainty, we performed an experiment to evaluate a cross-section of compressors for our problem.

For linearization, we have used row-major scan-line. The huge number of Unicode standard glyphs makes it very challenging to try different linearization methods to find the optimal approach. Moreover, other references that use NCD as an image similarity metric, [10, 11], have implemented row-by-row scanning. [12] after investigating various types of linearization (row-by-row, column-by-column, Hilbert-Peano and Self-describing Context-based Pixel Ordering) concluded that while different types lead to different NCD values, no technique was uniformly superior.

6 Design of Experiment

Unicode contains a huge number of characters arranged in code charts in two groups: scripts, symbols and punctuation. Scripts include Modern scripts; and Ancient scripts which are not in customary use [13]. For our experiment, we have selected a group of about 6,200 characters from Modern scripts excluding Hangul scripts. We did not utilize Hangul scripts, as they contain a large number of characters (11, 400) and ancient scripts. Covering 40 different scripts of Unicode, our character set is representative of a variety of modern languages. It covers European scripts such as Armenian, Coptic, Cyrillic; African scripts such as Ethiopic; Middle Eastern languages such as Arabic and Hebrew; South Asian Scripts such as Limbu and Devanagari; Southeast Asian scripts such as Myanmar and Thai; Central Asian scripts

such as Mongolian and Tibetan; Philippine scripts such as Hanunoo; East Asian languages such as Japanese and Yi; and American scripts such as Cherokee and Unified Canadian Aboriginal Syllables.

We rendered images for all of the characters to the same size. Since it was impossible to use only one font to show all the characters in our selection, we utilized a set of fonts (four Unicode fonts -- Arial Unicode MS, Code2000, Microsoft Sans Serif and GNU Unifont) to produce the glyphs.

In the next step, the NCD between each glyph and all other glyphs was measured. A higher NCD value (for $(image_1, image_2)$) expresses more dissimilarity between them. To select a compressor, we performed an experiment – basically, $NCD(image_i, image_j)$ where i is all the letters in our selected group. We ran this experiment several times for different 2-D compressors (PNG, GIF, JPEG and JPEG2000) and 1-D compressors (Gzip, Bzip2, PPMD, Deflate and LZMA). While using 2D compressors, we considered both horizontal and vertical concatenation methods to calculate NCD. Fig 10 shows histograms of the results for some compressors. In each graph, the horizontal axis represents the range of NCD values. Since in this experiment, the NCD value of each character against itself is measured, the best graph should have a peak around zero. Based upon this experiment, we have chosen LZMA compressor.

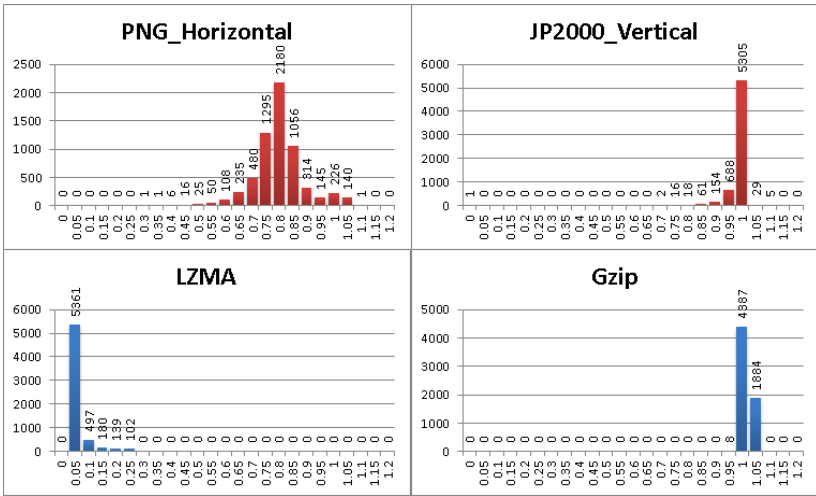


Fig. 10. Histograms of NCD values calculated by different compressors

7 Experimental Results

There are many homoglyphs in Unicode. Fig 11 represents some homoglyphs found for the Latin Letter ‘w’, the Arabic letter ‘Yeh’ and the Latin letter ‘o’. Homoglyphs in these examples belong to different scripts (Latin, Greek, Coptic, Cyrillic, Armenian, NKO, Bengali, Gujarati, Tamil, Telugu, Kannada, Malayalam, Unified Canadian Aboriginal Syllables, Thai, Lao, Myanmar, Georgian, Hangul Jamo, Ethiopic, Yi, Lisu, Phags-pa, Javanese, Deseret, etc). Since every typical glyph in the

Unicode has several homoglyphs, an attacker has many ways to trick users by substituting the glyphs within a URL. For example, assuming that the only homoglyphs of the Latin Letter ‘o’ are those shown in Fig 11 (which is not a correct assumption), a phisher is able to misrepresent www.google.com in more than 1360 different ways.

w	ŵ	u	o	ó	w	W	u	o	o	o	W	W	o	o	o	o	
0077	0175	026F	03C9	03CE	0461	051D	0561	0bf0	0baf	0CAC	0D27	0E1E	13B3	15EF	164	1883	198D
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
1B20	1C03	1E81	1E83	1E85	1E87	1E89	1E98	1F60	1F61	1F62	1F63	1F64	1F65	1F7C	1F7	1FA0	1FA1
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
1FA2	1FA3	1FA4	1FA5	1FF2	1FF3	1FF4	2CB1	2D0D	A4B3	A4EA	A86D	A99E	AA9D	10436	1D222		
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
0626	063D	063E	0649	064a	06CD	06D0	06CE	0777	FBFD	FE8A	FEF2						
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
00F2	00F3	00F4	00F6	014D	01A1	01D2	022F	03BF	03CC	04E7	0585	07CB	09F9	0AE6	0B6	0C66	0ED0
o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o	o
1090	101D	110B	12D0	1946	1A14	1A90	1BB0	1C40	1C5B	1ECD	1ECF	1F41	2D54	A4A8	A8D	AA50	ABF0

Fig. 11. Examples of homoglyphs in the Unicode

Our final purpose in this paper is a pair-wise NCD estimation between all 6,200 chosen characters; this produces more than 19,200,000 values. Fig 12 is a histogram of these results. Some examples of NCD values for some pairs of Unicode glyphs are displayed in Table 2. The two glyphs in the first row (o, o) have a very small NCD and are considered very similar. In contrast, glyphs in the last row (o, o), have a high NCD value of 1.04, and are considered dissimilar. However, specifying the amount of similarity required to call two glyphs *Homoglyphs* is very challenging and varies depending on different people’s perspectives. For example, the answer for the question “are the glyphs in row11 (o and o) homoglyphs?” changes from person to person.

Fig 13 represents the cumulative distribution function of the results. As can be seen, many pairs have significant NCD values. This means that most glyphs in the Unicode space are dissimilar. An elbow appears in the CDF at NCD=0.8. If someone decides this point as the similarity threshold, 634,461 out of 19,709,781 pairs are considered homoglyphs; this constitutes 3.2 percent of all records. The remaining 96.8% (19,075,320 pairs) are dissimilar. The existence of such a large number of homoglyphs in Unicode space does not seem reasonable and many counter-examples exist; see Table 2 for example, although two glyphs in row12 with NCD=0.71 have some similarity (two long vertical lines, etc.) from a compression viewpoint; they do not look very similar in users’ viewpoint. Fig 14 magnifies the left part of CDF graph (where 0 < NCD ≤ 0.75). Three further elbows can now be detected (0.6, 0.5 and 0.25). If we select the threshold to be 0.6, 0.48% of the pairs will be considered similar and 99.52% will be dissimilar. The ratio of dissimilar to similar pairs in this case is 19614321:95460 (205:1). 95,460 similar pairs is still a large number and again from Table 2, pairs with NCD values around 0.6 (e.g. o and o) are not really visually similar. The ratio increases to 392:1 for a threshold = 0.5; and to 3339:1 for a

threshold = 0.25. In the last case, 5,902 out of 19,709,781 (0.03%) of pairs are considered similar. This number is probably more realistic. As it can be seen in Table 2, pairs with NCD values less than 0.25 (rows 1-7) are visually very similar.

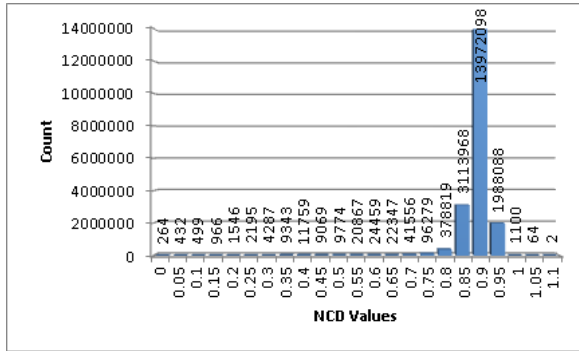


Fig. 12. The histogram of pair-wise NCD calculation of all characters in our set

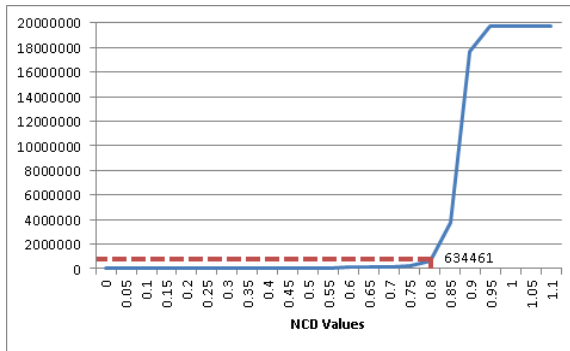


Fig. 13. Cumulative distribution function of pair-wise NCD calculations

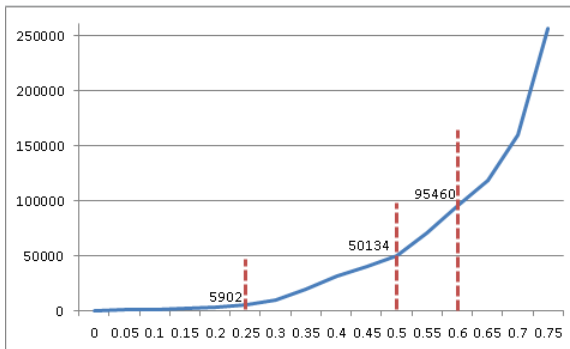


























Fig. 14. CDF of results with $0 < NCD \leq 0.75$

Table 2. Examples of NCD values between some characters

Row	Glyph1	NCD	Glyph2
1	0b95 (Tamil Letter Ka) 	0.027	 0be7 (Tamil Digit One)
2	A1E7 (YI Syllable Guox) 	0.132	 A1E8 (YI Syllable Guo)
3	00f2 (Latin Small Letter O with Grave) 	0.155	 04e7 (Cyrillic Small Letter O with Diaeresis)
4	03CE (Greek Small Letter Omega With Tonos) 	0.157	 0461 (Gyrillic Small Letter Omega)
5	0077 (Latin Small Letter W) 	0.224	 0175 (Latin Small Letter W with Circumflex)
6	0626 (Arabic Letter Yeh with Hamza Above) 	0.224	 064A (Arabic Letter Yeh)
7	038F(Greek Capital letter Omega With Tonos) 	0.234	 1FFC (Greek Capital letter Omega With Progegrammeni)
8	04C0 (Cyrillic Letter Palochika) 	0.337	 196C (Tai Le Letter Aue)
9	142A (Canadian Syllabic Final Down Tack) 	0.393	 1682 (Ogham Letter Luis)
10	079D(Thaana Letter Sheenu) 	0.417	 079E(Thaana Letter Saadhu)
11	1705 (Tagalog Letter Nga) 	0.601	 1730 (Hanunoo Letter Sa)
12	0389 (Greek ETA With Tonos) 	0.71	 1964(Tai Le Letter I)
13	0930 (Devanagari Letter RA) 	0.917	 0B05(Oriya Letter A)
14	0BF8 (Tamil As Above Sign) 	0.980	 174E (Buhid Letter La)
15	188C(Mongolian Ali Gali Tta) 	1.03	 A320 (Yi Syllable Su)
16	1696 (Ogham Letter Or) 	1.04	 0DA1 (Sinhala Mahaapraana Cayanna)

8 Conclusion and Future Work

In this paper, we propose a method to measure the amount of visual similarity between Unicode glyphs. This can help users to detect Unicode attacks. Currently, most available defenses against Homograph attacks concentrate on the detection of *text as text* attacks; however, modern web 2.0 system provides many opportunities for *text as an image* Homograph attacks. Therefore, an important issue in web security is to have a measure of similarity between a pair of Unicode glyphs. In this paper, we show that NCD (Normalized Compression Distance) can be a good metric for this purpose.

We designed an experiment and estimated this metric for a set of around 6,200 Unicode characters selected from more than 40 different modern languages. The result of this experiment showed that there is an inverse relationship between NCD value and the amount of similarity. The histogram and the CDF of the results

represent that most glyphs in the Unicode space are dissimilar and homoglyphs constitute a small percentage of the whole results. However, there is not any specific point in the NCD range by which the concept of ‘*Homoglyph*’ can be defined.

We have calculated NCD values for a set of around 6,200 Unicode characters so far. In future, we will expand it to the whole Unicode space and produce a Unicode Character Similarity List, which can be useful in various web security issues including the detection of phishing attacks. The process of detection, which is a part of our future work, starts by the segmentation of images in web pages that contain hyperlinks or other forms of interaction. Afterwards, considering the fact that each user prefers to use a particular language that they can select in their browser, we will identify all mixed-script homographs as well as single-script homoglyphs with respect to the preferred language. A glyph belonging to another language can be the sign of a potential attack, especially if it has a homoglyph in the preferred language. In the next step, marked homoglyphs will be passed to a decision making unit to determine if it is likely to be a real problem or not. The designing and implementation of this phishing detection strategy will be our next undertaking.

References

1. The Unicode Consortium.: The Unicode Standard, Version 5.0.0. Addison-Wesley, Boston (2007)
2. Unicode Security Considerations, <http://unicode.org/reports/tr36/>
3. Liu, W., Deng, X., Huang, G., Fu, A.Y.: An Anti-Phishing Strategy Based on Visual Similarity Assessment. *J. IEEE Internet Computing*. 10, 58–65 (2006)
4. Fu, A.Y., Liu, W., Deng, X.: Detecting Phishing Web Pages with Visual Similarity Assessment based on Earth Mover’s Distance (EMD). *J. IEEE Transactions on Dependable and Secure Computing* 3, 301–311 (2006)
5. Costello, A.: RFC 3492 - Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA), IETF (2003)
6. Krammer, V.: Phishing defense against IDN address spoofing attacks. In: Proceedings of the 2006 International Conference on Privacy, Security and Trust (PST 2006), New York (2006)
7. Fu, A.Y., Deng, X., Wenyin, L.: REGAP: A tool sfor Unicode-based web identity fraud detection. *J. Digital Forensic Practice* 1, 83–97 (2006)
8. Cilibrasi, R., Vitányi, P.: Clustering by compression. *J. IEEE Transactions on Information Theory* 51, 1523–1545 (2005)
9. Li, M., Vitányi, P.M.B.: An Introduction to Kolmogorov Complexity and its Applications. Springer, New York (1997)
10. Chen, T.C.: Detecting Visually Similar Web Pages: Application to Phishing Detection. Thesis (PhD). University of Alberta (2010)
11. Tran, N.: The Normalized compression distance and image distinguishability. In: The 19th IS&T/SPIE Symposium on Electronic Imaging Science and Technology, San Jose, vol. 6492, p. 64921D (2007)
12. Mortensen, J., Wu, J.J., Furst, J., Rogers, J., Raicu, D.: Effect of Image Linearization on Normalized Compression Distance. *SIP (Signal Processing, Image Processing and Pattern Recognition)* 61, 106–116 (2009)
13. The Unicode Standard Version 6.0.0, <http://www.unicode.org/versions/Unicode6.0.0/>

Holistic Approaches to Identifying the Sentiment of Blogs Using Opinion Words

Xiuzhen Zhang and Yun Zhou

School of Computer Science & IT, RMIT University
GPO Box 2476, Melbourne 3001, Australia
xiuzhen.zhang@rmit.edu.au, yun.zhou@student.rmit.edu.au

Abstract. Sentiment analysis aims to identify the orientation (positive or negative) of opinions or emotions expressed in documents. Opinion lexicons comprise opinion words expressing prior positive or negative sentiments. In most previous work documents are represented as bags of words and sentiment analysis has been cast a classification problem, where opinion lexicons are only used to enhance the classification models. In this paper we aim to establish the direct connection between document sentiment and opinion words in the documents. We propose two holistic approaches that consider the probability distribution of both opinion words and their polarity for analyzing document sentiment. Our extensive experiments on blogs of 12 topics show that our holistic models significantly improve baseline models using words and their polarity information separately, and is also superior to an existing approach combining both types of information.

Keywords: Opinion Mining, Sentiment Analysis, Text Classification, Naive Bayes.

1 Introduction

In recent years there has been an explosion of user-generated content on the Internet in the form of blogs. The term blogosphere has been used to refer to all the blogs on the Internet and their relationships. Blogs have been used by well-known international news agencies including BBC and CNN to discuss public opinions. Sentiment analysis aims to identify the positive or negative opinions or emotions expressed in blogs. For example, consider blogs on Cindy Sheean — the American anti-war activist whose son was killed during his service in the Iraq War in 2004. Sentiment analysis can be utilized to identify the favourable or unfavourable public opinions expressed in blogs about Cindy Sheean and the demonstrations she has been involved in.

In linguistic studies, sentiment analysis has been carried out mainly at the word, phrase or sentence levels [5,4,21,20,22]. Notably opinion lexicons have been developed that contain opinion words that are annotated with prior positive or negative polarity out of context [5,4,21]. For example in many opinion lexicons *beautiful* is a positive word while *horrid* is a negative word. Opinion lexicons

have been used in product review analysis to determine whether the opinion expressed on a product feature is positive or negative [6,2].

Sentiment analysis has also been performed at the document level [18,13,16,9]. Pang et al [13] cast the sentiment analysis of documents as a supervised text classification problem, where documents are represented as bags of words. In [18] identifying sentiment of reviews is carried out in an unsupervised setting.

Although recently there has been work on using opinion lexicons to augment text classification for document sentiment identification [16,9], opinion lexicons have not been used directly for document sentiment identification. In this paper we study the problem of using a generic opinion lexicon directly for identifying the sentiment polarity of blogs (documents), and investigate the best approach to leverage the word polarity information for blog sentiment identification. We propose two holistic approaches that consider the probability distribution of both opinion words and their polarity for analysing document sentiment. Our extensive experiments on blogs of 12 topics show that our holistic models significantly improve baseline models using words and word polarity information separately, and are also superior to an existing approach [9] combining both types of information.

2 Related Work

Opinion mining and sentiment analysis have attracted active research recently. Overview of developments in this area has been described in literature [8,12,17]. Generally there are two types of studies in this area, namely the linguistic approach and the learning approach.

Linguistic studies have mostly focused on identifying words and phrases that express subjectivity, where an important task is to generate dictionaries capturing the sentiment of words either manually or automatically [5,14,19,22]. Ng et al. [10] pointed out that automatic approaches often yield unsatisfactory lexicons, with either high coverage and low precision or vice versa.

In [18], Turney employed lexical knowledge phrases (adjectives followed by nouns, or adverbs followed by verbs) to develop an unsupervised learning algorithm for classifying reviews as positive (thumbs up) or negative (thumbs down). The polarity of phrases is computed by searching the Web to compute its similarity to the positive and negative reference words “excellent” and “poor” respectively. It is difficult to generalize the approach to blogs where such reference words do not exist.

Learning approaches to sentiment analysis differ in their learning paradigm (supervised or unsupervised), features for learning and the level of sentiment analysis. The focus of [2], [6] and [7] is at a micro level — the opinions for entities described in product reviews. In [2] and [6] opinion words around a product feature are used to determine the opinion orientation for the product feature. In [7] linguistic features such as part-of-speech, phrases internal formation patterns, and surrounding contextual clues of words/phrases are integrated into learning the sentiment orientation of product features. Wilson [20] formulated

the problem of identifying sentiment-bearing sentences from text as a supervised learning task.

For document sentiment analysis, Pang et al. [13] proposed to apply text classification (categorisation) models to classify sentiment for movie reviews. In their models movie reviews are represented as bags of words and the models outperform the naive polarity model of counting occurrences of positive and negative words. In our study, we show that making use of the lexical knowledge in the form of opinion word polarity together with training samples can achieve effective document sentiment classification.

Incorporating the polarity of opinion words into standard text classification has been studied recently [3,9,16]. In [3] and [16] word-class association (opinion word polarity is a special form of word-class association) was used for unsupervised learning. In [3] the word-class association (called labeled features) was used to constrain the model's prediction on unlabeled instances. In [16] the opinion word polarity was integrated with unlabeled documents for predicting document sentiment using standard regularized least squares. But none of the existing work analyzes the sentiment of documents using only the opinion lexicon.

Notably in [9], lexical information was used to refine text classification for sentiment classification. A Naive Bayes classification model was employed. A generative model for opinion word polarity distribution was proposed. We too apply the Naive Bayes model for classification and the generative model for lexicon polarity distribution. But there are several major differences. They proposed a linear pooling approach to combine words and their polarity probability distribution, but our experiments show that the approach is ineffective in our setting. This may be partly due to that their model is designed for the bags of words representation for document whereas we are only considering opinion words. We propose two new holistic approaches combining words and their polarity information for estimating the word probability distribution and for making decisions towards document sentiment. While their linear pooling approach obtains sentiment prediction accuracy comparable to the baseline Naive Bayes model, both our new approaches significantly improve the baseline Naive Bayes model.

3 The Opinion Lexicon

Opinion words are words expressing *prior* out of context positive or negative attitude for opinions or emotions. They are also called subjectivity words or sentiment words. Wilson et al. [22] compiled a comprehensive subjectivity lexicon of over 8000 subjectivity words (called subjectivity clues). The lexicon includes a list of subjectivity words from [5,14]. It also includes words from the General Inquirer positive and negative word lists [4] and subjectivity words from a dictionary and thesaurus [22].

Words in the subjectivity lexicon may also have objective usages. Words that are subjective in most contexts are *strongly subjective* and those that may only have certain subjective usages are *weakly subjective*. Subjective words are tagged with their prior polarity. The *positive* and *negative* tags are for positive and negative polarity respectively. The *both* tag is for words that can express both positive

Table 1. Some words in our opinion lexicon

Positive	acclaim, adore, affirm, befit, catalyst, dear, defer, encourage, fantastic, good, hero, loyalty, marvel, nice, perfect, radiant, sane, thrill, understand, want, yearn, zest
Negative	abash, abhor, accuse, admonish, agonize, beg, chao, defunct, excess, fear, grief, hell, insane, lose, malignant, nightmare, object, penalty, quarrel, racist, scandal, thwart, unfair, virus, yawn, zealou

and negative polarity. The *neutral* tag is for words expressing subjectivity but not obvious positive or negative polarity.

We apply some filtering criteria to the original subjectivity lexicon compiled by Wilson et al. to construct the sentiment lexicon for our research. We first remove the small fraction of words with the *both* or *neutral* tags from the original lexicon. As a result only positive and negative words are kept, and the weak subjective words are further removed. As a result, our final sentiment lexicon consists of 3218 unique words after stemming. In total there are 1067 positive and 2151 negative words in our final sentiment lexicon. This sentiment lexicon is a generic lexicon without any specific domain in mind, and it has reliable human annotations for sentiment polarity. Some sentiment words randomly chosen from our opinion lexicon are listed in Table 1.

Depending on contexts such as negation and word sense, the phrase in which an opinion word appears may express a sentiment polarity different from the prior polarity of the opinion word [22]. Even so we believe that the polarity of opinion words can produce prediction for the overall sentiment polarity of a document. As will be discussed in the next section, based on the assumption that positive words are more likely to be associated the positive class (for the overall document sentiment) and that negative words are more likely to be associated with the negative class, we propose a word polarity model based on the probability distribution of opinion word polarity in the lexicon. Our experiments (Section 7) show that such word polarity model can achieve reasonable accuracy for predicting document sentiment, and can be used to boost the performance of supervised sentiment classification.

4 Word Polarity Models for Document Sentiment Classification

We consider the prior polarity of opinion words for document sentiment analysis. Given a lexicon of positive and negative words, a naive approach to using this information is to measure the relative frequency of occurrences of positive and negative words in a document. Intuitively a document belongs to the positive class if it has more occurrences of positive words than negative words. We refer to this simple baseline model as the *naive polarity model*.

Given that the opinion lexicon is a collection of words whose prior polarity is reliably annotated by manual inspection of many documents containing the words, an alternative approach to using word polarity information is to derive the probability for word polarity with respect to document sentiment from the lexicon. We describe the approach proposed in [9]. The main idea to derive the probability of word polarity for the positive and negative class is based on the assumption that a positive (negative) document is more likely to contain positive (negative) words than negative (positive) words. A quantity termed *polarity level* is used to describe the ratio for this probability difference.

Let $P(w_+|+)$ and $P(w_+|-)$ denote respectively the probability that a positive word appears in a positive and negative document, and similarly $P(w_-|+)$ and $P(w_-|-)$ denote respectively the probability that a negative word appears in a positive and negative document. Given a lexicon of p positive words and n negative words,

$$p \times P(w_+|+) + n \times P(w_-|+) = 1$$

Assume that $P(w_+|+) = r \times P(w_-|+)$, where $r > 1$ is the polarity level. Substitute into the above equation and solve it we have

$$P(w_+|+) = \frac{1}{p + n/r} \quad (1)$$

$$P(w_-|+) = \frac{1}{p \times r + n} \quad (2)$$

Similarly

$$p \times P(w_+|-) + n \times P(w_-|-) = 1.$$

Assuming also that $P(w_-|-) = r \times P(w_+|-)$, we have

$$P(w_-|-) = \frac{1}{n + p/r} \quad (3)$$

$$P(w_+|-) = \frac{1}{n \times r + p} \quad (4)$$

Equations 1 to 4 are estimations for word polarity probability distribution in the positive and negative classes. Using these equations to estimate the probability distribution for any opinion word, the Naive Bayes theorem (described in Section 5) can be employed to classify document sentiment:

$$\operatorname{argmax}_{C_j} P(C_j) \times \prod_i P(w_i|C_j)$$

where $w_i \in \{w_+, w_-\}$ is the prior polarity of an opinion word and $C_j \in \{+, -\}$ is a class label. We refer to this model as the *lexicon polarity model*. As will be discussed in Section 7, our experiments show that this model produces more accurate document sentiment prediction than the naive polarity model.

5 The Baseline Naive Bayes Model for Sentiment Classification

Given a collection of documents with positive or negative sentiment labels as training data, the problem of document sentiment analysis can be cast as a classification problem. Following the Bayes theorem, a document D is classified as the sentiment class $C_j \in \{+, -\}$ as follows:

$$\begin{aligned} & \operatorname{argmax}_{C_j} (P(C_j|D)) \\ &= \operatorname{argmax}_{C_j} (P(C_j, D)/P(D)) \\ &\propto \operatorname{argmax}_{C_j} (P(D|C_j) \times P(C_j)) \end{aligned}$$

The prior probability $P(C_j)$ can be easily calculated from the training data. Given a lexicon (in our setting the opinion lexicon), under the assumption that words for a document are generated independently, the conditional probability of a document D given a class $C_j \in \{+, -\}$ is

$$P(D|C_j) = \prod_i P(w_i|C_j)$$

where w_i is an opinion word appearing in document D . Moreover in the Bayes generative model, a document is a multinomial distribution of words in the lexicon. $P(w_i|C_j)$ is estimated as follows:

$$P(w_i|C_j) = \frac{t_{ij} + 1}{\sum_j (t_{ij} + 1)} = \frac{t_{ij} + 1}{\sum_j (t_{ij}) + K} \quad (5)$$

where K is the number of words in the lexicon, and t_{ij} is the number of occurrences of the word w_i in all documents belonging to class C_j . In the above equation the word frequencies are smoothed by Laplace smoothing to eliminate zeros. In our implementation, the probability tends to be very small, and multiplying lots of such probabilities can result in floating-point underflow. Since $\log(x \times y) = \log x + \log y$, we perform all computations by summing logs of probabilities rather than multiplying probabilities.

We use the above word-based Naive Bayes model as the baseline for sentiment classification and simply refer to it as the *Naive Bayes model*.

6 Holistic Approaches to Sentiment Classification

Note that in the lexicon polarity model only the word polarity distribution is considered, whereas in the Naive Bayes model only the word distribution is considered. With a holistic approach to using opinion lexicons for blog sentiment analysis, the distribution of both words and word polarity should be considered. In [9] a linear pooling approach was proposed. In our setting, the word polarity distributions as described in Equations 1 to 4 and the word distribution as described in Equation 5 are combined by linear pooling as follows:

$$P(w_i|C_j) = \alpha_1 \times P_1(w_i|C_j) + \alpha_2 \times P_2(w_i|C_j)$$

where $P_1(\cdot)$ and $P_2(\cdot)$ denotes $P(w_i|C_j)$ for an opinion word w_i in document sentiment class $C_j \in \{+, -\}$ computed respectively using the lexicon polarity model and the Naive Bayes model. The weights α_1 and α_2 for each model are computed using a sigmoid function as follows:

$$\alpha_i = \log \frac{1 - err_i}{err_i}$$

where err_i ($i=1, 2$) is the error rate for each model. The α_i 's are indeed normalized to sum to one. The $P(w_i|C_j)$ resulted from linear pooling is used in the Naive Bayes model for predicting the sentiment of documents. This model forms the base holistic approach and is referred to as the *linear pooling model*.

As will be shown in Section 7, unfortunately linear pooling does not produce more accurate document sentiment prediction over the original Naive Bayes model. We will next describe two alternative approaches.

6.1 The Polarity Naive Bayes Model

We view that the probability that an opinion word appears in a sentiment class is the same as the probability that the word itself appears in the class and the word resumes its annotated prior polarity in the class. Consider a document D and a document sentiment class $C_j \in \{+, -\}$. For any opinion word $w_i \in D$ and the polarity $p_i \in \{w_+, w_-\}$ that it is associated with, assuming that $P(w_i|C_j)$ and $P(p_i|C_j)$ are independent and opinion words are independent, we have

$$P(D|C_j) = \prod_i P(w_i, p_i|C_j) = \prod_i P(w_i|C_j) \times P(p_i|C_j) \quad (6)$$

In the above equation $P(w_i|C_j)$ can be computed with the Naive Bayes model from training data using Equation 5 and $P(p_i|C_j)$ can be computed with the lexicon polarity model using Equations 1 to 4.

From Equation 6, Bayes theorem can then be applied to predict the sentiment class for any given document D . We refer to this model as the *polarity Naive Bayes model*. As will be described in Section 7, our experiments show that the polarity Naive Bayes model achieves more accurate document sentiment prediction than the baseline Naive Bayes model.

6.2 The Decision Aggregate Model

The linear pooling model and the polarity Naive Bayes model both use the probability derived from word polarity to improve the estimation of word probability. On the other hand however, the Naive Bayes model and the lexicon polarity model can be applied separately to predict the sentiment for documents. If each model produces prediction better than random guess, research on ensemble

learning [15, 11] has shown that aggregating the decisions from both models can often produce significantly more accurate prediction.

We combine the predictions by the baseline Naive Bayes model and the Lexicon Polarity model to predict the document sentiment. We refer to this model as the *decision aggregate model*. Following the Bayes theorem, a document D is classified sentiment class $C_j \in \{+, -\}$:

$$\operatorname{argmax}_{C_j} P(C_j) \times P(D|C_j).$$

As $P(C_j)$ is directly computed from the training data and not affected by the learning models, we only consider $P(D|C_j)$ when combining the two models. Suppose that $P_1(D|C_j)$ and $P_2(D|C_j)$ are the prediction results from the Naive Bayes model and the Lexicon Polarity model respectively. The two probabilities are aggregated as follows:

$$P(D|C_j) = \alpha_1 \times P_1(D|C_j) + \alpha_2 \times P_2(D|C_j),$$

where α_1 and α_2 are the weights computed from the error rates of the two models using the sigmoid function, as described earlier.

Generally aggregating decisions by individual models based on their errors on the training has been used in ensemble learning. The sigmoid weighting scheme was used in boosting [15].

7 Experiments

We conduct experiments to examine the performance of different models for blog sentiment analysis. We will examine the performance of the baseline Naive Bayes model. We will compare our two holistic models against the baseline Naive Bayes model and also against the linear pooling model [9]. All document sentiment classification accuracy reported below are obtained from 10-fold cross validation experiments on our experimental datasets.

7.1 Data Sets

Our experiments were conducted on the TREC Blog06 collection [11], which was used for both TREC-2006 and TREC-2007 Blog Track conferences. Fifty topics were selected for each conference. For a topic, documents are labeled with relevance judgments by professional annotators, with the following categories [11]: not judged (-1), not relevant (0), relevant (1), negative (2), mixed (3) and positive (4). For our sentiment classification task we only consider documents in the positive and negative categories.

The original Blog06 collection is a sample of the blogosphere crawled from 6 December 2005 to 21 February 2006. The whole collection is 148 GB in total, and comprises three components: 38.6 GB XML feeds, 88.8 GB permalink documents and 28.8GB HTML homepages. We only consider the permalink documents for our experiment. To prepare the datasets for experiment, we developed

Table 2. Datasets from 12 topics in the Blog06 collection

Topic	Description	#Positive Doc	#Negative Doc
853	Opinions on President Bush’s 2006 State of the Union address.	50	50
869	Worldwide opinions to the cartoons depicting the Muslim prophet Muhammad printed in a Danish newspaper.	107	107
872	Opinions on the movie Brokeback Mountain.	46	46
875	The public’s opinion on the TV program “American Idol”.	95	103
900	Opinions regarding the food at McDonald’s restaurants.	41	41
903	Opinions about Apple CEO Steve Jobs.	60	60
905	Opinions on the funeral of Coretta Scott King.	63	80
906	Opinions on the World Economic forum in Davos, Switzerland.	38	38
919	Opinions of the drug company Pfizer and its products.	76	82
922	Opinions of shock jock Howard Stern.	61	61
933	Opinions about the Challenger space shuttle disaster.	104	81
945	Documents that show opinions about Bolivia.	41	41

an in-house software to extract the main textual content from the permalink documents. The content of a blog post is defined as the content of the blog post itself and all comments on the post.

We selected 12 topics from the Blog06 collection where our opinion lexicon has relative good coverage for our experiments. For each topic, only opinion words that appear more than 17 times, and documents that contain more than 9 such opinion words are considered. Obviously 17 and 9 are arbitrary numbers to ensure that we can carry out large scale experiments. We further randomly select a roughly equal number of positive and negative documents for each topic. Table 2 summarizes the 12 datasets used in our experiments, and they cover opinion analysis on a wide range of topics, including political figures, entertainment program reviews, restaurants and companies and their products. Our comprehensive experiments will provide us good understanding of the performance of models in different domains.

7.2 Overview of Results

The classification accuracy for various models on 12 topics is summarized in Table 3. The table lists the sentiment classification accuracy for the baseline Naive Bayes model and the Lexicon polarity model, as well as three holistic models, including the linear pooling model, the polarity NB model and the decision aggregate model.

Table 3. Overview of results

topic	Naive Bayes	Lexicon polarity	Linear pooling	Polarity NB	Decision aggregate
853	82.00	56.00	84.00	82.00	85.00
869	57.01	55.61	57.48	57.94	59.35
872	77.17	55.43	81.52	77.17	77.17
875	60.61	52.02	63.64	64.14	63.13
900	65.85	53.66	69.51	68.29	68.29
903	56.67	52.50	60.83	56.67	68.29
905	61.54	55.94	60.14	61.54	61.54
906	63.16	48.68	50.00	63.16	65.79
919	71.52	56.96	71.52	71.52	71.52
922	59.84	49.18	57.38	59.84	60.66
933	61.08	61.08	65.95	65.41	65.41
945	54.88	50.00	54.88	54.88	54.88
Average	64.28	53.92	64.74	65.21	66.75

The best performance is achieved by the decision aggregate model, shown in the last column of the table. The model achieves an average accuracy of 66.75% and it produces significant improvement over the baseline Naive Bayes model of an average accuracy of 64.28% with a p-value of 0.007 in Wilcoxon signed rank test (the same test is used for all later discussions). The accuracy varies from a lowest of 54.88% for the topic of Bolivia (topic 945) to the highest of 85% for the topic of President Bush’s 2006 State of the Union address (topic 853). Across all topics, the decision aggregate model at least keeps the accuracy by Naive Bayes (NB), and very often it achieves significant improvement over the accuracy of NB. It achieves the highest relative improvement of 20.50% for the topic of Apple CEO Steve Jobs (topic 903). Overall our experimental results show that our approach of combining decisions from the Naive Bayes and lexicon polarity classification models is stable and can achieve consistent good classification performance.

The first two columns of Table 3 show the accuracy of two baseline models — the Naive Bayes classification model and the lexicon polarity model. It can be seen that Naive Bayes achieves relatively good accuracy across most domains, ranging from 54.88% for the topic of Bolivia(topic 945) to 82.00% for the topic of President Bush’s 2006 State of the Union address (topic 853). This confirms that sentiment classification can be cast as a text classification problem in machine learning, as was first demonstrated in [13] for sentiment analysis of movie reviews. On the other hand, the lexicon polarity model demonstrates modest accuracy across all domains, ranging from close to random guess (48.68%) for the topic of World Economic forum in Davos, Switzerland (topic 906) to 61.08% for the topic of the Challenger space shuttle disaster (topic 933).

The linear pooling model combines the Naive Bayes approach and lexicon polarity approach for estimating word probability distribution. The model produces modest improvement in accuracy for most topics but can also degrade the accuracy of NB. The most significant relative improvement in accuracy of 7.97% is seen on topic 933. On the other hand, a significant relative decrease of 26.32%

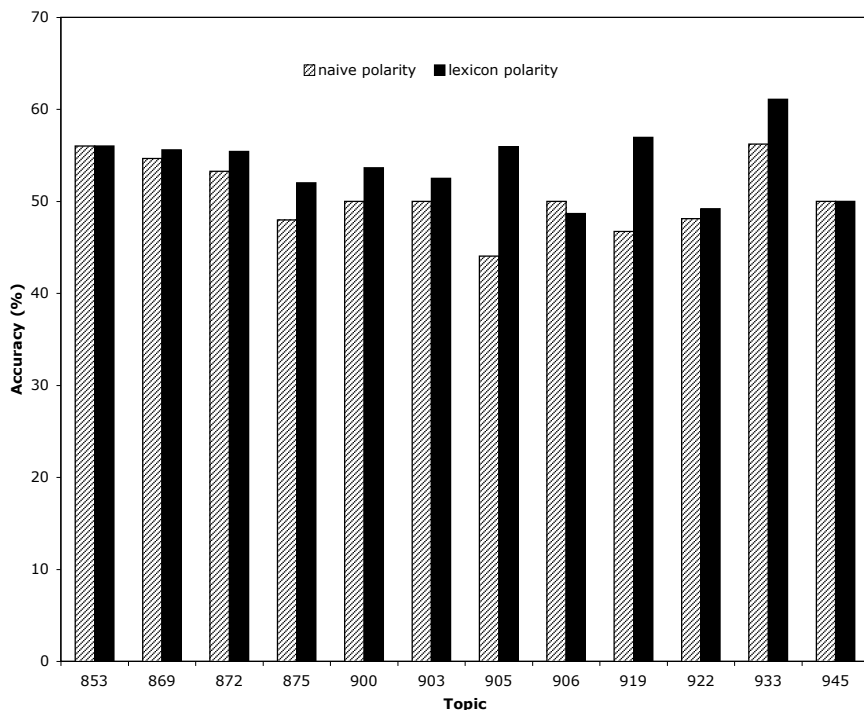


Fig. 1. The performance of the lexicon polarity model vs. the naive polarity model

is shown on topic 906. Overall the linear pooling model does not demonstrate statistically significant improvement over the Naive Bayes model.

The polarity NB model achieves robust sentiment classification for all topics. Compared with the baseline Naive Bayes model, the polarity NB model produces moderate improvement in accuracy on most topics. The largest relative improvement in accuracy is 7.09% for topic 933. Compared with the linear pooling model, it achieves more stable performance and never degrades the performance of NB for any topic. With an average accuracy of 65.21% the improvement in performance of the polarity NB model over the NB model is statistically significant (p -value = 0.05 in Wilcoxon signed rank test).

Generally our experiments demonstrate that the Naive Bayes model based on an opinion lexicon demonstrates good performance for sentiment classification. Although the model based on opinion word polarity only produce modest accuracy, the opinion word polarity provide complementary information towards the overall blog sentiment orientation. Our decision aggregate model that combines decisions from the Naive Bayes model and the lexicon polarity model can achieve consistent improvement in performance than Naive Bayes for blog sentiment classification.

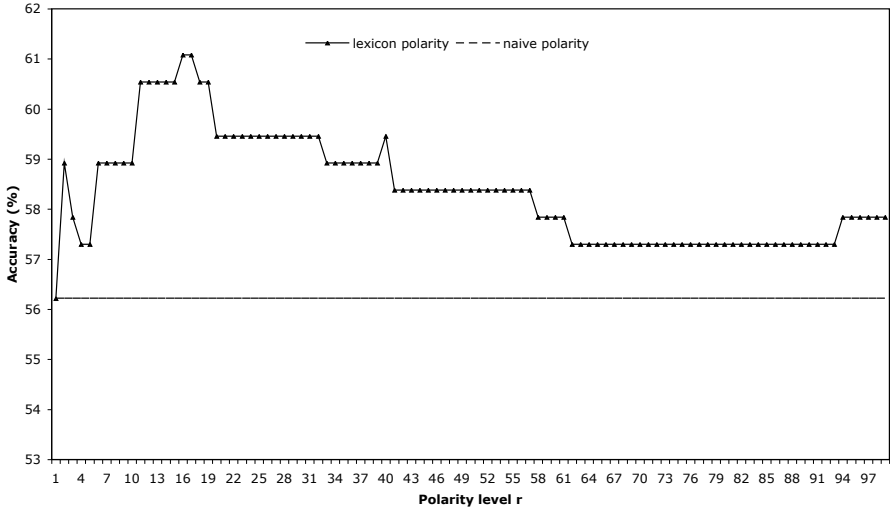


Fig. 2. Accuracy of the lexicon polarity model on topic 933 at different polarity levels

7.3 Results for the Lexicon Polarity Model

We compare the performance of the lexicon polarity model against the naive polarity model. The lexicon polarity model significantly outperforms the naive polarity model in terms of classification accuracy (p -value=0.006). Overall the average accuracy for the lexicon polarity model is 53.92% while that for the naive polarity model is 50.58%. The accuracy of the lexicon polarity model in contrast to the naive polarity model for each domain is shown in Fig. 1. In 8 out of 12 domains, the lexicon polarity model significantly outperforms the naive polarity model, and in the rest 4 domains, lexicon polarity model achieves similar level of accuracy as the naive polarity model.

Results of the lexicon polarity model in Fig. 1 are obtained by setting a suitable polarity level r (as described in Section 4) for each domain. Our experiments show that different settings for r result in moderate variation in the performance of the lexicon polarity model, and for different domains the lexicon polarity model achieves the best performance at a different r . Fig. 2 shows the performance of the lexicon polarity model with respect to different settings of r (from 1 to 100) for topic 933 (the Challenger space shuttle disaster), where the lexicon polarity model achieves the highest prediction accuracy. The naive polarity model accuracy does not vary with different setting for r and so it remains a straight line in Fig. 2. Accuracy of the lexicon polarity model varies from 56.22% to 61.08%. The model achieves the highest accuracy of 61.08% when r is 16 or 17, with an absolute increment of 4.86% over the naive polarity model.

8 Conclusions

In this paper we have investigated the problem of predicting blog sentiment orientation using only an opinion lexicon where opinion words are annotated as positive or negative. We have established that the probability distribution of words and their polarity in the opinion lexicon can be combined to accurately identify blog sentiment. We propose two holistic approaches to document sentiment classification making use of both the words and their polarity distribution information. Especially we propose a model that combines prediction from the Naive Bayes model for words in the lexicon and prediction from the generative model for word polarity in the lexicon. We conducted extensive experiments on blogs from 12 different domains to examine the performance of our proposed models. Our experiments show that our holistic approaches to blog sentiment identification achieve significantly more accurate sentiment prediction than the baseline Naive Bayes model.

Generic opinion lexicons are general knowledge independent of the application domains, and so word polarity distribution remains invariant across domains. This may have negatively affected the performance of the lexicon polarity model. In the future work we will study how to make the opinion lexicon adapt to application domains and how to accurately compute the polarity distribution based on the opinion lexicon for each domain.

Acknowledgment. The authors would like to thank Michael Spiteri for providing the software extracting posts from the permalink documents in the Blog06 collection.

References

1. Breiman, L.: Bagging predictors. *Machine Learning* 24(2), 123–140 (1996)
2. Ding, X., Liu, B., Yu, P.S.: A holistic lexicon-based approach to opinion mining. In: *Proceedings of the International Conference on Web Search and Web Data Mining*, pp. 231–240 (2008)
3. Druck, G., Mann, G., McCallum, A.: Learning from labeled features using generalized expectation criteria. In: *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 595–602 (2008)
4. General-Inquirer: The General Inquirer Home Page (2010), <http://www.wjh.harvard.edu/~inquirer/> (accessed September 22, 2010)
5. Hatzivassiloglou, V., McKeown, K.R.: Predicting the semantic orientation of adjectives. In: *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, p. 181 (1997)
6. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 168–177 (2004)
7. Jin, W., Ho, H.H., Srihari, R.K.: OpinionMiner: a novel machine learning system for web opinion mining and extraction. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1195–1204 (2009)

8. Liu, B.: Sentiment analysis and subjectivity, 2nd edn, Indurkha, N., Damerau, F.J. (2009)
9. Melville, P., Gryc, W., Lawrence, R.D.: Sentiment analysis of blogs by combining lexical knowledge with text classification. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1275–1284 (2009)
10. Ng, V., Dasgupta, S., Arifin, S.M.: Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In: Proceedings of the COLING/ACL on Main Conference Poster Sessions, p. 618 (2006)
11. Ounis, I., Rijke, M.D., Macdonald, C., Mishne, G., Soboroff, I.: Overview of the TREC-2006 blog track. In: Proceedings of TREC, vol. 6 (2006)
12. Pang, B., Lee, L.: Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval* 2(1-2), 1–135 (2008)
13. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL 2002 Conference on Empirical Methods in Natural Language Processing, vol. 10, pp. 79–86 (2002)
14. Riloff, E., Wiebe, J.: Learning extraction patterns for subjective expressions. In: Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, vol. 10, p. 112 (2003)
15. Schapire, R.E.: The strength of weak learnability. *Machine Learning* 5(2), 197–227 (1990)
16. Sindhvani, V., Melville, P.: Document-word co-regularization for semi-supervised sentiment analysis. In: Perner, P. (ed.) *ICDM 2008*. LNCS (LNAI), vol. 5077, pp. 1025–1030. Springer, Heidelberg (2008)
17. Tang, H., Tan, S., Cheng, X.: A survey on sentiment detection of reviews. *Expert Systems with Applications* 36(7), 10760–10773 (2009)
18. Turney, P.D.: Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 417–424 (2002)
19. Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation* 39(2), 165–210 (2005)
20. Wilson, T., Hoffmann, P., Somasundaran, S., Kessler, J., Wiebe, J., Choi, Y., Cardie, C., Riloff, E., Patwardhan, S.: OpinionFinder: a system for subjectivity analysis. In: Proceedings of HLT/EMNLP on Interactive Demonstrations, p. 35 (2005)
21. Wilson, T., Wiebe, J.: Annotating opinions in the world press. In: 4th SIGdial Workshop on Discourse and Dialogue (SIGdial 2003) (2003)
22. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, p. 354 (2005)

Characterizing Web Syndication Behavior and Content

Zeinab Hmedeh¹, Nelly Vouzoukidou², Nicolas Travers¹,
Vassilis Christophides², Cedric du Mouza¹, and Michel Scholl¹

¹ CEDRIC Laboratory - CNAM - Paris, France

`firstname.lastname@cnam.fr`

² FORTH/ICS and Univ. of Crete - Heraklion, Greece

`{christop,vuzukid}@ics.forth.gr`

Abstract. We are witnessing a widespread of web syndication technologies such as RSS or Atom for a timely delivery of frequently updated Web content. Almost every personal weblog, news portal, or discussion forum employs nowadays RSS/Atom feeds for enhancing *pull-oriented* searching and browsing of web pages with *push-oriented* protocols of web content. Social media applications such as Twitter or Facebook also employ RSS for notifying users about the newly available posts of their preferred friends. Unfortunately, previous works on RSS/Atom statistical characteristics do not provide a precise and updated characterization of feeds' behavior and content, characterization which can be used to successfully benchmark effectiveness and efficiency of various RSS processing/analysis techniques. In this paper, we present the first thorough analysis of three complementary features of real-scale RSS feeds, namely, publication activity, items structure and length, as well as, vocabulary of its content which we believe are crucial for Web 2.0 applications.

Keywords: RSS/Atom Feeds, Publication activity, Items structure and length, textual vocabulary composition and evolution.

1 Introduction

Web 2.0 technologies have transformed the Web from a publishing-only environment into a vibrant information place where yesterday's end users become nowadays content generators themselves. Web syndication formats such as RSS or Atom emerge as a popular mean for timely delivery of frequently updated Web content. According to these formats, information publishers provide brief summaries of the content they deliver on the Web, called *items*, while information consumers subscribe to a number of RSS/Atom *feeds* (i.e., streams or channels) and get informed about newly published items. Today, almost every personal weblog, news portal, or discussion forum employs RSS/Atom feeds for enhancing traditional *pull-oriented* searching and browsing of web pages with *push-oriented* protocols of web content. Furthermore, social media applications such as Twitter or Facebook also employ RSS for notifying users about the newly available posts of their preferred friends (or followees).

Unfortunately, previous works on RSS/Atom statistical characteristics [15,27,13] do not provide a precise and updated characterization of feeds' behavior and content which could be effectively used for tuning refreshing policies of RSS aggregators [24,22], benchmarking scalability and performance of RSS continuous monitoring mechanisms [19,9,7,8,25,5] or comparing various techniques for RSS items mining, recommendation, enrichment and archiving [3,26]. In this paper, we present the first thorough analysis of three complementary features of real-scale RSS/Atom feeds, namely, *publication activity*, *items structure and length*, as well as, *vocabulary of the textual content*.

Our empirical study relies on a large-scale testbed acquired over a 8 month campaign from March 2010 in the context of the French ANR project Roses¹. We collected 10,794,285 items originating from 8,155 productive feeds (spanning over 2,930 different hosting sites) out of 12,611 feeds, without communication and validation errors against RSS/Atom specifications, harvested from major RSS/Atom directories, portals and search engines (such as syndic8.com, Google Reader, feedmil.com, completeRSS.com, etc.). Then, we have identified six representative types of information sources delivering their content as RSS/Atom feeds: **Press** (for newspapers and agencies), **Blogs** (for personal weblogs), **Forums** (for discussion and mailing lists), **Sales** (for marketing web sites), **Social media** (for micro-blogging such as twitter, digg, yahoo! groups, and blogosphere) and **Misc.** (*e.g.*, for news sites with medical or city/group information as well as podcasts). To the best of our knowledge, we are the first to detail the composition of our testbed whose type and number of sources is representative of the web syndication universe. The acquired RSS/Atom items are stored into a local warehouse² using MySQL, where for each feed only one item occurrence is kept. Since their *pubDate* was not made always available by the originating RSS/Atom feeds, we timestamped items based on their acquisition time in the warehouse. The main conclusions drawn from our experimental study are:

1. As analyzed in section 2, 17% of RSS/Atom feeds produce 97% of the items of the testbed. In their majority, productive feeds (*i.e.* with >10 items per day) exhibit a regular behavior without publication bursts, thus are more predictable in their publication behavior. As expected, micro-blogging feeds from social media are more productive than those from blogs while press sources lie in between. The average publication rate among all feeds of the testbed has been measured to be 3.59 items a day;
2. As highlighted in section 3, the most popular RSS/Atom textual elements are *title* and *description* while the average length of items is 52 terms (which has not been reported so far in related work). It is clearly advertisement greater than bids (4-5 terms [10]) or tweets (15 terms at most [21]) but smaller than blogs (250-300 terms [16]) or Web pages (450-500 terms excluding tags [14]). In addition, re-publication of items across feeds is rare since we identified only 0.41% of duplicates among distinct feeds hosted by different sites;

¹ <http://www-bd.lip6.fr/roses>

² Available on line at deptmedia.cnam.fr/~traversn/roses

3. As studied in section 4, the total number of extracted terms from items written in English is 1,537,730 out of which only a small fraction (around 4%) is found in the WordNet dictionary. This is due to the heavy use in RSS/Atom textual elements of named entities (person and place names), URLs and email addresses as well as numerous typos or special-purpose jargon. We formally characterized the total vocabulary growth using Heaps' law, as well as the number of occurrences of the corresponding terms with a *stretched exponential distribution* used for the first time in the literature in this respect. We observed that the ranking of vocabulary terms does not significantly vary during the 8 month period of the study for frequent terms.

2 Feeds Analysis

In this Section we are interested in characterizing the composition of our testbed in terms of RSS/Atom feeds' type are originating from, as well as, in studying their publication activity. Although global statistics regarding Web2.0 activity are constantly monitored³, an in depth analysis of RSS feeds productivity was not already reported in the literature. Knowing that highly active feeds are more predictable in their publication behavior, would guide for instance resource allocation mechanisms to be tied to the feeds category.

2.1 Source Type

Six types of information sources delivering their content as RSS/Atom feeds were identified: **Press** (for newspapers and agencies), **Blogs** (for personal weblogs), **Forums** (for discussion and mailing lists), **Sales** (for marketing web sites), **Social media** (for micro-blogging such as twitter, digg, yahoo! groups, and blogosphere) and **Misc.** (*e.g.*, for news, medical, city/group information, podcasts). Then, the 8,155 productive feeds were classified under these six types using information available in feeds' title, category, link or in specific hosting sites.

Table 1. Source types of RSS/Atom feeds

Type	% of feeds	% of items	$\frac{\# \text{ items}}{\# \text{ feeds}}$
Social Media	1.77%	9.45%	7085.03
Press	9.99%	38.82%	5141.24
Forum	1.51%	3.62 %	3178.01
Sales	11.32%	15.49%	1811.92
Misc.	41.47%	25.47%	812.99
Blog	33.93%	7.14%	278.55

Table 1 depicts for each type, the corresponding percentage of feeds and items as well as the average number of items per feed. **Social media**, **Press** and **Forums** are more productive (with an average number of items per feed ranging

³ <http://thefuturebuzz.com/2009/01/12/social-media-web-20-internet-numbers-stats>

from 3178.01 to 7085.03) than **Sales**, **Misc** and **Blogs** (with less than 2000 items on average per feed). As discussed in the following sections, feeds’ behavior can be further refined by considering daily publication rates as well as the corresponding activity variability over time. We believe that the composition of our testbed is representative of the Web 2.0 universe. For instance the fact that blogs as opposed to Social media provide low productive feeds compared to their number, as well as the fact that numerous Press and Sales feeds are actually available, are reflected in the composition of our testbed.

2.2 Publication Activity

A time-agnostic characterization of RSS feeds activity has been originally proposed in [24] by studying the distribution of the number of feeds publishing at a given rate x . A similar power law behavior ax^b was observed for the 8,155 feeds, although with different coefficients $a = 1.8 \times 10^3$ and $b = -1.1$. This is due to the presence in our testbed of more productive feeds than in [24] (featuring more blog-originating feeds). A coarse-grained characterization of the temporal variation of feeds activity has been suggested in [27] which analyzes publication burstiness. With a similar burst definition of at least 5 times the average publication rate during a unit of time (a day), 89% of our feeds produce bursts. However, the remaining 11% of feeds without bursts produce more than 81% of the items. Thus, this burstiness measure is not sufficient for a precise characterization of feeds activity.

We prefer the *Gini coefficient* [20], denoted as G , to characterize the variability of feeds’ publication activity over time. G is adequate for time series (feeds are time series), since it does not only take into account the deviation from a mean value (as in the case of burstiness) but also the temporal variation of this deviation. It has been widely used for analyzing a variety of phenomena arising in economy, geography, engineering, or in computer science (e.g., self-join size, supervised learning). G is defined as:

$$G = \frac{\sum_{i=1}^n \sum_{j=1}^n |y_i - y_j|}{2 \times n \times \sum_{i=1}^n y_i}$$

with y_i denoting the number of items, sorted in increasing order, over the n days. A G value close to 1 suggests that the number of items of a feed significantly varies over time.

Figure 1 depicts in log-log-scale G vs the average publication rate for each of the 8,155 feeds. Feeds with a G value less than 0.02293 (below the horizontal dashed line) did not exhibit a single burst during the entire 8-month period. Three classes of feed activity are identified. The first one called “**productive class**”, comprises 614 feeds which produce more than 10 items a day with a very low temporal variation (G less than 0.03). The second one called “**moderate class**”, gathers 1,677 feeds that publish between 1 and 10 items per day with a moderately low temporal variation (G less than 0.1). The third one called “**slow class**”, represents the large majority of the feeds. They publish less than 1 item a day (0.23 on average) and exhibit a strong temporal variation in the number

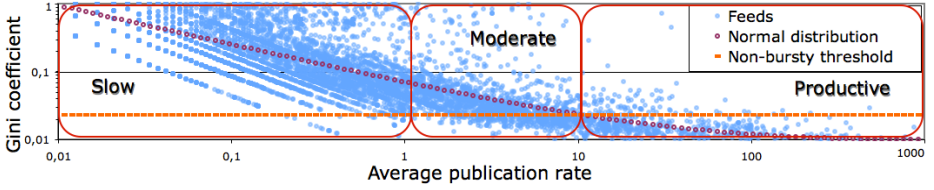


Fig. 1. Activity classes of RSS/Atom feeds

of items ($G = 0.32$ on average). The average G value as a function of the average publication rate p (dotted curve) is approximated by the following function:

$$G(p) = 5.53 \times 10^{-2} \times p^{-0.59} + 4.48 \times 10^{-3} \times p^{0.101}$$

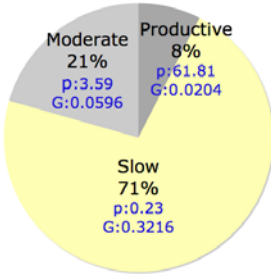


Fig. 2. Feeds per activity class

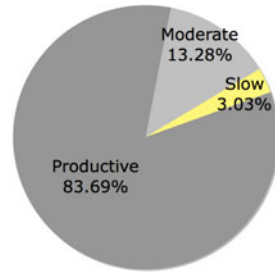


Fig. 3. Items per activity class

Unlike [15] reporting that 57% of the feeds have a publication rate > 24 items a day, in Figure 2 we can see that only 8% of our 8,155 feeds were actually very productive (with > 10 items a day) while 71% had a low publication activity. Note also that from the initially harvested 12,611 feeds around 35% did not publish any item during the entire 8-month period. Of course productive feeds, even though they represent a minority, account for the majority of the items harvested. We see in Figure 3 that feeds of productive class publish 83.69% of the total number of items while the majority of feeds from slow class produce only 3.03% of the items (Power Law behavior of feeds' publication rate).

Table 2. Source types and activity classes of feeds

Type	productive class			moderate class			slow class		
	%	p	G	%	p	G	%	p	G
Social Media	44.9%	65.33	.015	8.1%	5.57	.026	47.0%	0.18	.470
Press	27.4%	70.76	.020	43.0%	3.61	.036	29.6%	0.34	.237
Forum	21.0%	54.51	.018	20.2%	3.85	.042	58.8%	0.29	.348
Sales	8.3%	83.64	.022	19.0%	3.43	.122	72.7%	0.23	.454
Misc	2.7%	63.12	.024	12.9%	3.70	.059	84.4%	0.23	.338
Blog	4.3%	15.53	.019	24.7%	3.37	.056	71%	0.22	.267

Last but not least, Table 2 provides for each source type and activity class, the corresponding percentage of feeds, the average publication rate (p), as well as, the average Gini coefficient (G). Feeds from Social Media are almost evenly distributed in productive and slow classes with a temporal variation in their

publication rate which is more important in the latter than in the former class. This is related to *Twitter* like behavior with *Followers* and *Followees* (users can follow - slow class - or users can be followed and produce a lot - productive class). Press feeds exhibit a **moderate** and almost *regular* publication activity, while feeds originating from Forums, Sales, Misc and Blogs sites mainly exhibit a **slow** publication activity. It is worth also noticing that in this class, Sales are mostly bursty feeds while Blogs and Forums exhibit a more regular behavior.

3 Items Analysis

This section successively focuses on the analysis of items' structure and length as well as on the items' replication rate across RSS/Atom feeds which could be exploited for benchmarking scalability and performance of RSS continuous monitoring mechanisms.

3.1 Item Structure

The empirical analysis presented in Table 3, reveals that a great number of fields (tags) foreseen in the XML specification of RSS or Atom formats are actually not utilized in items. While title, description and link are present in almost all the items, pubDate is missing in around 20% of the items, and the language information is missing in 30% of the feeds. Almost 2/3 of the items are not *categorized* while author information is present in less than 8% of the items. It is worth noticing that 16% of fields contain errors or are used with a wrong format (mixing RSS or Atom fields). Other XML tags are only sparsely used in our testbed and are not reported here. In a nutshell, RSS/Atom items are characterized by the predominance of textual information as provided by title and description over factual information provided by category and author.

Table 3. Popularity of XML tags in RSS items

title	link	pubDate	desc.	Language
99.82%	99.88%	80.01%	98.09%	69.14% (feed)
author	category	GUID	ext.	RSS/Atom
7.51%	33.94%	69.50%	29.73%	16.48%

3.2 Item Length

Next, we focus on the length of items measured as the number of terms of the textual title and description fields. Items are short with a 52-terms size on the average (see Table 4). The high variance (11,885) of items length is mainly due to the large diversity of the description fields that can be either missing (or be a simple *wrt*) or oppositely be a long text (even an entire HTML document).

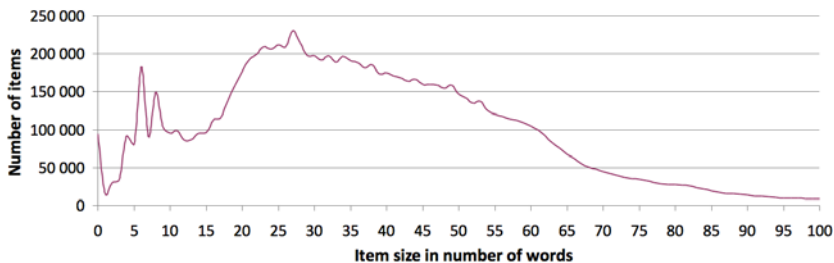
Figure 4 plots the number of items vs their length. A long-tail curve is observed in items length distribution as also reported in the literature for the size of Web

Table 4. Textual content characteristics in items

	title + description	title	description
Average	52.37	6.81	45.56
Max	10,262	235	10,256
Variance	11,885.90	12.97	11,821.36

documents [28]. 51.39% of the items have a length between 21 and 50 terms, and 14% between 8 and 20 terms. The peaks for length 6 and 8 are mainly due to **Sales** feeds (producing >55% of the items for these lengths) whose items respect a fixed-length pattern (*i.e.* items differ only by one or two terms).

The main conclusion from this analysis is that RSS/Atom items are longer than average advertisement bids (4-5 terms [10]) or tweets (~ 15 terms at most [21]) but smaller than the original blog posts (250-300 terms [16]) or Web pages (450-500 terms excluding tags [14]).

**Fig. 4.** Number of items per item length

3.3 Item Replication

Finally, we stick our attention to the replication of items either by feeds hosted by the same site (*intra-host* replication) or across different feeds (*inter-feed* replication). Replication detection is performed by exact matching of the item content based on a hash function. Out of the originally harvested 27 millions of items, 10.7 million distinct items per feed were extracted. Eliminating *intra-host* and *inter-feed* replication makes this number drop to 9,771,524 of items.

Table 5. % Hosts/items per intra-host replication

% of replication	< 10%	10 - 19%	20 - 29%	$\geq 30\%$
% of hosts	95.19%	1.09%	0.68%	3.04%
% of items	71.31%	10.88%	10.23%	7.58%

Table 5 reports the importance of *intra-host* replication that was measured for the 2,930 hosts (with distinct IPs) of the 8,155 feeds. 95% of the hosts (which publish 70% of the items) have less than 10% of replication. Only a few hosts

Table 6. % of items replicated across feeds

# distinct feeds	1	2	≥ 3
% items	99.51%	0.41%	0.08%

(less than 5%) of Sales, Press (especially news agencies) and Blogs feeds publish most of the replicated items (replication rate $> 30\%$).

Once intra-host replicates had been removed, *inter-feed* item replication was measured. Table 6 reports for each replicated item the number of *distinct* feeds in which it appears. Clearly replication across feeds in different hosts is negligible: it accounts for less than 0.5% of the total number of items. This behavior can be explained by the absence in the testbed of RSS aggregators such as GoogleReader⁴ or Yahoo Pipes⁵ which systematically replicate other feeds.

4 Vocabulary Analysis

This Section focuses on the analysis of the vocabulary of terms extracted from the title and description fields of RSS/Atom items. In order to deduce information regarding the quality of the employed vocabulary (valid terms/typos), we restrict our analysis to the English language, since a large majority of the analysed items in our testbed was written in English. In addition, as previous studies report, a similar behavior to that of the English language is exhibited for text corpora written in different languages (e.g. French, Korean [4], Greek [6], etc.). An automatic filtering of feeds (items) written in English turned out to be a difficult task given that language information was not always available.

7,691,008 “English” items were extracted as follows: an item is considered as English if the language tag exists in the feed metadata and is set to “en”, or the feed url belongs to an English spoken host (e.g. “us” or “uk”), or a “.com” feed without any language tag. Then, a lexical analysis of items’ textual contents was conducted using standard stemming tools (such as SnowBall⁶) and dictionaries (such as WordNet⁷) for the English language. In particular, we distinguish between V_W , the vocabulary of (61,845) terms appearing in the WordNet dictionary, from $V_{\overline{W}}$, the vocabulary of remaining (1,475,885) terms composed mostly of jargon, named entities and typos. Stop-words⁸, *urls* and *e-mail* addresses were excluded and for $V_{\overline{W}}$ the stemmed version of terms was kept while for V_W the terms in their base form as given by WordNet was added.

We believe that such a detailed characterization of feeds content will be beneficial to several Web 2.0 applications. For instance, knowledge regarding terms (co-)occurrence distribution could be helpful for efficient compression or indexing techniques of items textual content. The knowledge that the employed vocabulary is essentially composed of misspellings, spoken acronyms and morphological

⁴ www.google.com/reader

⁵ pipes.yahoo.com

⁶ snowball.tartarus.org

⁷ wordnet.princeton.edu

⁸ www.lextek.com/manuals/onix/stopwords2.html

variants will greatly affect the choice of effective ranking functions for filtering the incoming items while it will enable us to devise realistic workloads for measuring the scalability and performance of Web 2.0 analysis systems (e.g., keyword tracking, buzz measuring, keyword-association in online consumer intelligence).

4.1 Term Occurrences

The global vocabulary ($V=V_{\mathcal{W}}\cup V_{\overline{\mathcal{W}}}$) extracted from the English items reaches 1,537,730 terms. Figure 5 depicts the occurrences of terms belonging to $V_{\mathcal{W}}$ and to $V_{\overline{\mathcal{W}}}$ in decreasing order of their rank (frequency) in their respective vocabulary. As expected, terms from $V_{\mathcal{W}}$ are much more frequent than terms from $V_{\overline{\mathcal{W}}}$. The multiple occurrences in the items of the 5,000 most frequent terms of $V_{\mathcal{W}}$ (around 8% of its size) represent 87% of the total number of term occurrences from $V_{\mathcal{W}}$. The percentage of $V_{\mathcal{W}}$ terms appearing in the most frequent terms of V drops quickly: from 90% in the 5,000 first terms, to 78% for the next 5,000, to 50% after 20,000 terms and to only 3% for the remaining 1.5 million terms.

In the following, we are interested in characterizing formally the distribution of $V_{\mathcal{W}}$ and $V_{\overline{\mathcal{W}}}$ terms' occurrences. In this respect, Zipf's law distributions have been traditionally used in the literature [2,17] for various text corpora:

$$f(r) = \frac{K}{r^\theta}$$

where r is the term rank and θ and K are constants.

However, as can be seen in Figure 5 the corresponding curve for $V_{\mathcal{W}}$ has a significant deviation from Zipf's law, *i.e.*, from a straight line in log-log scale. This deviation is smaller for the $V_{\overline{\mathcal{W}}}$ curve. Similar deviations have been already reported for web related text collections [2,17,11,28,10] and few attempts have been made to devise more adequate distributions. [18] tried to generalize the Zipf's law by proposing a Zipf - Mandelbrot distribution while [13] suggested to use a Modified Power Law distribution. Although the latter laws can approximate the slow decrease at the beginning of the curves, their tail in log-log scale is close to a straight line, which does not fit neither to our terms' occurrences nor to any of the distributions in the aforementioned studies.

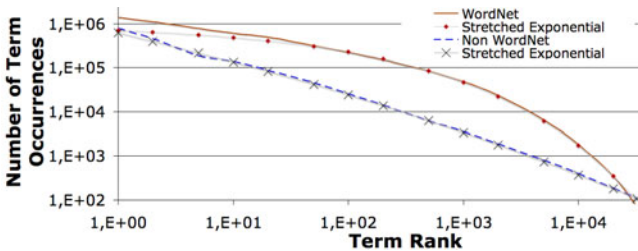


Fig. 5. Occurrences of terms from $V_{\mathcal{W}}$ and $V_{\overline{\mathcal{W}}}$

A *stretched exponential* distribution was found to better capture the tail of the distributions for both vocabularies. It is defined as follows:

$$f(r) = K \times e^{-(r/c)^\theta}$$

where θ expresses how fast the slope of the curve decreases, while constant c defines the curvature: for values closer to zero the distribution is closer to a straight line in log-log scale. To the best of our knowledge the stretched exponential distribution has been so far utilized to study physics phenomena but has never been associated with term distributions before, even though [12] suggests this distribution as an alternative to power laws. The constants that lead to the best fitting curves were obtained using the chi-square test (see Table 7). Lower values of chi-square indicate a better fit of the distribution to the empirical data.

Table 7. Chi-square for 4 distributions (deg. of freedom: 210 for V_W and 200 for $V_{\overline{W}}$)

	Zipf	Zipf Mandelbrot	Modified power law	Stretched Exponential
V_W	2,845	2,493	966	49.8
$V_{\overline{W}}$	11.5	11.5	9.2	9
Term co-currence	2,353	1,543	31,653	45

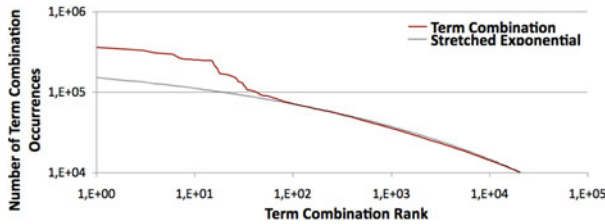


Fig. 6. Co-occurrences of terms from V

Figure 6 represents the occurrences of the 32,000 most frequent term combinations in items. These combinations are all pairs of terms, not necessarily consecutive, that appear in the item’s title or description regardless of the vocabulary they belong to. In contrast to [10], where the same test was performed for sponsored data, we find that this distribution does not follow the Zipf’s law. In Figure 6 the curve has been fitted using the stretched exponential distribution whose chi-square value is the lowest one (see table 7). The divergence of the empirical data from the stretched exponential distribution given above is limited to the 32 most frequent combinations of terms, which appear almost 4 times more than expected. These combinations are essentially due to high interests for some serials (*e.g.* like *Lost-fan*), a behavior also observed in [10].

Table 8. Terms’ number per occurrences

Occurrences	1	2	3	4	≥ 5
# terms	659,159	261,906	124,450	77,814	414,401

As the number of occurrences varies widely from one term to another (see Figure 5) in Table 8 we provide a summary of term occurrences distribution. Terms with less than 5 occurrences represent roughly 76% of the global vocabulary

size while terms with only one occurrence account for 45% of the vocabulary. A qualitative analysis over two samples of 5,000 terms comprising respectively the most and the less frequent terms of $V_{\overline{W}}$ showed that 61% of most frequent terms are named-entities and 27% are acronyms or shortcuts. On the other hand, less frequent terms are named-entities (35%), foreign-words (22%), composed and concatenation of words (22%) and misspelling (14%). Clearly, language of non-authoritative web document collections on diverse topics is subject to many kinds of imperfections and errors [1], like typos, mistake or slang, but also special-purpose jargon (*e.g.* technical terms in medicine or computer science).

4.2 Vocabulary Size

Figure 7 illustrates the size growth wrt the number of items progressively stored in our warehouse for the two vocabularies of terms. Clearly, the size of $V_{\overline{W}}$ evolves much faster than that of V_W : at the end of the 8-month period, it is 24 times bigger than the size of the former ($|V_W| = 61,845$ while $|V_{\overline{W}}| = 1,475,885$).

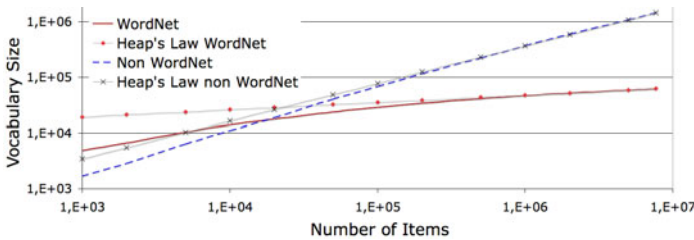


Fig. 7. Evolution of the two vocabularies

For V_W , a rapid increase is observed for the first 140,000 items to reach the 31,000 most frequent terms of V_W , while the size barely doubles up to the end, with the inclusion of less frequent terms. On the other hand, $V_{\overline{W}}$ has a smoother growth: for 140,000 items $V_{\overline{W}}$ comprises 88,000 terms, *i.e.* only 6% of the total $V_{\overline{W}}$ size which reaches almost 1.5 million terms after processing all items.

The vocabulary size growth is usually characterized by a Heap's law distribution [2,17] (see Figure 7), defined as follows:

$$|V(n)| = K \times n^\beta$$

where n is the number of collected items while K and β (values in $[0,1]$) are constants depending strongly on the characteristics of the analyzed text corpora and on the model used to extract the terms [28]. β determines how fast the vocabulary evolves over time with typical values ranging between 0.4 and 0.6 for medium-size and homogeneous text collections [2]. [28] reports a Heap's law exponent lying outside this range ($\beta=0.16$) for a 500 MB collection of documents from the Wall Street Journal. Table 9 specifies the constants chosen for Heap's laws approximating the global vocabulary growth as well as V_W and $V_{\overline{W}}$.

Clearly, the Heap's law exponent (β) of the global vocabulary is affected by the evolution of $V_{\overline{W}}$ rather than by V_W whose size is significantly smaller (attributed to the slow acquisition of less commonly used terms). The exponent for

$V_{\mathcal{W}}$ (0.675) slightly higher than those reported in the literature [2,17] indicates a faster increase of the vocabulary size due the aforementioned language imperfections of the items in our testbed. It should be stressed that this behavior is also exhibited by the evolution of vocabularies related to other user-generated textual content (such as web queries [29]). Finally, the high Heap’s law coefficient (K) for $V_{\mathcal{W}}$ is explained by the rapid vocabulary growth in the beginning due to the fast acquisition of the very popular terms.

Table 9. Heap Laws constants

	$ V_{\mathcal{W}} + V_{\mathcal{W}} $	$ V_{\mathcal{W}} $	$ V_{\mathcal{W}} $
K	51	7,921	33
β	0.65	0.13	0.675

This behavior is also observed in web queries vocabulary evolution [29,23] where the exponents of the Heap’s law exceed the upper bound of 0.6. In the matching Heap’s law of [29], the exponent of the vocabulary distribution for web queries was 0.69, while in [23] it was 0.8136. In both items and web queries, the text is sent once by the users or publishers and is never corrected (i.e. in a case of misspelling), which leads to a higher number of errors and therefore a faster than expected vocabulary evolution. Web documents, on the other hand, are frequently updated and possible misspellings are corrected.

4.3 Rank Variation

Since term occurrences evolves over time as new items arrive, their corresponding ranking could be also subject to change. Figure 8 illustrates the average variation of ranks during the last month of our study. A rank variation is measured by the difference of a term rank between week $t-1$ and week t . Obviously, this variation is proportional to the rank, i.e. less frequent terms are more likely to change their rank compared to the vocabulary of the previous week.

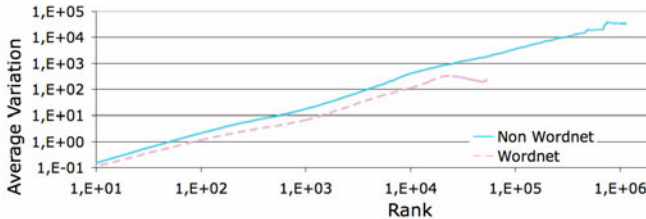


Fig. 8. Cumulative rank distance in V (last week)

We are finally interested in studying the weekly rank variation for specific rank ranges of $V_{\mathcal{W}}$ terms for the 8 months of our study using the *Spearman* metric [11]. The *Spearman* metric is the sum of rank variations for a range of ranks from a week to another:

$$S(t) = \sum_{i=1}^n |r(t)_i - r(t-1)_i|$$

where t is a given week, $r(t)_i$ and $r(t-1)_i$ are the rank of the term i at week t and $t-1$ respectively. Figure 9 depicts the weekly Spearman value for three classes of terms' ranks: (a) highly-frequent terms (ranks between 1 and 250), (b) commonly-used terms (10,000 to 10,250); and (c) rare ones (500,000 to 500,250). Note that the reference vocabulary for this experiment is chosen as the vocabulary obtained after the first three months of items' acquisition.

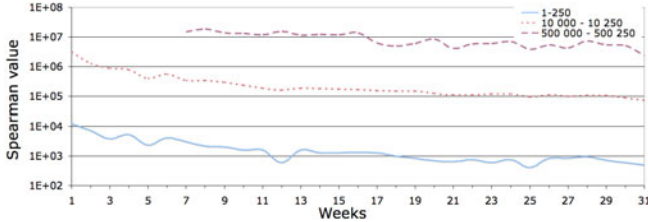


Fig. 9. Weekly Spearman Variation for three rank ranges of $V_{\overline{W}}$ terms

Surprisingly enough, we have observed that the ranking of vocabulary terms does not significantly vary during the 8-month period of the study for highly-frequent terms. As a matter of fact, the sum of rank variations of highly-frequent terms is 4 orders of magnitude less than the corresponding sum for rare terms (where commonly-used terms lie in between). We notice that all Spearman values stabilize after 20 weeks. This can be justified by the fact that the number of occurrences of each term decreases the impact of incoming terms on the rank.

5 Summary

In this paper we presented an in-depth analysis of a large testbed of 8,155 RSS feeds comprising 10.7 million of items collected over a 8-month period campaign. We proposed a characterization of feeds according to their publication rate and temporal variability highlighting three different activity classes. In addition, we focused on RSS items length, structure and replication rate. Last but not least, a detailed study of the vocabulary employed in a significant subset of the RSS items written in English was conducted along three lines: occurrence distribution of the terms, evolution of the vocabulary size and evolution of term ranks.

References

1. Ahmad, F., Kondrak, G.: Learning a Spelling Error Model from Search Query Logs. In: EMNLP (2005)
2. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press/Addison-Wesley (1999)
3. Bouras, C., Pouloupoulos, V., Tsogkas, V.: Creating Dynamic, Personalized RSS Summaries. In: ICDM, pp. 1–15 (2008)

4. Choi, S.-W.: Some statistical properties and zipf's law in korean text corpus. *JQL* 7(1), 19–30 (2000)
5. Haghani, P., Michel, S., Aberer, K.: The gist of everything new: personalized top-k processing over web 2.0 streams. In: *CIKM*, pp. 489–498. ACM, New York (2010)
6. Hatzigeorgiu, N., Mikros, G., Carayannis, G.: Word length, word frequencies and zipf's law in the greek language. *JQL* 8(3), 175–185 (2001)
7. Hristidis, V., Valdivia, O., Vlachos, M., Yu, P.S.: A System for Keyword Search on Textual Streams. In: *SDM* (2007)
8. Hu, C.-L., Chou, C.-K.: RSS Watchdog: an Instant Event Monitor on Real Online News Streams. In: *CIKM*, pp. 2097–2098 (2009)
9. Irmak, U., Mihaylov, S., Suel, T., Ganguly, S., Izmailov, R.: Efficient Query Subscription Processing for Prospective Search Engines. In: *USENIX*, pp. 375–380 (2006)
10. König, A.C., Church, K.W., Markov, M.: A Data Structure for Sponsored Search. In: *ICDE*, pp. 90–101 (2009)
11. Kumar, R., Vassilvitskii, S.: Generalized distances between rankings. In: *WWW*, pp. 571–580 (2010)
12. Laherrère, J., Sornette, D.: Stretched exponential distributions in nature and economy: "fat tails" with characteristic scales. *Eur. Phys. J. B* 2(4), 525–539 (1998)
13. Lambiotte, R., Ausloos, M., Thelwall, M.: Word Statistics in Blogs and RSS Feeds: Towards Empirical Universal Evidence. In: *CoRR* (2007)
14. Levering, R., Cutler, M.: The portrait of a common html web page. In: *ACM Symp. on Document Engineering*, pp. 198–204 (2006)
15. Liu, H., Ramasubramanian, V., Siner, E.G.: Client Behavior and Feed Characteristics of RSS, a Publish-Subscribe System for Web Micronews. In: *IMC*, p. 3 (2005)
16. Ma, S., Zhang, Q.: A Study on Content and Management Style of Corporate Blogs. In: *HCI*, vol. 15, pp. 116–123 (2007)
17. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press, Cambridge (2008)
18. Montemurro, M.A.: Beyond the zipf-mandelbrot law in quantitative linguistics. *Physica A* 300(3-4), 567–578 (2001)
19. Petrovic, M., Liu, H., Jacobsen, H.-A.: CMS-ToPSS: Efficient Dissemination of RSS Documents. In: *VLDB*, pp. 1279–1282 (2005)
20. Pitoura, T., Triantafillou, P.: Self-join size estimation in large-scale distributed data systems. In: *ICDE*, pp. 764–773 (2008)
21. Press, O.U.: Rt this: Oup dictionary team monitors twitterer's tweets (June 2009)
22. Roitman, H., Carmel, D., Yom-Tov, E.: Maintaining dynamic channel profiles on the web. *VLDB* 1(1), 151–162 (2008)
23. Schmidt-Maenz, N., Koch, M.: Patterns in search queries. In: *Data Analysis and Decision Support* (2005)
24. Sia, K.C., Cho, J., Cho, H.-K.: Efficient monitoring algorithm for fast news alerts. *TKDE* 19, 950–961 (2007)
25. Silberstein, A., Terrace, J., Cooper, B.F., Ramakrishnan, R.: Feeding frenzy: selectively materializing users' event feeds. In: *SIGMOD*, pp. 831–842 (2010)
26. Taddesse, F.G., Tekli, J., Chbeir, R., Viviani, M., Yetongnon, K.: Semantic-Based Merging of RSS Items. In: *WWW*, vol. 13(1-2), pp. 169–207 (2010)
27. Thelwall, M., Prabowo, R., Fairclough, R.: Are Raw RSS Feeds Suitable for Broad Issue Scanning? A Science Concern Case Study. *JASIST* 57(12), 1644–1654 (2006)
28. Williams, H.E., Zobel, J.: Searchable words on the Web. *JODL* 5(2), 99–105 (2005)
29. Zien, J.Y., Meyer, J., Tomlin, J.A., Liu, J.: Web Query Characteristics and their Implications on Search Engines. In: *WWW* (2001)

Software-as-a-Service in Small and Medium Enterprises: An Empirical Attitude Assessment

Till Haselmann and Gottfried Vossen

European Research Center for Information Systems (ERCIS)
University of Münster
Leonardo-Campus 3, D-48149 Münster, Germany
{haselmann,vossen}@ercis.uni-muenster.de
<http://ercis.uni-muenster.de/>

Abstract. Cloud service providers praise **Software-as-a-Service (SaaS)** as very attractive for **small and medium enterprises (SMEs)**. However, little research has been done on the **SMEs** perspectives on this matter. This paper presents the results of a survey conducted among German enterprises that tries to clarify their views of **SaaS**. Starting with the actual IT situation, we show the prerequisites as well as the reasons in favor of and against the use of **SaaS**. We then show that the views of **SMEs** differ in some significant respects from that of **CSPs**, and we deduce hypotheses that can help towards a better design and delivery of cloud software and services.

Keywords: Software-as-a-Service, SaaS, Cloud Computing, Small and Medium Enterprises, SME.

1 Introduction

Cloud service providers claim that **cloud computing** and particularly **Software-as-a-Service** are attractive models for **small and medium enterprises (SMEs)** that try to cut costs and at the same time increase the flexibility of their IT environment [18,25]. This paper presents the results of a survey whose goal was to clarify the view of the enterprises on this matter. We present agreements and differences of their views with those of the providers and present resulting conclusions.

Cloud computing is a new computing paradigm under which a **cloud service provider (CSP)** provides IT resources flexibly and on-demand as a service over a network, usually the Internet [18,19]. IT resources can be CPU time, network bandwidth, storage space, and also application instances [28]. All aspects of deployment, operation, and maintenance lie with the **CSP**, including, e.g., software updates and backups [2]. Depending on the type of resource provided, cloud services are typically categorized into one of three deployment models [18,19]: (1) **Infrastructure-as-a-Service (IaaS)**, where the **CSP** provides virtualized hardware resources that can be used in place of physical infrastructure; (2) **Platform-as-a-Service (PaaS)**, where the **CSP** sells

access to a software platform which allows building and executing custom logic in a cloud environment; (3) *Software-as-a-Service* (SaaS), where the CSP offers a software to the end user that can be accessed using a web browser.

According to expert [5,8,21] and also public opinion, one group of enterprises that ought to profit particularly well from SaaS products is the group of *small and medium enterprises* (SMEs). However, these are mainly theoretical deliberations as there has been little research on the SMEs' view on this matter. In order to address this issue, we have conducted an exploratory online survey among 55 German enterprises asking for their views on SaaS.

In light of this survey, our contributions are the following. We investigate the actual current IT situation in SMEs in Germany with respect to SaaS. We further show the prerequisites that SMEs expect from cloud software. Analogously, we present the relevant reasons in favor of and against the use of SaaS in the SME. Finally, we discuss the results showing that the SMEs' views on cloud software differs in some significant respects from that of the CSPs and deducing hypotheses that have relevant implications for the engineering of an optimal IT architecture.

The remainder of this paper is organized as follows. Section 2 sheds light on the special situation of SMEs in the German economy and provides a brief overview of the benefits of cloud computing. We then explain our research methodology in Section 3. Consequently, in Section 4 we present selected empirical results from the survey. The results are discussed in Section 5 where we formulate our hypotheses for further research. A conclusion in Section 6 wraps this paper up.

2 Basic Concepts and Related Work

2.1 Small and Medium Enterprises in Germany

SMEs play a very important role in Germany, both socially and economically. Providing 99.7% of the about 3.2 million German companies that are subject to turnover tax (VAT), they generate about 37% of the total revenue [12]. SMEs typically have very strong regional ties, forming a more robust financial foundation than large enterprises for the local communities in question [14]. Apart from that, SMEs contribute significantly to the social security system [12]. In essence, SMEs play a crucial role for the stability of the German economy through their large number, their independence, and their heterogeneity [14].

There are several definitions of what an SME is, with only slight variations [3]. For this study, we have chosen the SME definition by the European Union [11], which should facilitate comparisons with other studies in European countries. The classification provides three size classes – micro, small and medium – based on two criteria: (1) an upper bound on the number of employees as well as (2) an upper bound on the annual turnover or the total assets.

SMEs are in a situation that is considerably different from that of large companies, due to several characteristics. Important traits in this context are the lack of an IT strategy, limited financial resources, limited information skills, and often the presence of a solitary decision maker, i. e., the owner [4,6,7].

2.2 Software-as-a-Service for SMEs

The benefits of **cloud computing** for **SMEs** are quite similar to those for any other type of company [2,29]. However, they are more pronounced for **SMEs** in some respects because of their characteristic features. The principal argument for **cloud computing** is usually that it cuts cost in the enterprise, mainly because investments into hardware and maintenance can be reduced [1,2,23,29]. This also lowers the amount of money bound in capital expenditure, which is effectively converted into smaller periodic payments to the **CSP**, a model usually referred to as pay-per-use [28] or pay-as-you-go [2]. As a side-effect, the pay-per-use notion allows **SMEs** access to software that would otherwise be too expensive to purchase. Typically, **SMEs** are expected to have only semi-professional IT operations, which is why they can benefit significantly from the high standards in professionally operated data centers [10,18,22]. In addition, providing the typical cloud characteristics, such as elasticity and short-term contracts [19], cloud software can bring more flexibility for the **SMEs** [1].

Software-as-a-Service is essentially the successor of the older **application service provider (ASP)** concept [1]. A broad investigation of this was performed in [13] for **SMEs** in the midwest region of the USA, resulting in a model for **SME ASP** adoption. The proposed model was later refined stating that flexibility was a more important aspect to the adoption of **SaaS** in **SMEs** than topics like security, cost and capability [1]. In contrast, our results showed that security was the number one concern for **SMEs** from our sample. This divergence notwithstanding, our findings agree with the evidence from [1] that **SMEs** expect their **CSPs** to offer highly customizable software.

3 Research Methodology and Sample

The primary aim of this exploratory descriptive study was to elicit possible use cases for and problems with **SaaS** in **SMEs**. For that, we conducted an online survey using SurveyGizmo¹ during a 2-months period from August to September 2010.

The survey covered a variety of topics. First, the background of both the company and the respondent was clarified and the current situation of the IT in the **SME** was established. Then, the respondent was asked about prerequisites of using **SaaS** as well as reasons in favor of and against it. For each prerequisite, the respondent had to specify the relevance of the item on a rating scale ranging from 0 (“irrelevant”) to 5 (“indispensable”). Consequently, we asked to assess the relevance of the reasons in favor of or against **SaaS** using a similar scale that ranged from 0 (“irrelevant”) to 5 (“most important”). Thus, both scales were at least interval scaled, which justified the computation of sample means for the ratings. For each category, the list of items was based on a broad literature review and later enriched by further aspects found in influential online sources

¹ <http://www.surveygizmo.com>

Table 1. Contingency table of the complete sample by origin and enterprise size

Size Class	S1	S2	S3	Frequency	
				abs.	rel.
Micro	4	1	2	7	12.7
Small	11	1	0	12	21.8
Medium	18	3	1	22	40.0
Large	8	3	3	14	25.5
Freq. abs.	41	8	6	55	100.0
Freq. rel.	74.5	14.5	10.9	100	

(expert blogs, online reports) as well as preliminary expert interviews conducted by our researches. The lists were consequently reduced by fusing similar items.

The survey was open to invited participants only. We sent out invitations to the following three groups of enterprises. **(S1)** Having had access to the mailing list of the regional chamber of industry and commerce (IHK Nord Westfalen), we randomly selected a total of 1,250 **SMEs** from all industries, 20% of which are located in the Emscher-Lippe region while the remaining 80% are located in the Münsterland region. In order to receive more meaningful results, we ensured a reasonable amount of larger **SMEs** in the pool by restricting the size of the **SME** classes to 100 micro, 400 small and 750 medium enterprises. **(S2)** The Federal Association for IT, Telecommunications and New Media (BITKOM) included the invitation to the survey in their monthly e-mail newsletter to more than 1,300 recipients. **(S3)** Additional 30 **SMEs** were contacted directly because of existing contacts, e.g., from prior research projects. The selection was independent of size, industry and regional parameters.

In each case, the recipients were informed that the questionnaire required substantial knowledge about the IT landscape and the business side of the enterprise and were asked to forward the invitation to a suitable person. In order to ensure a common understanding of the term **Software-as-a-Service**, the survey included a concise definition.

In total, 55 enterprises completed the survey, the majority of which (~75%) was from group S1. The categorization into size classes was performed exclusively along the dimensions “number of employees” and “annual turnover” using the limits set by the **European Union** because the data about the total assets was not available. Due to missing data, six enterprises were categorized based on the numbers of employees only and one enterprise was categorized based on revenue only. The filtering of the recipients and the unambiguous introduction of the survey notwithstanding, a total of 14 large enterprises answered the survey. These were analyzed separately from the **SMEs** as a comparison group. Table II shows the structure of the complete sample by origin and enterprise size. As can be seen, there is an emphasis on S1, resulting in a regional focus on the Münsterland and Emscher-Lippe regions.

The results were analyzed using methods from descriptive statistics, primarily uni- and multivariate frequency distributions. More complex analyses were not

deemed feasible due to the small size of the sample. Most of the preparatory work and some of the analyses were performed as part of a bachelor thesis [24].

4 Empirical Results

As they constitute the main part of the survey, the analysis of the prerequisites, the advantages and disadvantages of SaaS will be presented in separate subsections. In our description, we will focus on the most interesting findings.

4.1 Actual IT Situation in the Investigated SMEs

As a first step in the analysis, the general IT landscape in the SMEs was investigated. This included a look at the IT infrastructure, the professional background of the IT staff as well as more aspects related to IT operations. Consequently, we inquired about the current software usage. Turning to SaaS, we then analyzed how cloud software was already deployed in the enterprises and whether there were any general reservations against SaaS. Some notable results are highlighted in the following paragraphs.

With regard to the IT infrastructure, staff and operations, the particular situation of SMEs vs. large enterprises became directly apparent: While 92.9% of the large enterprises operated their own data center(s), only 41.5% of the SMEs did. Most of the personnel involved with the IT in SMEs was expert staff. This was especially true for the IT services industry where almost nine out of ten (88.9%) employees were experts while other industries had a lower ratio of specialists with only 65% of the staff.

Another interesting finding was that the majority (68.3%) of the SMEs mainly use standard software. Of these, 28.5% (19.5% of the sample) had not even customized the software. On the contrary, only 12.2% used custom-made software. These facts are good news for SaaS providers since their products are typically customizable standard software by their nature.

SaaS was not yet used widely among the SMEs in the sample. Only about 27% of the enterprises used SaaS, only 17% thereof on a regular basis. Almost half of the SMEs (48.8%) had no plans of introducing SaaS at all. However, when layered by different industries, it becomes apparent that SaaS was used almost exclusively in IT services enterprises, where almost one out of three (31.6%) enterprises used SaaS on a regular basis already, an additional 15.8% at least occasionally. Another group of 31.6% was investigating the deployment of SaaS within the enterprise. Considering “other” industries, it was the other way around: 72.7% had no plans of deploying SaaS and less than 10% used SaaS at all, let alone regularly. Of those SMEs that had concrete plans of introducing SaaS, more than half were targeting a deployment within the next twelve months. The remainder was planning a medium-term deployment in twelve to 24 months. However, often the opinion of a solitary decision maker is the decisive factor for/against cloud services: In many SMEs (39%) there was general reluctance against the use of SaaS. These reservations appeared to be independent of

Table 2. Prerequisites for using SaaS in the SME with mean rating \bar{x} , number of valid responses N and the standard error of the mean $SE_{\bar{x}}$, sorted by descending relevance

Rank	Prerequisites for using SaaS	\bar{x}	N	$SE_{\bar{x}}$
1	Confidentiality of data with respect to 3rd parties	4.95	40	.03
2	Permanent availability of the software	4.63	40	.11
3	Confidentiality of data with respect to CSP	4.50	40	.16
4	CSP has clearly defined backup strategy	4.40	40	.15
5	Performance of the software (no latency during use)	4.38	40	.13
6	Stable prices for service usage	4.10	40	.14
7	Usability resembles traditional desktop applications	4.03	40	.19
8	Possibility of complete data export (e. g., database dump)	3.97	39	.18
9	Guaranteed geographical limits for cloud service	3.51	37	.28
10	Possible integration into existing non-SaaS software	3.42	38	.23
11	Possibility of changing the CSP	3.37	38	.23
12	Custom SLAs	3.28	39	.18
13	CSP is located in Germany	2.97	37	.23
14	Possible integration into other SaaS software	2.88	40	.21
15	Possibility of local installation of the SaaS software	2.87	39	.23
16	CSP is located in the EU	2.62	34	.25
17	CSP has the Safe Harbor seal	2.41	27	.26
18	CSP has the PrivacyMark seal	2.36	25	.29
19	CSP is certified according to ISO/IEC 27001	2.26	31	.27
20	CSP is certified according to SAS 70	1.80	30	.26

both size and industry and mainly concerned the security risks of outsourcing enterprise data (stated by 86.7%).

4.2 Perceived Prerequisites for SaaS

The respondents were asked to assign an importance rating to a list of 20 prerequisites. The list included various items from topics such as security, software usage and compliance in random order. Table 2 shows the detailed results for the prerequisites sorted by relevance. The following paragraphs explain the most relevant results.

Given the concerns about security mentioned in Section 4.1, it is not surprising that 95 % of the respondents rated the confidentiality of enterprise data towards third parties as indispensable. Furthermore, the SMEs from the sample showed a pronounced need for confidentiality towards the CSP as well. More than three out of four (77.5 %) SMEs regarded this as an indispensable precondition. Similarly important was that the provider had a clearly defined backup strategy, playing an important role (ratings 4–5) in the decision for 85 %.

Among the aspects concerning software usage, the most important prerequisite was the permanent availability of the SaaS solution. With 72.5 % of the respondents classifying it as indispensable and a total of 92.5 % in the rating categories 4–5, this aspect even received the second highest rating of all

prerequisites. At the same time, the performance of the application was rated only slightly less important.

By choosing a **CSP** that offers flexible data export, companies can reduce the risk of a vendor lock-in. Supporting expert claims that data lock-in is a great concern among cloud users [2], 41% of the **SMEs** found flexible data export options indispensable. Interestingly, one important part of the concern seemed to be the possible loss of control over their own data. While 89.7% of the respondents found the data export an important prerequisite for an **SaaS** solution (ratings 3–5), only 70.8% thought so of the possibility to change the **CSP**. The wish for flexible data export might also be related to the **SMEs** need to integrate the **SaaS** with other (non-cloud) software already used. In comparison, the integration with other **SaaS** was not a great concern.

Concerning compliance, many **SMEs** asked to be able to determine geographic boundaries for their data. Similarly to the concept of “regions” that the Amazon Web Services offer, the enterprises wanted to constrain their data geographically to certain countries, e. g., the **EU**. On the contrary, about one out of four (27%) enterprises regarded this aspect as rather irrelevant (ratings 0–2). It was unclear which reasons had led to the diverging ratings. Interestingly, it was of much less importance whether the **CSP** was actually based in Germany or the **EU**. While German providers seemed to benefit from a slightly better image, both aspects received ratings across the whole scale with modes at 2 (**EU**) and 3 (Germany).

Tailor-made **service level agreements (SLAs)** seemed to be rather important (median $\tilde{x} = 4$), although not vital for **SMEs**. Based on the sample, it seemed that the size of the enterprise has an influence on this. Smaller enterprises tended to rate this aspect as less important than larger enterprises. Even though custom **SLAs** appeared to be an important factor in the decision, there were very few companies (~10%) that rated this aspect “indispensable.”

4.3 Perceived Reasons for Using SaaS

As with the prerequisites, the respondents were asked to rate a randomly ordered list of 15 reasons for using **SaaS** by relevance. Table 3 shows the results, the most relevant of which are discussed in this subsection.

Supporting the results from Section 4.2, the most highly rated advantage concerned the (expected) high level of security in professionally operated data centers. Quite intuitively, a **CSP** that specializes in providing secure cloud software can provide higher levels of security than a data center operated by the **SME** could [18]. Since security concerns were very important to the **SMEs**, it was not surprising that one third (33.3%) of the enterprises regarded this aspect as very important (rating 4), 48.7% even assigned the highest importance rating. Only 7.8% of the respondents considered this an aspect of low importance (ratings 0–2).

With **SaaS**, the access to the software is performed using a standard web browser running on any platform and hardware. This easy access to the software, which also facilitates easy mobile access, was the second most important reason for using **SaaS**. Almost three out of four **SMEs** (74.4%) deemed this aspect very

Table 3. Top reasons for using SaaS in the SME with mean rating \bar{x} , number of valid responses N and the standard error of the mean $SE_{\bar{x}}$, sorted by descending relevance

Rank	Reason for using SaaS	\bar{x}	N	$SE_{\bar{x}}$
1	High level of security from professional data center operation	4.15	39	.19
2	Easy access to software through web browser	3.95	39	.18
3	CSP handles all tasks related to data/storage, incl. backups	3.92	38	.18
4	No additional hardware investments for own infrastructure	3.84	38	.17
5	Better scalability and performance of the software	3.81	37	.13
6	Software safe from problems with own IT infrastructure	3.69	39	.19
7	No license or software issues when replacing hardware	3.47	38	.22
8	Zero maintenance: software updates etc. handled by CSP	3.42	38	.22
9	SaaS is ready for use more quickly than traditional software	3.03	38	.23
10	Usability of software is like that of well-known web sites	2.97	38	.24
11	Pay-per-use instead of traditional pricing models	2.92	38	.21
12	Easier possibility of changing the CSP later on	2.87	39	.21
13	Risk-free testing of new software without installation	2.87	39	.24
14	Access to software from mobile devices	2.79	38	.25
15	Short-term deployment for a specific time only (e. g., project)	2.56	36	.31

important (ratings 4–5). 70% of these SMEs rated the importance as rather high (ratings 3–5), half of which even assigned ratings 4–5.

One commonly cited advantage of SaaS is the significantly reduced overhead of deploying and operating software and hardware [27]. The responses in the sample confirmed that the SMEs saw this as an important advantage, too. Most prominently, the outsourced data management, i. e., high-performance storage, redundant copies, secure backups, etc., was seen as a strong point of SaaS. It was closely followed by the elasticity, e. g., with regard to the number of users accessing the software. The SMEs also acknowledged that a professional CSP could offer a more reliable infrastructure. These three factors were regarded as quite important (ratings 4–5) by more than two thirds of the respondents. In addition, less than 16% of the SMEs considered these aspects rather not important (ratings 0–2), in case of the data management, this portion was even below 8%. Along the same lines, the SMEs appreciated the fact that they did not have to perform any maintenance of the software or hardware and did not have to tackle licensing problems when the hardware or software environment changed. Both aspects were rated as quite important (ratings 4–5) in 57.9% and 63.2% of the responses, respectively. Surprisingly, the licensing issues were considered slightly more important than software updates and upgrades, a fact that might indicate quite serious troubles with the software licenses in the SMEs.

Among the economic aspects of SaaS, the most prominent advantage was seen in the fact that many hardware investments for the IT infrastructure were no longer required because the infrastructure was provided by the CSP. 73.6% of the SMEs found this aspect very important (ratings 4–5). The new pricing model based on the pay-per-use notion and periodic payments seemed to be not

Table 4. Top reasons for not using SaaS in the SME with mean rating \bar{x} , number of valid responses N and the standard error of the mean $SE_{\bar{x}}$, sorted by descending relevance

Rank	Reason for not using SaaS	\bar{x}	N	$SE_{\bar{x}}$
1	SaaS does not work well with other software	3.63	38	.20
2	Loss of control over access to data	3.59	39	.23
3	Data Migration from existing application too tedious	3.56	39	.23
4	High latency when using software (page loads)	3.44	39	.25
5	Less control over backups of data	3.41	39	.22
6	SaaS software lacks some desired features	2.97	35	.25
7	Permanent broadband Internet connection required	2.92	38	.29
8	Usability of web application worse than desktop application	2.63	38	.26
9	Lack of acceptance among employees	2.58	38	.25
10	Pay-per-use is not attractive	2.49	35	.25
11	Technical implementation too much effort	2.15	39	.23
12	Restrictions from existing contracts	2.14	37	.21
13	Benefits of SaaS are not seen	2.03	37	.26
14	Previous CAPEX has to be amortized first	1.97	38	.23

as attractive to the SMEs as it is often presented [17,20]. Only 28.9% deemed this an important aspect (ratings 4–5).

4.4 Perceived Reasons Against Using SaaS

As the next part, the survey contained a randomly ordered list of 14 reasons for not using SaaS. Again the respondents were asked to assign the relevance for their decisions to each item. Table 4 shows the detailed results sorted by relevance. The remainder of this subsection explains the most relevant results.

From the point of view of the respondents, the most important reason for not using SaaS was that the cloud software did not work well together with other software in the enterprise. 84.2% of the SMEs rated this aspect as important (ratings 3–5), the majority of which (60.5%) even rated this as quite relevant (ratings 4–5). This aspect is emphasized by the SMEs stating that their number three reason for not moving into the cloud was the fact that the one-time data migration from an existing application into the cloud was too tedious. While more SMEs rated this aspect as not very relevant (ratings 0–2) than the previous item (25.6% vs. 15.8%), there were also more respondents who rated this aspect as most important (38.5% vs. 26.3%). All in all, the relevance of this aspect was not significantly lower than the first, showing the problems that the SMEs still see with the data migration into the cloud.

The second most relevant reason for not using SaaS was the perceived loss of control over the access to the enterprise data. Interestingly, 61.6% of the SMEs deemed this aspect highly relevant (ratings 4–5), while 30.8% of the respondents only assigned ratings 1–2. It is likely that the respondents who did not see this as very relevant trusted in the prerequisite of data confidentiality (cf. Section 4.2)

and thus in the **CSPs** ability to protect the enterprise data from unauthorized access. The related aspect of giving up control over the backup of the enterprise data and thus possible data loss, was judged a little less relevant with a mean rating of 3.41. Still, 53.8 % found this a very relevant reason, making it the reason number five on the list.

Last, not least, the **SMEs** seemed to expect high latencies from **SaaS**. More than half of the respondents (56.4 %) found this quite an important reason (ratings 4–5) for not using cloud software, a fact that makes a strong case for **CSPs** to provide maximum performance. At the same time, the **SMEs** planning on using **SaaS** must ensure that their Internet connection has enough bandwidth to support the expected usage, which may be a problem for some enterprises based in structurally weak, e. g., rural areas.

Interestingly, the group of **SMEs** that already use **SaaS** on a regular basis regards the reasons for not using **SaaS** generally less significant than those having only tried it or not using it at all. Table 5 shows the reasons with relevant discrepancies in the average ratings of these three groups (reasons not shown had similar ratings across the groups). For example, **SaaS** users rate the latency issue only with 2.00 on average, whereas testing users or non-users rated it significantly higher (3.89/3.86). On the other hand, the technical implementation is seen as no real problem for users who use or have tried **SaaS**, whereas non-users see this as problematic.

Table 5. Mean ratings \bar{x} of reasons for not using **SaaS**, layered by the current state of **SaaS** usage in the enterprise (ordering matches that of Table 4)

Reason for not using SaaS	Usage in enterprise		
	Regularly	Testing	None
Loss of control over access to data	2.62	3.40	3.90
Data Migration from existing application too tedious	2.92	3.10	3.87
High latency when using software (page loads)	2.46	3.80	3.87
Less control over backups of data	2.31	3.10	3.73
Permanent broadband Internet connection required	2.00	3.22	3.17
Lack of acceptance among employees	1.69	2.30	3.24
Technical implementation too much effort	1.23	1.90	2.63
Restrictions from existing contracts	1.67	1.67	2.40
Benefits of SaaS are not seen	1.31	1.70	2.70

5 Discussion of Derived Results and Limitations

The empirical results show that **SaaS** is still “terra incognita” for many **SMEs**: It is currently hardly used and most of them have no plans to introduce it. In this respect, the IT services industry is at the front line, using more **SaaS** than all other industries. Most likely, this is due to the service portfolio, which requires those **SMEs** to keep up with current developments in this field. Still, even for IT services, slightly more than half of the **SMEs** are not using any kind of **SaaS** yet,

and there are general reservations against using cloud software, mainly because of security concerns.

An interesting finding in this respect is that **SMEs** do not trust **CSPs**. Even though they expect very high levels of security from **CSPs**, they are still not willing to trust them with their data – something they would usually do with traditional outsourcing providers [9,16]. This line of thought is supported by the **SMEs** fear of losing control over their data when it is sent “into the cloud”. So, even though the **CSPs** are putting much effort into providing high security standards on the technological level, **SMEs** are reluctant to trust them. In addition, current means of certifying cloud security (e.g., SAS 70) are not recognized by the **SMEs**. These are strong indications that the problem is not purely technical, but also a problem of the organizational and possibly of the legal level. We thus argue:

H1: *Building trust into a **CSP** cannot be achieved by technological means alone. Instead, additional instruments, specifically at the organizational and legal levels, have to be considered in the engineering of cloud services to convince **SMEs**.*

Almost as important as the security and trust aspect are issues concerning the availability and performance of the **SaaS**. These are perceived as insufficient by many respondents and thus considered reasons not to use **SaaS**. **SMEs** are expecting both availability and performance to match that of a locally operated application, which poses a difficult challenge to **SaaS** providers. On the contrary, these reservations are less pronounced for those enterprises that had already used **SaaS** regularly. There are many possible explanations for this, including, e.g., **SMEs** having an unjustly negative view of **SaaS** that is ameliorated when cloud software is put to the test, but also that only those cloud applications that provide good performance are actually put to use in **SMEs**. Our data, however, does not allow any specific conclusions in this respect.

The ratings of the perceived advantages of **SaaS** were not as conclusive. The main advantages seen by **SMEs** were related to improved security, better scalability, and less maintenance effort. This is rather surprising because the **CSPs** main argument is usually that cloud services cut costs [15]. Coinciding with this, **SMEs** had not generally adopted any novel pay-per-use license models; less than 10% made use of them. We suspect that **SMEs** regard **SaaS** as a means of improving the overall IT situation, but not as a fundamentally new concept. More generally, the results suggest that **SMEs**, who are mostly operating their servers on premise, can increase the quality of service in all regards, especially with regard to disaster prevention and recovery, relying on professional **CSPs** without increasing the IT spending. Hence, we argue:

H2: ***SMEs** do not see **SaaS** as a means for quick wins or cutting costs, but as an instrument for sustainable improvements in the IT landscape supporting their business. Hence, short-term commitments are less interesting for them than long-term relationships to **CSPs**.*

SaaS is standard software by nature and matches well the main type of software used in SMEs. However, each SME has individual requirements for SaaS, e. g., concerning the migration of data or integration into other software, but also individual qualifications, e. g., the knowledge of its IT staff. It is, thus, a vital aspect that the software offers customization abilities [1,26]. Especially the integration with other software seems to be a concern for SMEs. For the SMEs in our sample, the integration into non-cloud software was much more important than that into other SaaS products. This is probably so because most software currently used in the enterprises is still non-SaaS and, thus, there is not much need for integration with other SaaS products. This also indicates that the majority of the SMEs does not regard SaaS as an isolated addition to the existing application landscape, but rather as a true alternative to existing software that is often tightly integrated with other systems in the enterprise. Independently of the question of how high the individual complexity of the software integration is, the results indicate that SMEs generally require the possibility of integration from the CSP – and assess current solutions as insufficient. Hence, providers should work on better ways of easy integration, e. g., by stringent application of service orientation. It therefore also makes sense that CSPs offer additional services, such as help with data migration from legacy applications or custom integration solutions, in order to facilitate the deployment of SaaS in the SMEs. We therefore argue:

H3: *The best product for SMEs contains both the customizable SaaS and additional services that facilitate the deployment within the enterprise, both on a technical and organizational level.*

It is clear that the limited size as well as the regional focus of our sample inhibit a deduction of any results representative for all SMEs in the federal state of North-Rhine Westphalia, let alone all of Germany or beyond. Especially when trying to identify groups within the sample, the small size does not allow many meaningful results, e. g., about certain industries apart from the IT services industry. In addition, the voluntary participation in the survey makes it likely that the sample is biased in the sense that the respondents are already working the topic “cloud computing” or are at least very interested in doing so. All in all, only trend statements can be drawn from the data we have collected. Nevertheless, the statements made in this paper can be seen as good indicators for the general mindset of SMEs towards SaaS and provide some helpful insights for cloud service designers. Even though the sample is restricted to German SMEs, we believe the results will be of interest for the international community as well.

6 Conclusion and Future Work

In spite of obvious limitations w.r.t. a generalization of our results, some important insights can still be elicited. It is apparent that SaaS is not going to be a “revolution in business” for SMEs for the foreseeable future. However, it is

becoming an important concept in the IT architecture of the enterprises and offers a large market, especially when **CSPs** succeed in motivating the **SMEs** from industries other than IT services to adopt **SaaS**. Developing IS in the cloud asks for better integration with non-cloud and even legacy applications as well as for a more fine-grained security approach, possibly complemented by organizational or legal activities.

Still, there is need for future work. First, we plan on verifying our hypotheses by conducting focused interviews and follow-up studies. In addition, it will be important to apply the findings to existing **CSP** to identify specific areas for improvement, particularly at an organizational level.

References

1. Altaf, F., Schuff, D.: Taking a flexible approach to asps. *Commun. ACM* 53, 139–143 (2010), <http://doi.acm.org/10.1145/1646353.1646389>
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: A view of cloud computing. *Commun. ACM* 53, 50–58 (2010)
3. Ayyagari, M., Beck, T., Demirguc-Kunt, A.: Small and medium enterprises across the globe. *Small Business Economics* 29(4), 415–434 (2007)
4. Ballantine, J., Levy, M., Powell, P.: Evaluating information systems in small and medium-sized enterprises: issues and evidence. *European Journal of Information Systems* 7, 241–251 (1998), <http://dx.doi.org/10.1038/sj.ejis.3000307>
5. BMWi: Action program cloud computing. Federal Ministry of Economics and Technology (BMWi) (October 2010) (in german)
6. Chen, L., Haney, S., Pandzik, A., Spigarelli, J., Jesseman, C.: Small business internet commerce: A case study. *Information Resources Management Journal (IRMJ)* 16(3), 17–41 (2003)
7. Clasen, J.P.: Small and medium enterprises in a crisis: An entrepreneurial concept of turnaround management as an option of coping with the crisis. Ph.D. thesis, Hochschule St. Gallen, Bamberg (1992) (in German)
8. Currie, W.L.: A knowledge-based risk assessment framework for evaluating web-enabled application outsourcing projects. *International Journal of Project Management* 21(3), 207–217 (2003)
9. Dibbern, J., Goles, T., Hirschheim, R., Jayatilaka, B.: Information systems outsourcing: a survey and analysis of the literature. *SIGMIS Database* 35, 6–102 (2004)
10. Dimopoulos, V., Furnell, S., Jennex, M., Kritharas, I.: Approaches to it security in small and medium enterprises. In: *Australian Information Security Management Conference*, Perth (November 2004)
11. EU: Commission recommendation 2003/361/ec. *Official Journal of the European Union* 124, 36 (May 2003)
12. Geisen, B., Hebestreit, R.: Mid tier businesses: The power of diversity. In: *Federal Ministry of Economics and Technology (BMWi)* (March 2009) (in german)
13. Grandon, E.E., Pearson, J.M.: Electronic commerce adoption: an empirical study of small and medium us businesses. *Information & Management* 42(1), 197–216 (2004), <http://www.sciencedirect.com/science/article/B6VDO-4C4BNWB-2/2/e053b8d5%85e4fcd9405f1f90b35e60f5>

14. Hamer, E.: The economic role of small and medium enterprises (in German). In: Pfohl, H.C. (ed.) *Business Economics of Small and Medium Enterprises*, 4th edn., pp. 25–50. Erich Schmidt Verlag (2006) (in German)
15. Kittlaus, H.B., Schreiber, D.: SaaS – how can smes benefit? *Wirtschaftsinformatik & Management* (02), 36–42 (2010) (in German)
16. Lee, J.N., Huynh, M.Q., Hirschheim, R.: An integrative model of trust on it outsourcing: Examining a bilateral perspective. *Information Systems Frontiers* 10(2), 145–163 (2008)
17. Ma, D., Seidmann, A.: The pricing strategy analysis for the “Software-as-a-service” business model. In: Altmann, J., Neumann, D., Fahringer, T. (eds.) *GECON 2008*. LNCS, vol. 5206, pp. 103–112. Springer, Heidelberg (2008)
18. Mather, T., Kumaraswamy, S., Latif, S.: *Cloud Security and Privacy*. O’Reilly Media, Sebastopol (2009)
19. Mell, P., Grance, T.: The nist definition of cloud computing v1.5. Tech. rep., National Institute of Standards and Technology (NIST) (2009), <http://csrc.nist.gov/groups/SNS/cloud-computing/> (online)
20. Menken, I.: *SaaS – The Complete Cornerstone Guide to Software as a Service Best Practices*. Emereo Pty Ltd (November 2008)
21. Münzl, G., Przywara, B., Reti, M., Schäfer, J., Sondermann, K., Weber, M., Wilker, A.: *Cloud computing - evolution in technology, revolution in business*. BITKOM Guide (October 2009) (in German)
22. Patnayakuni, R., Seth, N.: Why license when you can rent? risks and rewards of the application service provider model. In: *Proceedings of the 2001 ACM SIGCPR Conference on Computer Personnel Research, SIGCPR 2001*, pp. 182–188. ACM, New York (2001), <http://doi.acm.org/10.1145/371209.371233>
23. Rittinghouse, J., Ransome, J.F.: *Cloud Computing: Implementation, Management, and Security*. CRC Press Inc., Boca Raton (2009)
24. Röpke, C.: *Software-as-a-Service for small and medium enterprises*. Bachelor thesis, Westfälische Wilhelms-Universität Münster (November 2010) (in German)
25. Santos, N., Gummadi, K.P., Rodrigues, R.: Towards trusted cloud computing. In: *HOTCLOUD, USENIX* (2009)
26. Schubert, P.: Personalizing e-commerce applications in smes. In: *Proceedings of the Ninth Americas Conference on Information Systems (AMCIS)*, Tampa, FL, pp. 737–750 (2003)
27. Sun, W., Zhang, K., Chen, S.K., Zhang, X., Liang, H.: Software as a service: An integration perspective. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) *ICSOC 2007*. LNCS, vol. 4749, pp. 558–569. Springer, Heidelberg (2007)
28. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A break in the clouds: Towards a cloud definition. *Computer Communication Review* 39(1), 50–55 (2009)
29. Velte, T., Velte, A., Elsenpeter, R.C.: *Cloud Computing: A Practical Approach*. McGraw-Hill Professional, New York (2009)

Trust-Based Probabilistic Query Answering

Achille Fokoue¹, Mudhakar Srivatsa¹, and Robert Young²

¹ IBM T.J. Watson Research Center, USA
{achille,msrivats}@us.ibm.com

² Defence Science and Technology Labs, UK
riyoung@mail.dstl.gov.uk

Abstract. On the semantic web, information from several sources (with disparate trust levels) may be fused with the goal of answering complex queries from end users. In such context, it is critical to aggregate information from heterogeneous sources and support trust-based query answering over the integrated knowledge base. In this paper, we describe a scalable probabilistic query answering over complex, uncertain and partially consistent knowledge bases. The key contributions in this paper are fourfold. First, our approach provides an intuitive query answering semantics over a probabilistic knowledge base. Second, our approach tolerates inconsistencies in knowledge bases. Third, we propose and evaluate a scalable error-bounded approximation query answering over a large knowledge base. Finally, we empirically show the value of taking into account trust information in the query answering process.

1 Introduction

Emerging semantic web knowledge representation standards such as OWL have been successfully used in knowledge intensive domains ranging from health care and life sciences [20] to tactical military settings [6] [5]. In such domains information from several information sources (with disparate trust levels) may be fused with the goal of answering queries (retrieving relevant information) from end users. Hence, it becomes important to aggregate information from heterogeneous information sources and support trust-based query answering over the integrated knowledge base (see Figure 1).

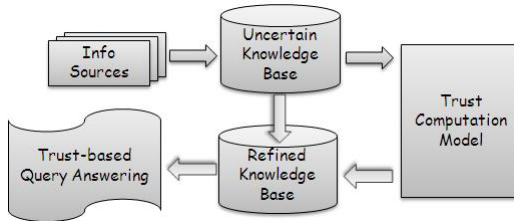


Fig. 1. Trust-based Query Answering

There are several approaches in literature to assess trust in information sources based on recommendations (eBay recommendation system [21], NetFlix movie

ratings [19], EigenTrust [13], inconsistencies in the knowledge base [7][16][23][9], etc. A vast majority of such approaches represent trust as a numeric metric over the range $[0, 1]$ – and interpret the trust value as a probabilistic measure of uncertainty in a piece of information that is reported by a data source. In this paper, our goal is to exploit such trust information to refine the knowledge and support meaningful query answering over the integrated knowledge base.

In particular, we interpret trust as a probability measure in a data item and present a solution that quantifies probabilities associated with various query answers; for instance, given a query “who does employee emp_1 work for?”, the answer to that query may be of the form $\{(org_1, p_1), \dots, (org_n, p_n)\}$ – a list of organizations and the probability that emp_1 works for that organization. Such an approach allows the querier to probabilistically interpret various possible query answers and allows the system to suitably tradeoff completeness (e.g., stop searching for query results when $\sum_{i=1}^n p_i > thr$, for some $thr < 1.0$ – assuming that an employee can work for at most one organization) and precision of query results (e.g., compute p_i within say 95% confidence) with the cost (e.g., latency) of query answering.

Various authors have proposed probabilistic extensions of less expressive knowledge representation formalisms (e.g., probabilistic RDF [11][22]). More recently, probabilistic extensions of Description Logics (DL), the theoretical foundation of OWL, have been proposed to enable reasoning under uncertainty. By and large, these approaches provide a tight integration of a particular DL (in most cases, a inexpressive one: \mathcal{ALC} in [10] and [12], $\mathcal{Classic}$ in [15], \mathcal{FL} in [24], etc.) with a given probabilistic reasoning formalism (e.g., Bayesian Network in [15], cross entropy minimization in [12]). The tightness of integration often imposes severe limitations on the underlying DL in order to assure decidability or scalability, and makes it harder to extend the approach to new DLs. Also, these approaches typically do not support uncertainty in the assertional part of the knowledge base, or impose some important restrictions on it. Probabilistic extensions of expressive DL formalisms often add an exponential overhead: in [17], for example, a probabilistic reasoning task reduces to an exponential number of corresponding classical reasoning tasks. As a result, optimized implementations such as [14] can barely scale to hundreds of probabilistic statements.

Recently, [3] has introduced a flexible, loosely coupled and orthogonal integration of Description Logics (DL) and Bayesian Network (BN) formalisms without limiting, as previous integrations [24][15] did, the expressiveness of the underlying DL. In particular, they support uncertainty in both terminological and assertional part of the knowledge base, and can be extended without any fundamental change to any decidable subset of first order logic.

Unfortunately, past work on probabilistic extensions of DL still suffer from the following shortcomings. First, they do not tolerate inconsistencies which may naturally arise when aggregating data from two unreliable sources, one asserting a fact f and the other its negation. In such a scenario, the entire knowledge base will be found unsatisfiable and no meaningful answers to any query will be derived. Second, although tractability of query answering with respect to data-complexity can be obtained on inexpressive DL such as $DL-Lite$, the combined

complexity, even for the inexpressive *DL-Lite*, is at least exponential in the number of random variables in the BN. Finally, as we will illustrate in this paper, the query answering semantics adopted in some of the past approaches (e.g., [3]) might lead to some counterintuitive results.

In this paper, we present a scalable unsatisfiability tolerant probabilistic query answering mechanism using Bayesian Description Logics that addresses the above limitations. Our key contributions are fourfold. First, our approach provides a more intuitive query answering semantics over a probabilistic knowledge base. Second, our approach tolerates inconsistencies in knowledge base by computing a measure of unsatisfiability and by performing query answering over satisfiable subspaces of the knowledge base. Third, we propose an error-bounded approximation algorithm for scalable probabilistic query answering over a large and very expressive knowledge base. This approach, which gets rid of the exponential overhead introduced by the probabilistic extension and can be made as precise as needed, is proven effective on probabilistic adaptation of the expressive UOBM benchmark ontology [18]. Finally, we empirically show the value of taking into account trust information in the query answering process.

2 Background

2.1 Bayesian Network Notation

A Bayesian network is a well-known probabilistic graphical model that encodes in a directed acyclic graph probabilistic dependencies between random variables. We briefly recall the Bayesian network notations used in the remainder of the paper. V : set of all random variables in a Bayesian network (e.g., $V = \{V_1, V_2\}$). $D(V_i)$ (for some variable $V_i \in V$): set of values that V_i can take (e.g., $D(V_1) = \{0, 1\}$ and $D(V_2) = \{0, 1\}$). v : assignment of all random variables to a possible value (e.g., $v = \{V_1 = 0, V_2 = 1\}$). $v|X$ (for some $X \subseteq V$): projection of v that only includes the random variables in X (e.g., $v|\{V_2\} = \{V_2 = 1\}$). $D(X)$ (for some $X \subseteq V$): Cartesian product of the domains $D(X_i)$ for all $X_i \in X$.

2.2 Bayesian Description Logics

Bayesian Description Logics [3] is a class of probabilistic description logic wherein each logical axiom is annotated with an event which is associated with a probability value via a Bayesian Network. In this section, we describe Bayesian DL at a syntactic level followed by a detailed example. A probabilistic axiom over a Bayesian Network BN over a set of variables V is of the form $\phi : e$, where ϕ is a classical DL axiom, and the probabilistic annotation e is an expression of one of the following forms: $X = x$ or $X \neq x$ where $X \subseteq V$ and $x \in D(X)$. Intuitively, every probabilistic annotation represents a scenario (or an event) which is associated with the set of all value assignments $V = v$ with $v \in D(V)$ that are compatible with $X = x$ (that is, $v|X = x$) and their probability value $Pr_{BN}(V = v)$ in the Bayesian network BN over V . Simply put, the semantics of a probabilistic axiom $\phi : X = x$ is as follows: when event $X = x$ occurs then ϕ holds.

A probabilistic knowledge base $K = (\mathcal{A}, \mathcal{T}, BN)$ consists of :

- A Bayesian Network BN over a set of random variables V
- A set of probabilistic Abox axioms \mathcal{A} of the form $\phi : e$, where ϕ is a classical Abox axiom. In classical DL, an Abox axiom is an axiom describing the propriety of instances in the knowledge base. In \mathcal{SHIN} DL (the core of OWL standard), an Abox axiom is expressed in one of two forms: 1) $C(a)$, stating that the individual a is an instance of the class C (e.g., $Man(Socrates)$ asserts that *Socrates* is an instance of the class *Man*); or 2) $R(a, b)$, stating that the individual a is related to b through the relationship R (e.g. $isStudentOf(Plato, Socrates)$ asserts that the individual *Plato* is related to *Socrates* by the relation $isStudentOf$).
- \mathcal{T} in the definition of $K = (\mathcal{A}, \mathcal{T}, BN)$ denotes a set of probabilistic Tbox axioms \mathcal{T} of the form $\phi : e$, where ϕ is a classical Tbox axiom. In classical DL, a Tbox axiom is an axiom describing relationship between concepts and relations in the domain of discourse. The kinds of constructors allowed in Tbox axioms determine the expressivity and the computational properties of a particular DL. For example, in the very expressive \mathcal{SHIN} DL, new concepts can be defined as the intersection, complement, or union of existing concepts, and by using universal or existential quantification on existing relations (e.g., a *HappyFamily* can be defined as $HappyFamily = Family \sqcap \forall member. Happy$ - a family whose members are all happy). The Tbox may also define constraints on relations: for example, a relation can be defined as transitive, inverse of another relation, symmetric, etc.

$\phi : p$, where $p \in [0, 1]$, is often used to directly assign a probability value to an classical axiom ϕ . This is an abbreviation for $\phi : X_0 = true$, where X_0 is a boolean random variable in the BN independent of all other variables, not mentioned in any other annotation, and such that $Pr_{BN}(X_0 = true) = p$.

The following example illustrates how this formalism can describe road conditions that are influenced by probabilistic events such as weather conditions:

$$\mathcal{T} = \{SlipperyRoad \sqcap OpenedRoad \sqsubseteq HazardousCondition, \\ Road \sqsubseteq SlipperyRoad : Rain = true\}$$

$$\mathcal{A} = \{Road(route9A), OpenedRoad(route9A) : TrustSource = true\}$$

In this example, the Bayesian network BN consists of three variables: *Rain*, a boolean variable which is true when it rains; *TrustSource*, a boolean variable which is true when the source of the axiom $OpenedRoad(route9A)$ can be trusted; and *Source*, a variable which indicates the provenance of the axiom $OpenedRoad(route9A)$. The probabilities specified by BN are as follows:

$$Pr_{BN}(TrustSource = true | Source = 'Mary') = 0.8$$

$$Pr_{BN}(TrustSource = true | Source = 'John') = 0.5$$

$$Pr_{BN}(Rain = true) = 0.7, Pr_{BN}(Source = 'John') = 1$$

The first Tbox axiom asserts that an opened road that is slippery is hazardous. The second Tbox axiom indicates that when it rains, roads are slippery. Finally,

the Abox axioms assert that *route9A* is a road and, assuming that the source of the statement *OpenedRoad(route9A)* can be trusted, *route9A* is opened.

Informally, probability values computed through the Bayesian network ‘propagate’ to the ‘DL side’ as follows. Each assignment v of all random variables in BN (e.g., $v = \{Rain = \text{true}, TrustSource = \text{false}, Source = \text{‘John’}\}$) corresponds to a primitive event ev (or a scenario). A primitive event ev is associated, through BN , to a probability value p_{ev} and a classical DL KB K_{ev} which consists of all classical axioms annotated with a compatible probabilistic annotation (e.g., $SlipperyRoad \sqcap OpenedRoad \sqsubseteq HazardousCondition$, $Road \sqsubseteq SlipperyRoad$, $Road(route9A)$). Intuitively, the probability value associated with the statement ϕ (e.g., $\phi = HazardousCondition(route9A)$) is obtained by summing p_{ev} for all ev such that the classical KB K_{ev} entails ϕ (e.g., $Pr(HazardousCondition(route9A)) = 0.35$).

3 Unsatisfiability Tolerant Bayesian DL

Inconsistencies inevitably arise when integrating information originating from disparate sources. Unfortunately, traditional Bayesian DL (as well as classical first order logic) does not tolerate inconsistencies in the knowledge base as illustrated in the following example.

A Simple Example. Let KB be the probabilistic knowledge base defined as follows: $KB = (\mathcal{T}, \mathcal{A} \cup \{\top \sqsubseteq \perp : X = \text{true}\}, BN)$ where \mathcal{T} is a classical Tbox and \mathcal{A} is a classical Abox such that the classical knowledge base $cKB = (\mathcal{T}, \mathcal{A})$ is satisfiable; BN is a Bayesian Network over a single boolean random variable X , and the probability $Pr_{BN}(X = \text{true}) = 10^{-6}$ that X is true is extremely low. Under past probabilistic extensions to DL, the KB is completely unsatisfiable, and nothing meaningful can be inferred from it. This stems from the fact that when X is true, the set of classical axioms that must hold (i.e., $\mathcal{T} \cup \mathcal{A} \cup \{\top \sqsubseteq \perp\}$) is unsatisfiable. However, the event $X = \text{true}$ is extremely unlikely, and, therefore, it is unreasonable to consider the whole probabilistic KB unsatisfiable.

Intuitively, the likelihood of events, whose set of associated classical axioms is unsatisfiable, represents the degree of unsatisfiability of a probabilistic KB, denoted $DU(K)$ ($U(K)$ denotes the set of unsatisfiable events, which is a subset of $D(V)$). In the example, $DU(K) = 10^{-6}$ and $U(K) = \{X = \text{true}\}$. Informally, the refined inconsistency tolerant semantics removes from consideration inconsistent interpretations (i.e., those associated with unsatisfiable events) and normalize with respect to consistent interpretations.

In the remaining of this section, we briefly present the main elements of the formal semantics of the unsatisfiability tolerant DL needed for the discussion of our approach to query answering. For a detailed formal definition $U(K)$, $DU(K)$, and the refined unsatisfiability tolerant semantics, we refer the interested reader to our technical report [8].

For $v \in D(V)$, we say that v is compatible with the probabilistic annotation $X = x$ (resp. $X \neq x$), denoted $v \models X = x$ (resp. $v \models X \neq x$), iff $v|X = x$ (resp. $v|X \neq x$). A probabilistic annotation e represents a scenario (or an event) which

is associated with the set of all value assignments $V = v$ with $v \in D(V)$ that are compatible with e .

Next, we define an annotated interpretation as an extension of a classical first-order logic interpretation by assigning a value $v \in D(V)$ to V . An annotated interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is defined in a similar way as a first-order interpretation except that the interpretation function $\cdot^{\mathcal{I}}$ also maps the set of variables V in the Bayesian Network to a value $v \in D(V)$. An annotated interpretation \mathcal{I} satisfies a probabilistic axiom $\phi : e$, denoted $\mathcal{I} \models \phi : e$, iff $V^{\mathcal{I}} \models e \Rightarrow \mathcal{I} \models \phi$.

Now, we define the notion of probabilistic interpretation, as a finite probabilistic distribution over annotated interpretations.

Definition 1. (From [3]) *A probabilistic interpretation Pr is a probability function over the set of all annotated interpretations that associates only a finite number of annotated interpretations with a positive probability. The probability of a probabilistic axiom $\phi : e$ in Pr , denoted $Pr(\phi : e)$, is the sum of all $Pr(\mathcal{I})$ such that \mathcal{I} is an annotated interpretation that satisfies $\phi : e$. A probabilistic interpretation Pr satisfies (or is a model of) a probabilistic axiom $\phi : e$ iff $Pr(\phi : e) = 1$. We say Pr satisfies (or is a model of) a set of probabilistic axioms F iff Pr satisfies all $f \in F$.*

Finally, partial consistency is defined as follows

Definition 2. *Let $K = (\mathcal{T}, \mathcal{A}, BN)$ be a probabilistic KB such that BN is a Bayesian Network over a set of variables V . A probabilistic interpretation Pr (as per Definition 1) satisfies (or is a model of) a probabilistic KB $K = (\mathcal{T}, \mathcal{A}, BN)$ to a degree d , $0 < d \leq 1$ iff the following holds:*

- (i) Pr is a model as $\mathcal{T} \cup \mathcal{A}$, and (ii) $d = 1 - DU(K)$
- (iii) for $v \in V$,

$$\left(\sum_{\mathcal{I} \text{ s.t. } V^{\mathcal{I}}=v} Pr(\mathcal{I}) \right) = \begin{cases} 0 & \text{if } v \in U(K) \\ \frac{Pr_{BN}(V=v)}{d} & \text{if } v \notin U(K) \end{cases}$$

A probabilistic knowledge base $K = (\mathcal{T}, \mathcal{A}, BN)$ is consistent to the degree d , with $0 < d \leq 1$, iff there is a probabilistic interpretation that satisfies K to the degree d . It is completely inconsistent (or satisfiable to the degree 0), iff $DU(K) = 1$.

4 Query Answering Semantics

In this section, we present how query answering can be carried out over partially satisfiable probabilistic knowledge bases. First, we briefly recall the query answering semantics defined in [3] and explain its shortcomings.

Definition 3. (From [3]) *An annotated interpretation \mathcal{I} satisfies (or is a model of) a ground query $\psi : e$, denoted $\mathcal{I} \models \psi : e$, iff $V^{\mathcal{I}} \models e$ and $\mathcal{I} \models \psi$. The probability of a ground query $\psi : e$ in Pr , denoted $Pr(\psi : e)$, is the sum of all $Pr(\mathcal{I})$ such that \mathcal{I} is an annotated interpretation that satisfies $\psi : e$. An answer for a probabilistic query $Q = \psi : e$ to a probabilistic knowledge base*

$K = (\mathcal{T}, \mathcal{A}, BN)$ is a pair (θ, pr) consisting of a ground substitution θ for the variables in ψ and some $pr \in [0, 1]$ such that $Pr(\psi\theta : e) = pr$ for all models Pr of K . An answer (θ, pr) for Q to K is positive iff $pr > 0$.

This definition, which requires all probabilistic models Pr of K to assign the exact same probability $Pr(\psi\theta : e) = pr$ to an answer (θ, pr) , is counter-intuitive as illustrated in the following example.

Example 1. Let $K = (\mathcal{A}, \mathcal{T}, BN)$ be the probabilistic KB defined as follows:

$$\begin{aligned}\mathcal{A} &= \{A(a) : X = \text{true}, D(a) : X = \text{false}\} \\ \mathcal{T} &= \{A \sqsubseteq C : X = \text{true}\}\end{aligned}$$

and BN is a Bayesian Network over a single boolean variable X such that $Pr_{BN}(X = \text{true}) = Pr_{BN}(X = \text{false}) = 0.5$. Consider the query $C(y)$ asking for all the instances y of C .

The classical knowledge base $cK_1 = (\mathcal{A} = \{A(a)\}, \mathcal{T} = \{A \sqsubseteq C\})$ associated with the primitive event $X = \text{true}$ entails $C(a)$, and the classical knowledge base $cK_2 = (\mathcal{A} = \{D(a)\}, \mathcal{T} = \emptyset)$ associated with the primitive event $X = \text{false}$ does not entail $C(a)$. One would expect $(y \rightarrow a, 0.5)$ to be a solution to the query. Unfortunately, according to the query answering semantics, a cannot be a solution, because there exist two probabilistic models Pr and Pr' of K such that $Pr(C(a)) \neq Pr'(C(a))$:

- Pr assigns non-zero probability to only two annotated interpretations \mathcal{I}_T and \mathcal{I}_F and $Pr(\mathcal{I}_T) = Pr(\mathcal{I}_F) = 0.5$. \mathcal{I}_T is defined as follows : its domain $\Delta^{\mathcal{I}_T} = \{\alpha\}$, $A^{\mathcal{I}_T} = C^{\mathcal{I}_T} = \{\alpha\}$, $D^{\mathcal{I}_T} = \emptyset$, $a^{\mathcal{I}_T} = \alpha$, $V^{\mathcal{I}_T} = (X = \text{true})$. \mathcal{I}_F is defined as follows : its domain $\Delta^{\mathcal{I}_F} = \{\alpha, \beta\}$, $A^{\mathcal{I}_F} = \{\beta\}$, $C^{\mathcal{I}_F} = \{\alpha\}$, $D^{\mathcal{I}_F} = \{\alpha\}$, $a^{\mathcal{I}_F} = \alpha$, $V^{\mathcal{I}_F} = (X = \text{false})$.
- Pr' assigns non-zero probability to only two annotated interpretations \mathcal{I}'_T and \mathcal{I}'_F and $Pr'(\mathcal{I}'_T) = Pr'(\mathcal{I}'_F) = 0.5$. $\mathcal{I}'_T = \mathcal{I}_T$. \mathcal{I}'_T is the same as \mathcal{I}_T previously defined. \mathcal{I}'_F differs from \mathcal{I}_F only in that it maps C to an empty set instead of $\{\alpha\}$: $\Delta^{\mathcal{I}'_F} = \{\alpha, \beta\}$, $A^{\mathcal{I}'_F} = \{\beta\}$, $C^{\mathcal{I}'_F} = \emptyset$, $D^{\mathcal{I}'_F} = \{\alpha\}$, $a^{\mathcal{I}'_F} = \alpha$, $V^{\mathcal{I}'_F} = (X = \text{false})$.

The annotated interpretation \mathcal{I}_T and \mathcal{I}'_T obviously satisfies $C(a)$ - they have to otherwise Pr and Pr' would not be models of K . One the other hand, an annotated interpretation with non-zero probability and which maps X to *false* does not have to satisfy $C(a)$. \mathcal{I}_F does satisfy $C(a)$, making $Pr(C(a)) = 1$, while \mathcal{I}'_F does not satisfy it, making $Pr'(C(a)) = 0.5$. So, we have two models which disagree on the probability value to assign to $(y \rightarrow a)$ as an answer, therefore according to the query answering semantics of Definition 3 $(y \rightarrow a)$ cannot be an answer, which is counterintuitive.

4.1 Meaningful Query Answering Semantics

Requiring all probabilistic models of K to agree on the probability value assigned to each ground query is too constraining. A more appropriate semantics, which

aligns better with our intuition, consists in defining the probability value associated with an variable assignment θ answer to a query $\psi : e$ as the infimum of $Pr(\psi\theta : e)$ for all models Pr of K . The following definition formally introduces the notion of meaningful answers.

Definition 4. *An meaningful answer for a probabilistic query $Q = \psi : e$ to a probabilistic KB $K = (\mathcal{T}, \mathcal{A}, BN)$ satisfiable to a degree d ($d \neq 0$), is a pair (θ, p) consisting of a ground substitution θ for the variables in ψ and some $p \in [0, 1]$ such that $p = \text{infimum}\{Pr(\psi\theta : e) | Pr \text{ is model of } K\}$.*

For partially unsatisfiable probabilistic knowledge bases the *meaningful* query answering semantics of Definition 4, considers only answers for satisfiable primitive events v (i.e., $v \notin U(K)$) since, by definition, for $v \in U(K)$ and an interpretation \mathcal{I} s.t. $V^{\mathcal{I}} = v$, $Pr(\mathcal{I}) = 0$. Intuitively, for a solution (θ, pr) to a query $\psi : e$, pr represents the weighted fraction (weighted by $Pr_{BN}(V = v)$) of satisfiable primitive events v s.t. $v \models e$ and θ is a solution to ψ in the classical knowledge base that must hold given v . This intuition is confirmed by Theorem 1 in the next section.

4.2 Computational Properties

In this section, we introduce key results that demonstrate how query answering tasks can be carried out on partially satisfiable knowledge bases by reducing them to query answering over classical knowledge bases.

First we introduce some important notations:

Notation 1. *Let $K = (\mathcal{T}, \mathcal{A}, BN)$ be a probabilistic knowledge base. For every $v \in D(V)$, let \mathcal{T}_v (resp., \mathcal{A}_v) be the set of all axioms ϕ for which there exists a probabilistic axiom $\phi : e$ in \mathcal{T} (resp., \mathcal{A}), such that $v \models e$. K_v denotes the classical knowledge base $(\mathcal{T}_v, \mathcal{A}_v)$. Informally, K_v represents the classical knowledge base that must hold when the primitive event v occurs.*

A key aspect of the query answering semantics presented in Definition 4 is the ability to compute the infimum of the probability of a grounded query over the set of all probabilistic models. The following critical theorem shows how such infimum can be computed precisely:

Theorem 1. *Let $K = (\mathcal{T}, \mathcal{A}, BN)$ be a probabilistic knowledge base satisfiable to the degree d ($d \neq 0$). Let $Q_g = \psi : e$ be a grounded query.*

$$\text{infimum}\{Pr(\psi : e) | Pr \text{ is a probabilistic model of } K\} = \frac{1}{d} \sum_{v \in \Omega} Pr_{BN}(v)$$

with $\Omega = \{v | K_v \text{ is satisfiable and } K_v \models \psi \text{ and } v \models e\}$

The formal proof of this theorem is established in [8] (see proof of Lemma 2 in appendix section).

Theorem 1 provides a concrete mechanism to perform query answering against a partially satisfiable probabilistic KB. However, it is highly intractable since they require an exponential number, in the number of variables in BN , of classical query answering tasks. We will address this issue in the next section.

5 Error-Bounded Approximate Query Answering

A Bayesian network based approach lends itself to fast Monte Carlo sampling algorithms for scalable query answering over a large probabilistic knowledge base. In particular, we use a *forward sampling* approach described in [2] to estimate $pr = \sum_{v \in \Omega} Pr_{BN}(V = v)$ (recall Theorem 1) that generates a set of samples v_1, \dots, v_n from BN (each sample is generated in time that is linear in the size of BN) such that the probability pr can be estimated as $\widehat{pr}_n = \frac{1}{n} * \sum_{i=1}^n I(v_i \in \Omega)$, where $I(z) = 1$ if z is true; 0 otherwise. One can show that \widehat{pr}_n is an unbiased estimator of pr such that $\lim_{n \rightarrow \infty} \sqrt{n} * (\widehat{pr}_n - pr) \rightarrow \mathcal{N}(0, \sigma_z^2)$, where $\mathcal{N}(\mu, \sigma^2)$ denotes a normal distribution with mean μ and variance σ^2 and σ_z^2 denotes the variance of $I(z)$ for a boolean variable z . We observe that the maximum variance $\sigma_{z_{max}}^2$ for a boolean random variable is 0.25, which allows one to compute, for a given acceptable error margin and confidence, the sample size. For example, to compute answers within $\pm 5\%$ error margin with a 95% confidence, the sample size $n = 396$ is necessary. The sketch of our query answering algorithm (prob-QueryAnswering) is shown below. Its soundness and completeness follow from Theorem 1.

PROBQUERYANSWERING($K, \psi : e, \epsilon, \eta$)

Input: $K = (\mathcal{A}, \mathcal{T}, BN)$ a probabilistic KB, $\psi : e$ a probabilistic query, ϵ absolute error, η confidence

Output: set of answers (θ, p) of the query $\psi : e$

- (1) $d \leftarrow 1 - DU(K)$ (see [8] for details on $DU(K)$)
- (2) **if** $d = 0$
- (3) **return** \emptyset
- (4) $size \leftarrow$ compute the size of the sample given ϵ and η
- (5) $sample \leftarrow$ generate a random sample of size $size$ given BN
- (6) $R \leftarrow \emptyset$ (R is a multiset)
- (7) **foreach** v **in** $sample$
- (8) **if** $v \models e$ **and** K_v is satisfiable
- (9) $R \leftarrow R \cup \text{CLASSICALQUERYANSWERING}(K_v, \psi)$
- (10) **return** $\{(\theta, \frac{\text{number of occurrences of } \theta \text{ in } R}{d \times size}) \mid \theta \in \text{distinct}(R)\}$

In some applications it may be sufficient to return ground substitutions θ whose probability pr exceeds a threshold thr . In this case, we argue that it is easier to decide if pr exceeds a threshold thr , rather than accurately estimate the true value of pr itself. With slight abuse of notation, we overload pr to denote a random variable that represents our belief in the true probability associated with a ground substitution θ . Hence, our goal is to select answers (θ, pr) such that $\Pr(pr > thr) > \eta$ (where η is the confidence level); symmetrically, one can also discard answers (θ, pr) such that $\Pr(pr < thr) > \eta$. We note that since $\widehat{pr}_n = pr + X$, where $X \approx \mathcal{N}(0, \frac{\sigma_{z_{max}}^2}{n})$, $\Pr(pr > thr) = \Pr(X < \widehat{pr}_n - thr) = \frac{1}{2} \left(1 + \text{erf} \left(\frac{\sqrt{n} * (\widehat{pr}_n - thr)}{\sqrt{2} * \sigma_{z_{max}}} \right) \right)$, where erf denotes the Gaussian error function. Setting $\Delta_{n_{min}} = \text{erf}^{-1}(2\eta - 1) * \frac{\sqrt{2} * \sigma_{z_{max}}}{\sqrt{n}}$, θ is a valid (resp. invalid) substitution if $\widehat{pr}_n - thr > \Delta_{n_{min}}$ (resp. $\widehat{pr}_n - thr < -\Delta_{n_{min}}$).

Using $\eta = 95\%$ and $n = 25$ (resp. $n = 100$), we require $\Delta_{n_{min}} = 0.17$ (resp. $\Delta_{n_{min}} = 0.085$). For instance, let $thr = 0.75$ and consider an answer θ whose true probability $pr > 0.92$; while it requires $n = 396$ samples to estimate the true probability pr with 95% confidence and under 5% error, one can conclude that the true probability exceeds $thr = 0.75$ with 95% confidence using only $n = 25$ samples. Similarly, one can eliminate answers whose true probability is below 0.57 using only $n = 25$ samples. Hence, one can initially accept (resp. reject) answers whose probability significantly exceeds (resp. falls below) the threshold thr ; only answers whose probability is in the neighborhood of the threshold thr requires more samples to validate them. One can also show that the expected value of n in order to attain the desired confidence level η is given by $\hat{n} = \frac{2 * \sigma_{z_{max}}^2 * (erf^{-1}(2\eta - 1))^2}{(pr - thr)^2}$. For example, setting $\eta = 95\%$, $thr = 0.75$, a substitution θ with $pr = 0.85$ may be adjudged as a valid answer using sample size $n = 60$.

6 Experimental Evaluation

To evaluate our approach, we performed experiments assessing, on the one hand, the performance and scalability of our probabilistic query answering algorithm and, on the other hand, the value (in terms of precision/recall) of taking into account, during query answering, uncertainty originating from difference in the trustworthiness of data sources.

6.1 Performance and Scalability

We have developed a prototype implementation, PSHER, that extends our SHER reasoner [4] to support Bayesian *SHIN* query answering. We evaluated it on the UOBM benchmark [18] which was modified to *SHIN* expressivity (the core of OWL 1.0 DL), and its Abox was modified by randomly annotating half of the axioms with probability values. We issued instance retrieval queries for a subset of concepts [5] in the ontology. The results are reported for 1, 10, and 30 universities, which are referred to as UOBM-1, UOBM-10 and UOBM-30. The runs were made on a 64 bit 2.4 GHz AMD 4-core processor 16G RAM Linux machine (only 2 cores were used: one for the SHER process, and the other for the DB2 process). The Abox was stored in a DB2 database.

As expected, Table II shows that PSHER preserves SHER scalability characteristics: it still scales sublinearly. For the experiments whose results are reported in Table II, we computed the probability values for all answers, without any threshold, with an absolute error of 0.1 and a confidence of 0.9. The resulting large number of classical KBs (72) to consider for each query explains the relatively high absolute runtime in Table II. Note that, Pronto [14], the only other implementation of a probabilistic extension of a very expressive DL, can barely scale to hundreds of probabilistic axioms.

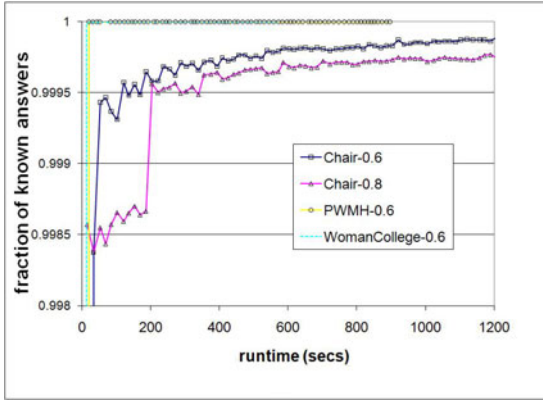
¹ SHER is the only OWL reasoner able to scale to UOBM-30. It is available for download on IBM Alphaworks: <http://www.alphaworks.ibm.com/tech/sher>

² Non-atomic complex concepts.

Table 1. PSHER on probabilistic UOBM KBs

Dataset	type axioms	role axioms	avg time (mins)	stdev (mins)	range (mins)
UOBM-1	25,453	214,177	4	1	3-5
UOBM-10	224,879	1,816,153	22	10	14-34
UOBM-30	709,159	6,494,950	61	26	41-94

As described in the previous section, one can enhance the performance of PSHER using an application specified threshold (thr) for selecting answers. In that scenario, for an instance retrieval query and a given confidence level η , at every stage of the evaluation, each individual in the Abox is in one of three states: rejected when its probability is already known to be below thr ; accepted when its probability is already known to be above thr ; or unknown when not enough samples have been processed to conclude with confidence η . We conducted instance retrieval experiments with thresholds set at 0.6 and 0.8. As shown in Figure 2, the overwhelming majority of individuals are quickly in a known state (rejected or accepted). For example, for the concept ‘Chair’ with threshold 0.8, 99.95% of the individuals are in a known state in less than 3.3 mins. Most of the rest of the time is spent computing the status of a small number of individuals whose probability is close to thr .

**Fig. 2.** Query Answering with threshold on UOBM10

Finally, parallelization is another way to significantly reduce the absolute numbers presented in Table 1, and our approximate reasoning approach naturally lends itself to efficient parallelization.

6.2 Trust-Based Query Answering

To illustrate the benefits of a trust-based query answering approach, we evaluate how query answering completeness, as measured by recall, and soundness, as

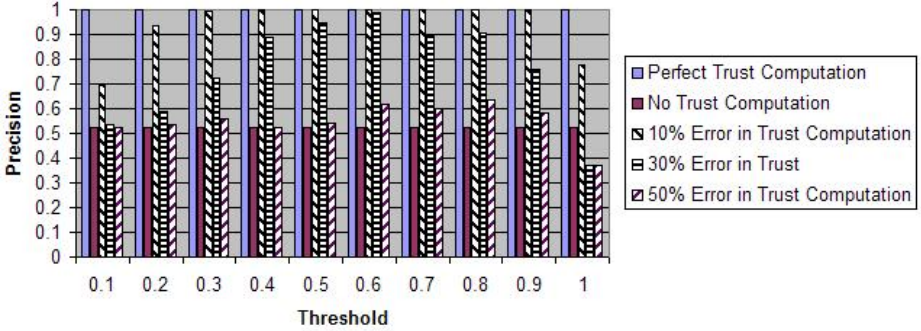


Fig. 3. Precision

measured by precision, are affected by either the lack of trust inference or imprecision in the trust inference computation. For that purpose, we assume that only half of the axioms in UOBM-1 can be trusted, while the others are assumed to be incorrect. A classical query answering against the classical knowledge base \mathcal{K}_g consisting of the randomly selected trustworthy axioms of UOBM-1 provides the ground truth for our evaluation. Next, we define the following refinement of UOBM-1 to reflect the absence of trust computation and varying levels of precision in the trust computation module:

- $(\mathcal{K}_{ptc}, BN_{ptc})$ is a refinement of UOBM-1 where trusted axioms (i.e., axioms in \mathcal{K}_g) are assigned a trust value of 1 and all other axioms have a trust value of 0. $(\mathcal{K}_{ptc}, BN_{ptc})$ corresponds to the refinement performed based on the results of a perfect trust computation. Formally, $\mathcal{K}_{ptc} = \{\phi : T_c = \mathbf{true} | \phi \in \mathcal{K}_g\} \cup \{\phi : T_f = \mathbf{true} | \phi \in (UOBM_1 - \mathcal{K}_g)\}$ where the Bayesian Network BN_{ptc} consists of two independent boolean variables T_c and T_f such that $Pr_{BN}(T_c = \mathbf{true}) = 1$ and $Pr_{BN}(T_f = \mathbf{true}) = 0$
- $(\mathcal{K}_{ntc}, BN_{ntc})$ represents the knowledge base obtained when no trust computation is performed. Basically all axioms are taken at face value. Formally, \mathcal{K}_{ntc} is exactly the same as \mathcal{K}_{ptc} defined previously, and BN_{ntc} consists of the two independent boolean variables T_c and T_f such that $Pr_{BN}(T_c = \mathbf{true}) = 1$ and $Pr_{BN}(T_f = \mathbf{true}) = 1$.
- $(\mathcal{K}_{tc}^e, BN_{tc}^e)$ is a refinement of UOBM-1 performed based on the results of a trust inference computation with a uniform error rate of $e\%$, where $e \in \{10, 30, 50\}$. Formally, \mathcal{K}_{tc}^e is exactly the same as \mathcal{K}_{ptc} defined previously, and BN_{tc}^e consists of the two independent boolean variables T_c and T_f such that $Pr_{BN}(T_c = \mathbf{true}) = 1 - e/100$ and $Pr_{BN}(T_f = \mathbf{true}) = e/100$.

Figures 3 and 4 show the weighted average of precision and recall, respectively, for membership query answering performed on 15 concepts with an absolute error $\pm 10\%$ and a confidence of 90%. The results are grouped by the threshold thr used in selecting answers, and all the previously defined trust-based refinements of UOBM-1 are considered.

In Figure 3, as expected, query answering precision decreases with the error rate of the trust inference computation. However, except for the very high

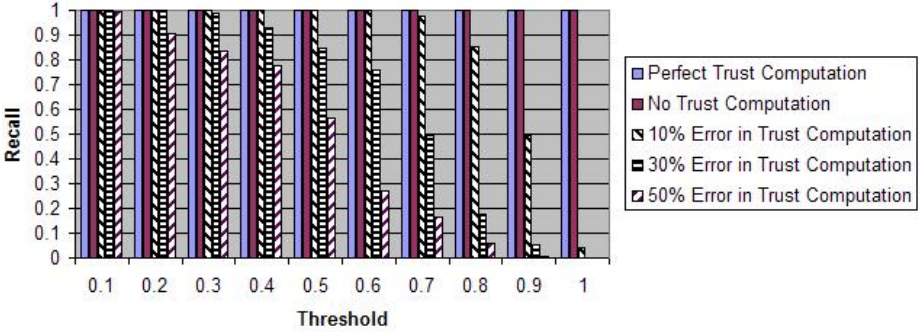


Fig. 4. Recall

threshold of 1, it is always better than query answering without trust inference, which has a constant precision of about 50%, even in case of a high error rate of 50%. For recall, taking every axiom at face value (i.e., in absence of trust inference computation), already guarantees a recall of 100% at all thresholds. Such recall can only be matched at all thresholds in the case of perfect trust inference computation as illustrated in Figure 4. In practice, recall is more sensitive (especially at high thresholds) to the error rate of the trust inference computation (e.g. in Figure 4, at threshold 0.9, recall is almost equal to zero for refinements based on trust inference computation with error rate greater than 30%).

In terms of combined F-Score, for a trust inference computation with an error rate of less than 10%, which is the typical range for our trust computation approach presented in [8], trust-based query answering significantly outperforms query without trust for all thresholds that are less than or equal to 0.8. At thresholds greater than 0.9, trust-based query answering with trust inference error rate of 10% becomes inferior to query answering without trust because the threshold clearly falls within the margin of error of both the trust inference module and, in our experiment, the accuracy of our error-bounded probabilistic query answering module.

The main lesson from the results presented in this section is that, provided that the query answering threshold is properly set with respect to the error rate of the trust inference computation module and the query answering module, trust-based query answering significantly outperforms in terms of F-Score query answering without taking into account axiom trustworthiness.

7 Conclusion

In this paper, we have described a loosely coupled integration of Description Logics and Bayesian Network formalisms to support scalable probabilistic reasoning over a large, uncertain knowledge base. This work has highlighted the need for a sound query answering semantics over an incomplete and inconsistent knowledge base. We have described an error-bounded approximation algorithm for scalable query answering and measurement of inconsistency over an

uncertain knowledge base. Our initial experiments have shown the expressiveness and scalability of our approach using the UOBM benchmark.

Acknowledgements. Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.: The Description Logic Handbook. Cambridge University Press, Cambridge (2003)
2. Cheng, J., Druzdzel, M.J.: AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks. *Journal of AI Research* (2000)
3. D’Amato, C., Fanizzi, N., Lukasiewicz, T.: Tractable reasoning with bayesian description logics. In: Greco, S., Lukasiewicz, T. (eds.) SUM 2008. LNCS (LNAI), vol. 5291, pp. 146–159. Springer, Heidelberg (2008)
4. Dolby, J., Fokoue, A., Kalyanpur, A., Kershenbaum, A., Schonberg, E., Srinivas, K., Ma, L.: Scalable semantic retrieval through summarization and refinement. In: AAAI, pp. 299–304 (2007)
5. Fikes, R., Ferrucci, D., Thurman, D.: Knowledge associates for novel intelligence (kani) (2005), https://analysis.mitre.org/proceedings/Final_Papers_Files/174_Camera_Ready_Paper.pdf
6. Fokoue, A., Srivatsa, M., Rohatgi, P., Wrobel, P., Yesberg, J.: A decision support system for secure information sharing. In: ACM SACMAT, pp. 105–114 (2009)
7. Fokoue, A., Srivatsa, M., Young, R.: Assessing trust in uncertain information. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010, Part I. LNCS, vol. 6496, pp. 209–224. Springer, Heidelberg (2010)
8. Fokoue, A., Srivatsa, M., Young, R.: Trust Inference and Query Answering over Uncertain Information. Technical Report RC25157, IBM Research (2011)
9. Guo, Y., Heflin, J.: An Initial Investigation into Querying an Untrustworthy and Inconsistent Web. In: Workshop on Trust, Security and Reputation on the Semantic Web (2004)
10. Heinsohn, J.: Probabilistic description logics. In: UAI 1994 (1994)
11. Huang, H., Liu, C.: Query evaluation on probabilistic RDF databases. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) WISE 2009. LNCS, vol. 5802, pp. 307–320. Springer, Heidelberg (2009)
12. Jaeger, M.: Probabilistic reasoning in terminological logics. In: KR 1994 (1994)
13. Kamvar, S., Schlosser, M., Garcia-Molina, H.: EigenTrust: Reputation management in P2P networks. In: WWW Conference (2003)

14. Klinov, P., Parsia, B.: On improving the scalability of checking satisfiability in probabilistic description logics. In: Godo, L., Pugliese, A. (eds.) SUM 2009. LNCS, vol. 5785, pp. 138–149. Springer, Heidelberg (2009)
15. Koller, D., Levy, A., Pfeffer, A.: P-classic: A tractable probabilistic description logic. In: AAAI 1997, pp. 390–397 (1997)
16. Kuter, U., Golbeck, J.: SUNNY: A New Algorithm for Trust Inference in Social Networks, using Probabilistic Confidence Models. In: AAAI 2007 (2007)
17. Lukasiewicz, T.: Expressive probabilistic description logics. *Artif. Intell.* 172(6-7), 852–883 (2008)
18. Ma, L., Yang, Y., Qiu, Z., Xie, G., Pan, Y.: Towards a complete OWL ontology benchmark. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 124–139. Springer, Heidelberg (2006)
19. Netflix. Netflix Prize, <http://www.netflixprize.com/>
20. Patel, C., Cimino, J.J., Dolby, J., Fokoue, A., Kalyanpur, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: Matching patient records to clinical trials using ontologies. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 816–829. Springer, Heidelberg (2007)
21. Schafer, J.B., Konstan, J., Riedl, J.: Recommender Systems in E-Commerce. In: ACM Conference on Electronic Commerce (1999)
22. Udea, O., Subrahmanian, V.S., Majkic, Z.: Probabilistic rdf. In: IRI (2006)
23. Xiong, L., Liu, L.: Supporting reputation based trust in peer-to-peer communities. *IEEE TKDE* 16(7) (July 2004)
24. Yelland, P.M.: An alternative combination of bayesian networks and description logics. In: KR, pp. 225–234 (2000)

Top-K Possible Shortest Path Query over a Large Uncertain Graph^{*}

Lei Zou, Peng Peng, and Dongyan Zhao

Peking University, Beijing, China

{zoulei, pengpeng, zdy}@icst.pku.edu.cn

Abstract. This paper studies the top-k possible shortest path (kSP) queries in a large uncertain graph, specifically, given two vertices S and T , a kSP query reports the top-k possible shortest paths from S to T . Different from existing solutions for uncertain graph problems, we adopt the *possible worlds* semantics. Although *possible worlds* semantics have been studied widely in relational databases, the graph structure leads to more complexity in computational model. The main contribution of this paper lies in developing lower and upper bounds for the probability that one path is the shortest path in an uncertain graph, based on which, a filter-and-refine framework is proposed to answer kSP queries. We demonstrate the superiority of our method by extensive experiments.

1 Introduction

As a fundamental problem, shortest path query has many significant applications, such as web-based mapping services [1] and network routing. Therefore, an extensive study has been made over shortest path queries [2,3,4], especially in deterministic graphs, in which, vertices, edges and edge weights are known with certainty. However, in some applications, the graph structure may have some uncertainty. For example, a road network can be modeled as a graph, in which, each edge denotes a highway. In practice, some highways are likely to be blocked. In other words, the probability that these edges do not exist is high. Therefore, route planning should consider both edge weights and uncertainties to provide more reasonable paths for users. However, recent GPS-enabled path services ignore the uncertainty inherent in traffic.

In order to model the uncertainty, for each highway e , we introduce $\overline{Pr}(e)$ to denote the probability that e is blocked, i.e., the probability that edge e does not exist. Thanks to roadside sensors and other monitoring techniques, $\overline{Pr}(e)$ can be obtained from historical traffic data. Figure 1(a) shows a running example of a road network. Edge weights, probabilities $Pr(e)$ and $\overline{Pr}(e)$ are given in Figure 1(b). Given two vertices S and T , although path $P_1(e_1, e_2, e_4)$ is shorter than path $P_2(e_1, e_3)$, the probability that P_1 is blocked is much larger than that of P_2 . In this case, P_2 should be ranked better than P_1 .

We can find some other applications of the above uncertain graph model. In Yago projects [5], authors build a large RDF graph by extracting information from Wikipedia. Each edge (fact) in the RDF graph is assigned a *confidence value* that depends on the

^{*} This work was supported by NSFC under Grant No.61003009 and RFDP under Grant No. 20100001120029.

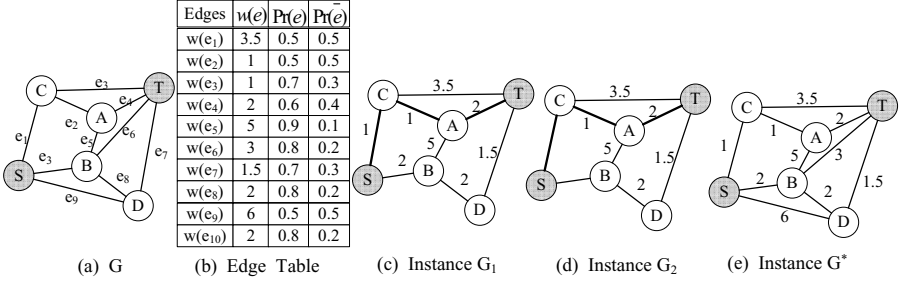


Fig. 1. Running Example

estimated accuracy with which the fact was extracted from Wikipedia [6]. Actually, the confidence value is the same as the probability $Pr(e)$ in our uncertain graph model.

Uncertain graphs have been considered from different applications for several decades [7,8]. However, few works adopt the *possible worlds* semantics. For example, they use some simple statistic aggregates to model the uncertainty, such as [9] studied the expected shortest paths in dynamic stochastic networks. However, the simple aggregation may miss uncertain inherent. Hua and Pei studied probabilistic path queries in road networks in [10], but, their method does not consider the interplay among all uncertain paths.

In this paper, we adopt the *possible worlds* semantics, the classical model in uncertain database community, to model an uncertain graph. Specifically, an uncertain graph is viewed as a set of instances (possible worlds). Each possible world is a combination of edges. Given an uncertain graph G in Figure 1(a), Figures 1(c-d) show two different instances G_1 and G_2 . Note that there are $O(2^{|G|})$ instances for an uncertain graph G . Obviously, each instance G_i also has a member probability $Pr(G_i)$, which is computed by assuming the existence of edges in this instance, and the absence of all other edges. As we know, given two vertices S and T in an uncertain graph G , for different instances, there are different shortest paths between S and T . For example, the shortest path in instance G_1 is $P_1(e_1, e_2, e_4)$, but the shortest path in instance G_2 is $P_2(e_1, e_3)$, as shown in 1(c-d), respectively. Given two vertices S and T , a path P is called the shortest path in G if and only if P is the shortest path in at least one instance of G . Thus, given two vertices S and T in an uncertain graph G , there are many different shortest paths between them. The problem to be addressed in this paper is called *top-k possible shortest path queries* (kSP for short), which is to find k paths with the k largest probabilities to be the shortest path in G .

Given an uncertain graph G , let G^* be the instance that all edges in G exist. We also call G^* as the *certain version* of G . Figure 1(e) shows an example of G^* . Given two vertices S and T , let P_n be the n -th shortest path from S to T in G^* . A key issue in kSP is how to compute the probability that path P_n is the shortest path in G , i.e., $Pr(P_n = SP(G))$. Obviously, the event that path P_n is the shortest path in G equals to the event that path P_n exists in G but none of paths P_i exists, $i = 1, \dots, n-1$, where P_i is the i -th shortest path in G^* . Formally, we have the following equation:

$$\Pr(P = SP(G)) = \Pr\left(\bigcap_{i=1}^{i=n-1} \overline{E(P_i)} \cap E(P_n)\right) \quad (1)$$

where $\overline{E(P_i)}$ denotes the event path P_i does not exist and $E(P_n)$ denotes the event path P_n exists.

Note that, in an uncertain graph G , $\overline{E(P_i)}$ and $E(P_n)$ are not atomic events, since paths P_i (P_n) are composed by several edges. Thus, paths P_{i_1} and P_{i_2} ($1 \leq i_1, i_2 \leq n-1$) may have common edges. In this case, we have to use inclusion-exclusion principle to evaluate Equation 1. This is the key difference between kSP query and top-k queries over uncertain relational tables [8]. As we know, inclusion-exclusion principle has the exponential time complexity, thus, it is quite expensive to answer kSP queries by the inclusion-exclusion principle. More details are discussed in Section 2.

In order to improve the query performance, we adopt the filter-and-refine framework. The main contribution in this paper is to define proper bounds, which have the linear computation complexity. Then, bases on the bounds, we can find candidate paths for kSP queries without any dismissal. In the refining process, for each candidate path P , we utilize Monte Carlo (MC) simulation to estimate $Pr(P = SP(G))$. Finally, the top-k possible shortest paths are reported to users.

To summarize, in this work, we make the following contributions:

1. Based on the classical top-k shortest path algorithms over deterministic graphs, we propose a filter-and-refine framework to answer the top-k possible shortest path queries in a large uncertain graph.
2. We design two different lower bounds to estimate the probability that one path P is the shortest path over an uncertain graph G , in which, the first one is based on the path independence assumption and the second one is based on the approximate “set-cover” based method (it has the linear time complexity).
3. An upper bound that is decreasing with regard to path lengths is proposed. The stop strategy of our kSP query algorithm depends on the monotone decreasing property of the upper bound.
4. Extensive experiments over both real and synthetic datasets confirm the superiority of our method.

2 Problem Definition

Definition 1. An uncertain graph G is denoted as $G = \{V, E, W, F_W, P, F_P\}$, where (1) V is a set of vertices, and (2) $E \subseteq V \times V$ is a set of edges, and (3) W is a set of edge weights, and (4) the weight function F_W defines the mapping $F_W : E \rightarrow W$, and (5) P is a set of edge probability, and (6) the probability function F_P defines the mapping $F_P : E \rightarrow P$, which denotes the existence probability of an edge in E .

Let us recall the uncertain graph G in Figure 1(a). There are 2^{10} instances of G . Due to space limit, we only show two instances of G , as shown in Figure 1(c) and (d), respectively. The shortest path between S and T is (S, C, A, T) in instance G_1 , as shown in Figure 1(c), but the shortest path is (S, B, T) in instance G_2 , as shown in Figure 1(d).

Given an uncertain graph G , we use G^* to denote the certain version of G , i.e., the instance that all edges exist. For example, given an uncertain graph G in Figure 1, its corresponding certain version G^* is given in Figure 1(e).

Definition 2. Given an uncertain graph G and one instance G_i of G , the probability of instance G_i is defined as:

$$\begin{aligned} \Pr(G_i) &= \left(\prod_{e \in G_i} \Pr(e)\right) \times \left(\prod_{(e \notin G_i \wedge e \in G)} \overline{\Pr(e)}\right) \\ &= \left(\prod_{e \in G_i} \Pr(e)\right) \times \left(\prod_{(e \notin G_i \wedge e \in G)} (1 - \Pr(e))\right) \end{aligned}$$

where $\Pr(e)$ denotes the existence probability of edge e .

Definition 3. Given a path P from vertex S to T in an uncertain graph G , P is called the shortest path over G if and only if P is the shortest path in at least one instance of G . The probability that P is the shortest path from S to T in G is denoted as $\Pr(P = SP(G, S, T))$.

$$\Pr(P = SP(G, S, T)) = \sum_{G_i \in PW(G) \wedge P = SP(G_i, S, T)} \Pr(G_i) \quad (2)$$

where $P = SP(G_i, S, T)$ denotes the event that P is the shortest path from S to T in an instance G_i , and $PW(G)$ denotes all possible instances of G .

Definition 4. (Problem Definition) Given two vertices S and T in an uncertain graph G and a parameter k , a top-k possible shortest path query (kSP) reports the k shortest paths P_j ($j = 1, \dots, k$) over G , where $\Pr(P_j = SP(G, S, T))$ are the k largest among all possible shortest paths from S to T in G .

Note that, when vertices S and T are clear from the context, we use $\Pr(P = SP(G))$ and $P = SP(G_i)$ to simplify the notations. Given an uncertain graph G in Figure 1, path (S, C, A, T) is the shortest path from S to T in 2^7 instances of G . Obviously, it is quite expensive to compute $\Pr(P = SP(G))$ according to Equation 2 since its complexity is $O(2^{(|E(G)| - |P|)})$. Thus, it is impossible to enumerate all possible instances of G to answer a top-k possible shortest path query.

3 Computational Model

Given a path P , P also refers to all edges in the path. We use $E(P)$ to denote the event that path P exists, i.e., the event that all edges in P exist. We use $\Pr(E(P))$ to denote the probability that the event $E(P)$ happens. Thus, we have the following equation about $\Pr(E(P))$:

$$\Pr(E(P)) = \sum_{G_i \in PW(G) \wedge P \subseteq G_i} \Pr(G_i) = \prod_{e \in P} \Pr(e)$$

Furthermore, we also use $\overline{E(P)}$ to denote the event that P does not exist, i.e., the event that *not all edges* in P exist. Formally, we have the following equation:

$$\Pr(\overline{E(P)}) = 1 - \Pr(E(P)) = 1 - \prod_{e \in P} \Pr(e)$$

To address the top-k possible shortest path query, we have the following computation model:

1) Given two vertices S and T in an uncertain graph G , let P_1 denote the shortest path from S to T in G^* , i.e., the certain version of G . The event that P_1 is the shortest path in G is equivalent to the event that P_1 exists in G . Therefore, we have the following equation:

$$\Pr(P_1 = SP(G)) = \Pr(E(P_1)) = \prod_{e \in P_1} \Pr(e) \quad (3)$$

2) Let P_n denote the n -th shortest path from S to T in G^* . The event that P_n is the shortest path in G is equivalent to the event that P_n exists but none of paths $P_{n'}$ ($n' = 1, \dots, n-1$) exists in G . Therefore, we have the following equations.

$$\Pr(P_n = SP(G)) = \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i)} \cap E(P_n)) \quad (4)$$

In order to compute Equation 4, we have the following process:

$$\begin{aligned} \Pr(P_n = SP(G)) &= \Pr((\bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \cap E(P_n)) \\ &= \Pr((\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}) \cap E(P_n)) \end{aligned} \quad (5)$$

where $(P_i - P_n)$ denotes all edges that are in P_i but not in P_n , and $\overline{E(P_i - P_n)}$ denotes the event that not all edges in $(P_i - P_n)$ exist. Since the events $\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}$ and $E(P_n)$ are independent from each other. Thus, we have the following process:

$$\begin{aligned} \Pr(P_n = SP(G)) &= \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)} \cap E(P_n)) \\ &= \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}) \times \Pr(\bigcap E(P_n)) \\ &= (1 - \Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))) \times \prod_{e \in P_n} \Pr(e) \end{aligned} \quad (6)$$

In order to compute the above equation, we need to use inclusion-exclusion principle to compute $\Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))$, as shown in Equation 7

$$\Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n)) = \sum_{i=1}^{i=n-1} (-1)^{k-1} \sum_{I \subset \{1, \dots, n-1\}; |I|=k} \Pr(A_I) \quad (7)$$

where $A_I = \bigcap_{i \in I} E(P_i - P_n)$.

4 Framework

In order to answer a kSP query efficiently, we adopt the filter-and-refine framework. Specifically, given two vertices S and T in an uncertain graph G and a parameter k , we first find all candidate paths for the kSP query. Then, for each candidate path P_n , we compute the probability $\Pr(P_n = SP(G))$ by Equations 6 and 7 to find the answers. According to the framework, there are two important issues to be addressed: First, we need to find all candidates efficiently without enumerating all possible paths from S to T ; Second, it is very expensive to evaluate Equation 7 directly due to its exponential complexity. We need a cheap method to evaluate $\Pr(P_n = SP(G))$ instead of the inclusion-exclusion principle.

We first illustrate the intuition about our method. Given a path P from S to T , $w(P)$ denotes the length of P . P can be the shortest path in G if and only if path P exists and

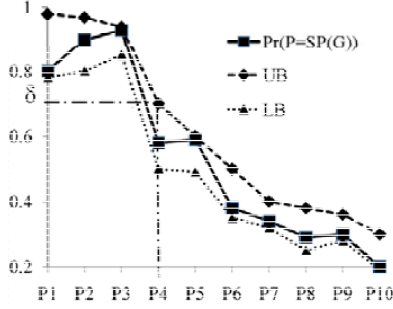


Fig. 2. Bounds

none of paths P' , where $w(P') < w(P)$, exists. Intuitively, given a path P , if $w(P)$ is large, the probability that P is the shortest path from S to T (i.e., $\Pr(P = SP(G))$) is small. Although the general trend of $\Pr(P = SP(G))$ is decreasing with regard to $w(P)$, $\Pr(P = SP(G))$ is not monotonically decreasing with regard to $w(P)$, as shown in Figure 2. In order to answer a kSP query efficiently, we have the following steps:

1) Given a path P , find a lower bound and an upper bound for $\Pr(P = SP(G))$, denoted as $LB(\Pr(P = SP(G)))$ and $UB(\Pr(P = SP(G)))$, respectively, where $UB(\Pr(P = SP(G)))$ is monotonically decreasing with regard to $w(P)$. Figure 2 demonstrates the bounds.

2) We use Yen's algorithm [11] to compute the top- k' shortest paths in G^* progressively. Note that, k' is not an input parameter, which depends on the following stop condition. In each step of Yen's algorithm, we can find a path P_n ($n = 1, \dots$) that is the n -th shortest path over G^* . We evaluate the lower bound and the upper bound for $\Pr(P_n = SP(G))$, i.e., $LB(\Pr(P_n = SP(G)))$ and $UB(\Pr(P_n = SP(G)))$, respectively. During the process, we always keep a threshold δ to be the k -th largest lower bound so far. If $UB(\Pr(P_n = SP(G))) < \delta$, it means that we have found all candidate paths without any dismissal. In this case, we can stop the algorithm.

Take Figure 2 for example. In order to find the top-3 possible shortest paths, we can find $\delta = UB(\Pr(P_4 = SP(G)))$, since there are three paths whose lower bounds are larger than δ . In this case, for any path P_i ($i > 4$), $UB(\Pr(P_i = SP(G))) < \delta$. Therefore, all paths P_i ($i > 4$) can be pruned safely.

3) In order to improve query performance, for each candidate, we compute $\Pr(P_i = SP(G))$ by the Monte Carlo (MC) simulation.

Following the above framework, we discuss the three steps in Sections 5 to 7, respectively.

5 Probability Bounds

Given a path P_n that is the n -th shortest path from S to T , we discuss how to define the proper bounds for the probability $\Pr(P_n = SP(G))$.

5.1 The Lower Bound

In this subsection, we discuss how to find the lower bounds for $\Pr(P_n = SP(G))$. The first lower bound is derived when we assume that $(P_1 - P_n), \dots, (P_{n-1} - P_n)$ have no common edges, where $(P_i - P_n)$ ($i = 1, \dots, n-1$) denotes all edges in P_i but not in P_n . For purposes of presentation, we also call $(P_i - P_n)$ as a path in the following presentation, which refers to all edges in $(P_i - P_n)$. If these paths $(P_i - P_n)$ have many common edges, the first lower bound is not tight. In order to address this issue, we propose an approximate “set-cover” based method (it has the linear time complexity) to evaluate the lower bound. This lower bound favors the situation that there are many common edges between $(P_i - P_n)$ ($i = 1, \dots, n-1$).

The First Lower Bound. According to Equation 5 we can get that $\Pr(P_n = SP(G)) = \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}) \cdot \Pr(E(P_n))$. It is easy to know that events $\overline{E(P_{i_1} - P_n)}$ and $\overline{E(P_{i_2} - P_n)}$ ($i_1 \neq i_2$) are positive correlated, since the occurrence of the event $\overline{E(P_{i_1} - P_n)}$ may lead to the occurrence of the event $\overline{E(P_{i_2} - P_n)}$, if there are some common edges between $(P_{i_1} - P_n)$ and $(P_{i_2} - P_n)$. Since events $\overline{E(P_i - P_n)}$ ($1 \leq i \leq n$) are pairwise positive correlated, thus, we know that $\Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}) \geq \prod_{i=1}^{i=n-1} \Pr(\overline{E(P_i - P_n)})$.

Theorem 1. (Lower Bound 1) Given a path P_n that is the n -th shortest path from S to T in G^* , the lower bound for $\Pr(P_n = SP(G))$ is denoted as $LB1(\Pr(P_n = SP(G)))$, which is defined as follows:

$$\begin{aligned} \Pr(P_n = SP(G)) &\geq LB1(\Pr(P_n = SP(G))) \\ &= \left(\prod_{i=1}^{i=n-1} (1 - \prod_{e \in (P_i - P_n)} \Pr(e)) \right) \cdot \prod_{e \in P_n} \Pr(e) \end{aligned} \quad (8)$$

Proof.

$$\begin{aligned} \Pr(P_n = SP(G)) &= \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}) \cdot \Pr(E(P_n)) \\ &\geq \prod_{i=1}^{i=n-1} \Pr(\overline{E(P_i - P_n)}) \cdot \Pr(E(P_n)) \\ &= \prod_{i=1}^{i=n-1} (1 - \Pr(E(P_i - P_n))) \cdot \Pr(E(P_n)) \\ &= \left(\prod_{i=1}^{i=n-1} (1 - \prod_{e \in (P_i - P_n)} \Pr(e)) \right) \cdot \prod_{e \in P_n} \Pr(e) \\ &= LB1(\Pr(P_n = SP(G))) \end{aligned} \quad (9)$$

□

Obviously, the smaller of $|(P_{i_1} - P_n) \cap (P_{i_2} - P_n)|$ ($1 \leq i_1 \neq i_2 \leq n$), the more tight of the lower bound. On the extreme, $|(P_{i_1} - P_n) \cap (P_{i_2} - P_n)| = 0$, $\Pr(P_n = SP(G))$ equals to the lower bound.

The Second Lower Bound. Although $LB1(\Pr(P_n = SP(G)))$ provides a lower bound for $\Pr(P_n = SP(G))$, it is not tight when some paths have many common edges. In order to address this issue, we introduce another lower bound that is based on the set cover problem [12].

Let us consider the top-3 shortest paths P_1, \dots, P_3 from S to T in G^* , as shown in Figure 3(b). According to Equation 5, we know $\Pr(P_3 = SP(G)) = \Pr(\bigcap_{i=1}^{i=2} \overline{E(P_i - P_3)} \cap E(P_3)) \cdot \bigcap_{i=1}^{i=2} \overline{E(P_i - P_3)}$ denotes the event that not all edges

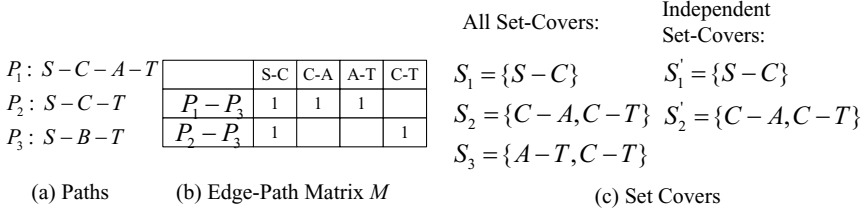


Fig. 3. Set-Cover-Based Lower Bound

in $P_i - P_3$ ($i = 1, 2$) exist. We build an edge-path matrix M , as shown in Figure 3(b), where each row corresponds to one path $P_i - P_3$ and each column corresponds to one edge. $M[i, j] = 1$ means that the j -th edge exists in the i -th path $P_i - P_3$ in M .

Definition 5. (Set Cover) Given an edge-path matrix M and a set of edges $S = \{e\}$, a path P is covered by S if and only if $P \cap S \neq \phi$, i.e., there is at least one common edge between P and S .

S is a set cover over M if and only if S satisfies the following statements:

- 1) For each path P in M , P is covered by S ; and
- 2) S cannot cover all paths if any edge $e \in S$ is removed from S .

Given an edge-path matrix M in Figure 3(b), there are three different set covers, in total, over M , as shown in Figure 3(c). Given a set S of edges, we use $E(\overline{S})$ to denote the event that *none* of edges in S occurs. Note that, event $E(\overline{S})$ is different from event $\overline{E(S)}$, since the later refers to the event that *not all* edges in S exist.

Lemma 1. Given an edge-path matrix M over paths $(P_i - P_n)$, $i = 1, \dots, n-1$, all set covers over M are denoted as S_1, \dots, S_m . The event that $\bigcap_{i=1}^{i=n} \overline{E(P_i - P_n)}$ equals to the event that $\bigcup_{i=1}^{i=m} E(\overline{S_i})$.

According to Lemma 1, we can know that $Pr(\bigcap_{i=1}^{i=n} \overline{E(P_i - P_n)}) = Pr(\bigcup_{i=1}^{i=m} E(\overline{S_i}))$. Obviously, it is impossible to enumerate all set covers over M , due to its exponential time complexity. We propose to find some pairwise independent set covers. Two set covers are *pairwise independent* if and only if they have no common edges. For example, we can find two independent set covers S'_1 and S'_2 in Figure 3(d). Given an edge-path matrix M over paths $(P_i - P_n)$, $i = 1, \dots, n-1$, in order to find all pairwise independent set covers $S'_1, \dots, S'_{m'}$, we propose the following steps: First, we find a set cover S over M by the greedy minimal set cover algorithm [12]. Then, we remove all edges in S from M , and iterate the above step over M until that $M = \phi$ or no set cover can be found. In this way, we can find pairwise independent set covers $S'_1, \dots, S'_{m'}$.

Lemma 2. Given an edge-path matrix M over paths $(P_i - P_n)$, $i = 1, \dots, n-1$, all set covers over M are denoted as $\{S_1, \dots, S_m\}$. $\{S'_1, \dots, S'_{m'}\}$ are pairwise independent set covers over M . The following equations holds:

$$1) \{S'_1, \dots, S'_{m'}\} \subseteq \{S_1, \dots, S_m\}; \text{ and } 2) Pr(\bigcup_{i=1}^{i=m'} E(\overline{S'_i})) \geq Pr(\bigcup_{i=1}^{i=m} E(\overline{S_i}))$$

According to Lemmas 1 and 2, we have the following lower bound.

Theorem 2. (Lower Bound 2) Given an edge-path matrix M over paths $(P_i - P_n)$, $i = 1, \dots, n-1$, $S'_1, \dots, S'_{m'}$ are pairwise independent set covers over M . The lower bound for $\Pr(P_n = SP(G))$ is denoted as $LB2(\Pr(P_n = SP(G)))$, which is defined as follows:

$$\begin{aligned} \Pr(P_n = SP(G)) &= \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)}) \cdot \Pr(E(P_n)) \\ &\geq LB2(\Pr(P_n = SP(G))) = LB2(\Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)})) \cdot \Pr(E(P_n)) \quad (10) \\ &= \Pr(\bigcup_{i=1}^{i=m'} E(\overline{S'_i})) \cdot \Pr(E(P_n)) \end{aligned}$$

where $LB2(\Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i - P_n)})) = \Pr(\bigcup_{i=1}^{i=m'} E(\overline{S'_i}))$

Although we can use the inclusion exclusion principle to compute $\Pr(\bigcup_{i=1}^{i=m'} E(\overline{S'_i}))$ in Equation 10, the computation has the exponential time complexity $O(2^{m'})$. Note that, the events $E(\overline{S'_i})$ are pairwise independent. We propose an iterative algorithm to compute $\Pr(\bigcup_{i=1}^{i=m'} E(\overline{S'_i}))$ with the linear time complexity. Specifically, we have the following process:

- 1) $\Pr(\bigcup_{i=1}^{i=1} E(\overline{S'_i})) = \Pr(E(\overline{S'_1})) = \prod_{e \in S'_1} (1 - \Pr(e))$;
- 2) $j = 2, \dots, m'$

$$\begin{aligned} &\Pr(\bigcup_{i=1}^{i=j} E(\overline{S'_i})) \\ &= \Pr(\bigcup_{i=1}^{i=j-1} E(\overline{S'_i})) + (1 - \Pr(\bigcup_{i=1}^{i=j-1} E(\overline{S'_i}))) \times \Pr(\overline{E(S'_j)}) \\ &= \Pr(\bigcup_{i=1}^{i=j-1} E(\overline{S'_i})) + (1 - \Pr(\bigcup_{i=1}^{i=j-1} E(\overline{S'_i}))) \times \prod_{e \in S'_j} (1 - \Pr(e)) \end{aligned}$$

where $\Pr(\bigcup_{i=1}^{i=j-1} E(\overline{S'_i}))$ has been evaluated in the previous step.

Assume that all paths $(P_i - P_n)$, $i = 1, \dots, n-1$, have many common edges. The set $\{S'_1, \dots, S'_{m'}\}$ is closer to the set $\{S_1, \dots, S_m\}$. In this case, Equation 10 provides a more tight lower bound than the first lower bound. Let us consider one example in Figure 3. The exact value of $\Pr(P_3 = SP(G))$ is 0.3872. According to Theorem 2, we know that $LB2(\Pr(P_3 = SP(G))) = \Pr(E(\overline{S'_1}) \cup E(\overline{S'_2})) \times \Pr(E(P_3)) = 0.575 \times 0.64 = 0.368$. However, according to Theorem 1, $LB1(\Pr(P_3 = SP(G))) = \Pr(\overline{E(P_1 - P_3)}) \times \Pr(\overline{E(P_2 - P_3)}) \times \Pr(E(P_3)) = 0.5525 \times 0.64 = 0.3536$. Obviously, the second lower bound is more tight.

5.2 The Upper Bound

In this subsection, we discuss how to find an upper bound for $\Pr(P = SP(G))$. According to the framework discussed in Section 4, the upper bound must be monotonically decreasing with $w(P)$.

Theorem 3. Upper Bound. Given a path P_n that is the n -th shortest path from S to T , the upper bound for $\Pr(P_n = SP(G))$, is defined as follows:

$$\begin{aligned} \Pr(P_n = SP(G)) &\leq UB(\Pr(P_n = SP(G))) \\ &= (1 - \sum_{i=1}^{i=n-1} LB(\Pr(P_i = SP(G)))) \quad (11) \\ &= (1 - \sum_{i=1}^{i=n-1} \text{Max}(LB1(\Pr(P_i = SP(G))), LB2(\Pr(P_i = SP(G)))) \end{aligned}$$

where $LB(\Pr(P_i = SP(G)))$ denotes a lower bound for $\Pr(P_i = SP(G))$.

Proof. According to Equation 5, we have the following equation:

$$\begin{aligned} \Pr(P_n = SP(G)) &= \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i)} \cap E(P_n)) \\ &= \Pr(E(P_n) | \bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \times \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \end{aligned} \quad (12)$$

Furthermore, the following two events are equal to each other.

$$\begin{aligned} &\overline{\bigcap_{i=1}^{i=n-1} \overline{E(P_i)}} \\ &= E(P_1) \cup (\overline{E(P_1)} \cap E(P_2)) \cup \dots \cup (\bigcap_{i=1}^{i=n-2} \overline{E(P_i)} \cap E(P_{n-1})) \end{aligned}$$

For any two elements $\bigcap_{i=1}^{i=k} \overline{E(P_i)} \cap E(P_k)$ and $\bigcap_{i=1}^{i=k'} \overline{E(P_i)} \cap E(P_k)$ in the right part of the above equation, we know that $(\bigcap_{i=1}^{i=k} \overline{E(P_i)} \cap E(P_k)) \cap (\bigcap_{i=1}^{i=k'} \overline{E(P_i)} \cap E(P_k)) = \emptyset$. According to the principle of additivity, we have the following equation:

$$\begin{aligned} \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) &= 1 - \sum_{i=1}^{i=n-2} \Pr(\bigcap_{j=1}^{j=i-1} \overline{E(P_j)} \cap E(P_i)) \\ &= 1 - \sum_{i=1}^{i=n-1} \Pr(P_i = SP(G)) \end{aligned} \quad (13)$$

According to Equations 12 and 13, we have the following equation:

$$\begin{aligned} \Pr(P_n = SP(G)) &= \Pr(E(P_n) | \bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \times \Pr(\bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \\ &= \Pr(E(P_n) | \bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \times (1 - \sum_{i=1}^{i=n-1} \Pr(P_i = SP(G))) \\ &\leq \Pr(E(P_n) | \bigcap_{i=1}^{i=n-1} \overline{E(P_i)}) \times (1 - \sum_{i=1}^{i=n-1} LB(\Pr(P_i = SP(G)))) \\ &\leq 1.0 \times (1 - \sum_{i=1}^{i=n-1} LB(\Pr(P_i = SP(G)))) \end{aligned}$$

Theorem 4. *The upper bound in Equation 14 is monotonically decreasing with $w(P)$.*

Proof. It is straightforward to be proven according to Equation 11.

6 Finding Candidate Paths

In this section, we discuss how to find candidate paths for a kSP query. We use Yen's algorithm to find top- k' shortest paths over graph G^* (the certain version of G) progressively. Note that, k' is not an input parameter. It depends on a stop condition, as discussed shortly. In order to better understand our method, we briefly review Yen's algorithm. It first builds a shortest path tree rooted at T by Dijkstra's algorithm. For each neighbor u of S , the algorithm inserts $[p = (S, u), w(p)]$ into a set U , where p is a path formed by vertices S and u . Then, a path $p \in U$ is found to minimize $w(p) + D(u_m, T)$, where u_m is an end point of p except for S and $D(u_m, T)$ is the shortest path distance between u_m and T . It can be proved that path P_1 composed by path p and the shortest path between u_m and T must be the shortest path from S to T .

In order to find the top-2 shortest path, we remove path $p = (S, u_m)$ from U and insert $[p = (S, u_m, u), w(p)]$ into U , where u is a neighbor of u_m and $w(p)$ is the distance of path p . If neighbor u is on the shortest path between S and T , we ignore the insertion. Furthermore, if p is not a simple path, we ignore p . If p is not a simple path, we ignore p . Iteratively, a path $p \in U$ is found to minimize $w(p) + D(u_m, T)$,

where u_m is an end point of p except for S and $D(u_m, T)$ is the shortest path distance between u_m and T . It can be proved that path P_2 composed by path p and the shortest path between u_m and T must be the top-2 shortest path from S to T . We iterate the above process to find the top- k' shortest paths progressively.

In each step of the above algorithm, we can find a path P_n that is the n -th shortest path over G^* . We evaluate the lower bound and the upper bound for $Pr(P_n = SP(G))$. We always keep a threshold δ to be the k -largest lower bound so far. When $UB(Pr(P_n = SP(G))) < \delta$, the algorithm can stop. It means that all paths P' , where $w(P') > w(P_n)$, can be pruned safely. Otherwise, we insert P_n into candidate paths CP and update the threshold δ .

7 Computing the Probability

According to the framework in Section 4, we need to evaluate $Pr(P_n = SP(G))$ for each candidate path P_n . According to Equation 5, the key problem is how to compute $Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))$. Obviously, it is quite inefficient to evaluate it by Equation 7, since it has the exponential time complexity. In this section, we use Luby-Karp's algorithm [13] (Algorithm 1), a classical Monte Carlo simulation, to evaluate $Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))$. Note that, this verification step is not our contribution in this paper. For each candidate path P_n , we just simply use the existing Monte Carlo simulation to estimate $Pr(P_n = SP(G))$. The main contribution of our work lies finding candidate paths efficiently. Theorem 5 means that more iteration steps lead to more accurate estimation. According to Equation 14, we can control iteration number N to meet the accuracy requirement.

Algorithm 1. Compute the probability by Luby-Karp Algorithm

Require: **Input:** paths $P_i, i = 1, \dots, n$.

Output: $Pr(P_n = SP(G))$.

1: $Cnt = 0$; and $S = Pr(E(P_1 - P_n)) + \dots + Pr(E(P_{n-1} - P_n))$;

2: **repeat**

3: randomly choose $i \in \{1, \dots, n-1\}$, with probability $Pr(E(P_i - P_n))/S$

4: For all edges e , according to $Pr(e)$, choose a random truth assignment s.t. $E(P_i - P_n)$ is true.

5: **if** for all $j < i-1$ $E(P_j - P_n)$ is false **then**

6: $Cnt = Cnt + 1$

7: **until** N times

8: $\tilde{p} = \frac{Cnt}{N} \times S$

9: $\tilde{Pr}(P_n = SP(G)) = (1 - \tilde{p}) \times \prod_{e \in P_n} Pr(e)$

10: Report $\tilde{Pr}(P_n = SP(G))$

Theorem 5. [13] N is the number of steps excuted by the Luby and Karp algorithm. $\tilde{Pr}(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))$ is the estimation value computed by the Luby and Karp algorithm. Let $\delta > 0$.

$$\varepsilon = \sqrt{4(n-1) \log(2/\delta)/N}; p = Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n));$$

$$\tilde{p} = Pr(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))$$

Then, we have the following equation about the estimation accuracy:

$$\Pr(p \in [\tilde{p} - \varepsilon, \tilde{p} + \varepsilon]) > 1 - \delta \quad (14)$$

According to the estimated probability $\widetilde{\Pr}(\bigcup_{i=1}^{i=n-1} E(P_i - P_n))$, for each candidate path P_n , it is easy to compute $\Pr(P_n = SP(G))$ by Equation 5. Then, the top-k possible shortest paths are reported to users.

8 Experiments

In this section, we evaluate our methods in both real and synthetic datasets. As far as we know, there is no existing work on top-k possible shortest path queries. In [14], Yuan et al. proposed threshold-based shortest path queries over uncertain graphs. In order to enable comparison, we transfer top-k possible shortest path queries into threshold-based one with decreasing threshold. In the following section, we refer the method as “Decreasing Threshold-Based Method” (DT for short). Specifically, we first set threshold θ to be a large value. According to the algorithm in [14], if we can find the top-k possible shortest paths, then we report them and terminate the algorithm. Otherwise, we decrease θ and repeat the above process until we obtain the top-k answers. All experiments are implemented by standard C++ and conducted on a P4 1.7G machine of 1G RAM running Windows XP.

8.1 Datasets

We use two large real datasets in our experiments: 1) Yago (<http://www.mpi-inf.mpg.de/yago-naga/yago/>) extracts facts from Wikipedia and integrates them with the WordNet thesaurus. According to the Yago dataset, we build a large uncertain graph, in which, each vertex denotes an entity and each edge denotes the property between two entities. Edge weight is assigned as ‘1’. Edge probability is assigned according to the method in [6]. There are 365142 vertices and 512071 edges in Yago graph. 2) OMIM (<http://www.ncbi.nlm.nih.gov/omim>) is a protein-protein interaction network. There are 4163 vertices and 11504 edges. Each edge is associated with one probability to model the confidence of one protein-protein interaction.

Furthermore, we also use two classical random network models Erdos Renyi Model (ER) and Scale-Free Model (SF). ER is a classical random graph model. It defines a random graph as $|V|$ vertices connected by $|E|$ edges, chosen randomly from the $|V|(|V| - 1)$ possible edges. SF defines a random network with $|V|$ vertices satisfying power-law distribution in vertex degrees. In this work, we set power-law distribution parameter $\gamma = 2.5$ to simulate real complex networks [15].

8.2 Evaluating Lower and Upper Bounds

In this section, we evaluate the lower and upper bounds on two real uncertain graphs. We show two lower bounds (LB1 and LB2) in Figure 4(a) and 4(b), respectively. Furthermore, we also show the exact probability value $\Pr(P = SP(G))$. From Figure 4(a), we know that LB1 is much better than LB2 in OMIM graph, but LB2 is better

than LB1 in Yago graph. As we know, the average degree in OMIM (2.76) is much larger than Yago (1.4). Therefore, top-k paths in OMIM have fewer common edges than that in Yago. In this case, as mentioned in Section 5 LB1 is better than LB2. On the other hand, if there are many common edges between top-k paths, LB2 is better than LB1, like Yago graph.

Furthermore, from Figures 4(a) and 4(b), we know that although the exact probability value is not monotonically decreasing with path length $w(P)$, the upper bound is a monotonic function.

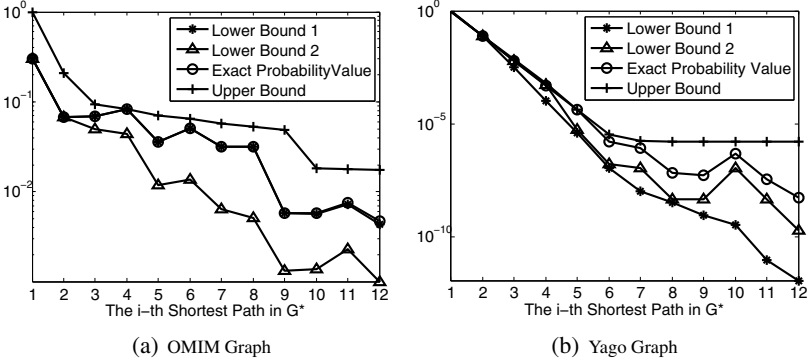


Fig. 4. Evaluating Bounds

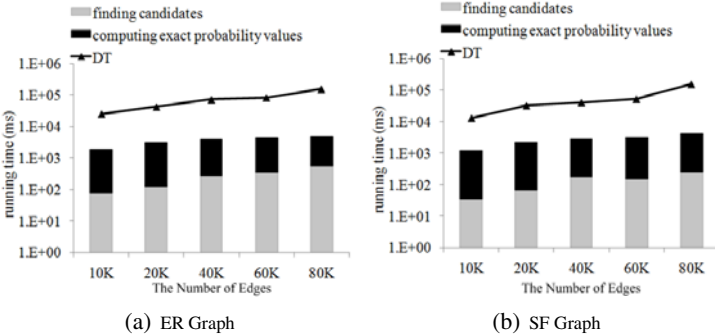


Fig. 5. Running Times VS. Graph Size

8.3 Performance vs. Graph Size

In this subsection, we evaluate the performance with graph size, i.e., $|E|$. In ER graphs, we fix the $\frac{|E|}{|V|}$ as 2.0 and vary $|E|$ from 10K to 80K vertices. Figure 5(a) shows the running time for top-10 possible shortest path queries. As discussed in Section 4, there are two steps: finding candidates and evaluating the probability values. We show the

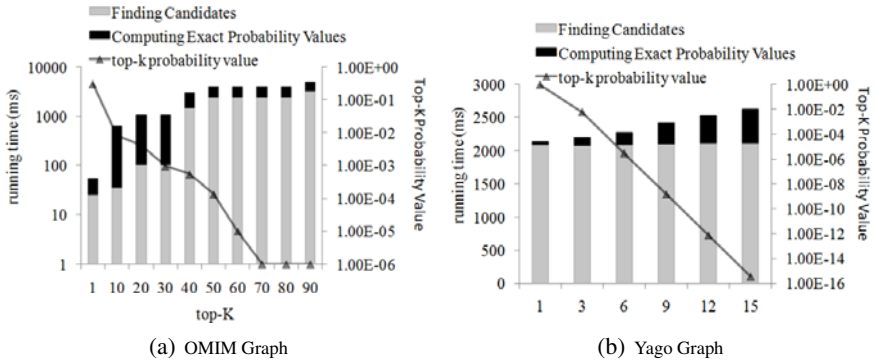


Fig. 6. Performance VS. K

running time for each step in Figure 5. Note that, we also show the running time of the decreasing threshold-based method (DT for short) in Figure 5(a), from which, we know that our method is faster than DT by orders of magnitude. We also report the performance in SF graphs in Figure 5(b).

8.4 Performance vs. K

In this subsection, we evaluate the performance with regard to K in OMIM. Figure 6(a) shows that the probability value is very small ($< 10^{-3}$) when $K > 30$. We can also find the similar problem in Yago graph. This observation means that K should be small in practice. Otherwise, the result paths have very small probabilities. Figure 6 also shows the running time with regard to K .

9 Conclusions

In this paper, we propose a top-k possible shortest path queries over a large uncertain graph. The main contribution of this work lies in developing lower and upper bounds for the probability computation. Based on the bounds, we propose a filter-and-refine framework to find answers. Extensive experiments confirm the superiority of our method.

References

1. Samet, H., Sankaranarayanan, J., Alborzi, H.: Scalable network distance browsing in spatial databases. In: SIGMOD Conference (2008)
2. Cohen, E., Halperin, E., Kaplan, H., Zwick, U.: Reachability and distance queries via 2-hop labels. *SIAM J. Comput.* 32(5) (2003)
3. Chan, E.P.F., Lim, H.: Optimization and evaluation of shortest path queries. *VLDB J* 16(3), 343–369 (2007)
4. Jing, N., Huang, Y.-W., Rundensteiner, E.A.: Hierarchical encoded path views for path query processing: An optimal model and its performance evaluation. *IEEE Trans. Knowl. Data Eng.* 10(3) (1998)

5. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: WWW (2007)
6. Kasneci, G., Suchanek, F.M., Ifrim, G., Ramanath, M., Weikum, G.: Naga: Searching and ranking knowledge. In: ICDE (2008)
7. Frank, H.: Shortest paths in probabilistic graphs. *Operations Research* 17 (1969)
8. Soliman, M.A., Ilyas, I.F., Chang, K.C.-C.: Top-k query processing in uncertain databases. In: ICDE (2007)
9. Sigal, C.E., Pritsker, A.A.B., Solberg, J.J.
10. Hua, M., Pei, J.: Probabilistic path queries in road networks: traffic uncertainty aware path selection. In: EDBT (2010)
11. Yen, J.Y.
12. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: *Introduction to algorithms*. The MIT Press, Cambridge (2001)
13. Karp, R.M., Luby, M.: Monte-carlo algorithms for enumeration and reliability problems. In: FOCS (1983)
14. Yuan, Y., Chen, L., Wang, G.: Efficiently answering probability threshold-based shortest path queries over uncertain graphs. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010. LNCS, vol. 5981, pp. 155–170. Springer, Heidelberg (2010)
15. Albert, R., Barabási, A.-L.: Statistical mechanics of complex networks. *Reviews of Modern Physics* 74, 47–97 (2002)

Training a Named Entity Recognizer on the Web

David Urbansky, James A. Thom, Daniel Schuster, and Alexander Schill

Dresden University of Technology,

RMIT University

{david.urbansky,daniel.schuster,alexander.schill}@tu-dresden.de,

james.thom@rmit.edu.au

Abstract. In this paper, we introduce an approach for training a Named Entity Recognizer (NER) from a set of seed entities on the web. Creating training data for NERs is tedious, time consuming, and becomes more difficult with a growing set of entity types that should be learned and recognized. Named Entity Recognition is a building block in natural language processing and is widely used in fields such as question answering, tagging, and information retrieval. Our NER can be trained on a set of entity names of different types and can be extended whenever a new entity type should be recognized. This feature increases the practical applications of the NER.

1 Introduction

Named Entity Recognition (NER) is an information extraction task that the Message Understanding Conference 6 (MUC-6) originally defined in 1996. The goal of the task was to identify six types of entities. These were names of people, organizations, geographic locations, times, currencies, and percentage expression in texts [12]. NER is an important task in the chain of natural language processing, and other techniques such as co-reference resolution, machine translation, language understanding, and question-answer finding build upon it. With the trend toward a more entity-centric search paradigm, NER will become even more important in the future. Essentially, there are three types of NER systems: Those that use hand-crafted rules, those that use supervised machine learning algorithms, and those that use unsupervised machine learning [23]. Hand-crafted rules are often more precise, but yield lower recall. Moreover, they require a lot of manual work. When there are annotated corpora available, the preferred technique is to use supervised machine learning. Often we do not have corpora to train a named entity recognizer, and compiling one is a very time consuming task. Consider an application that wants to detect mentions of mobile phone names in news feeds. The usual tagged corpora contain only four entity types – person, organization, location, and miscellaneous. Training the NER on such a corpus would not be beneficial since mobile phone names fall under the category miscellaneous, as would digital cameras, cars, movie names, etc. We would therefore need to label the mobile phone names in a corpus of news feed texts. If our application advances and we also want to recognize mentions of digital

cameras, we need to update that corpus and label all mentions of digital cameras as well. This scenario parallels the ambitions of the WebKnox research project [28] in which the goal is to extract unknown entities from the web.

In this paper we try to answer the following research questions:

- How much labeled data is necessary to train a well-performing NER?
- How well do state-of-the-art NERs perform on web texts recognizing many different entity types?
- How well does a NER perform when it is automatically trained on the web?

Our contributions are answers to these questions using two datasets, one with 4 entity types (CoNLL) and one with 22 entity types (TUDCS4). It is important to note that almost all related work limits its findings to a very small number of entity types as found in the MUC or CoNLL datasets. First, we analyze the performance of three state-of-the-art NERs based on the size of the training data and the number of entity types that need to be recognized. Second, we propose a technique to train a NER on a set of seeds automatically, and show how well this approach performs compared to supervised learning. Another contribution is the release of our dataset on the research platform Areca¹ to allow other researchers use the corpus. Furthermore, our developed Named Entity Recognizer and all related tool are available free for research purposes².

2 Related Work

There are three major approaches to named entity recognition. First, there are lexicon-based recognizers that can only detect entities that they have stored in their lexicon or gazetteer [21, 14]. Often there are ambiguous entities such as “Paris”, which is both a city (and there are many cities called Paris) and a forename. Using lexicons is therefore not simple and matches need to be disambiguated. This approach is not of interest to us, however, since our goal is to recognize unknown entities. Second, there are hand-crafted rules for NERs [18, 6]. One such rule could be “the uppercase sequence of words after the token ‘Mr.’ is a name”. Ripper [7] is a system used to create such if-then rules. Rule-based systems were mainly popular in the early days of NER research and resulted in high precision. This approach does not scale well, however, since rules need to be created by experts and must be maintained for every new entity type that should be recognized [19]. For instance, the example rule for detecting person names using the prefix “Mr.” has to be refined when the NER is supposed to differentiate between various kinds of people, such as politicians, actors, or professors. Third, there are statistical machine learning approaches that have become increasingly popular. These approaches can broadly be divided into unsupervised and supervised machine learning. In unsupervised machine learning, the learning algorithm does not know the entity types and clusters the detected mentions.

¹ <http://areca.co>

² <http://palladian.ws>

This approach often relies on patterns and lexical resources such as WordNet [22]. [10] researched the task of detecting any type of entity in web texts instead of learning particular predefined classes. However, they do not propose a solution for entity classification, but rather only provide a means to delimit boundaries. The most common approach is supervised machine learning in which the NER is trained on a labeled corpus of text, e.g. “<PER>John Hiatt</PER> is a musician from <LOC>Indiana</LOC>”. The NER builds a model from the training data and uses it to recognize the learned entity types for untagged texts later. Many machine learning techniques have been applied including hidden markov models [3], maximum entropy models [4], Support Vector Machines [2], decision trees [26], and conditional random fields [17]. Also, various researchers [30, 15, 16] have shown that combining several weak classifiers yields a stronger single classifier. Apart from these three main approaches, a number of hybrid methods have been developed which combine rules, lexicons, and machine learning, for example [19].

The problem of creating labeled training was quickly recognized and researchers have since tried to ease the compilation of training data. One method to train a NER is to use a list of seed entities per entity type. The contexts around the seeds can be gathered and used for learning rules. There were several experiments with the size of the seed lists varying from only 7 per entity type [9] up to 100 [5]. [8] show that only as few as 7 seeds are needed to create well-performing rules. [9] have tested the performance using between 40 and 100 seeds and found that the larger the number of seeds, the higher the F-Measure due to an increased recall. On the other hand, [5] showed that precision increased when using smaller lists with only a slight drop in recall. Seed lists are therefore an easy and fast way to train a NER, but as [19] pointed out, the recall of those systems is usually quite low. The recall problem can be countered using the so-called “bootstrapping” technique, which means that the rules learned from the contexts of the seed entities are again used to find other entities which are then again seeds for the next learning phase [9, 24]. [27] use the web to overcome the sparseness problem with labeled data. In particular, they improve the best CoNLL 2003 NER F-Measure by more than 1% by using search engine hit counts to improve the boundary detection of the token-based NER. Furthermore, they use Hearst pattern [13] queries and WordNet with entity candidates to perform disambiguation between ambiguous entities.

Most approaches that we have reviewed research the problem on a very narrow domain of about four classes which are typically person, organization, location, and miscellaneous. Some papers introduce other types such as products [24], which is still very broad and insufficient for complex applications [11]. We want to be able to recognize a wide variety of different entity types without creating and maintaining a training corpus. [10] have realized this problem and researched the problem of “Web NER”, that is, the problem of detecting entities of any type. While they perform only entity delimitation by finding the correct boundaries around the entities but not the entity type, we want to scale the named entity

detection and classification on the web level using a flat ontology of 22 entity types.

3 Palladian NER

Our hypothesis states that we can train a NER from the web using only seed lists for each type as learning input. In order to test our hypothesis, we designed a named entity recognizer that can be trained with sparsely-annotated training data. Usually, supervised NERs need to be trained on completely annotated input data such as the CoNLL 2003 dataset and each token must be labeled by an expert. In the case of the CoNLL corpus, this means that every token has either the label `PER` (person), `LOC` (location), `ORG` (organization), `MISC` (miscellaneous), or `O` (no entity / outside). As discussed in the previous section, compiling such a training set is tremendously labor intensive. Our NER must therefore be able to learn from sparsely-annotated texts in which not every token has a label. Thus, we have an open world instead of a closed world as in supervised learning. If a token has no label, we can not assume that it is not an entity (`O`).

3.1 Training the NER

We divide the training on sparse data into three parts: *creating the training data*, *training the text classifier*, and *analyzing the contexts* around the annotations.

Creating the Training Data. The input for our named entity recognizer is not a completely annotated corpus but rather a set of *seed entities*. For example, for the concept Actor we could use “Jim Carrey” as one of the seeds. While there are no limitations on the choosing of the seeds, popular, and more importantly, unambiguous entities with many mentions on the web are the preferred input. Figure 3.1 shows the algorithm for creating the training data. We have n concepts $C = [S_1, \dots, S_n]$, each with a set seeds S . We now query a search engine³ with the exact seed name and its concept and take the top URLs from the result set. Each of the result URLs is downloaded and we extract the textual content of the page using the Palladian Content Extractor [29], removing the header, footer, and navigational elements to acquire only the “main content” of the web page. Next, we annotate all our seed entities for the given concept in the text and save it. We use XML annotations in the form of `<TYPE>seed</TYPE>`. The last operation is cleaning and merging the annotated text files to one. In the cleaning process, we remove all lines that have no annotated seed, are shorter than 80 character, or have no context around the seed.

Train the Text Classifier. After generating the training data we train a dictionary-based text classifier using the seed entities and the surrounding contexts that we found on the web. Unlike many other NERs, we do not employ a large feature vector of numeric features such as TFxIDF, token position, or length. Instead, we treat the entity classification purely as a text

³ We used <http://www.bing.com> in our experiments.

```

begin
  mentions := 10
  for each concept in C
    S := seeds for concept
    for each seed in S
      URLs := querySearchEngine(seed,concept,mentions)
      for each url in URLs
        webPage := download(url)
        text := extractPageText(webPage)
        for each seed in S
          text := annotateSeed(seed,text)
          save(text)
        end
      cleanAndMerge()
    end
  end
end
end
end
end

```

Fig. 1. Generating Training Data

classification problem. The input for our text classifier is a sequence of characters that is then divided into character level n-grams. The dictionary is built by counting and normalizing the co-occurrences of one n-gram and an entity type. The dictionary might then resemble Table 1 in which each column is an entity type (Person, Location, and Product) and each row is a 7-gram. In each cell, we now have the learned relevance for each n-gram and entity type $relevance(ngram, entityType)$. The sum of the relevances in each row must add up to one. The n-gram “r. John” is more likely to indicate a Person (see Table 1: $relevance(\text{“r. John”}, Person) = 0.9$) than Location (see Table 1: $relevance(\text{“r. John”}, Location) = 0.05$) while the n-gram “son jr.” is a little bit more likely to be a Person than a Product.

Table 1. N-Gram dictionary with relevances for entity types

n-gram	Person	Location	Product
r. John	0.9	0.05	0.05
Indiana	0.1	0.85	0.05
son jr.	0.5	0.1	0.4

We build four dictionaries of this kind. One holds only on the seed names and their relevance with the entity types. A second one uses the complete context before and after the annotation within a certain window size. A third one uses only the three context words before and after the mention of the seed entity. A fourth one holds case signatures for all tokens in the training data. We store three case signatures, “A” for completely uppercase words, “Aa” for capitalized words, and “a” for lowercase words. We will show how we use these dictionaries to classify entity candidates in Section 3.2.

Analyze Contexts. Apart from training a text classifier on the context around the seeds, we also build a dedicated context dictionary with context patterns that we find adjacent to the entities. The rationale behind this approach is that the sequence of words before or after a word are good indicators for the entity type. For example, consider the following phrases: “X traveled to Y”, “X was born in Y”, or “X came back from Y”. All of these two to three word contexts indicate that Y might be a location when used as a left context pattern and that X might be a person when seen as a right context pattern. We therefore build a dictionary similar to the one shown in Table 1, but with context phrases instead of n-grams. This approach is similar to [11]. We use context phrases with a length between one and three words. Also, we map all numeric expressions to the word “NUM”. This gives us a higher recall in context phrases with numbers. For instance, it does not matter whether the phrase is “X paid 2 dollars” or “X paid 3 dollars” so we capture “paid NUM dollars” as the context phrase. The context dictionary will be used in the last step of the entity recognition in Section 3.2.

3.2 Using the NER

Once we have trained our named entity recognizer on the automatically generated training data, it is ready to be used. We divide the recognition problem into three parts: *entity detection*, *entity classification*, and *post processing*. These steps are executed sequentially on each given text.

Entity Detection. In the entity detection phase we need to find entity candidates in the text. An entity candidate is a sequence of characters of an unknown type. The output of the entity detection phase is a list of candidate annotations. For instance, given the text “John Hiatt is a great musician from Indiana, USA”, we expect the entity detector to tag the candidates as “<CANDIDATE>John Hiatt</CANDIDATE> is a great musician from <CANDIDATE>Indiana</CANDIDATE>, <CANDIDATE>USA</CANDIDATE>”. Our NER is supposed to work for English language texts only. We therefore use a rule-based detection with regular expressions to find possible entity mentions. In particular, our expression covers sequences of capitalized words, but also allows a few lower-cased words. For example it covers “of” in order to detect “United States of America”. For each detected entity candidate, we perform the same feature generation as shown in Section 3.1.

Entity Classification. The second phase is the classification of the candidates. To classify an entity candidate, we again create all n-grams ($3 \leq n \leq 7$), look up the relevance scores in the dictionary, and assign the entity type with the highest score S to the candidate. The score for each entity type and given entity candidate is calculated as shown in Equation 1 and 2 where $N_{Candidate}$ is the set of n-grams for the given entity candidate and T is the set of possible types that the NER was trained to recognize.

$$S(\text{type}|\text{candidate}) = \sum_{n \in N_{\text{candidate}}} \text{relevance}(n, \text{type}) \quad (1)$$

$$\text{type} = \arg \max_{\text{type} \in T} S(\text{type}|\text{candidate}) \quad (2)$$

Post Processing. In the last phase of entity recognition we filter the classified entities, change their boundaries if necessary, and reclassify their types by using the learned patterns. We apply the following post processing steps.

- We remove all entities that are date fragments such as “Monday” and “July”. This step is necessary because these fragments are capitalized like the entities we actually want to detect, and because they fall under the tagging rules of the entity candidate detector. Of course, we lose a little recall since there are people named “April” and movies with the name “August”, but the precision rises significantly more than the recall drops when applying this filter.
- We remove those date fragments from other entity candidates and change their boundaries. For instance, “July John Hiatt” becomes “John Hiatt”.
- In the entity detection phase, we generate many entity candidates that consist of words at the beginning of a sentence. For instance with “This is a sentence”, we would generate “This” as a candidate entity, when it is not. To filter out these incorrect detections, we employ the case dictionary that we have generated in the training phase. We calculate the ratio of uppercase to lowercase occurrences and remove the entity if the ratio is lower than or equal to one. The rationale behind this approach is that capitalized words at the beginning of sentences might appear more often in a lowercase form indicating that they are not entities. The word “This” from the example has a very low uppercase to lowercase ratio since it most often appears in the lowercase form. We borrow this idea from [20].
- We switch the classified entity type when we found the entity in the training data but classified it incorrectly.
- We now apply the information from the context dictionary that we trained. Again, we take the context around the candidate and look up the score of each entity type matching the context words. We merge the score outcomes of the entity classification, the context classification, and the context words to reassign the most likely entity type to the candidate. For example, if we have classified “Paris” as a person but now the context “born in Paris” suggests that the candidate is much more likely a location, the context classifiers overrule the candidate-only classifier and switch the label. We now use our set D of dictionaries. These are (1) candidate-only, (2) context n-grams, and (3) context patterns. As shown in Equation 3 we combine the scores. We then use Equation 2 to get the most likely entity type for the candidate.

$$S(\text{type}|\text{candidate}) = \sum_{d \in D} S_d(\text{type}|\text{candidate}) \quad (3)$$

- In the last step, we use the learned left context information to change the boundaries on multi-token candidates. For example, we have detected the candidate “President Obama”, but knowing that the token “President” appeared many times before the actual entity in the training set, we drop this token and change the boundaries so that the candidate represents “Obama” only.

4 Evaluation

4.1 Datasets

We use two datasets for our evaluation, the CoNLL 2003 and the TUD 2011 dataset. In this section, we will describe the properties, similarities, and differences between the datasets.

CoNLL 2003. The CoNLL 2003 dataset⁴ was developed for the shared task on CoNLL in 2003. The participants in that task were asked to provide a language-independent named entity recognition tool to work on English and German data recognizing the types: person, organization, location, and miscellaneous. In our evaluation we only use the English part of the dataset. The dataset also comes with additional data such as list of names, POS-tags, and non-annotated data which we do not use in our experiments. The data is separated into training (68%), validation (for tuning the systems, 17%), and test (15%). A Reuters corpus of news wire articles was annotated by the University of Antwerp for this dataset.

While the CoNLL 2003 dataset is a valuable resource it comes with a number of drawbacks. Most importantly, only three entity types in addition to MISC type are tagged. We have noticed that the MISC type very often represents a nationality identifier such as “Spanish”. Since there are many more entities that could fall into this category (products, buildings, landmarks, etc.) it is questionable whether the type of articles covers enough variety. As we have argued in the motivation of this paper, this categorization is too broad for many real world applications. Therefore, the performance levels that NERs can reach on this dataset might be misleading. It is unclear whether they would work with a much finer grained classification. Furthermore, we found that some documents in the dataset do not seem to resemble a real text but rather look as if they were scraped table contents.

TUD 2011. The TUD 2011 dataset⁵ was developed at the Dresden University of Technology to evaluate Named Entity Recognition on a finer grained level. Furthermore, it was important to us that we did not get data solely from one source otherwise the writing tends to be too homogeneous. We therefore used different web pages as sources for the dataset. Two annotators spent about 100 hours annotating 22 entity types in the dataset. In the first step, 20 seed entities

⁴ <http://www.cnts.ua.ac.be/CoNLL2003/ner/>

⁵ <http://areca.co/7/Web-Named-Entity-Recognition-TUDCS4>

per entity type were collected from web sources. We then used the algorithm from Figure 3.1 to automatically find possible mentions of the seed entities on the web. Documents for each entity type were then hand tagged for about four hours. We tried to assign the most specific type to each entity; if there was no fitting type we went up in the hierarchy and if nothing matched there, we had to assign MISC. For example, “Jim Carrey” would be annotated as Actor while “Bill Gates” would be a Person since there is no specific matching type. The 127 documents are separated into training (50%) and test (50%) sets.

Table 2 shows the number of tagged entities per type in the training and test part of the dataset. While the five top level types are almost evenly distributed, the 17 more specific types show more variation in the number of annotated instances. This variation is due to the fact that country names, for example, also occur in documents about airplanes, but not the other way around. Altogether, 3,400 mentions were annotated.

Table 2. Number of tagged entities per type

Entity Type	Training	Test	Total	Entity Type	Training	Test	Total
Person	216	166	382	Car	19	14	33
Politician	54	66	120	Comp. Mouse	18	21	39
Actor	59	28	87	Movie	28	30	58
Athlete	60	78	138	Mobile Phone	24	24	48
Location	178	145	323	Organization	201	231	432
Country	143	119	262	Newspaper	70	39	109
City	142	142	284	Team	60	75	135
Lake	28	14	42	University	14	17	31
Airport	40	30	70	Restaurant	19	20	39
Product	64	44	108	Band	37	20	57
Airplane	92	118	210	Misc	243	150	393

4.2 Comparisons

In this section, we evaluate our named entity recognizer on the two datasets and compare it with state-of-the-art NERs. In particular, we compare it to LingPipe⁶ [1], which uses a character language model on top of a hidden markov model, OpenNLP⁷, which uses a maximum entropy approach for tagging, and the Illinois LBJ Tagger⁸ [25], which is based on conditional random fields. All these recognizers work in a supervised manner, needing manually-labeled training data. Palladian works either in supervised mode (named just “Palladian” in the charts) but can learn from sparsely-annotated training data on the web, too (named “Palladian Web” in the charts). We use the MUC evaluation scores for precision, recall, and the F1 measure. Furthermore, it is important to note that

⁶ <http://alias-i.com/lingpipe>

⁷ <http://incubator.apache.org/opennlp/>

⁸ http://cogcomp.cs.illinois.edu/page/software_view/4

all evaluation is done on “unseen” data, that is, we ignored annotations in the test set which could have been learned in the training data. We do this to avoid as much dictionary learning as possible since our goal is to extract new entities from text, not primarily to recognize what we know already.

Evaluation on CoNLL 2003. Figure 2 shows the comparison of the four NERs on the CoNLL data. All taggers were trained on 68% training data and tested on 15% test data from the dataset. This graphic shows what we can reach when using supervised learning with the training data.

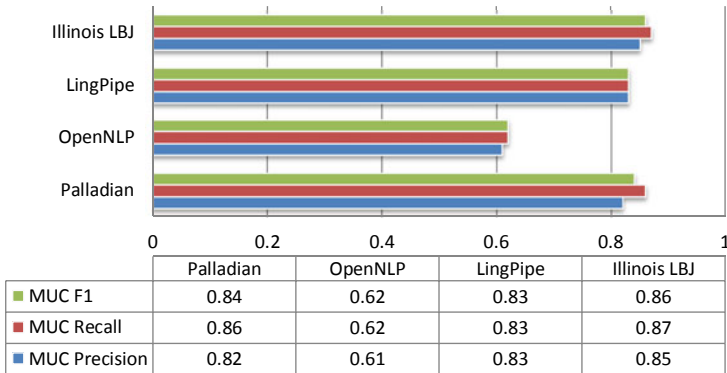


Fig. 2. Comparison of recognizers with supervised learning on CoNLL

Figure 3 shows the performance of the NERs based on the size of the training data. We evaluated the performance by continuously increasing the training set by ten documents (in total, CoNLL provides about 600 training documents). We can see that all NERs benefit from more training data. The dashed line signifies the performance of Palladian Web – Palladian in web training mode. We used 1 to 50 seed entities per type and automatically generated training data using the algorithm from Figure 3.1. The upper x-axis shows the number of seeds that were used for training Palladian Web. While this approach works **without** hand-labeled training data, the F1-Score is about 29% below the maximum F1-Score of the best tagger when using completely annotated training data. Still even for 20 training documents the web training approach still outperforms LingPipe and up to over 250 training documents it beats OpenNLP’s performance.

Evaluation on TUD 2011. Figure 4 shows the comparison of the four NERs on the TUD data. All taggers were trained on 50% training data and tested on the other 50% test data from the dataset. It is clear that the performance is dramatically worse for all the NERs compared to the CoNLL dataset. The worsened performance stems from a smaller number of training documents, more diverse training data, and more importantly the higher number of entity types that need to be recognized. From this chart, we can conclude that even state-of-the-art NERs perform poorly on diverse web text when the task is to extract unseen

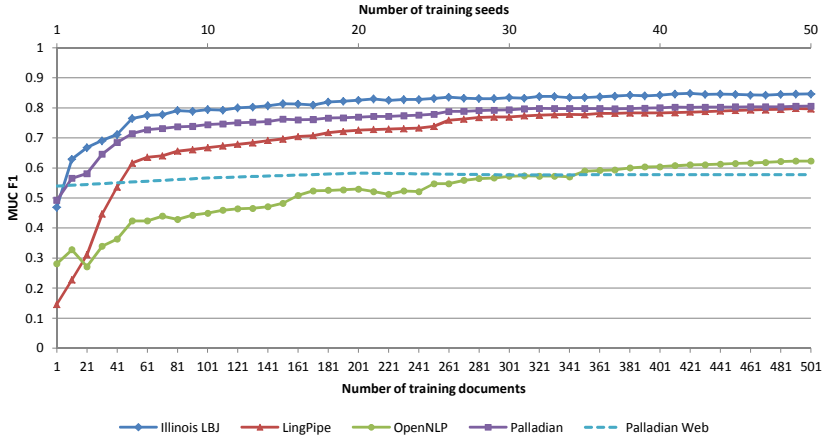


Fig. 3. Comparison of recognizers based on the size of training data on CoNLL

entity mentions. The performance could be improved by allowing gazetteers, but this is not our focus.

Figure 5 shows the performance of the NERs based on the size of the training data from the TUD dataset (in total, the dataset provides about 60 training documents). We increased the number of training documents by five for each evaluation. We can see a similar behavior of the taggers as in Figure 3. We trained the Palladian Web tagger the same way as before, supplying it with automatically generated training data. It takes the Illinois LBJ tagger about ten training documents to reach the performance of the Palladian Web tagger trained on 10 seeds. This time, however, the number of training documents is too small for LingPipe and OpenNLP to reach the automatically trained tagger’s performance at all. Training Palladian Web with 50 seed entities per type performs only about 9% worse than the Illinois LBJ after 56 training documents.

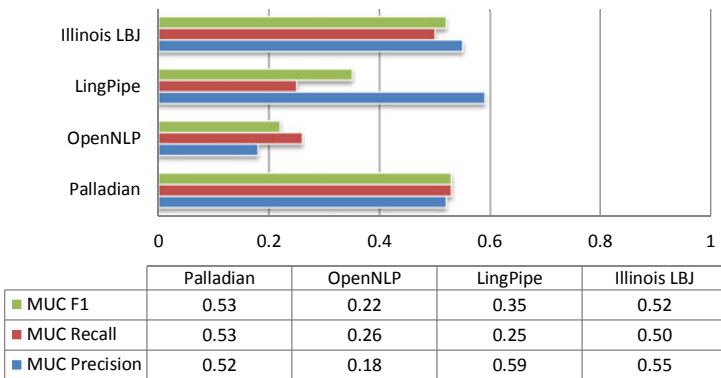


Fig. 4. Comparison of recognizers with supervised learning on TUD

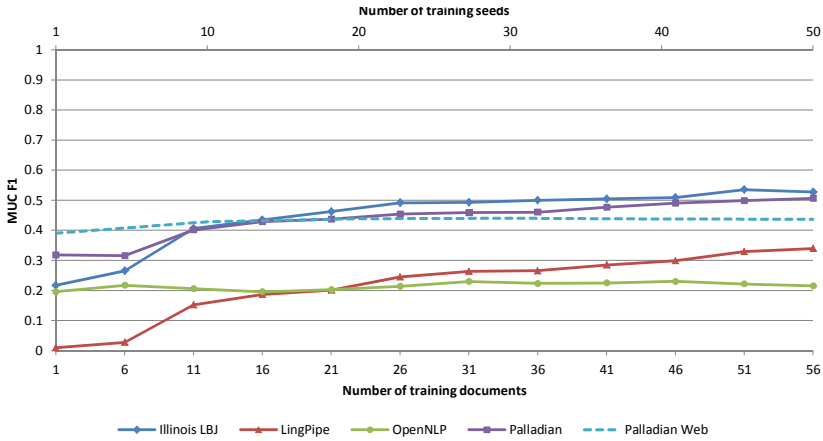


Fig. 5. Comparison of recognizers based on the size of training data on TUD

We can see a promising direction of automatically training the NER on the web since acquiring training data is extremely costly, and when a new entity type should be extracted, the complete data needs to be manually re-annotated.

5 Conclusion and Open Questions

In this paper, we have presented the named entity recognizer Palladian, which can be automatically trained on the web requiring only a small number of seed entities per entity type. While this approach is outperformed by training supervised state-of-the-art NERs on manually-labeled training data, it is rather competitive when little training data is given and many different entity types should be extracted. Furthermore, we have shown that named entity extraction which focuses on recognizing unknown entities is a rather difficult problem when the task is done on heterogeneous web texts and a larger number of entity types need be recognized. In our future studies, we will combine the presented NER with the entity extraction techniques from [28] to improve the quantity and quality of the entity extractions.

References

- [1] Alias-i. Lingpipe 4.0.1 (2011), <http://alias-i.com/lingpipe>
- [2] Asahara, M., Matsumoto, Y.: Japanese named entity extraction with redundant morphological analysis. In: Proceedings of Human Language Technology Conference (HLT-NAACL), pp. 8–15 (2003)
- [3] Bikel, D.M., Miller, S., Schwartz, R., Weischedel, R.: Nymble: a high-performance learning name-finder. In: Proceedings of the Fifth Conference on Applied Natural Language Processing, pp. 194–201. Association for Computational Linguistics Morristown, NJ (1997)

- [4] Borthwick, A., Sterling, J., Agichtein, E., Grishman, R.: NYU: Description of the MENE named entity system as used in MUC-7. In: Proceedings of the Seventh Message Understanding Conference (MUC-7), vol. 6 (1998)
- [5] Buchholz, S., van den Bosch, A.: Integrating seed names and n-grams for a named entity list and classifier. In: Proceedings of the Second International Conference on Language Resources and Evaluation, pp. 1215–1221 (2000)
- [6] Chiticariu, L., Krishnamurthy, R., Li, Y., Reiss, F., Vaithyanathan, S.: Domain adaptation of rule-based annotators for named-entity recognition tasks. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, pp. 1002–1012 (2010)
- [7] Cohen, W.W.: Fast effective rule induction. In: Machine Learning, International Workshop then Conference, pp. 115–123. Morgan Kaufmann, San Francisco (1995)
- [8] Collins, M., Singer, Y.: Unsupervised models for named entity classification. In: Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, pp. 189–196 (1999)
- [9] Cucerzan, S., Yarowsky, D.: Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In: Proceedings of the Workshop on Very Large Corpora at the Conference on Empirical Methods in NLP., pp. 90–99 (1999)
- [10] Downey, D., Broadhead, M., Etzioni, O.: Locating complex named entities in web text. In: Proceedings of IJCAI (2007)
- [11] Fleischman, M., Hovy, E.: Fine Grained Classification of Named Entities. In: Proceedings of the 19th International Conference on Computational Linguistics, vol. 1, pp. 1–7. Association for Computational Linguistics (2002)
- [12] Grishman, R., Sundheim, B.: Message understanding conference-6: A brief history. In: Proceedings of the 16th conference on Computational linguistics, vol. 1, pp. 466–471. Association for Computational Linguistics, Morristown (1996)
- [13] Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th Conference on Computational Linguistics, vol. 2, pp. 539–545. Association for Computational Linguistics, Morristown (1992)
- [14] Iacobelli, F., Nichols, N., Hammond, L.B.K.: Finding new information via robust entity detection. In: Proactive Assistant Agents (PAA 2010) AAAI 2010 Fall Symposium (2010)
- [15] Klein, D., Smarr, J., Nguyen, H., Manning, C.D.: Named entity recognition with character-level models. In: Proceedings of CoNLL, vol. 3 (2003)
- [16] Kozareva, Z., Bonev, B., Montoyo, A.: Self-training and co-training applied to spanish named entity recognition. In: Gelbukh, A., de Albornoz, Á., Terashima-Marín, H. (eds.) MICAI 2005. LNCS (LNAI), vol. 3789, pp. 770–779. Springer, Heidelberg (2005)
- [17] McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: Seventh Conference on Natural Language Learning (CoNLL) (2003)
- [18] McDonald, D.D.: Internal and external evidence in the identification and semantic categorization of proper names. In: Corpus Processing for Lexical Acquisition, pp. 21–39 (1996)
- [19] Meulder, F.D., Daelemans, W., Hoste, V.: A named entity recognition system for dutch. *Language and Computers* 45(1), 77–88 (2002); ISSN 0921-5034
- [20] Millan, M., Sánchez, D., Moreno, A.: Unsupervised Web-based Automatic Annotation. In: Proceeding of the 2008 conference on STAIRS 2008: Proceedings of the Fourth Starting AI Researchers' Symposium, pp. 118–129. IOS Press, Amsterdam (2008)

- [21] Milne, D., Witten, I.H.: Learning to link with wikipedia. In: Proceeding of the 17th ACM Conference on Information and Knowledge Management, pp. 509–518. ACM, New York (2008)
- [22] Nadeau, D., Sekine, S.: A Survey of Named Entity Recognition and Classification. *Named Entities: Recognition, Classification and Use*, pp. 3–28 (2009)
- [23] Nadeau, D., Turney, P.D., Matwin, S.: Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In: *Proceedings of the Canadian Conference on Artificial Intelligence*, pp. 266–277. Springer, Heidelberg (2006)
- [24] Niu, C., Li, W., Ding, J., Srihari, R.K.: Bootstrapping for named entity tagging using concept-based seeds. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003-short papers, vol. 2*, pp. 73–75. Association for Computational Linguistics (2003)
- [25] Ratnov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pp. 147–155. Association for Computational Linguistics (2009)
- [26] Sekine, S.: NYU: Description of the Japanese NE System used for MET-2. In: *Proceedings of the Seventh Message Understanding Conference (MUC-7)* (1998)
- [27] Szarvas, G., Farkas, R., Ormándi, R.: Improving a state-of-the-art named entity recognition system using the world wide web. *Advances in Data Mining. Theoretical Aspects and Applications*, 163–172 (2007)
- [28] Urbansky, D., Feldmann, M., Thom, J.A., Schill, A.: Entity extraction from the web with webKnox. In: Snášel, V., Szczepaniak, P.S., Abraham, A., Kacprzyk, J. (eds.) *Advances in Intelligent Web Mastering - 2. AISC*, vol. 67, pp. 209–218. Springer, Heidelberg (2010)
- [29] Urbansky, D., Muthmann, K., Katz, P., Reichert, S.: *Palladian: A toolkit for Internet Information Retrieval and Extraction*. Website (May 2011), <http://www.palladian.ws/documents/palladianBook.pdf>
- [30] Wu, D., Ngai, G., Carpuat, M., Larsen, J., Yang, Y.: Boosting for named entity recognition 20, 1–4 (2002)

Exploiting Available Memory and Disk for Scalable Instant Overview Search

Pavlos Fafalios and Yannis Tzitzikas

Institute of Computer Science, FORTH-ICS, Greece, and
Computer Science Department, University of Crete, Greece
{fafalios,tzitzik}@csd.uoc.gr

Abstract. *Search-As-You-Type* (or *Instant Search*) is a recently introduced functionality which shows predictive results while the user types a query letter by letter. In this paper we generalize and propose an extension of this technique which apart from showing on-the-fly the first page of results, it shows various other kinds of information, e.g. the outcome of results clustering techniques, or metadata-based groupings of the results. Although this functionality is more informative than the classic search-as-you type, since it combines *Autocompletion*, *Search-As-You-Type*, and *Results Clustering*, the provision of real-time interaction is more challenging. To tackle this issue we propose an approach based on pre-computed information and we comparatively evaluate various index structures for making real-time interaction feasible, even if the size of the available memory space is limited. This comparison reveals the memory/performance trade-off and allows deciding which index structure to use according to the available main memory and desired performance. Furthermore we show that an incremental algorithm can be used to keep the index structure fresh.

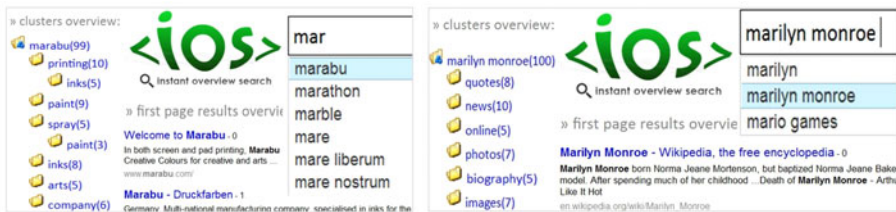
1 Introduction

Autocompletion services help users in formulating queries by exploiting past (and logged) queries. Recently, Google adopted *Instant Search*, a new *Search-As-You-Type* (for short *SAYT*) enhancement that apart from allowing the user to find out what is popular given the current input string, it also shows an overview of the top hits (first page of results) as the user types a query. In particular, one or more possible matches (actually “completions”) of the typed query are computed and immediately their results are presented to the user. This immediate feedback allows the user to stop typing if the desired results have already been recommended. If on the other hand the sought results are not there, the user can continue typing or change what he has typed so far. In general, we can say that the user adapts his query on the fly until the results match what he wants. Although the usefulness of this functionality has not been measured or proved (we did not manage to find any work that reports the results of a user study), it is true that people type slowly but read quickly (a glance at another part of the

page typically takes only a few milliseconds¹), implying that the user can scan a results page while he types. Furthermore, we observe an increasing interest on providing such functionality evidenced by the emergence of several systems e.g. *EasySearch*², *Keyboardr*³, or *Wiki Instant*⁴.

Apart from the benefit for the user's side, SAYT is also beneficial for the WSE (Web Search Engine), in the sense that the suggested answers have been pre-computed, thus the engine has to evaluate less queries at run time.

In this paper we generalize and propose a more powerful search-as-you-type functionality which apart from showing on-the-fly only the first page of results of the guessed query, it can show several other kinds of supplementary information that provide the user with a better overview of the search space. We shall call this paradigm of searching *Instant Overview Searching*, for short IOS. Regarding the supplementary information, we focus on *results clustering* since it can provide informative overviews of larger parts of the answer. Such "*Cluster-As-You-Type*" functionality, can be considered as an instance of IOS. To grasp the idea, Fig. 1 shows an indicative screen dump. Consider a user who wants to find information about the biography of *Marilyn Monroe* and for this reason he starts typing the query "marilyn monroe biography" letter by letter. After having typed the first three letters, i.e. the string "mar", a set of query completions are prompted as shown in Fig. 1a. At the same time, the first page of results and the cluster label tree that correspond to the top suggestion "marabu" appear instantly (at the main area and the left bar of the screen respectively). At that point the user can either continue typing or select a suggestion from the list. If he selects one of the suggestions the first page and the clustering of the results appear immediately. Moreover, the user is able to click on a cluster's name and get the results of only that cluster.



(a) Start typing a query. (b) Select the suggestion "marilyn monroe".

Fig. 1. IOS indicative screen dumps

Suppose the user continues typing and presses the key "i". The list of suggestions is refreshed according to the new input "mari", the top suggestion is now different (i.e. "marilyn") and thus the first page and the cluster label tree

¹ <http://www.google.com/instant/>

² <http://easysearch.webs.com/home.htm>

³ <http://keyboardr.com/>

⁴ <http://wikinstant.com/>

change according to the new top suggestion. Moreover, since there is the suggestion “marilyn monroe” that matches what he would type, the user selects it (Fig. 1b) and then the cluster label tree and the first page of the query “marilyn monroe” appears immediately. We observe that the cluster label tree contains a cluster with label “biography (5)”. The user clicks on that cluster and gets the results that are relevant to his information need. Notice that the user typed only 4 letters and made only 2 clicks in total. Furthermore, he has seen how the results about Marilyn Monroe are clustered.

It is not hard to see that efficiency and real-time interaction is challenging. To tackle this challenge, we propose enriching the trie structure [13] which is used for query autocompletion with the various pre-computed supplementary information. For example, and for the case of clustering, for each query in the trie we keep the outcome of the results clustering algorithm, specifically the cluster label tree (the cluster label tree is a tree of strings, each being the label, readable name, for a cluster). This choice greatly reduces the required computation at interaction time, however it greatly increases the space requirements of the trie. For this reason in this paper we describe and comparatively evaluate various index structures. Essentially each index structure is actually a method for partitioning the information between main and secondary memory. We have conducted a detailed performance evaluation according to various aspects such as system’s response time and main memory cost, which reveals the main performance trade-off and assists deciding which index structure to use according to the available main memory and desired performance. Furthermore we show that an incremental algorithm can be used to keep the index structure fresh in affordable time.

Finally we should clarify that the overall effectiveness of the provided functionality, apart from its efficiency, depends on the quality of the pre-computed information, i.e. on the quality of (a) the ranking method that determines query suggestions, (b) the hits in the first page of results, and (c) the labels returned by the clustering. However we should stress that the index structures that we introduce and their analysis can be used with *any* autocompletion, ranking and clustering method, making our approach widely applicable.

The rest of this paper is organized as follows. Section 2 motivates IOS by sketching applications. Section 3 discusses related works. Section 4 introduces the index structures and Section 5 reports extensive comparative experimental results. Finally, Section 6 concludes and identifies issues that are worth further research.

2 Applications of IOS

In this section we describe in brief two main forms of IOS; over a *meta* web search engine (MSE), and over a *standalone* web search engine (WSE).

For instance, [8] describes a MSE over Google that clusters at real time the retrieved results according to the words that appear in the title and the snippet of each result. Such a system could pre-compute and store the cluster label

tree and the top- k results for each query of the log file. In this way the system could provide an overview of the results allowing the user to adapt his query while he is typing. Suppose that a user searches for “apple iphone” and starts typing that string. After having typed “apple”, he notices that there is a cluster with name *apple iphone* with 15 results. Instantly, he can click on that cluster label and view the results. If the results do not satisfy his information need, he can continue typing. However we should note that in the context of a MSE the delivered results depend on the results of the selected underlying search engines. Since the results of the same query may be different at different points in time, the pre-computed and stored cluster label tree of the query results may be different from the cluster label tree of the current query results. This means that a particular stored term/label of a cluster may not exist in the cluster label tree of a future search of the same query. Therefore, when a user clicks on a cluster’s term, the system may not be able to perform the query and then focus on the selected cluster unless the results of this cluster (i.e. its top-10 hits) have been also stored beforehand. Of course that approach increases the amount of pre-computed information that has to be stored and fetched. As we shall see, the index structures that we will introduce can be used to provide real-time interaction also in such cases.

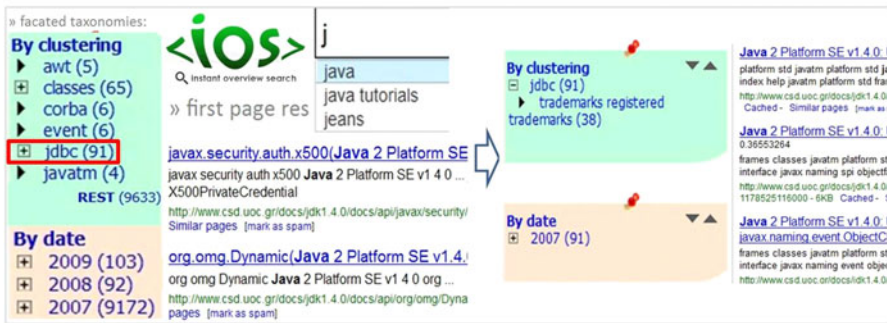


Fig. 2. IOS over a faceted search engine

In the context of a standalone WSE, the engine can pre-compute and store not only the cluster label tree, but also the facets (metadata-based groupings) of the top- k results of each logged query. For example, in [12] the engine characterizes the top- k results according to five facets: *by clustering*, *by domain*, *by date*, *by filetype* and *by language*. The provision of such information during typing can accelerate the search process. For instance, consider a user seeking for SIGIR 2010 papers. While he types this query letter by letter, he notices that for the top suggestion “SIGIR”, there is the term *2010* below the facet *By date* with 50 results. This means that he can directly click on that term and get the corresponding results (on user click the engine executes the top suggested query and directly focuses, i.e. restricts the answer on the clicked term/facet). Two screendumps of a prototype application we have developed are given at Fig. 2.

It follows that IOS can be used for accelerating the performance of multifaceted browsing interfaces [4,3,6].

3 Background and Related Work

A SAYT system computes answers to keyword queries as the user types in keywords letter by letter. Fig. 3 illustrates the architecture of such a system. At each keystroke of the user the browser sends (in AJAX style) the current string to the server, which in turn computes the top suggestions and returns to the browser the best answers ranked by their relevancy to the top-suggested query.

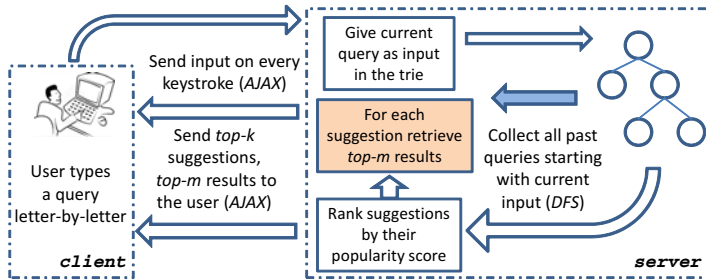


Fig. 3. A Search-As-You-Type (SAYT) System

There are several works describing letter-by-letter searching for various kinds of sources, e.g. for relational databases [5,15,10,11], or documents [2,9]. To cope with the efficiency problem, most use pre-computed results and in-memory indexes (i.e. the entire index is loaded in main memory). The only work that does not load everything in main memory is the semantic search engine *CompleteSearch* [5] [1]. That work differs from ours in the following aspects: (1) *CompleteSearch* is not a generic search engine, but is based on specific known datasets (like Wikipedia and DBLP) with predefined semantic categories, (2) its suggestion system is word based, not query based, i.e. it suggests only words that match user's current input, not whole queries, (3) it focuses on compression of index structures, especially in disk-based settings, while IOS uses partial in-memory indexes, (4) in general *CompleteSearch* helps users to formulate a good semantic query while IOS helps users to locate directly and fast what they are looking for.

4 Index Structures

To tackle the requirements of IOS, we propose enriching the trie that is used for autocompletion with the results of the pre-processing steps. Specifically, and for the scenario of Fig. 1, for each query stored in the query log file, we extend its entry in the trie by two additional strings. The first string contains the first

⁵ <http://search.mpi-inf.mpg.de>

page of results of the guessed query (i.e. an HTML string containing for each hit its title, snippet and URL). The second string contains the *faceted dynamic taxonomy*, for short facets (or only the cluster label tree in the case of a MSE). Note that we prefer to store both strings in HTML format in order to save time while presenting the results to the user (no need for any post-processing).

Obviously, such enrichment significantly increases the size of the trie, since for each query we have to save two additional long strings. Specifically, in the original trie for each logged query of n characters, the trie keeps a node of about n bytes (usually $n = 16$). However, the string that represents the cluster label tree (or the facets) can be about 30,000 bytes and the string that represents the first page of results is about 40,000 bytes (including the characters of the HTML code for both strings). Below we propose methods and index structures for managing this increased amount of data.

The first idea is to adopt the *trie partitioning* method proposed in [7]. According to that method, only one “subtrie” (of much smaller size) is loaded in main memory. Since users seldom change their initial queries [14], dividing the trie in this way implies that once the appropriate subtrie has been loaded (during the initial user’s keystrokes), the system can compute the completions of the subsequent requests using the same subtrie. Now suppose the case where we do not have enough main memory to load the enriched trie (or subtrie). In such case we can save the results of the preprocessing steps (e.g. facets, cluster label tree, first hits), in one or more different files. Consequently, the trie for each query entry has to keep only three numbers: (a) one for the file, (b) one for the bytes to skip and (c) one for the bytes to read. Obviously, one should use *random access files* for having fast access to the pre-computed information. This approach greatly reduces the size of the trie, however we have to perform additional disk accesses for reading the pointing file.

Note that this approach could be used in the context of a MSE, in order to store also the results of each cluster (as we discussed previously). We could further reduce the size of the trie that is loaded in the main memory by combining the last two approaches (trie partitioning and trie with indices to external files). Such approach requires very small amount of main memory, however it requires more time for loading the appropriate subtrie in the main memory and reading the data from the pointing file.

To clarify the pros and cons of the aforementioned approaches, we decided to comparatively evaluate the following approaches (depicted at Fig. 4):

- (a) **(SET) Single Enriched Trie.** The entire enriched trie in main memory.
- (b) **(STIE) Single Trie with Indices to External files.** Single (query) trie in main memory with pointers to external random access files where a pointer consists of 3 numbers (file number, bytes to skip, and bytes to read).
- (c) **(PET) Partitioned Enriched Trie.** The *enriched* trie is partitioned to several subtrees using the partitioning proposed in [7].
- (d) **(PTIE) Partitioned Trie with Indices to External files.** The (query) trie is partitioned ([7]) and each subtrie contains pointers to external random access files.

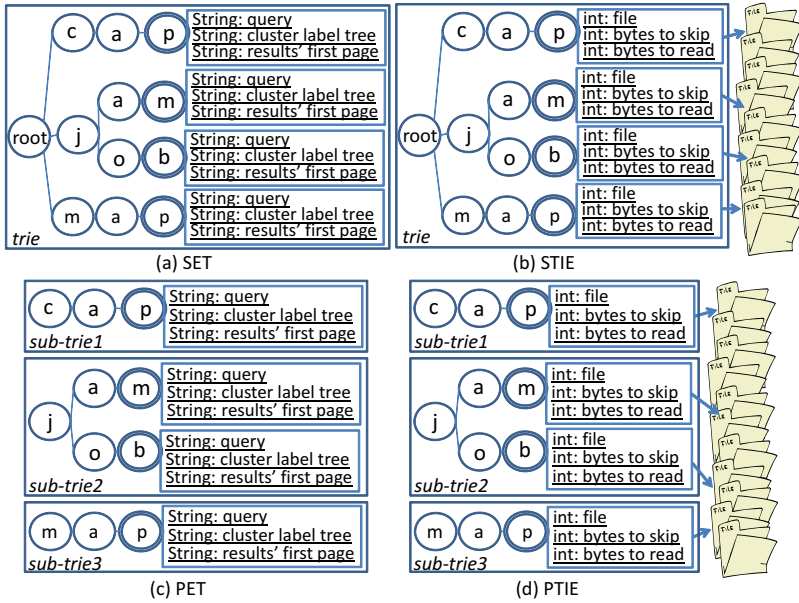


Fig. 4. Index implementation approaches

5 Experimental Evaluation

The objective is to comparatively evaluate the previous approaches, to identify the more scalable one(s) and to clarify the main performance trade-offs.

Evaluation Aspects. We evaluated the above approaches according to the following aspects:

[*Trie Size to be loaded in main memory*] We will measure the size of the trie that has to be loaded in main memory. The key requirement for those approaches which are based on a single trie, is to fit in memory, since loading is done only once (at deployment time), so the loading time of the trie does not affect the performance at user requests. Instead, the loading time is very important for those approaches relying on multiple subtries, since each user request may require loading the appropriate subtrie from the disk.

[*Average retrieval time*] We will measure the average time for retrieving the suggestions, the results and the clusters/facets for a specific current user's query. This time does not contain the network's delay time and the javascript running time.

[*Construction Time and Update Time*] We will measure the time required to process the query log and construct the corresponding index. This task has to be done once, however it should be redone periodically since the log file changes. We should clarify at this point that in contrast to the problem of inverted file construction, in our case we do not have main memory problems at construction time, specifically the most space consuming part of the pre-computed information (i.e. the outcome

of results clustering and the first page of hits) can be directly stored in the external file(s) at construction time, thus there is no need for creating and merging partial indices (as it is the case of inverted file construction). We will also investigate the time required to *update* the index structures (after query log change) instead of constructing them from scratch.

Table 1. Query Log Files

Num. of log's queries	Num. of unique queries	Avg num. of words per query	Num. of distinct words	Avg num. of chars per query
1,000	578	2.23	950	15.5
10,000	5,341	2.3	6,225	16
20,000	10,518	2.34	10,526	16.2
40,000	20,184	2.35	17,179	16.2

Data Sets and Setup. We used 4 query log files of different sizes. One with 1,000, one with 10,000, one with 20,000 and one with 40,000 queries. Each file is a subset of a random log sample of about 45,000 fully anonymized queries from a *real* query log file (from Excite⁶ WSE). Table 1 illustrates some features of these files. We should note that it is not necessary to have a very big set of distinct queries for an IOS functionality. Note that in WSE query logs the query repetition frequency follows a Zipf distribution [16], i.e. there are few queries that are repeated frequently and a very large number of queries which have very small frequency. Obviously the latter are not very useful to be logged and used as suggestions for an IOS functionality in the sense that they will not be suggested as completions (their rank will be very low). Regarding trie partitioning, we created subtrees each corresponding to the first two characters ($k = 2$) of the queries. As shown in [7], for $k = 2$ the partitioning yields subtrees whose sizes are very close (close to ideal). All experiments were carried out in an ordinary laptop with processor Intel Core 2 Duo P7370 @ 2.00Ghz CPU, 3GB RAM and running Windows 7 (64 bit). The implementation of all approaches was in Java 1.6 (J2EE platform), using Apache Tomcat 6 and Ajax JSP Tag Library.

Results on Trie's Sizes. Fig. 5 illustrates how the size of the trie that is loaded in the main memory grows in each approach.

In the SET approach (Fig. 5a), the size of the trie increases linearly with the query log size and can get quite high. In particular, we observe that SET requires about 40 MB for storing the results and the clusters of a query log file of 1000 queries (578 distinct queries), i.e. it needs about 70 KB per query.

In the PET approach (Fig. 5b), we examine how the size of a subtree grows as a function of the entries (results and clusters of a query), we decide to store in it. Since the subtrees do not have the same size, the diagram presents the worst case (max size), the best case (min size) and the mean case (average size) of a subtree's size (for a query log file of 40,000 queries). However, the smaller this number is, the more subtrees have to be created. For example, selecting to

⁶ <http://www.excite.com>

store 50 entries in each subtrie, 170 tries have to be created (note that this also depends on the number of distinct substrings of size k , for more see [7]). On the other hand, selecting to store 800 entries per subtrie, only 24 subtrees are created.

In the STIE approach (Fig. 5c), the number of entries we select to store in each separate file does not affect the size of the trie that is loaded in the main memory. We observe that this approach leads to a very small trie's size. In particular, STIE requires about 1,8 MB for a query log file of 1,000 queries (578 distinct queries), i.e. it needs only about 3 KB per query. For a log file of 40,000 queries, selecting to store 50 entries in each file leads to the creation of 404 external files (with average size of about 3.5 MB each one). At the same time, selecting to store 400 entries in each file, 51 files have to be created (with average size of about 28 MB each one).

In the PTIE approach (Fig. 5d), we combine trie partitioning and trie with indices to external files in order to further reduce the size of the trie. As in PET, we examine how the number of the entries we select to store in each subtrie affects its size (as we mentioned above, the number of entries we select to store in each separate file does not affect the size of the trie). Since the subtrees have not the same size, the diagram presents the worst case (max size), the best case (min size) and the mean case (average size) of a subtrie's size (for a query log file of 40,000 queries and selecting to store 400 entries in each external file). We observe that even for the worst case, the size of a subtrie is extremely low.

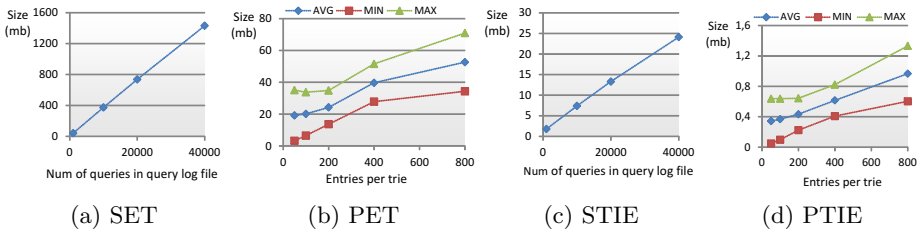


Fig. 5. Size of the Trie

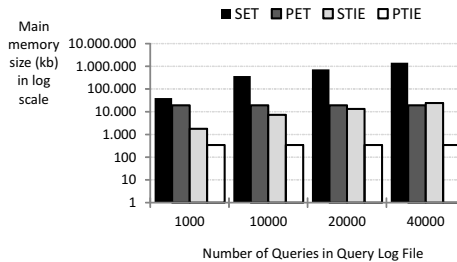


Fig. 6. Comparison of the trie's size

Conclusion. Fig. 6 compares the four approaches where the y-axis is in logarithm scale. For the PET and PTIE approach, we choose to depict the best case (50 entries per trie), so the average size of a subtrie is constant and independent of the query log size. As expected, SET requires the more main memory space, which can be very big for large query logs. The best approach (regarding only the size of the trie) is PTIE with great difference from the others. STIE follows but for query logs of smaller than 40,000 queries. Finally, PET requires less space than STIE only for very large query logs files (more than 40,000 queries).

Average Retrieval Time. Fig. 7 depicts the average retrieval time for each implementation approach (for the PET approach, the corresponding diagram concerns a query log file of 40,000 queries).

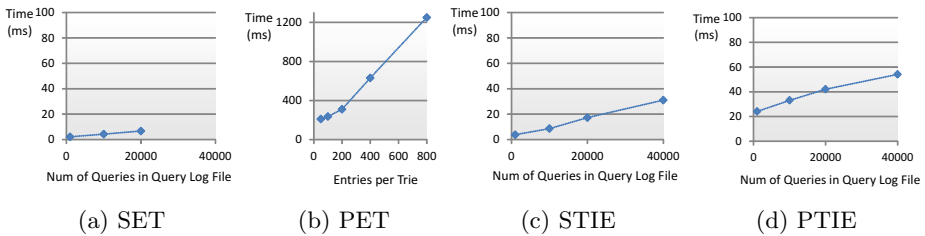


Fig. 7. Average retrieval time

We can see that SET has very low average retrieval time (almost constant), taking only a few milliseconds to retrieve the required data even for very large query log files (Fig. 7a). However, it requires that the entire trie fits the main memory. In PET, the average retrieval time depends on the size of the appropriate subtrie that needs to be loaded in the main memory, i.e. the number of entries per subtrie. We observe that for all cases, PET is much slower than SET (Fig. 7b). The STIE approach was implemented using *random access files* and therefore the average retrieval time does not depend on the size of the external file. We observe that this approach is very efficient even for large query log files with average retrieval time lower than 40 ms (Fig. 7c). Finally, as in STIE approach, PTIE was implemented using random access files. For this reason, its average retrieval time depend mainly on the size of the query log file (Fig. 7d). We observe that this approach is a bit worse than STIE. The reason is that it requires one more disk access in order to find and load the appropriate subtrie.

Synopsis. Fig. 8b compares the average retrieval time of all approaches (for PET approach, we consider the best case of 50 entries per subtrie). It is obvious that the SET approach is much more efficient than all the other approaches. However, as mentioned above, for large query log files its size is huge and it does not fit in main memory. For this reason, one may argue that the best approach is STIE, as it combines low trie size and very fast retrieval time. Moreover, PTIE is a very good approach as it offers very small trie size and efficient retrieval time.

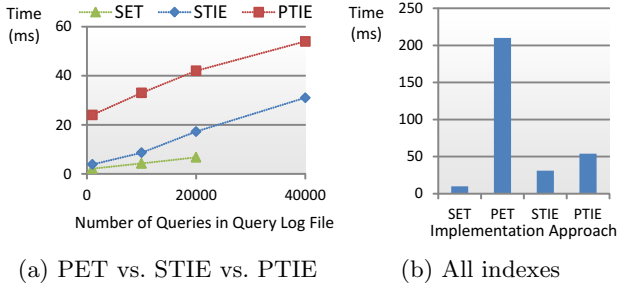


Fig. 8. Comparison of the average retrieval time

Finally, PET approach is the worst although its retrieval time is not unacceptable (about 200 ms). Fig. 8a compares only the SET, STIE and PTIE approaches, excluding PET (as it is independent on the size of the query log file). We observe that in all approaches, the average retrieval time is very low. SET is the more efficient approach, followed by STIE, and finally by PTIE.

At last we should mention that experiments over bigger synthetic query logs yield the same conclusions⁷.

5.1 Selecting the Right Index

The main conclusion is that the proposed functionality is feasible for real time interaction even if the log file and the preprocessed information have considerably high size. The selection of the implementation approach depends on the available main memory, the size of the log file, and the size of the preprocessed information. Below we describe criteria that should be used and in the right order.

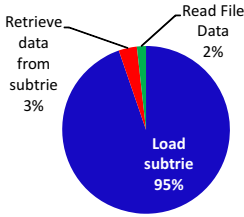
1/ If the entire SET fits in memory then this is the faster choice since no loading has to be done during user's requests.

2/ If SET does not fit in memory then, the next choice to follow is STIE since it offers faster retrieval time in comparison to PET and PTIE. However note that STIE is feasible only if the trie of the query log fits in main memory (which is usually the case), if not then PTIE approach has to be used.

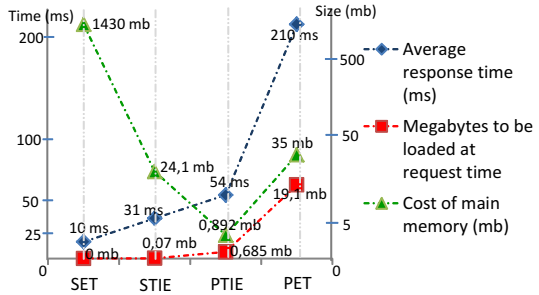
3/ Finally, we could say that the more scalable approach is PTIE, since it can be adopted even if the available main memory has very small size. Furthermore, the experiments showed that PTIE is very efficient (with retrieval time lower than 60 ms) and can be used even with very large query log files. Fig. 9a analyzes the retrieval time of PTIE's main tasks. However, more information has to be loaded to main memory at request time in comparison to SET and STIE. This limits the throughput that is feasible to achieve. This problem can be alleviated by adopting a caching scheme, an issue for future research.

Fig. 9b summarizes the results and illustrates the trade-off between (a) response time, (b) amount of bytes to be loaded in main memory for serving one

⁷ For example over a synthetic query log with 230,000 distinct queries STIE gave average resp. time 80ms while PTIE slightly higher: 80-90ms.



(a) Retrieval time analysis of PTIE.



(b) Summarized results for all approaches.

Fig. 9

request, and (c) cost (amount) of main memory that should be available. The values concern a query log of 40,000 queries (20,184 unique queries). For the PET approach we consider the best case of 50 entries per subtrie.

We observe that SET does not load anything at request time, while STIE loads only the bytes that has to read from the external file (about 70 KB). For this reason, these two approaches achieve very fast response time. PTIE needs about 54 ms for loading the appropriate subtrie (of average size about 622 KB) and reading the results from the external file, while PET spends about 210 ms to load the appropriate subtrie (of average size about 19,1 MB) that contains all the needed information. On the other side, SET requires vast amount of main memory contrary to the other three approaches that require much less.

5.2 Construction and Update

Construction. For the construction of the trie, the main tasks are (a) the analysis of the query log file, (b) the execution of each distinct query in order to get the first page and the cluster label tree of the results and (c) the creation of the trie’s file. Of course the time required by these tasks depend on the particular query evaluation or clustering method that it is employed. Table 2 reports the average times of these tasks for various sizes of the query log.

Table 2. Trie’s Construction Time

Number of log’s queries	Query log file analysis time (ms)	Results and clusters retrieval time (ms)	Trie creation time (ms)	Total time (sec)
1,000	4	592,515	1,259	594
10,000	9	5,415,150	10,156	5,425
20,000	12	10,802,970	19,950	10,823
40,000	16	21,105,780	34,760	21,141

The results are almost the same for all index approaches and for this reason we present only the results of the SET approach (for the STIE and PTIE approach,

the creation of the external files has very small time impact and for the PET and PTIE approach, the time for creating all the subtrees is almost equal to the time SET requires to create a single big trie).

We observe that the task requiring the most time is the retrieval of the results and their clustering. For example, for the query log of 1,000 queries (578 distinct queries), we can see that the retrieval of the results and their clustering takes about 592 seconds, i.e. around 1 second per query.

Update. Note that the index should be updated periodically (based on the contents of the constantly changing query log and the new fresh results of the underlying WSE). One policy is to update the index daily. Since the construction takes some hours (as we saw earlier) it is worth providing an *incremental* method. An incremental approach is to create the trie of the new query log file and then merge the old “big” trie with the new one which is much smaller. If an entry of the new trie does not exist in the initial trie then we just add the new query with all its data to the initial trie. If however an entry of the new trie exists in the initial then we have to update its results, its clusters and its popularity (which is used by the autocompletion algorithm). As we have seen earlier, the time for creating a trie depends on the size of the query log file (about 1 second for each query in our setting). For example, if the new query log has 250 distinct queries then its trie creation time is about 4 minutes.

For testing the time required for merging two tries, we run an experiment with an initial trie of about 11,000 distinct queries (767 MB) and a new trie of 250 distinct queries (14 MB). The execution time was about 2 minutes. We can see that the incremental approach is much more efficient (6 minutes versus about 3 hours).

6 Conclusion

In this paper we introduced a generalized form of search-as-you-type functionality which apart from suggesting query completions and showing fast the first page of results, it shows various other kinds of supplementary information for giving the users an overview of the information space. The effectiveness of this functionality depends on the quality of the pre-computed information (e.g the quality of the ranking method for the query suggestions, the quality of the hits in the first page of results, the quality of the labels returned by the clustering). In any case, the provision of such services at real time significantly increases the amount of information that has to be stored and fetched at run time. For this reason we focused on the problem of efficiency, and we comparatively evaluated various index structures which partition the pre-computed information in various ways. This comparison reveals the main performance trade-off and allows deciding which index structure to use according to the available main memory and desired performance. Contrary to past works, the proposed structure (PTIE) partitions the index and can be adapted to a system’s main memory capacity. This means that one can exploit large amounts of pre-computed information

(even images produced by visualization algorithms). We should also note that the performance is independent of the size of the collection; it is affected only by the size of the query log (in particular by the number of distinct queries), which often has a limited size as we discussed earlier. Furthermore, and since the index should be updated periodically based on the contents of the constantly changing query log and the new fresh results of the underlying WSE, we described an incremental approach for updating the index. Finally we should clarify that the proposed implementation method can be used with any ranking, clustering or autocompletion method. Issues which are worth further research include: (a) *caching* mechanisms for further reducing the information that has to be loaded at user request time and thus increasing the throughput that can be served, and (b) methods for *automatically* selecting the more beneficial index according to details of the particular setting (e.g. size of cluster label tree, size of first page of results, etc), the available main memory, and desired performance.

Acknowledgements. This work was partially supported by the EU project SCIDIP-ES (FP7-283401).

References

1. Bast, H., Chitea, A., Suchanek, F., Weber, I.: Ester: efficient search on text, entities, and relations. In: SIGIR 2007, pp. 671–678. ACM, New York (2007)
2. Bast, H., Weber, I.: Type less, find more: fast autocompletion search with a succinct index. In: SIGIR 2006, pp. 364–371. ACM, New York (2006)
3. Basu Roy, S., Wang, H., Das, G., Nambiar, U., Mohania, M.: Minimum-effort driven dynamic faceted search in structured databases. In: CIKM 2008. ACM, New York (2008)
4. Dakka, W., Ipeirotis, P., Wood, K.: Automatic construction of multifaceted browsing interfaces. In: CIKM 2005, pp. 768–775. ACM, New York (2005)
5. Ji, S., Li, G., Li, C., Feng, J.: Efficient interactive fuzzy keyword search. In: WWW 2009, pp. 371–380. ACM, New York (2009)
6. Kashyap, A., Hristidis, V., Petropoulos, M.: Facetor: cost-driven exploration of faceted query results. In: CIKM 2010, pp. 719–728. ACM, New York (2010)
7. Kastrinakis, D., Tzitzikas, Y.: Advancing search query autocompletion services with more and better suggestions. In: Benatallah, B., Casati, F., Kappel, G., Rossi, G. (eds.) ICWE 2010. LNCS, vol. 6189, pp. 35–49. Springer, Heidelberg (2010)
8. Kopidaki, S., Papadakos, P., Tzitzikas, Y.: STC+ and NM-STC: Two novel online results clustering methods for web searching. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) WISE 2009. LNCS, vol. 5802, pp. 523–537. Springer, Heidelberg (2009)
9. Li, G., Feng, J., Zhou, L.: Interactive search in xml data. In: WWW 2009, pp. 1063–1064. ACM, New York (2009)
10. Li, G., Ji, S., Li, C., Feng, J.: Efficient type-ahead search on relational data: a tastier approach. In: SIGMOD 2009, pp. 695–706. ACM, New York (2009)
11. Li, S., Yu, W., Gu, X., Jiang, H., Fang, C.: Efficient interactive smart keyword search. In: Chen, L., Triantafyllou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 204–215. Springer, Heidelberg (2010)

12. Papadakos, P., Kopidaki, S., Armenatzoglou, N., Tzitzikas, Y.: Exploratory web searching with dynamic taxonomies and results clustering. In: Agosti, M., Borbinha, J., Kapidakis, S., Papatheodorou, C., Tsakonas, G. (eds.) ECDL 2009. LNCS, vol. 5714, pp. 106–118. Springer, Heidelberg (2009)
13. Shang, H., Merrettal, T.: Tries for approximate string matching. *IEEE Knowledge and Data Engineering*, 540–547 (1996)
14. Silverstein, C., Marais, H., Henzinger, M., Moricz, M.: Analysis of a very large web search engine query log. In: *SIGIR 1999*, pp. 6–12. ACM, New York (1999)
15. Wu, H., Li, G., Li, C., Zhou, L.: Seaform: search-as-you-type in forms. In: *VLDB 2010*, pp. 1565–1568 (2010)
16. Xie, Y., O'Hallaron, D.: Locality in search engine queries and its implications for caching. In: *IEEE INFOCOM 2002*, pp. 1238–1247. IEEE, Los Alamitos (2002)

Unsupervised User-Generated Content Extraction by Dependency Relationships

Jingwei Zhang, Yuming Lin, Xueqing Gong, Weining Qian, and Aoying Zhou

Institute of Massive Computing, Software Engineering Institute,
East China Normal University, Shanghai, China
{gtzhjw, ymlinbh}@gmail.com, {xqgong, wnqian, ayzhou}@sei.ecnu.edu.cn

Abstract. User-generated contents are very valuable for event detection, opinion mining and so on, but the extraction of those data is difficult because users are given strong power to present their contents in Web 2.0 pages. Compared to machine-generated contents, user-generated contents are very personalized, which often take on complex styles, combine various information and embed much noise. Users' deep participation makes data acquisition environment a great change and breaks the hidden assumption of traditional extraction methods, which is that Web pages should be relatively regular. The traditional extraction methods can not adapt complex user-generated contents well. In this paper, we consider user-generated contents as unstable contents and advise an unsupervised approach to extract high-quality user-generated contents without noise. Those stable information in machine-generated contents, which are often omitted by traditional extraction methods, are firstly picked up by a two-stage filtering operation, page-level filtering and template-level filtering. Path accompanying distance is then defined to compute the dependency relationships between unstable information and stable information, which guide us to locate user-generated contents. Our approach gives a full consideration on structures, contents and the dependency information between stable and unstable contents to assure the extraction accuracy of user data. The whole process does not need any artificial participation. The experimental results show its good performance and robustness.

1 Introduction

For Web data analyzers, the extraction of user-generated contents is a precondition of their work, the quality of extraction results is a basic guarantee for their research. As an important source of user-generated contents, Web 2.0 permit users to express fully their personalization in Web pages, which has become a mixed data environment by merging user-generated contents with machine-generated contents. Compared to those regular machine-generated contents, such as template-based data, Web 2.0 pages are containing more and more "malformed" data and noise generated by different users, for example, microblogs, Web forums, Social Networks, etc. can accept complex user-generated contents

and give every user powerful support to present their personalized contents. Data acquisition environment are becoming more and more opaque because of users' deep participation.

Unlike machine-generated contents, user-generated contents often take on complex structures and uncertainty, which are very unfriendly for Web data extraction. In Web 2.0, user-generated contents can be presented by any styles, merged with some extra contents, such as some links or signatures, added some images to show users' moods, even inserted some advertisements and so on. Much noise co-occurs with valuable user contents. Fig.1 gives out some situations of user-generated contents, (a),(b) and (c) are from some classical Web forums, (d) and (e) are from two popular microblogs. The contents in red boxes is user-generated contents, but only those restricted by blue boxes is required, and the contents in green boxes are machine-generated contents. The required user contents are surrounded by various noise, such as images, citations, and so on. Especially, those required user contents themselves also have different characteristics, for example, (b) uses different colors to highlight contents, the relative position of required data and citations are different in (a) and (d), (c) presents nothing for user contents except an image, the contents in (d) and (e) are long text but (a) is only a single one and (b) is a combination of multiple contents. More "exciting" content schemas are being and have been made by wisdom users. In fact, users have made a big obstacle for existing data extraction approaches when they are creating much wealth for Web.



Fig. 1. Illustrations of different user-generated contents

The primary task of traditional Web data extraction is to detach structured data from HTML presentation data, they often have a hidden assumption, which is that the Web pages should be relatively regular and can be formalized by logic-like languages or regular expressions. Users' deep participation introduces too

much irregularities and uncertainty, which destroys the above assumption. The existing unsupervised methods often depend on the repeated structural characteristics, they can not work well on user-generated contents since users do not give a stable structure for their contents, which also decides structural information are not enough to extract user-generated contents. The supervised methods seem to adapt user-generated contents well through labeling large amount of examples, but the labeling task itself is difficult even infeasible to cover all situations for large-scale users, especially even the same user is still uncertain in content presentation, which decides that supervised methods have poor scalability for the extraction of user-generated contents.

User-generated contents with a wealth of features cause complex structures in Web pages, the existing extraction methods need to be reexamined. In this paper, we advise an unsupervised method to extract user-generated contents from Web 2.0 pages. Two-stage filtering, page-level filtering and template-level filtering, combined with tag distribution and fan-out-fan-in ratio computing, are used to locate the template areas and find stable information, then path accompanying distance and text density are applied to extract user-generated contents. The rest of this paper is organized as follows. Section 2 briefly reviews the related work in Web data extraction. In section 3, some related concepts and our goals are presented. Two-stage filtering is introduced to detect template areas in section 4. The extraction of user-generated contents is detailed in section 5. The experimental evaluations are reported in section 6. In section 7, we draw conclusions.

2 Related Work

For traditional Web data extraction, there are three primary strategies to locate interesting data. The simplest is to write some rules manually based on some observations for extraction [9], which is very tedious to adapt different templates. Later, some supervised machine learning methods are introduced, most of which still make use of the structural information to induce template wrappers or logic rules [2] [12] [13] [7] [8] [14] [6]. Except requiring rich labeled pages, those supervised methods also need to define some new logic languages to describe induced rules, such as Elog [2], HEL [9], RAQuery [7] and so on, which decide the rules' expressive ability. Though [11] does not label any pages, it incorporates a lot of page-level and site-level information to model Markov logic networks for extraction.

In order to avoid the artificial repeated work, some unsupervised methods are provided by [1], [4], which are only suitable for those pages with high template utilization. [10] doesn't depend on structural information, which considers Web pages as pure text and uses tag ratio to extract contents, which does not model Web pages into DOM trees and make use of template information directly. More work in the fields of Web data extraction have been covered by two surveys [3], [5].

Both above supervised methods and unsupervised methods conform to the assumption that Web pages are relative regular, unsupervised methods impose another strong requirement, which is that the same template should be used multiple times in a Web page. But today, Web data environment changes greatly

because of users' deep participation. Pure structural information are not enough for user-generated content extraction. Those stable information are completely omitted, which are redundant for extraction results, but can provide helpful information during extraction process. The complexity of user-generated contents are not considered enough.

3 Problem Setting

Web data extraction has two primary tasks, one is to discover the template and the other is to remove noise in template areas. Here, we focus on the extraction task of Web 2.0 pages, such as Web forums, microblogs and so on, the extraction target are user-generated contents and some user-related information, for example, those contents are covered by blue boxes and green boxes in Fig 1.

A Web page is a mixture of machine-generated contents and user-generated contents. A template area is a basic information unit in Web pages, which is a collection of user-related information, user-generated contents and some noise. Considering the characteristics of both the contents and the structures, a template area can be divided into three parts, stable region, semi-stable region and unstable region, which are defined as following,

Stable region the contents located by the same path in different template areas are same.

Semi-stable region the contents located by the same path in different template areas are very different.

Unstable region the contents with the same nature are located by similar but different paths, every of those similar paths often exists in several but not all templates.

Here, stabilization implies two meanings, the first is whether the contents are same, the second is whether the structures, usually paths, are consistent. The contents in stable regions are often automatically generated by machine, such as some guiding links or advertisements. Those in semi-stable regions are often generated by database queries or Javascript, such as user name, registration date and other user-related information. Those in unstable regions are entirely generated by users, which are very irregular. Obviously, the unstable regions destroy the regularity of the templates, which makes the formalization of the templates difficult. The contents in unstable regions are a core of Web 2.0 since they are the outcome of users' active involvement. It is a key and valuable task to extract those data in Web 2.0 pages.

Consider the data sources and the extraction targets, there are four combinations in template areas, which are listed in Table 1. Here, required data are just those data that we want to extract, redundant data are noise.

Problem Definition. The objective of Web 2.0 data extraction is to identify all template areas in Web pages, remove stable regions, capture the contents except noise in semi-stable and unstable regions from template areas.

Table 1. Relationship between stability and template data types

	machine-generated	user-generated
redundant data	stable	unstable(noise)
required data	semi-stable	unstable

In order to realize the extraction process, we model Web pages into DOM trees. Here, we cover some related concepts that will be used in the following sections.

path, given any leaf node in a DOM tree, there is an ordered sequence of node tag from root to this leaf node, if every tag in this sequence is followed by a corresponding index, the sequence is called a path of this tree. $IN(page1, path1)$ returns true if $path1$ is a path of $page1$. $LOC(page1, path1)$ returns the content located by $path1$ in $page1$.

path consistency, given two paths $path1$ and $path2$, if all tags and indexes in every step are same, we say that $path1$ and $path2$ are consistent. If only all tags in every step are same, we say that $path1$ and $path2$ have consistent features. $PC(path1, path2)$ returns true if $path1$ and $path2$ satisfied path consistency, else returns false. $PFC(path1, path2)$ corresponds to path feature consistency.

content type consistency, given two texts $c1$ and $c2$, if they describe the same type of contents, we say that $c1$ and $c2$ are type consistency. $CTC(c1, c2)$ returns true if $c1$ and $c2$ satisfy type consistency, else returns false. $CC(c1, c2)$ returns true if $c1$ and $c2$ are completely same.

4 Template Area Detection

In this section, we use two key filtering operations, page-level filtering and template-level filtering, to obtain the template areas in Web pages. The comparison between different Web pages can help us to remove most noise outside template areas and get most paths existing in template areas. Based on the results, the tag distributions in different levels are computed, singular distribution and uniform distribution are used to decided the template splitting points under the help of every node’s fan-out-fan-in ratio. The comparison between different template areas can detach all stable and semi-stable contents from unstable contents.

4.1 Identifying Template Splitting Point

In a Web page, we restrict the scope of template areas through finding the splitting points of template areas. Since Web pages are modeled as DOM trees, template splitting points are some non-leaf nodes in DOM tree, which are in the same level of DOM tree, such as the pink nodes in Fig 2. A template area are just the part from the template splitting point to all its descendant nodes in the DOM tree, such as the gray parts in Fig 2.

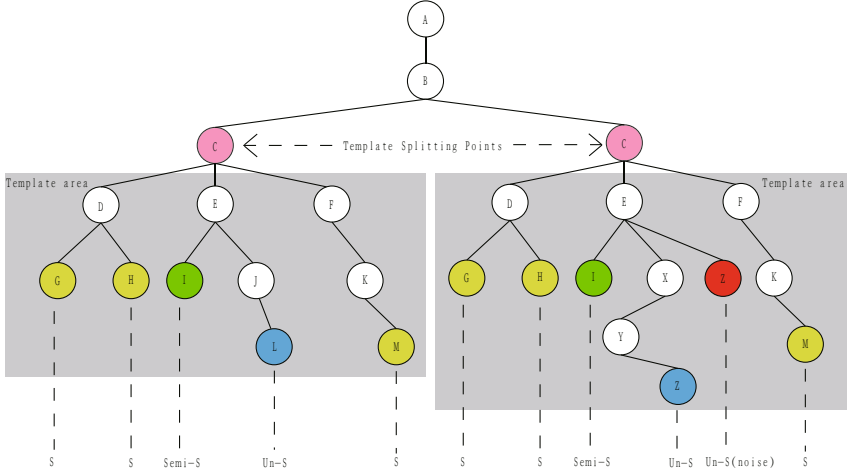


Fig. 2. An example for two simplified template areas (S: stable path; Semi-S: semi-stable path; Un-S: unstable path)

In order to locate the template splitting points accurately, we firstly carry out comparison between Web pages and remove most noise outside the templates. For similar but different Web pages, we have the following conclusions,

- If the contents located by the same path are also identical in different pages, the contents are often invalid.
- If the contents located by the same path are different in different pages, the contents will be a part of some template area with a high probability.

According to the above properties, we can compare two or more similar pages and retain those paths locating distinct contents in different pages, then use those paths to obtain the locations of splitting points. The intermediate results will be used for computing tag distribution. Because most of those paths are template-related, the front tags often show singular distribution with their indexes, the first approximately uniform distribution will be considered as the template points. Obviously, higher the templates frequency in Web pages are, more obvious the distribution phenomenon will be.

If the tag distribution can not give a clear signal for template splitting points, the fan-out-fan-in ratio will be computed to decide the concrete splitting points. Given a node of a DOM tree $node$, $TagName(node)$ deontes the tag name of $node$, $NodeLayer(node)$ denotes the layer number of $node$, $NodeSet(tagname, i)$ denotes a set of nodes who are in i th layer and have same tag name with $tagname$, $ChildNode(node)$ denotes the child node set of $node$. The fan-out-fan-in ratio of $node$ can be defined as,

$$FOFI(node) = \frac{|\bigcup_{n \in NodeSet(TagName(node), NodeLayer(node))} ChildNode(n) |}{|NodeSet(TagName(node), NodeLayer(node)) |}$$

The fan-out-fan-in ratio describes a relationship of node's output and input, template splitting points often have high fan-out-fan-in ratio since a template has multiple paths with a common path head. The tag distribution and fan-out-fan-in are controlled by two thresholds, which are given empirically. The detailed algorithm are presented in Algorithm 1.

Algorithm 1: Identify Template Splitting Points

Input: two Web pages, $p1$ and $p2$

Output: the tag and level of splitting points

```

1: get all paths from  $p1$  and inert them into  $pSet$ 
2: WHILE  $pSet$  is not null
3:   get one path from  $pSet$ ,  $path1$ 
4:   IF(IN( $p2$ ,  $path1$ ))
5:     IF(CTC(LOC( $p1$ , $path1$ ),LOC( $p2$ , $path1$ ))
      && !CC(LOC( $p1$ , $path1$ ), LOC( $p2$ , $path1$ )))
6:       add  $path1$  into  $pathSet$ 
7:     END IF
8:   ELSE add  $path1$  into  $pathSet$ 
9:   END IF
10: remove  $path1$  from  $pSet$ 
11: END WHILE
12: compute tag and index distribution in  $pathSet$ 
13: find the first place who satisfies the given thresholds for approximately
    uniform distribution and the fan-out-fan-in ratio
14: return the tag and level of the located nodes

```

4.2 Template Level Filtering

In this stage, we will focus on the content classification in discovered template areas, stable contents and semi-stable contents will be covered accurately.

We use both path and content characteristics to refine the above initial template areas. The front rules are still suitable for comparison between template areas to construct the basic parts of template areas. We firstly scan all template areas and obtain all sub-paths, namely the part of paths in template areas, then three rules are applied to these paths,

- If a path occurs in all template areas and the contents located by the path are also same, the path is a stable path.
- If a path occurs in all template areas and the contents located by the path are very different, the path is a semi-stable path.
- If a path occurs in some template areas, but does not occur in other template areas, the path is unstable.

When carried out template filtering, we will reserve those paths meeting the first two rules and remove those paths meeting the third rules. Obviously, the stable part and semi-stable part in template areas are identified after template comparison, but it also removes all user-generated contents when filtered all noise.

5 User-Generated Content Extraction and Noise Filtering

Both stable regions and semi-stable regions can be solved effectively by page filtering and template filtering. In this section, we try to extract those contents hiding in unstable regions, which is a difficult work because users often merge much noise with their contents, just like those parts limited in red boxes and blue boxes in Fig 1.

For users' strong expressive ability in unstable regions, there are two key problems for solution. One is path redundancy, the other is path similarity. When users are writing some comments or replies in Web pages, they are often used to edit the text format to let their contents more eye-catching, and they are also possible to add some extra contents, such as images, links and their signatures, to express their emotions. The first factor causes path similarity, which means that these paths do not completely satisfy path consistency and path feature consistency, but the contents located by them are the same type, these paths often own a common header but different tails for users' personalized editing. The second causes path redundancy, the contents located by these paths are also redundant. In unstable regions, we must remove redundant paths and group similar paths to serve for extraction, but it is a difficult work to distinguish between redundant paths and similar paths in unstable regions since redundant paths are often also similar. It is invalid to decide whether a path should be reserved or removed only according to path characteristics themselves.

In order to detach valuable contents from noise effectively, we introduce path accompanying distance, which makes use of the dependency relationships between stable information discovered in the filtering stage and unstable information to determine whether a path is noise path. Path accompanying distance is a kind of computation of interdependent information, which is defined as,

Path Accompanying Distance(PAD). Given two paths $path1$ and $path2$, their PAD means the simultaneous frequency of $path1$ and $path2$ in a group of template areas.

$$PAD(path1, path2) = \frac{|\{T_i | \forall i, IN(T_i, path1) \wedge IN(T_i, path2)\}|}{|\{T_i | \forall i, IN(T_i, path1) \text{ is true}\}|}$$

Here, T_i denotes a template area. If a group of similar paths have a long path accompanying distance with a stable path, those paths will be the paths locating valuable user contents with high probability because users often contribute more frequent valuable contents than noise. Under path accompanying distance, path

similarity has two meanings, one is the degree of path feature consistency, the other is their individual accompanying distance related to a fixed objective, which is often the stable information in template areas. Combined with path feature consistency, path similarity can be formalized as,

$$PS(path1, path2) = w_1 * PFC(path1, path2) + w_2 * PAD(path1, path2)$$

Here, w_1 and w_2 are weights for path feature consistency and accompanying distance, which are set empirically. Usually, redundant paths have a degree of path feature consistency, but a worse accompanying distance, similar paths have a big path feature consistency and a long accompanying distance, the two factors can be used to distinguish similar paths and redundant paths.

A strict path distance has been proposed in [12], which can be used to compute path feature consistency. The strict path feature consistency judges two paths either consistent or not, which can not tell us the probability of their similarity. Most of user-generated contents have similar paths, but they do not conform to strict path feature consistency. We relax the strict path feature consistency by computing the similarity probability of two paths based on Jaccard similarity, which can be denoted as,

$$PFC(path1, path2) = \frac{|TL(path1) \cap TL(path2)|}{|TL(path1) \cup TL(path2)|}$$

Here, TL is a function that is used to divide a path into a group of pairs, every pair is composed of a tag name and the tag's layer number in the given path. For example, `"/A[1]/B[1]/C[1]/E[1]/J[1]/L[1]"` is a path in Fig. 2, $TL("/A/B/C/E/J/L") = \{< A, 1 >, < B, 2 >, < C, 3 >, < E, 4 >, < J, 5 >, < L, 6 >\}$.

Base on the results of the two-stage filtering, we have the following information, a group of template areas, a set of unstable regions, and stable information. We traversal all unstable regions and collect all paths, every one of which is used to compute their accompanying distance with stable information in all template areas. Because every unstable path is often low frequency, a backtracking operation is also carried out during the computation of accompanying distance. When an unstable path has a low accompany distance, we will truncate the path from tail to head step by step and compute the accompanying distance for those sub-paths. Those paths with a low value on accompanying distance will be washed out. Text density is also used to help us identify extracted data since user-generated contents are often long text, which are mostly applied in the stop condition of backtracking operations. When removed all noise paths from unstable regions, the common part of these unstable regions is abstracted, which are then used to extract user-generated contents. The detailed algorithm is presented in Algorithm 2.

Algorithm 2: Distinguish valid paths and noise paths in unstable regions**Input:** a set of unstable regions, UR , and stable information, SI **Output:** a path set, $pathSet$

-
- 1: traversal all unstable regions and get the path set, $pSet$
 - 2: carry out backtracking operation for every path in $pSet$, add all sub-paths into $pSet$, text density decides where backtracking should stop.
 - 3: compute the accompanying distance with SI of every path in $pSet$
 - 4: compute the path similarity of all paths in $pSet$ and filter paths with low similarity
 - 5: group paths in $pSet$ based on their similarity into $pathGroup$
 - 6: FOR every group in $pathGroup$
 - 7: get the maximum common part of all paths, cp , according to their accompanying distance
 - 8: $pathSet = pathSet \cup cp$
 - 9: END FOR
 - 10: RETURN $pathSet$
-

6 Experiments

In this section, we report the experiment results by applying our method to extract semi-stable contents and unstable contents in Web 2.0 sites. The data sources are from popular Chinese Web forums, liba¹, QQ², 163³, tianya⁴ and so on.

The first group of experiments are carried out to evaluate the overall extraction performance on 22 groups of Web pages, which are from the different sections of those forums and cover different user data cases. The number of template areas are between 10 and 40 in every Web page, the number of extracted items is from tens to hundreds. We use recall, precision and F_β measure as the evaluation metrics, β is set to 2 since high recall can assure the integrity of user contents. The recall, precision and F2-score on 22 groups of Web pages are shown in Fig. 3, which correspond to the left y-axis. The number of the expected extraction items are also listed on Fig. 3, which corresponds to the right y-axis. Though the expected extraction items vary, our method has a good and stable performance on recall, precision and F2-score. Most of the expected items can be extracted accurately, especially the stable precision shows that our method can effectively avoid noise during extraction.

Another group of experiments are carried out to compare the effects of unstable items extraction. We pick up 5 groups of Web pages, the proportion of unstable items is from 16% to 25%. We compare the extraction performance on semi-stable contents and unstable contents. The methods without considering accompanying distance is firstly applied on this dataset, and then accompanying distance

¹ <http://bbs.libaclub.com>² <http://bbs.qq.com>³ <http://bbs.163.com>⁴ <http://www.tianya.cn/bbs/>

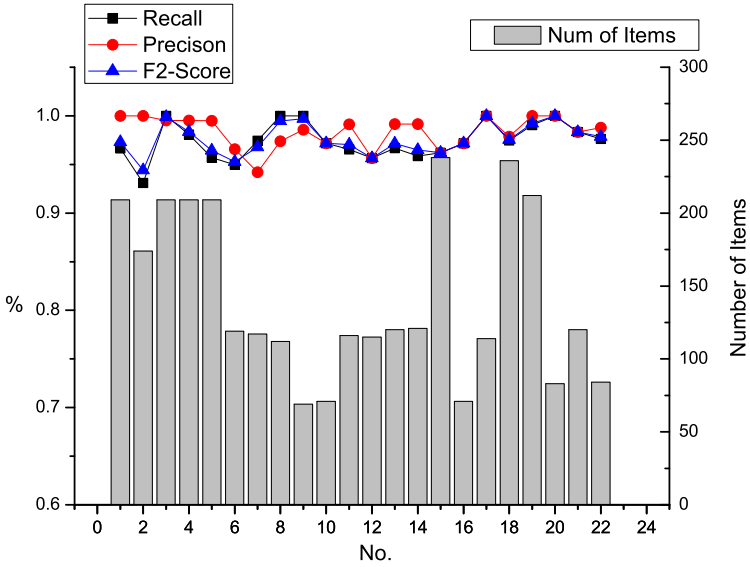


Fig. 3. Extraction Performance(The right ordinate corresponds to the number of expected extraction items)

are introduced to extract semi-stable contents and unstable contents separately. The experimental results are presented in Fig 4(a), which are composed of two parts corresponding to the left and right ordinates. The left ordinate represents recall shown by lines, the blue line is the recall of extraction methods embedded

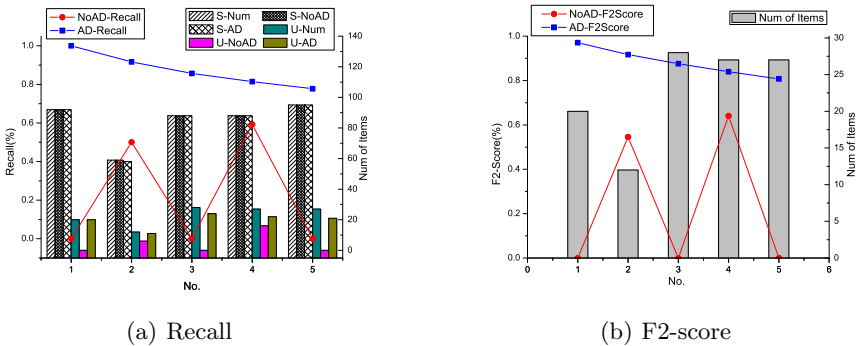


Fig. 4. Extraction performance comparison on semi-stable and unstable contents(NoAD: without using accompanying distance; AD: using accompanying distance. S-Num: the number of expected semi-stable items; S-NoAD: the number of extracted semi-stable items by NoAD methods; S-AD: the number of extracted semi-stable items by AD; U-Num: the number of expected unstable items; U-NoAD: the number of extracted unstable items by NoAD methods; U-AD: the number of extracted unstable items by AD).

accompanying distance, the red line corresponds to the recall of extraction without applying accompanying distance. The right ordinate represents the number of extraction items shown in column bars. The six column bars in turn indicates the number of expected semi-stable items, extracted semi-stable items without using accompanying distance, extracted semi-stable items with accompanying distance, expected unstable items, extracted unstable items without accompanying distance, extracted unstable items with accompanying distance. The accompanying distance can effectively solve the unstable content extraction without any interference on the extraction of semi-stable contents. Fig. 4(b) shows the F2-score of the two methods on unstable contents. The accompanying distance can give a strong power to assure a good performance on those user-generated contents even where filtering operations can not work.

7 Conclusion

In this paper, we have presented an unsupervised method to extract semi-stable contents and unstable contents from Web 2.0 pages. Most existing methods have a hidden requirement that Web pages should be relatively regular, which is broken by unlimited user-generated contents. Those existing methods can not work well on user-generated contents with various styles and much noise.

Our method introduces path accompanying distance to extract user-generated contents, which makes full use of the dependency relationships between stable contents and unstable contents to decide whether a content should be extracted. The stable contents are often omitted by traditional extraction methods because they are not the extraction targets, but the dependency relationships between the stable contents and unstable contents give a strong signal to distinguish valid user contents from noise and improve the quality of the extracted results. The experimental results on 22 groups of Web pages from different Web 2.0 sites show that our method has a big improvement and robustness on extraction of user-generated contents.

Acknowledgments. The authors sincerely thanks Dr. Peng Cai for helpful discussion and advice to complete this paper. This work is partially supported by National Science Foundation of China under grant numbers 60833003, 61070051, and 60803022, National Basic Research (973 program) under grant number 2010CB731402, and National Major Projects on Science and Technology under grant number 2010ZX01042-002-001-01.

References

1. Arasu, A., Garcia-Molina, H.: Extracting structured data from web pages. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, SIGMOD 2003, pp. 337–348. ACM, New York (2003)
2. Baumgartner, R., Flesca, S., Gottlob, G.: Visual web information extraction with lixt0. In: Proceedings of the 27th International Conference on Very Large Data Bases, VLDB 2001, pp. 119–128. Morgan Kaufmann Publishers Inc., San Francisco (2001)

3. Chang, C.-H., Kayed, M., Girgis, M.R., Shaalan, K.F.: A survey of web information extraction systems. *IEEE Trans. on Knowl. and Data Eng.* 18, 1411–1428 (2006)
4. Crescenzi, V., Mecca, G., Merialdo, P.: Roadrunner: Towards automatic data extraction from large web sites. In: *Proceedings of the 27th International Conference on Very Large Data Bases, VLDB 2001*, pp. 109–118. Morgan Kaufmann Publishers Inc., San Francisco (2001)
5. Flesca, S., Manco, G., Masciari, E., Rende, E., Tagarelli, A.: Web wrapper induction: a brief survey. *AI Commun.* 17, 57–61 (2004)
6. Gulhane, P., Madaan, A., Mehta, R.R., Ramamirtham, J., Rastogi, R., Satpal, S., Sengamedu, S.H., Tengli, A., Tiwari, C.: Web-scale information extraction with vertex. In: Abiteboul, S., Böhm, K., Koch, C., Tan, K.-L. (eds.) *ICDE*, pp. 1209–1220. IEEE Computer Society, Los Alamitos (2011)
7. Han, W.-S., Kwak, W., Yu, H.: On supporting effective web extraction. In: Li, F., Moro, M.M., Ghandeharizadeh, S., Haritsa, J.R., Weikum, G., Carey, M.J., Casati, F., Chang, E.Y., Manolescu, I., Mehrotra, S., Dayal, U., Tsotras, V.J. (eds.) *ICDE*, pp. 773–775. IEEE, Los Alamitos (2010)
8. Muslea, I., Minton, S., Knoblock, C.A.: Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems* 4, 93–114 (2001)
9. Sahuguet, A., Azavant, F.: Building light-weight wrappers for legacy web data-sources using w4f. In: *Proceedings of the 25th International Conference on Very Large Data Bases, VLDB 1999*, pp. 738–741. Morgan Kaufmann Publishers Inc., San Francisco (1999)
10. Weninger, T., Hsu, W.H., Han, J.: Cetr: content extraction via tag ratios. In: *Proceedings of the 19th International Conference on World Wide Web, WWW 2010*, pp. 971–980. ACM, New York (2010)
11. Yang, J.-M., Cai, R., Wang, Y., Zhu, J., Zhang, L., Ma, W.-Y.: Incorporating site-level knowledge to extract structured data from web forums. In: *Proceedings of the 18th International Conference on World Wide Web, WWW 2009*, pp. 181–190. ACM, New York (2009)
12. Zhang, J., Zhang, C., Qian, W., Zhou, A.: Automatic extraction rules generation based on xpath pattern learning. In: *The 1st International Symposium on Web Intelligent Systems and Services (WISS 2010)*. Springer, Heidelberg (2010)
13. Zheng, S., Song, R., Wen, J.-R., Giles, C.L.: Efficient record-level wrapper induction. In: *Proceeding of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009*, pp. 47–56. ACM, New York (2009)
14. Zheng, S., Song, R., Wen, J.-R., Wu, D.: Joint optimization of wrapper generation and template detection. In: Berkhin, P., Caruana, R., Wu, X. (eds.) *KDD*, pp. 894–902. ACM, New York (2007)

Sentiment Analysis with a Multilingual Pipeline

Daniella Bal¹, Malissa Bal¹, Arthur van Bunningen², Alexander Hogenboom¹,
Frederik Hogenboom¹, and Flavius FrasinCAR¹

¹ Erasmus University Rotterdam

PO Box 1738, NL-3000 DR, Rotterdam, The Netherlands

{daniella.bal,malissa.bal}@xs4all.nl,

{hogenboom,fhogenboom,frasinCAR}@ese.eur.nl

² Teezir BV

Kanaalweg 17L-E, NL-3526 KL, Utrecht, The Netherlands

Arthur.van.Bunningen@teezir.com

Abstract. Sentiment analysis refers to retrieving an author’s sentiment from a text. We analyze the differences that occur in sentiment scoring across languages. We present our experiments for the Dutch and English language based on forum, blog, news and social media texts available on the Web, where we focus on the differences in the use of a language and the effect of the grammar of a language on sentiment analysis. We propose a multilingual pipeline for evaluating how an author’s sentiment is conveyed in different languages. We succeed in correctly classifying positive and negative texts with an accuracy of approximately 71% for English and 79% for Dutch. The evaluation of the results shows however that usage of common expressions, emoticons, slang language, irony, sarcasm, and cynicism, acronyms and different ways of negation in English prevent the underlying sentiment scores from being directly comparable.

1 Introduction

The Web, being available to a continually expanding audience all over the world, dramatically increases the availability of data in the form of reviews, previews, blogs, forums, social media, etc. Because of increased transparency and convenience of gathering data on the Web, global competition between companies is also highly increased. In this expanding competitive landscape it becomes more important for a company to understand its market and target audience. With the vast amounts of information available on the Web it is impossible for companies to read everything that is written about their position between competitors and competing products manually. Moreover, a company’s staff would have to master many languages to read all the relevant texts.

Sentiment analysis comes to answer this need, as it can be used as a marketing tool to find out what people are generally saying about their products or products of their competitors. Sentiment analysis – also referred to as opinion mining – is the process of retrieving the sentiment or opinion of the author from a text [13]. Yet, for international companies it is not only important to know what people are saying about them in one language; they need to know as well

what is said about them across the world in many languages. Being able to analyze documents written in multiple languages gives companies the opportunity to assess local opinions which can be of great help when setting out a marketing plan across different geographies. It can help companies understand where clients adore or dislike their products. It can also help them understand why people in different areas around the world think differently about their products and where they need to focus their marketing efforts. However, it is still not possible to accurately compare the results of sentiment analysis across different languages. Existing work on sentiment analysis either does not focus on the comparability of sentiment across languages or assumes sentiment across languages to be comparable. We argue that the assumption of comparability of sentiment analysis across languages is too simplistic.

Therefore, we propose to utilize a novel language-independent method for analyzing sentiment in order to uncover structural differences in the way sentiment is conveyed in different languages. These insights may contribute to future endeavors in the field of cross-language sentiment analysis. To this end, our framework identifies sentiment in any considered language in the same way, albeit based on language-specific sentiment lexicons. We focus on documents written in the English and Dutch language. Here, documents are constructs of natural language text in the form of news articles, blog posts, forum posts or reviews, but also comments on a review or messages on a social media platform.

The remainder of this paper is structured as follows. Sections 2 and 3 discuss related work on sentiment analysis and our derived hypotheses respectively. Sections 4 and 5 then discuss our framework for multilingual sentiment analysis and its implementation, respectively. The evaluation of the proposed framework is discussed in Section 6. Finally, we conclude in Section 7.

2 Related Work

Current research on sentiment analysis in different languages [1,4,5,6,8,9,11,14] focuses mainly on how to create new sentiment lexicons [9,14] in a different language or on how to create a new pipeline for a new language [1,5,6,8], both by mainly using machine learning techniques.

Moens et al. [11] analyze the creation of different pipelines used for different languages with minimal human effort to develop them. They stress the choices one has to make when applying a machine learning approach and discuss differences in choices between different languages. Additionally, several hypotheses are tested to optimize the output of the algorithm that computes the sentiment scores. Moens et al. recommend a pipeline consisting of three layers. The three-layer pipeline is meant to create a fast way of computing sentiment scores. The first layer is very fast in its computations but is not highly accurate. When a computation is not accurate (measured by thresholds) the text will be passed on to the next more precise, but also slower, computation layer. The process is repeated on the third layer. If still no accurate score is computed, the score of layer two is kept.

Bautin et al. [4] analyze cross-lingual sentiment by machine translation. They translate all considered texts to the English language and perform sentiment analysis on the translated results. The authors assume that the results are comparable and that the errors made by the machine translation do not significantly influence the results of the sentiment analysis.

Wan [14] focuses on the creation of a lexicon in Chinese, based on an English lexicon using a co-training approach. The idea of the co-training approach is to have two sets of documents – one in English and one in Chinese. These sets can be split in annotated and unannotated documents. The goal is to obtain a set of all unannotated English texts and all annotated Chinese texts. This is achieved by machine translation of all annotated English texts to Chinese and all unannotated Chinese texts to English. These two sets are then used to compute a classifier for Chinese texts and the classifier is tested on all Chinese reviews. To test the validity of the classifier, the reviews are translated to English and tested against the existing English classifier. If classified correctly in both languages, the classifier reflects the positive or negative polarity of the documents.

The research reported on by both Wan [14] and Bautain [4] has been on the area of creating a new sentiment analysis framework from an existing one and comparability of the sentiment of documents written in different languages is assumed. However, we believe this assumption is risky, as the grammar of languages is different to such extents that the best performing self-learning algorithms for classifying positive and negative sentiment texts in different languages are not the same [11]. This would also imply that, with the current research, companies can still not effectively assess local opinions. Differences in opinion score might as well be the result of differences that occur in the sentiment lexicon or the grammar of the language.

In this paper, we focus on creating a pipeline for sentiment analysis that uses multiple languages and takes language specifics into account. We thus aim to make the obtained results comparable across languages. Optimizing the accuracy of sentiment scores for single languages is already subject of widespread research and is out of our scope. We address the question whether it is possible to create a pipeline for sentiment analysis that can handle documents in multiple languages without sacrificing accuracy on sentiment scores. In order to answer this question, we aim to assess whether the sentiment scores of documents from different languages are comparable and whether differences in grammar or usage of language influence the working of the pipeline and the results.

3 Hypotheses

Two hypotheses could be derived from the related work, both of which can help to answer the research question. First, we hypothesize that the difference in pipelines for different languages causes a difference in the final document score. Moens et al. [11] show that the optimal pipeline for different languages looks different. This is caused by the grammar of the language. An example of these differences can be found in the way different languages handle negation.

For example, English negation is most accurately found by extending the effect of a negation word until the first punctuation, while in the Dutch language it is better to use a set window frame of words around the negation word [11]. Since we are comparing the document scores of documents written in different languages we propose a general pipeline, handling both languages, to minimize these differences in results.

Our second hypothesis is that differences in the way people express themselves cause unwanted differences in the sentiment analysis. An important difference between languages is in the way of speech. For example, when an Englishman says: “that is not bad at all”, he means he is very enthusiastic about it, yet for the sentiment analysis it is just the negation of bad. This may give documents lower overall sentiment scores while the text is either very positive or very negative. Additionally, a Dutch person would prefer to say “Dit is een goede Universiteit” (This is a good University) rather than “Ik houd van deze Universiteit” (I love this University). For English people, the opposite holds true. We want to quantify how much effect these differences have on the final document sentiment score and how we can compensate for these differences [11].

4 SAMP Framework Design

To support our research goals, we have developed a framework for Sentiment Analysis with a Multilingual Pipeline (SAMP), which is composed of three parts. Each of the three parts and their respective goal is discussed in the sections below. We start with a quick overview of the framework and give a short introduction to the different components discussed.

4.1 Overview

As shown in Fig. 1 the framework consists of three main components:

1. **Language Selection.** Determine the language of the text. Select either Dutch or English, based on the likelihood of the text to be written in the respective languages.
2. **Text Preparation.** Text preparation consists of two parts. The first part is text cleaning. This involves replacing all capital letters with small letters, remove diacritics, etc. until a clean text remains. The second part involves word typing, i.e. finding the Part-Of-Speech (POS) of all words and determining whether each word exists in the lexicon to find the word category (i.e., opinion or modifier).
3. **Sentiment Score Computation.** The sentiment score computation is divided in three separate steps. The first step is document sentiment scoring: calculate the sentiment score of a document. The second step is document relevance scoring, i.e., determining the relevance of a document with respect to an arbitrary topic. Step three is topic sentiment score computation, which involves computing a weighted average of the relevance and sentiment score of a batch of documents with respect to the topic of this collection.

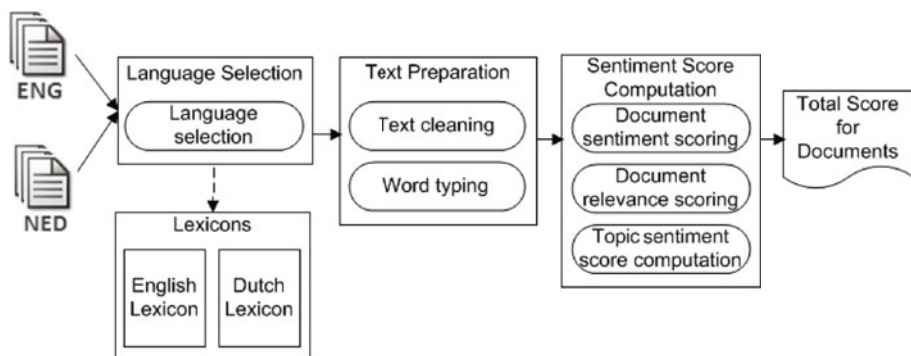


Fig. 1. The SAMP framework overview

4.2 Language Selection

Algorithm 1 shows the algorithm used for the language selection. The input is a document or a list of documents. For each document, the language is determined as Dutch or English. The language is found with the help of letter n -grams. For example, if one would make letter 3-grams for the sentence “The cat was happy.”, the grams are: ([The] [he_] [e_] [ca] [cat] [at_] [t_w] [wa] [was] [as_] [s_h] [ha] [hap] [app] [ppy] [py.]), where “_” is represents an empty space. Some n -grams are language-specific, e.g., [the] for English and [sch] for Dutch.

Nowadays, the usage of English words in the Dutch language – especially in short texts like tweets with a maximum of 140 characters – may confuse the document language detection components. We assume all documents to be either Dutch or English, but never a mix of both. The language with the highest likelihood is selected.

Algorithm 1. Language Selection.

input : Documents from batch D_{batch}
output: Lexicon Lex with the sentiment lexicon associated with the documents’ language

- 1 $nGrams = \text{findNGrams}(D_{batch})$;
- 2 $probDutch = \text{compareTo}(nGrams, Lexicon.NL)$;
- 3 $probEnglish = \text{compareTo}(nGrams, Lexicon.EN)$;
- 4 **if** $probDutch > probEnglish$ **then**
- 5 $Lex = Lexicon.NL$;
- 6 **else**
- 7 $Lex = Lexicon.EN$;
- 8 **end**
- 9 **return** Lex

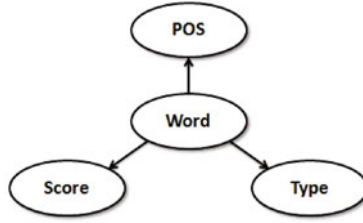


Fig. 2. Database entities

We use two similar sentiment lexicons for English and Dutch. Both lexicons are simple structured databases with the entities in the database shown in Fig. 2. The Dutch lexicon is provided by Teezir. For each entry in the Dutch lexicon, the English lexicon contains a manual translation with the same meaning as the Dutch entry. A manual translation helps us provide two similar lexicons in order to rule out biased results due to differences in lexicons. The lexicon has been translated by three experts until they reached consensus.

4.3 Text Preparation

Before computing the sentiment of a batch of documents, we preprocess the documents by means of Algorithm 2. First of all, stop words like “a”, “an”, “the”, etc., are removed, as they are of no use in the sentiment analysis and they unnecessarily affect the relevance score of a text. Furthermore, the documents are cleaned from diacritics (e.g., “ongeëvenaard” is replaced with “ongeevenaard”) in order to optimize the likelihood of matches with words in the sentiment lexicon. Additionally, words are tagged with their associated POS. In this process, non-textual elements (e.g., tags, images, or videos) are not parsed by a POS tagger. The result is a batch of documents consisting of plain text which can be parsed and labeled.

Algorithm 2. Text preparation.

input : Documents from batch D_{batch} and their sentiment lexicon Lex
output: A batch of preprocessed documents D_{proc}

- 1 $D_{proc} = D_{batch}.CleanText();$
- 2 **foreach** *sentence* **in** D_{proc} **do**
- 3 **foreach** *word* **in** *sentence* **do**
- 4 $POS = sentence.FindPOS(word);$
- 5 **end**
- 6 $D_{proc}.addPOS(POS);$
- 7 **end**
- 8 **return** D_{proc}

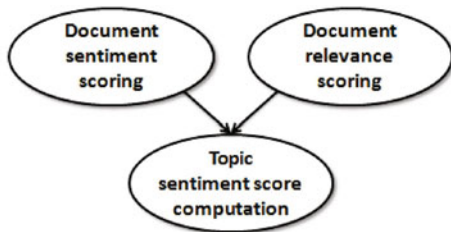


Fig. 3. Total sentiment score computation

All words and their associated POS are subsequently compared to the lexicon of the matching language. Each word’s sentiment score thus retrieved is considered in the sentiment computation, while taking into account the word type. We consider two types of words here – opinion terms and modifier terms. Modifier terms modify opinion terms. For example, in the sentence “The book was not good”, “not” modifies the opinion term “good”. For some words, it depends on their POS whether they are a modifier term or an opinion term. For example, in the sentence “The movie was pretty good”, the adjective “pretty” is a modifier term as it increases the sentiment of the opinion term “good”. Conversely, in the sentence “The blonde girl is very pretty”, the adverb “pretty” is an opinion term as it expresses sentiment associated with the blonde girl.

4.4 Determine Sentiment Score

Fig. 3 shows the steps in creating a total score for a batch of documents. First, the sentiment of a document is determined. The sentiment S_i for document i is a function of all n opinion terms and their modifiers:

$$S_i = \frac{\sum_{o=1}^n \text{modifierterm}_o \times \text{opinionterm}_o}{n}, \quad (1)$$

assuming that opinion terms and modifier terms form couples. When a word does not have an opinion term, the value of the couple is set to 1.

In order to compute the sentiment associated with a batch of documents with respect to a certain topic, the sentiment scores S_i of each individual document i are weighted for their associated document relevance R_i :

$$\text{Topic Sentiment Score} = \frac{\sum_{i=1}^n S_i R_i}{\sum_{i=1}^n R_i}, \quad (2)$$

where the document relevance with respect to the query is computed based on the PL-2 standard [3] combined with document length normalization. Document length normalization means that longer texts have a relatively higher relevance (e.g., a news article is more relevant than a tweet of at most 140 characters).

5 SAMP Implementation

In order to evaluate our framework, we created a program that can parse texts in both Dutch and English and is designed to help the user find differences in the way texts are parsed and labeled. The application is written in *C#* and pre-processes texts with a maximum-entropy based POS tagger, which can process English as well as Dutch texts. Our implementation has three major user interfaces. We distinguish between a Specific Result Screen (SRS), a General Result Screen (GRS), and a Graph Screen (GS).

5.1 Specific Result Screen

The SRS, depicted in Fig. 4, is created to give more insight in the process of classifying a text as positive or negative. Annotated texts can be analyzed in both languages in order to analyze whether the pipeline works properly. Together with the GRS (described below), this screen also provides insight in the differences between languages. Because of the nature of the screen, documents can easily be analyzed manually. This analysis can reveal differences in the use of language, like those discussed in Section 6.

The text is displayed with positive opinion terms in green, negative opinion terms in red and modifier terms in purple ①. Below the text, a list of words is produced ② together with the type of the words (opinion terms or modifier terms) ③ and their score ④. Below the list, the screen shows the computation ⑤ and the document score ⑥. This way, the user can validate whether or not the correct words are flagged as sentiment words, the words have the correct score and the computation of the document score is correct.

5.2 General Result Screen

The GRS (Fig. 5) is designed to provide more insight in the process of computing sentiment scores for batches of documents. It shows the average score of the documents ①, how many documents constitute the score ② and a list of documents with their score, sorted on relevance ③. The user can thus analyze whether the documents are correctly labeled and ranked on relevance. This screen can help to provide answers for the second type of experiments.

5.3 Graph Screen

The GRS gives the possibility to show the GS presented in Fig. 6. The GS visualizes the distribution of document-level sentiment scores in the corpus. For example, if the score for both Dutch and English equals 0.50, it is not required that people in both countries think the same. It may very well be that, in The Netherlands, half of the texts result in a score of 0 and the other half results in a score of 1, while in England all the texts yield a 0.50 score.

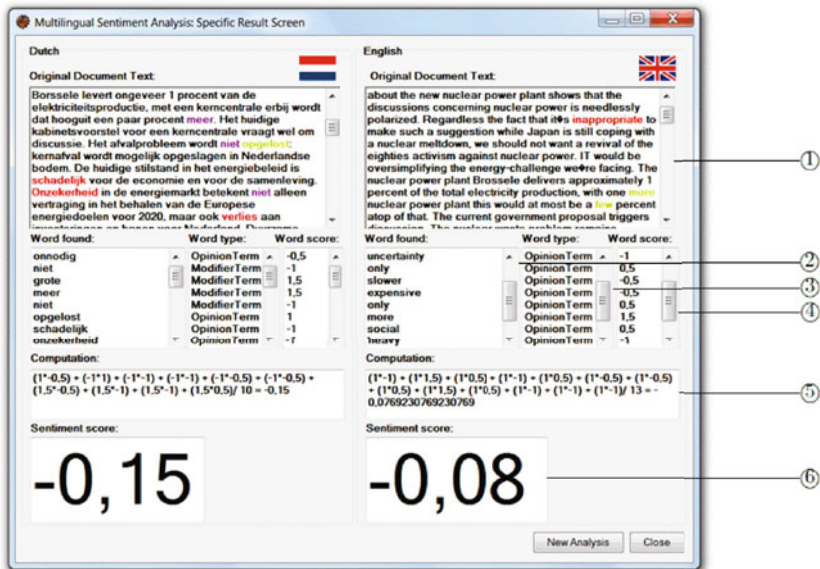


Fig. 4. Specific Result Screenshot

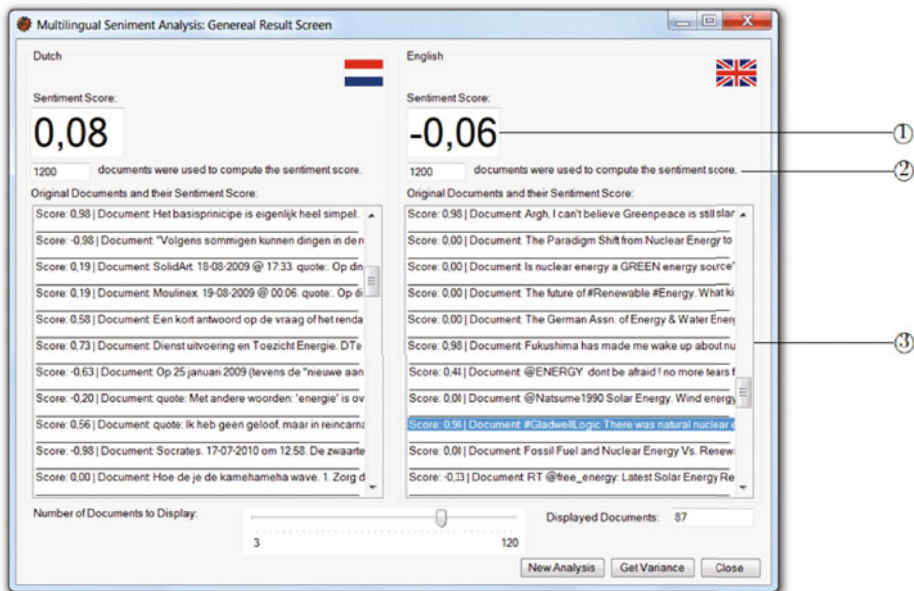


Fig. 5. General Result Screenshot

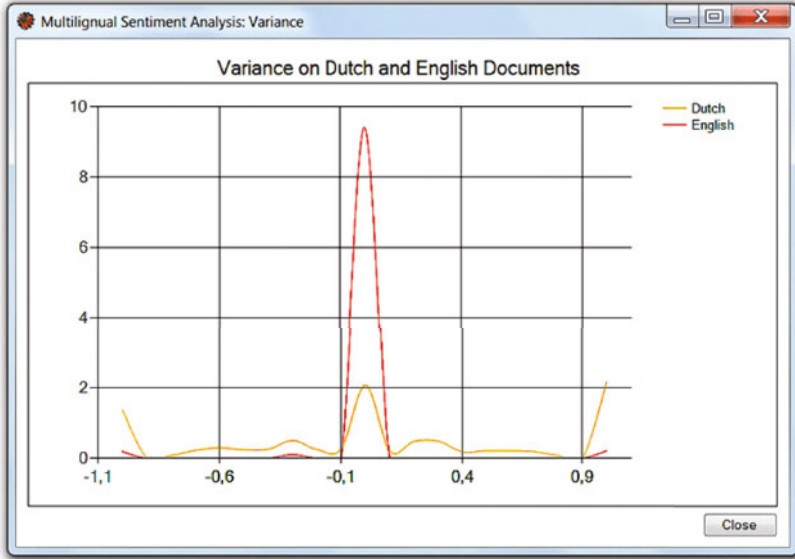


Fig. 6. Graph Screen

6 Evaluation

For the evaluation of the performance of our framework, we first of all use 75 randomly selected Dutch movie reviews from the Dutch movie website FilmTotaal [7]. These reviews have been annotated on a five-star scaling from 1 to 5 by the respective writers of the reviews. We consider reviews with a rating of 1 or 2 stars to be negative, whereas reviews with higher ratings are considered to be positive.

Additionally, we use 75 randomly selected English movie reviews from the IMDb movie website [10]. These reviews have been annotated by their respective writers on a five-star scaling from 0 to 4. Pang and Lee [12] have already created positive and negative subsets of these reviews, which are readily available to us.

The sources of both considered sets of reviews have similar audiences between the age of 18 and 54, with no kids and some degree of college education. Our source of English reviews is slightly more favored by men compared to the source of Dutch reviews, whereas the source of Dutch reviews is favored by slightly better educated visitors [2].

For further (qualitative) analysis of differences in the way in which sentiment is conveyed in different languages, we consider two additional batches of documents, varying from news articles to blog posts, social media, reviews, etc. One batch consists of documents about the Xbox Kinect, whereas the other batch contains documents about nuclear power. These documents have been randomly selected from Dutch and English topic-related forums. Each document in this

collection is available in both English and Dutch. Documents have been translated and annotated for sentiment by three experts until they reached consensus. Per topic, half of the documents was originally published in Dutch, whereas the other half of documents was originally published in English.

The first additional batch of Kinect documents is chosen since the Kinect is relatively new in the console game industry. Therefore, a lot of information is generated on the Web in the form of product reviews, comments on social media and news articles. The great variety in information about the topic and the large daily amount of new information generated, results in a high likeliness of useful and reliable results when applying sentiment analysis.

In order to ensure that the results are not topic-dependent, our second additional batch contains documents about a topic completely different from consumer products. After the recent earthquake and subsequent tsunami disaster in Japan, nuclear power has recently become a well-debated issue. Different countries have different opinions about nuclear power mainly correlated to their dependency on nuclear power.

On our Dutch and English movie corpora, our pipeline shows to be approximately 79% and 71% accurate, respectively, when classifying documents as either positive (for sentiment scores equal to or higher than 0) or negative (for negative sentiment scores). Table 11 shows the confusion matrix for the Dutch and English movie corpora. Table 12 shows the recall, precision and F_1 measure for both positive and negative annotated texts for English and Dutch, as well as the overall accuracy per language and the macro-level F_1 measure. The accuracy of the language selection is approximately 95% for both languages.

Further analysis of our considered sets of documents reveals that we still miss the identification and correct interpretation of specific expressions like: “A piece of cake” and “Easy peasy”. These sentences often carry sentiment, very different from the sentiment found when parsed and labeled with a normal dictionary. Some of the encountered Dutch expressions include: “Daar gaat Shahada *de mist mee in*” (“This is where Shahada messes up”), “Hij *blijkt niet in staat om ...*” (“Apparently he is not capable of ...”) and “Deze 3D animatiefilm, *is in geen enkel opzicht het geld of de tijd waard*” (“This 3D animation is not worth spending any of your time and money on in any respect”), where the text in italics highlights the common expressions.

The next observation is the common use of slang language in documents. A few examples are: “Nevertheless, the cinematography (Sven Nykvist) and the sets (Mel Bourne) were pretty *blah*.” and “If you like Russian style humor or if you like Monty Python style British humor, you will probably *go gaga* over this show.”, where the italic part of the sentence highlights the slang.

We continue by pointing out the difference in expressing negation in English and Dutch. In English, negation tends to be more ambiguous than in Dutch. For example, in English, the word “no” can be used as a negation keyword, whereas the Dutch equivalent for “no” (“nee”) cannot. Moreover, in English, negation can occur in the form of a verb with the affix “n’t”, which further complicates negation in the English language, as compared to negation in Dutch.

Table 1. Movie review classifications (columns) per target classification (rows)

	Positive	Negative
	Dutch/English	Dutch/English
Positive	36/31	7/15
Negative	9/7	23/22

Table 2. Performance measures for movie review classifications

	Precision	Recall	F_1 measure	Accuracy	Macro F_1 measure
	pos/neg	pos/neg	pos/neg		
Dutch	0.80/0.77	0.84/0.72	0.82/0.74	0.79	0.78
English	0.82/0.59	0.67/0.76	0.74/0.66	0.71	0.80

Furthermore, the Dutch language knows semantics that are very hard to identify using our pipeline. For instance, the sentence “*Misschien dat sommige kinderen nog wel zullen lachen om de stupide grappen, maar of ouders daar nu blij mee moeten zijn?*” is a (rhetorical) question which carries no direct sentiment. Conversely, it suggests that the answer to the question would be negative and therefore the sentence appears to the reader to have a negative sentiment. Although the sentence cannot be directly translated in its true form a rough translation would be: “Maybe some of the children would laugh at such easy jokes, but would parents be happy with that?”.

Many acronyms like “lol” (“laughing out loud”), “afaik” (“as far as I know”) and “ga” (“go ahead”), but also emoticons like :D (happy emoticon), :((crying emoticon) and -.- (extremely bored emoticon) are used to express feelings. Those emoticons and acronyms have great influence on the sentiment of small texts and often help humans to recognize the true sentiment of a text. For example, let us consider the next sentence that we came across in the Kinect batch: “That’s... Freaking Awesome!”. This sentence, as our parser interprets it, appears to show a very positive sentiment towards the Kinect. However, the actual sentence was: “That’s... Freaking Awesome! -.- ”, which shows a very strong negative sentiment towards the Kinect. In other words, the emoticon allows the reader to understand that the comment is meant as sarcasm, but the parser is oblivious of this additional information.

Of course, we also found many errors in the parsing of ways of speech like sarcasm, irony and cynicism. The sentence “What a fine movie this turned out to be!” could either be interpreted as a compliment because someone really thinks the movie turned out to be fine or it could be an expression of sarcasm when someone thinks the movie is rubbish. The current sentiment analysis will always interpret the sentence as having the first meaning.

Furthermore, in Dutch it is possible to split a verb. For example, in the sentence “De help-desk loste het probleem maar niet op.” (“The help-desk failed to solve the problem”), “loste” and “op” are together a conjugation of the verb “oplossen” (“to solve”). Yet, the parser does not recognize this as such and misinterprets the word “loste” as “unloaded”.

Finally, our results show that in English, the stars that are used to rate a review are better reflected by the sentiment score that is given to the document, as the English language is more based on explicitly mentioned sentiment. Conversely, the Dutch tend to have a more reserved way of expressing themselves. However, the results shown on the GS (Fig. 6) proved to be highly influenced by the large number of 0-sentiment documents in the chosen batches. Nevertheless, it did show for both batches that English documents either have no sentiment or very strong positive or negative sentiment while the sentiment in Dutch documents shows a more uniform distribution across the -1 to 1 scale.

7 Conclusions and Future Work

Our evaluation indicates that differences in grammar and usage of language indeed influence the results, e.g., because of common expressions, usage of slang language, difference in negation, Dutch semantics, acronyms and emoticons, sarcasm irony and cynicism, splitting of verbs in Dutch and the extremes in the English explicit way of expressing sentiment. Even though the sentiment classification in both languages remains comparable, it is currently not possible to compare the overall sentiment scores directly, as these scores are affected by many different language-specific phenomena, preventing these scores to be trustworthy representations of authors' sentiment. However, this does not imply that these differences cannot be dealt with. Looking at the current accuracy of approximately 79% for Dutch and 71% for English, we do believe it is possible to create a sentiment analysis pipeline that can handle documents of multiple languages, even without losing accuracy. These results suggest a need for normalization of sentiment scores in order to make them comparable across languages. Alternatively, differences across language can be dealt with explicitly.

To further implement multiple languages in a single pipeline, we would like to implement some of the findings in this paper, including detection of common expressions, acronyms and emoticons, difference in negation and usage of slang language. This will leave the challenges concerning semantics, sarcasm, irony and cynicism, that are very different for every language, to future research.

References

1. Abbasi, A., Chan, H., Salem, A.: Sentiment Analysis in Multiple Languages: Feature Selection for Opinion Classification in Web Forums. *ACM Transactions on Information Systems* 26(3) (2008)
2. Alexa Internet Inc.: Alexa the Web Information Company (2011), <http://www.alexa.com/>
3. Amati, G., van Rijsbergen, C.: Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness. *ACM Transactions on Information Systems* 20(4), 375–389 (2002)
4. Bautin, M., Vijayarenu, L., Skiena, S.: International Sentiment Analysis for News and Blogs. In: 2nd International Conference on Weblogs and Social Media (ICWSM 2008), pp. 19–26. AAAI Press, Menlo Park (2008)

5. Dai, W., Xue, G., Yang, Q., Yu, Y.: Co-clustering Based Classification. In: 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2007), pp. 210–219. ACM, New York (2007)
6. Dai, W., Xue, G.-R., Yang, Q., Yu, Y.: Transferring naive bayes classifiers for text classification. In: 22nd Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence (AAAI 2007), pp. 540–545. AAAI Press, Menlo Park (2007)
7. FilmTotaal: Film Recensies en Reviews op FilmTotaal (2011), <http://www.filmtotaal.nl/recensies.php>
8. Gliozzo, A., Strapparava, C.: Cross Language Text Categorization by Acquiring Multilingual Domain Models from Comparable Corpora. In: ACL Workshop on Building and Using Parallel Texts (ParaText 2005), pp. 9–16. ACL (2005)
9. Hofman, K., Jijkoun, V.: Generating a Non-English Subjectivity Lexicon: Relations that Matter. In: 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2009). pp. 398–405. ACL (2009)
10. IMDb.com Inc.: The Internet Movie Database (IMDb) (2011), <http://www.imdb.com/>
11. Moens, M.-F., Boiy, E.: A Machine Learning Approach to Sentiment Analysis in Multilingual Web Texts. *Information Retrieval* 12(5), 526–558 (2007)
12. Pang, B., Lee, L.: A Sentimental Education: Sentiment Analysis using Subjectivity Summarization based on Minimum Cuts. In: 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), pp. 271–280. ACL (2004)
13. Pang, B., Lee, L.: Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval* 2(1), 1–135 (2008)
14. Wan, X.: Co-Training for Cross-Lingual Sentiment Classification. In: Joint Conference of the 47th Annual Meeting of ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL 2009), pp. 235–243. ACL (2009)

Achieving Multi-tenanted Business Processes in SaaS Applications

Malinda Kapuruge, Alan Colman, and Jun Han

Faculty of Information and Communication Technologies,
Swinburne University of Technology, Melbourne, Australia
{mkapuruge, acolman, jhan}@swin.edu.au

Abstract. With the emergence of Cloud Computing and maturity of Service Oriented Architecture (SOA), the Software-as-a-Service (SaaS) delivery model has gained popularity, due to advantages such as lower startup cost and reduced time to market. A SaaS vendor owns and takes the responsibility of maintaining a single application for multiple clients/tenants who may have similar but also varying requirements. Business process modeling (BPM) approaches can be used to package service offerings to meet these varying requirements on a shared basis. However the customizations in those business processes can be challenging. In this paper we discuss the challenges arising from single-instance multi-tenancy, and present our approach to defining business processes in SaaS applications to address those challenges.

Keywords: SaaS, Cloud, SOA, BPM, Multi-tenancy.

1 Introduction

Software-as-a-Service (SaaS) is a software delivery model that lets a customer (tenant) remotely utilize hosted software over the internet and pay according to a certain subscription package [1, 2]. SaaS is being increasingly adopted due to its lower startup costs and higher reliability compared to the *on-premise* software development and delivery model. In contrast to the *on-premise* model, a SaaS vendor [3] owns, hosts and maintains the software application and the underlying infrastructure in the hosting environment, freeing the tenants from those burdens [4].

It has been observed that SOA and SaaS are very closely related architectural models [3]. Service Oriented Architecture (SOA) plays an important role in realizing the SaaS concepts in the enterprise architecture[5]. In a SaaS delivery model the SOA principles can be used to easily integrate existing available services to broaden business offerings, especially when the SaaS vendor alone cannot provide all the required functionalities. Also, in order to specify the order in which the services are delivered and how the dependent services should be invoked, a Business Process Modeling (BPM) mechanism is typically employed. The advantages of BPM such as automated enactment, re-design and verification can be leveraged in delivering Software-as-a-Service simultaneously to multiple tenants in native multi-tenant environments[6].

Consequently, there is an interest in using BPM languages to orchestrate service delivery in SaaS applications. However, *single instance multi-tenant*(SIMT)

applications[7] brings additional challenges to BPM. A SaaS vendor has to capture the commonalities in process definitions, maintain different variations as required by tenants, and achieve effective isolation in modifications to protect the integrity of tenants' requirements and the application as a whole. Most work on meeting similar challenges has occurred in providing virtual individualized data service to tenants. However defining workflows and processes is also a key aspect of SOA, and one that needs addressing if SOA systems are to be deployed as *true* multi-tenanted SaaS applications. In this paper we discuss the relevant challenges in detail and introduce our approach, *Serendip4SaaS* to defining business processes in SaaS applications to address these challenges.

In order to analyze the problem, a motivational scenario is given in Section 2. We will also highlight the challenges and requirements for BPM in SIMT applications. The section 3 gives an overview of our approach. In Section 4, we discuss how our approach is capable of meeting those challenges. Section 5 introduces our prototype implementation. Related works are analyzed in Section 6. Finally, we conclude the paper in Section 7.

2 Problem Analysis

In order to understand the problem and motivate the discussion, we first present an expository scenario. We will then discuss the challenges of BPM in multi-tenant SaaS applications based on the presented scenario.

2.1 Motivation

A Road Side Assistance service benefits motorists by providing emergency assistance when there is a car breakdown. Software systems are being used to coordinate the activities such as towing, repairing etc. Possible businesses that need such a service may include insurance companies, car sellers, and travel agents, providing road side assistance as a value added service to attract clients. For businesses whose core-business is not Road Side Assistance, running and maintaining such a software system and the infrastructure is an onerous task. In such situations it is cost effective and efficient to use an external service and the underlying infrastructure of such kind on a subscription basis [8].

In order to meet this market need, *RoSaaS.com* provides road side assistance as a service. *RoSaaS* (Road Side Assistance as a Service) provides different service packages for its customers depending on their requirements. *RoSaaS* as the SaaS vendor contracts various third party service providers, including garages (GR), tow car (TC) services, taxis (TX) and paramedics (PM). Other service providers such as case handling officers (CO) could be either handled internally by *RoSaaS* or outsourced as appropriate. The *RoSaaS* software handles the complexity of the underlying operations such as requesting, meditating and monitoring third-party and internal services and binding/unbinding service endpoints etc. However, tenants (e.g. car-sellers, travel agents) can request customizations due to changing business goals.

An overall view of the *RoSaaS* business model is given in Fig. 1(a). The lowest layer represents the internal and external (third party) service providers/collaborators. Then *RoSaaS.com* needs to integrate these third party services according to a well-defined choreography. Such choreography defines an *acceptable* ordering and

scheduling of activities in the composition. Based on the offerings of the *RoSaaS* platform different subscriptions/customizations can be delivered to its subscribing customers. The top layer represents the end users, i.e. motorists (MM) that ultimately use the road side assistance, e.g. a traveller or an insurance policy holder.

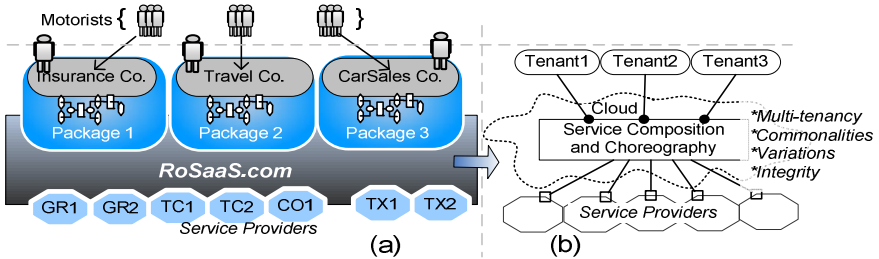


Fig. 1. RoSaaS: (a) Business model, (b) SaaS realization (application architecture)

2.2 Challenges and BPM Requirements

SaaS vendors as business entities have to depend on third party service providers to provide certain functionalities (e.g. towing and repairing) as they alone might not be able to meet all customer requirements. Such dependencies are becoming common as more and more enterprise applications are now exposed and delivered as services [6, 9]. The SaaS realization of the RoSaaS business model is given in Fig. 1(b). The service composition is required to combine the various third-party lower level services, and deliver application level services in a multi-tenant environment, on demand.

In *multi-tenancy* one application instance is utilized by all its tenants [7]. Each tenant interacts with the system as if they are the sole user [6]. A tenant can request modifications to its *package* to suit their changed business objectives. However, these modifications are applied on a single shared application instance. Subsequently modifications could be available to other tenants who use the same application instance. In some cases this might be a necessity, e.g. applying a patch/upgrade. However, in other cases, modifying a common application instance can challenge the integrity of the application instance, compromising the objectives of RoSaaS and other tenants. For example, a request by the tenant, *CarSellerCo* to add additional pre-condition for towing, might delay the execution and thereby hinder the objectives of another tenant such as *InsuranceCo* who utilize the same application instance. Therefore achieving effective isolation in process modifications is a must.

One naïve solution to achieve isolation of process modifications is to allocate dedicated process definitions for each and every tenant, i.e. similar to lower level of the *maturity model* [10]. However, it should be noted that tenants of a SaaS application have common business interests. Hence, there can be *significant overlapping* in those business processes. Having separate process definitions, leads to code duplication. Such duplication deprives the SaaS vendor from exploiting the benefits of SIMT. The SaaS vendor has to apply modifications repeatedly to these process definitions, which is not efficient and could be error prone. Therefore the BPM approach and the language utilized in RoSaaS should be able to *capture commonalities* in behavior. These common *behaviors* or *process segments* can then be reused to define complete *road side assistance business process definitions* as part of the software packages for

individual tenants. As an example, the *towing behavior* could be common to all customer processes and consequently the code/script can be shared among different process definitions.

Although *capturing commonalities* is essential and beneficial to any SaaS vendor, it is allied with two accompanying challenges. Firstly, while the tenants have common business requirements, these requirements may slightly vary in practice. As such, RoSaaS cannot assume a common code/script can continue to serve all the tenants in the same manner. As an example, even though the abovementioned *Towing Behavior* is common to all customers, during design time or runtime a tenant might request a change in pre-conditions to start towing. Therefore the BPM approach should be able to **allow variations**, while capturing the commonalities.

Secondly, capturing commonalities might lead to *invalid boundary crossings*. To elaborate, suppose that tenant *CarSellerCo* requests a modification to the *Towing Behavior* of a road side assistance process. However, this modified business process might violate a business requirement of another tenant such as *InsuranceCo*, because the *InsuranceCo*'s package shares the same behavior. Moreover, as the RoSaaS is a service composition, such changes might lead to violations to the business models of some of its collaborating third party service providers such as *Tow cars* and *Garages*.

To support business processes in single-instance multi-tenant and service oriented applications, in general, it is essential that the following requirements are fulfilled.

- Req 1. *Commonalities* in behavior of the SaaS application and its service delivery should be captured to reduce redundancy.
- Req 2. *Variations* in behavior should be allowed as tenant requirements are similar but not the same.
- Req 3. *Invalid boundary crossings* should be prevented. In this sense, the BPM approach should be capable of identifying the impact of a change from one tenant on both other tenants and the underlying collaborator services.

3 The Approach

In this section we will first give an overview of our process modeling approach. Then we will discuss how the above objectives are achieved in the next section by providing more details.

In order to address the above mentioned requirements, we propose an *organizational* approach to modeling business processes in SaaS applications. We envision a SaaS application instance as an **organization** that provides an abstraction of the underlying collaborating services. Then we define acceptable *behaviors* (e.g. Towing, Repairing) of the organization in a *declarative* manner on top of this organizational structure. These declarative behaviors are then used to construct a business process (view) in the package for each subscribing tenant in the form of an Event-driven Process Chain (EPC) [11].

As shown in Fig. 2, we identify three layers in the SaaS application design. The lowest layer defines the service-service interactions in the application system. We use the ROAD [12, 13] design concepts such as contracts and roles to modularize these interactions in a declarative manner. ROAD allows the definition of **roles** and the **contracts** between roles in a composition/system. A role is a position description and should be *played* by a business service like *web services* hosted in the Garages, Tow

cars, Taxis or even a client application in Case Officers' mobiles. The contracts among roles capture the mutual obligations and responsibilities among these roles in terms of allowed **interactions**. A Role player performs **tasks** by interacting with other role players[14]. As an example, a task in the road side assistance process is *Tow()*, which is an execution step carried out by a bound *towing service*. The towing service needs to interact with other role players, e.g. case officer, garage services, who perform tasks too. However the execution of these tasks needs to be ordered.

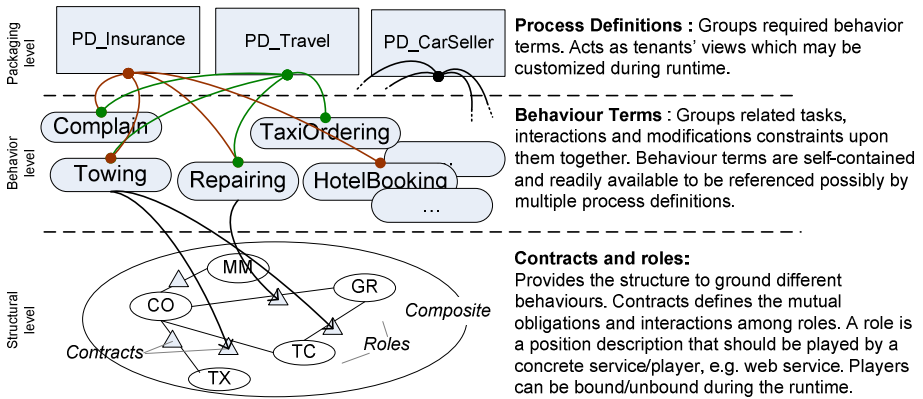


Fig. 2. Different levels of the RoSaaS organization

To achieve that, as shown in Fig. 2, we define different behaviors which we call **behavior terms** on top of structured interactions in the lower organizational layer. Each behavior term specifies an acceptable behavior (ordering of tasks) of the organization. For example, *when towing is required, the case officer (CO) should notify the tow car (TC); upon that notification, the TC should perform towing*. These behavior terms are self-contained and group related tasks that need to be performed by services playing roles in the organization. They also capture any constraints (**behavior constraints**) that govern the order of execution of tasks. This allows ensuring that specific orderings of tasks are not violated during runtime customizations.

At the top layer, **process definitions** make use of such predefined behavior terms. A process definition (a tenant's view of the application or organization behavior) is allocated to each tenant's package. Multiple process definitions can share the same behavior term. These views can be customized at runtime to suit tenants' changed requirements. However such changes are applied in the same organizational structure (single application instance) and should respect the constraints defined in related behavior terms. A business process of a particular tenant can also add additional constraints (**process constraints**) to ensure the goals of the tenant are not violated upon runtime customizations.

Fig. 3 presents a meta-model summarizing the concepts we discussed above.

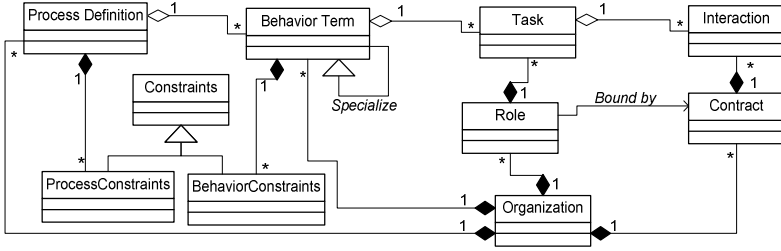


Fig. 3. The Serendip4SaaS meta-model

4 Addressing Requirements

In this section we demonstrate how our approach fulfills the three requirements, defined in section 2.2, of BPM in multi-tenant environments, namely, capturing commonalties, allowing variations and preventing invalid boundary crossings.

4.1 Capturing Commonalties (Req 1) via Behavior Modeling

Tenants of a SaaS application will share some degree of common interest. Therefore the business processes defined to serve the tenants naturally show a *significant overlapping* in the required tasks. As in our scenario, tasks and their ordering associated with car *towing* and *repairing* can be common to many process definitions. As mentioned before, defining dedicated multiple business processes for each tenant can lead to redundancy and need to be avoided. In our approach, behavior terms provide the basis for *modularity* and *re-use* and thereby capture the commonalties across tenants’ business processes.

```

BehaviorTerm Towing{
  Task SendTowReq{
    EPre TowReqd;
    EPost TowReqSent & PickupLocKnown;
    PerfProp 2h;
    Roblig CO;
  };
  Task Tow{
    EPre PickupLocKnown & DestinationKnown;
    EPost CarTowed ;
    PerfProp 24h;
    Roblig TC;
  };
  Task PayTow{
    EPre CarTowed & TowAcked;
    EPost TCPaid;
    PerfProp 2h;
    Roblig CO;
  };
  //More tasks ...
  Constraint c1: (CarTowed>0)->(TCPaid>0);
  Constraint c2: (TowReqSent>0)->[4h,24h](CarTowed>0);
  //More constraints...
};
    
```

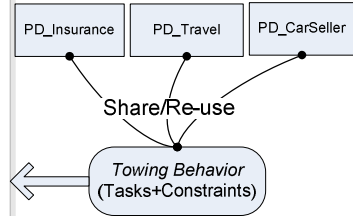


Fig. 4. A sample behavior term

The sample shown in Fig. 4 presents three tasks (*SendTowReq*, *Tow*, *PayTow*) associated with the *Towing* behavior, possibly shared by multiple process definitions.

Each task specifies certain attributes, which includes pre-conditions (EPpre), post-conditions (EPpost), performance properties (PerfProp) and Obligated role (Roblig).

For example, the above *Tow* task specifies “when the pickup location is known and the destination is known (=EPpre), the TC role is obliged to perform task *Tow* within 24 hours. Once the towing is complete, it will trigger events *CarTowed* and *TowAked*(=EPpost)”. Note that the task dependencies are represented via events. For example, task *SendTowReq* triggers event *PickupLocationKnown*, which is a pre-condition of task *Tow*. This means task *Tow* should happen after task *SendTowReq*. Similarly, task *PayTow* happens after task *Tow*.

A behavior term also specifies *modification constraints* on how these tasks should be carried out. We use TCTL[15] to specify such constraints. As an example the first constraint specifies that “every *CarTowed* event should eventually be followed by *TCPaid* event”. We will discuss the use of constraints in section 4.3 in detail.

Two sample process definitions *PD_Insurance* and *PD_Travel* are shown in Fig. 5. A process definition can refer to a behavior term using the attribute *BehaviorTermRef*. As shown, both the *PD_Insurance* and *PD_Travel* re-use the behavior terms *Towing* and *Repairing*, because both tenants *InsuranceCo* and *TravelCo* require to provide towing and repairing for their customers i.e. policy holders and travellers. Specifying such re-usable behavior terms allows capturing the commonalities in different processes.

Also each process definition may specify its own process level constraints using attribute *Constraint*. As an example the *PD_Insurance* specify specific constraint that *an end user request through InsuranceCo should be fulfilled within 5 days*. Such process level constraints protect tenants’ goals (See section 4.3).

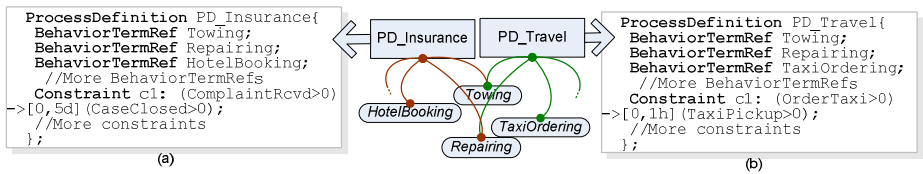


Fig. 5. Sample process definitions

When a process definition group multiple behavior terms, the framework dynamically constructs an EPC graph[11] to present the view of the process definition for the corresponding tenant. The EPC graph is constructed by merging behavior terms using the event dependencies. Fig. 6 shows a section of a complete EPC graph. As shown, the common events (e.g. *CarTowed*) of two tasks are mapped together to construct the process view.

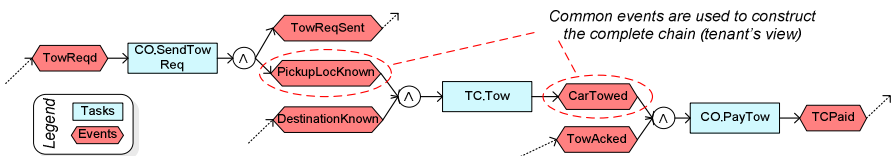


Fig. 6. Process view (EPC) construction from declarative task definitions

In summary, the key characteristic to fulfill *Req1* is the *re-usability* and *modularity* provided by the behavior terms. Furthermore, the *use of events* to define task dependencies helps to achieve loose-coupling among these modularized behavior terms.

4.2 Allowing Variations (*Req 2*) via Behavior Specialization

In SaaS applications it is essential to address the *variations in behavior* as pointed out in section 2.2. As mentioned, process definitions can have specific behavior terms to allow variations, while sharing some common behavior terms. However, it is likely that during runtime these common behavior terms might fail to facilitate the unforeseen variations. As an example, after some time, a tenant (*CarSellerCo*) might require additional conditions to start *tow*; another tenant (*TravelCo*) might require a new *notification* to be sent to the traveller/motorist, once the *tow* is complete.

To support such variations we use *behavior specialization*. In this sense, a behavior term can *specialize* another already defined behavior term to create a new one by specifying the additional properties or overriding existing properties. We call the new behavior a **child** of the already defined **parent**. The parent behavior can either be a *concrete* or an *abstract* (i.e. cannot be packaged/instantiated) behavior term.

Shown in Fig. 7 is a new behavior called *Towing2*, which *specializes* the already defined *Towing* behavior, previously shown in Fig. 4. The child *Towing2* term specifies only those tasks and attributes that would override those of the parent *Towing* term. As shown, the attribute **EPpre** has been changed to delay the *tow* task until the taxi picks up the motorist as required by the tenant *CarSellerCo*.

By specializing (extending) *Towing*, the new *Towing2* will inherit,

1. All the other attributes specified in task *Tow*. E.g., EPpost, PerfProp, Roblig.
2. All the other tasks specified in *Towing*, e.g., SendTowReq, PayTow.
3. All the constraints specified in *Towing*, e.g., c1, c2.(See Fig. 4).

Now the process definition *PD_CarSeller* can refer to the new behavior term *Towing2* instead of *Towing*. The framework will identify these specializations and recognize the inheritance hierarchy to complete the partially defined tasks and attributes of child behavior term. Then the framework will dynamically build the process view as mentioned in the previous subsection. The variation on the constructed EPC graph due to switch to *Towing2* is shown in Fig. 8.

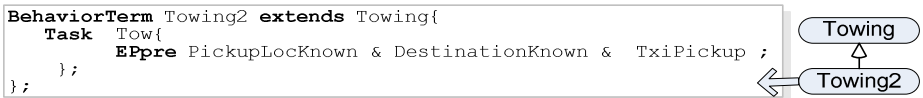


Fig. 7. Specializing behavior (property change of a task)

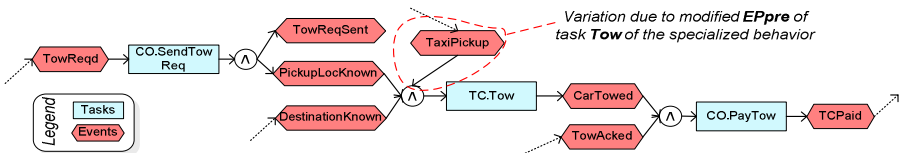


Fig. 8. Variation in *CarSellerCo*'s process due to switch to *Towing2*

Apart from specializing attributes of a Task, a behavior term can add additional task(s) too. For example, if a *TravelCo* service requires an alert to be sent to Motorist upon towing, apart from other usual interactions, a new behavior term *Towing3*, may specify an additional Task *AlertMM* as shown in Fig. 9. Now instead of using *Towing*, the *PD_Travel* may refer to *Towing3*. The variation is shown in Fig. 10.

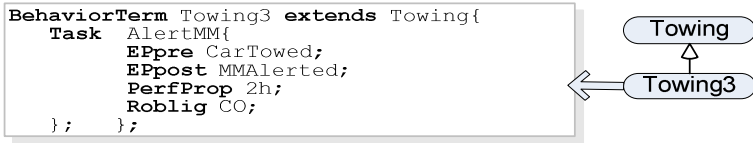


Fig. 9. Specializing behavior (additional task)

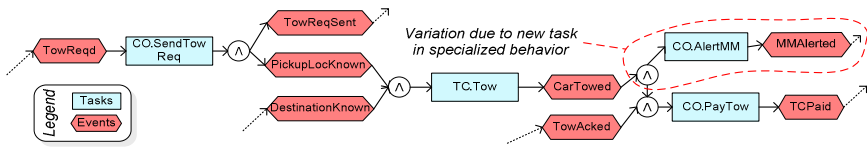


Fig. 10. Variation in *TravelCo*'s process due to switch to *Towing3*

The advantage of such specialization is that a parent behavior can capture the *commonalities* of behavior whilst the children specialize them to capture the *variations*. This mechanism keeps the redundancy in the script to a minimum (e.g., behavior terms *Towing2* and *Towing3* only specified additional properties). Also, since the child inherits all the properties from the parent, the modification constraints defined in the parent can act as a guard to ensure the variation does not lead to any violations (see next section). Furthermore, such specializations can be used in a similar fashion to keep multiple variations consistent over variations in underlying collaborating services.

In summary, the key to achieve specialization is the declarative nature of the behavior terms. Rather than using imperative workflow modeling languages such as EPC [11], we use a declarative language, which made it possible to extend/override the declaratively specified properties. Nonetheless, for visualization purposes we dynamically construct the workflow in the form of an EPC graph by merging the referenced behavior terms together, specialized or otherwise.

4.3 Preventing Invalid Boundary Crossing (*Req 3*) via Two-Level Constraints

Tenants might demand modifications to their process views as shown earlier. However, these modifications are applied in a single application instance. Such a modification might potentially violate a goal of another tenant or even an underlying collaborator. We call them *invalid boundary crossings*. To prevent such *invalid boundary crossings*, our language provides two levels of constraint specifications.

Behavior level constraints are defined in behavior terms, e.g., in *Towing*, irrespective of the enclosing business processes. Once a process definition refers to the behavior term as shown in Fig. 5, the integral modification behavior constraints (both specified and inherited) are applicable. The goal is to ensure that the underlying service-service collaborations are not affected by the customizations to the processes.

Process level constraints are defined in business processes, e.g., in *PD_Insurance*. The goal is to ensure that the modifications/patches to underlying service-service collaborations do not affect the goal of the tenant, e.g., *InsuranceCo*.

For example, in Fig. 4, the behavior term *Towing* has a constraint: “every *CarTowed* event should eventually be followed by *TCPaid* event”. But in some situations additional constraints need to be defined in a much broader *Process* level, across multiple aggregated behavior terms. For example in Fig. 5 the process definition *PD_Insurance* defines a constraint: “the total time from car-is-towed to car-is-repaired needs to be within 5 days”. Here the towing activities and events are defined in a behavior term *Towing* while car repair activities and events are defined in another behavior term *Repairing*.

Suppose the tenant *CarSellerCo* wants to add an additional event to the pre-condition of *tow* task. However this modification results in an alteration in *Towing* behavior shared by another tenant, e.g., *InsuranceCo*. But new pre-condition might delay the start of *tow* task and thereby increase the estimated time to complete a case, possibly violating *InsuranceCo*’s goals. Detecting such invalid boundary crossings without an automated validation can be a time consuming and a tedious task.

Therefore, as shown in Fig. 11, prior to applying a modification requested by a tenant to a behavior term β , two different types of validations are carried out by the framework to analyze the impact of the modification.

Validation 1. All the constraints defined in the behavior β are not violated.

Validation 2. All the constraints defined in process definitions (*sharing process definitions*) that share behavior term β are not violated.

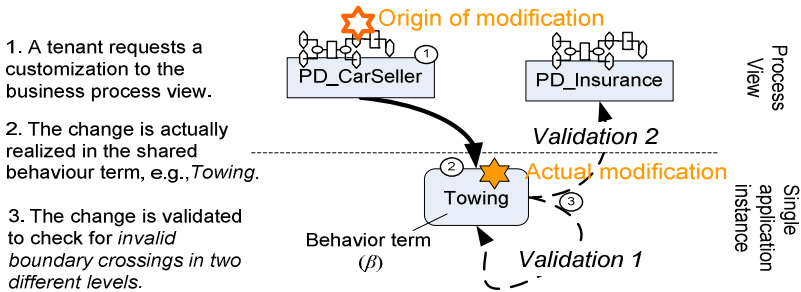


Fig. 11. Two level constraint validation

To formally validate the integrity, we convert the constructed EPC into a Time-Petrinet (TPN) [16] according to the translation rules introduced by van der Aalst et al.[17]. A TPN is a directed bipartite graph that can specify firing intervals of transitions. Places, transitions, firing intervals in a translated TPN are analogous to the events, tasks and estimated time for completing tasks. We use TCTL (CTL with time properties)[15] to specify the constraints. TCTL properties has been used to validate TPNs previously [16]. After mapping the elements (i.e. events, time units) defined in the constraints into the *places* and *firing intervals* of generated TPN, we validate whether the generated Petri-net conforms to the defined set of constraints. To achieve this, we have implemented a wrapper module for *Romeo on-the-fly model checker* [18] which performs the transformation and validation on the fly. If the validation has a

negative outcome, the change is rejected and the violated constraints are shown. Such a validation will occur every time a change is made to the behavior terms to ensure the integrity of the SaaS application instance is not compromised. Upon such modification failures, RoSaaS designer might take further actions, such as looking for a possibility of relaxing a constraint or specializing the behavior term to the client so that other tenants are not affected.

In summary, the way we incorporate temporal constraints into *behavior terms* and *process definitions* means *only the relevant set* of constraints will be considered[14]. Such well-scoped validation reduces the number of constraints that need to be validated without unnecessarily restricting the possible modifications to behaviors. This is an improvement compared to setting up a global set of constraints (applicable for all the processes and collaboration behaviors) to protect the integrity of an application.

5 Implementation

Fig. 12 presents an overview of the implementation framework for our approach to realizing multi-tenanted business processes. Initially the SaaS designer/vendor defines the allowed behaviors using a set of behavior terms at design time. During the runtime, tenants can request the customizations using the *Process Customization and Visualization Tools*. However, the actual customization is realized in the *Model Provider Factory (MPF)*, which maintains all behavior terms including their specializations in a single runtime. The *SaaS designer* too can apply the patches and upgrades to the core behaviors apart from constructing and modifying the views during the runtime. A screenshot of the GUI of *Designer tool* is given in Fig. 13.

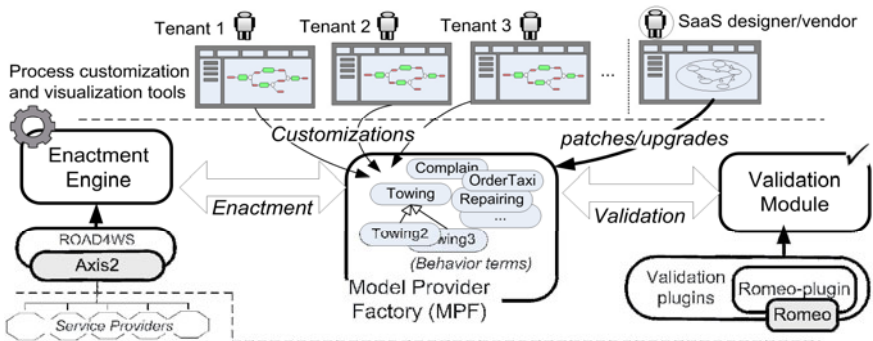


Fig. 12. Implementation Architecture

The *MPF* uses the *Validation Module* to validate the correctness of the defined behaviors and process definitions upon modifications, i.e. both vendor's patches and tenants' customizations. As mentioned in section 4.3, we use the *Romeo on-the-fly model checker*[18] to validate the TCTL[15] constraints against the generated TPN[16]. If a modification violates some constraints, i.e., goals of other tenants or underlying collaborators, it will be rejected. Then the issue will be escalated to the RoSaaS designer pinpointing the affected behavior terms and violated constraints.

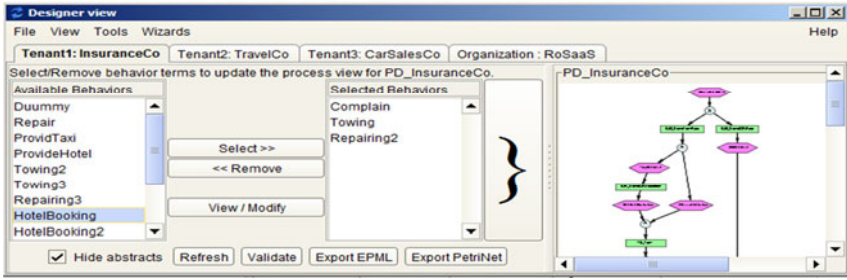


Fig. 13. A screen shot of the Designer Tool of the prototype

The *Enactment Engine* uses the *behavior terms* as grouped by *process definitions* to enact and maintain process instances (as parts of the single SaaS application). For example, the engine will enact a process instance of the definition *PD_Insurance* (Fig. 5) to handle a case of a *Motorist* who is a policy-holder of *InsuranceCo*. We use *ROAD4WS* [13], which is the web service realization of *ROAD*[12] and an extension to Apache Axis2, for binding and interacting with third party service providers such as web services hosted in garages, tow-car companies etc.

6 Related Work

WS-BPEL [19, 20], which is considered the de-facto standard for web service integration, is an obvious candidate for process centric SaaS applications. However, the imperative nature of WS-BPEL limits its adaptability and maintainability, which are prerequisite for SaaS integration/applications. For example, in a multi-tenant environment, supporting each tenant with a dedicated BPEL process can lead to redundancy and potential inconsistency across the processes, compromising the adaptability and maintainability of the SaaS application. One solution for this issue is to use explicit opaque tokens in abstract BPEL[20]. However these abstract descriptors need to be converted to executable BPEL processes prior to enactment. Once enacted, the concrete process cannot be changed dynamically. Therefore the solution does not provide the required agility of adaptation as required by a SaaS application.

More recently a few approaches have been suggested to overcome such issues. Mietzner et al. proposes a customization process for service-oriented SaaS applications via variability descriptors [21]. In this sense, a provided template is further customized based on tenant requirements. Later these templates can be transformed into BPEL processes models. Similarly the VxBPEL language extension for BPEL attempts to allow a process to be customized based on variability points [22]. However, the variability points are fixed via the appropriate parameters during runtime. Furthermore, there are also many aspect oriented approaches[23, 24] that have taken similar paths to address the issue.

These *template-based* or *aspect-oriented* approaches help to identifying the commonalities and variations to a certain degree. For example, the aspects/rules viewed via point-cuts in the AO4BPEL [23] represents the volatile part, while the abstract process which defines the point-cuts represents the fixed part. Similarly in [21], the variability points represents the volatile part while the provided template

represents the fixed part. However, these approaches suffer from a *common weakness*, i.e., they assume that the fixed part and the volatile part can be well-identified at the design time. In contrast, our approach does not rely on such an assumption. Rather than differentiating the volatile and fixed at the design time, we use concepts such as *behavior specialization* to further customize the business processes. As such, the customization on definitions is not limited the design time and can be carried out at runtime. This is beneficial for SaaS vendors and tenants, who might not be able to foresee all the variations upfront.

Inheritance has been used earlier in defining business processes[25]. However, our use of inheritance is for specializing self-contained behaviors defined within an organization, providing reusability and modularity at a finer-grained level. This is in clear contrast to using inheritance for a complete workflow. A SaaS vendor can serve the tenants with different variations of behavior by selecting the specialized behavior terms to form processes suitable for specific tenants on-demand.

Note that the use of temporal constraints is not new[16]. Nonetheless the way we modularize such temporal constraints using the modularization and controlled adaptation provided by the organizational structure in order to define and enact business processes for multiple tenants is a key difference between our approach and the rest. This allows customizations to business processes on demand without compromising the integrity or maintainability of the SaaS application instance. The vendor can utilize the existing already defined behaviors to create customizations by changing or *specializing* them to adjust to latest business and operating conditions but with the necessary checking required.

A summary of comparison is given in Table 1.

Table 1. A summary of comparison

Approach <i>Feature</i>	Kuo [26]	Charfi [23]	Graml [24]	Michiel [22]	Grivas [2]	Mietzn er[21]	Serendip 4SaaS
<i>Customizability</i>	+	+	+	+	+	+	+
<i>Process support</i>	-	+	+	+	-	-	+
<i>Support for single- instance multi-tenancy</i>	-	-	-	-	-	+	+
<i>Capturing commonalities in behavior</i>	-	~	~	~	~	~	+
<i>Facilitating variability in behavior</i>	-	~	~	~	~	~	+
<i>Prevent invalid boundary crossings</i>	-	-	-	-	-	-	+

+ exist, ~ exist with limitations, - does not exist/not applicable

7 Conclusion and Future Work

In this paper we have presented an approach that defines a SaaS application as an organization-based service composition and delivers its service offerings with variations to multiple tenants, via Business Process Modeling (BPM). Such a SaaS application or composition (instance) utilizes third party collaborator services and offers the tenants similar but varying functionalities, while respecting the business rules of all the stakeholders. In particular, we have highlighted how our BPM approach achieves *single-instance multi-tenancy* and meet the requirements of supporting

commonalities and *variations* while preventing *invalid boundary-crossings* between tenant processes when changes are realized. Our approach includes a process modeling language and an implementation framework that helps SaaS vendors and tenants in modeling and managing their applications. The work left for future includes, achieving similar requirements in the data-flow aspects of process modeling and improving the tool support for a better user experience.

Acknowledgments. This work is partly supported by Smart Services CRC, Australia.

References

1. Liang-Jie, Z., Qun, Z.: CCOA: Cloud Computing Open Architecture. In: IEEE International Conference on Web Services (ICWS), pp. 607–616 (2009)
2. Grivas, S.G., Uttam Kumar, T., Wache, H.: Cloud Broker: Bringing Intelligence into the Cloud. In: IEEE 3rd International Conference on Cloud Computing (CLOUD), pp. 544–545 (2010)
3. Laplante, P.A., Jia, Z., Voas, J.: What’s in a Name? Distinguishing between SaaS and SOA. *IT Professional* 10, 46–50 (2008)
4. Waters, B.: Software as a service: A look at the customer benefits. *Digital Asset Management* 1, 32–39 (2005)
5. Sathyan, J., Shenoy, K.: Realizing unified service experience with SaaS on SOA. In: Communication Systems Software and Middleware and Workshops, COMSWARE 2008, pp. 327–332 (2008)
6. Mietzner, R., Leymann, F., Papazoglou, M.P.: Defining Composite Configurable SaaS Application Packages Using SCA, Variability Descriptors and Multi-tenancy Patterns. In: Internet and Web Applications and Services (ICIW), pp. 156–161 (2008)
7. Chang Jie, G., Wei, S., Ying, H., Zhi Hu, W., Bo, G.: A Framework for Native Multi-Tenancy Application Development and Management. In: Enterprise Computing, CEC/EEE, pp. 551–558 (2007)
8. Campbell-Kelly, M.: Historical reflections. The rise, fall, and resurrection of software as a service. *Communications ACM* 52, 28–30 (2009)
9. Barros, A., Dumas, M.: The Rise of Web Service Ecosystems, vol. 8, pp. 31–37. IEEE Computer Society, Los Alamitos (2006)
10. Chong, F., Carraro, G.: Architecture Strategies for Catching the Long Tail, MSDN Library. Microsoft Corporation (2006)
11. Scheer, A.-W.: Business Process Engineering: Reference Models for Industrial Enterprises. Springer-Verlag New York, Inc., Secaucus (1994)
12. Colman, A., Han, J.: Using role-based coordination to achieve software adaptability. *Science of Computer Programming* 64, 223–245 (2007)
13. Kapuruge, M., Colman, A., King, J.: ROAD4WS – Extending Apache Axis2 for Adaptive Service Compositions. In: Enterprise Computing Conference (EDOC). IEEE Press, Los Alamitos (2011)
14. Kapuruge, M., Colman, A., Han, J.: Controlled flexibility in business processes defined for service compositions. In: Services Computing (SCC), pp. 346–353. IEEE Press, Los Alamitos (2011)
15. Baier, C., Katoen, J.-P.: Principles of Model Checking. The MIT Press, Cambridge (2008)
16. Boucheneb, H., Hadjidj, R.: CTL* model checking for time Petri nets. *Theoretical Computer Science* 353, 208–227 (2006)

17. van der Aalst, W.M.P.: Formalization and Verification of Event-driven Process Chains. Department of Mathematics and Computing Science. Eindhoven University of Technology (1999)
18. Gardey, G., Lime, D., Magnin, M., Roux, O.: Romeo: A tool for analyzing time petri nets. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 418–423. Springer, Heidelberg (2005)
19. Weerawarana, S., Curbera, F., Leymann, F., Storey, T., Ferguson, D.F.: Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging and More. Prentice Hall PTR, Englewood Cliffs (2005)
20. OASIS: Web Services Business Process Execution Language Version 2.0. (2006), <http://docs.oasis-open.org/wsbpel/v2.0/>
21. Mietzner, R., Leymann, F.: Generation of BPEL Customization Processes for SaaS Applications from Variability Descriptors. In: Services Computing (SCC), pp. 359–366 (2008)
22. Michiel, K., Chang-ai, S., Marco, S., Paris, A.: VxBPEL: Supporting variability for Web services in BPEL. *Information and Software Technology* 51, 258–269 (2009)
23. Charfi, A., Mezini, M.: Hybrid web service composition: business processes meet business rules. In: International Conference on Service Oriented Computing, pp. 30–38. ACM, New York (2004)
24. Graml, T., Bracht, R., Spies, M.: Patterns of business rules to enable agile business processes. In: Enterprise Distributed Object Computing Conference, vol. 2, pp. 385–402 (2008)
25. van der Aalst, W.M.P., Basten, T.: Inheritance of workflows: an approach to tackling problems related to change. *Theory of Comp. Sci.* 270, 125–203 (2002)
26. Kuo, Z., Xin, Z., Wei, S., Haiqi, L., Ying, H., Liangzhao, Z., Xuanzhe, L.: A Policy-Driven Approach for Software-as-Services Customization. In: 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, pp. 123–130 (2007)

TOAST: A Topic-Oriented Tag-Based Recommender System

Guandong Xu¹, Yanhui Gu², Yanchun Zhang¹, Zhenglu Yang³,
and Masaru Kitsuregawa³

¹ School of Engineering and Science, Victoria University, Australia

² Dept. of Information and Communication Engineering, University of Tokyo, Japan

³ Institute of Industrial Science, University of Tokyo, Japan

Abstract. Social Annotation Systems have emerged as a popular application with the advance of Web 2.0 technologies. Tags generated by users using arbitrary words to express their own opinions and perceptions on various resources provide a new intermediate dimension between users and resources, which deemed to convey the user preference information. Using clustering for topic extraction and incorporating it with the capture of user preference and resource affiliation is becoming an effective practice in tag-based recommender systems. In this paper, we aim to address these challenges via a topic graph approach. We first propose a Topic Oriented Graph (TOG), which models the user preference and resource affiliation on various topics. Based on the graph, we devise a Topic-Oriented Tag-based Recommendation System (TOAST) by using the preference propagation on the graph. We conduct experiments on two real datasets to demonstrate that our approach outperforms other state-of-the-art algorithms.

1 Introduction

Tag-based services, e.g., Del.icio.us¹, Last.fm², and Flickr³, have undergone tremendous growth in the past several years. All of the above services allow users to express their own opinions on resources with arbitrary words. A primary concern in personalized recommender systems is to present users with instrumental means for navigating the resources that are most relevant to their real information needs. In social annotation systems, tags serve as an intermediate metadata between users and resources conveying the users' navigational preference, therefore the key challenge in social annotation recommender systems is to accurately capture user preferences through tags and make use it for personalized recommendation.

A typical social annotation system has three kinds of entities: user, resource and tag. The user prefers some resources and annotates with some words. The

¹ <http://del.icio.us/>

² <http://www.last.fm/>

³ <http://flickr.com/>

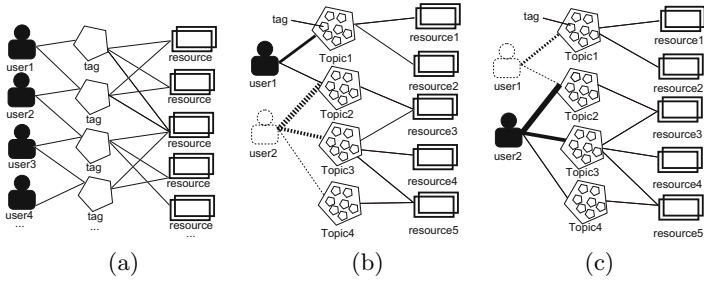


Fig. 1. Tag-based Recommender System

basic intuition behind the social annotation recommender systems is that if a user u has used a tag t for many times, and the tag t has been annotated on a resource r for many times, then we believe that the user u has strong interest on the resource r , which should be recommended to u . Therefore in this case, tags serve as an intermediary between users and resources. Figure 1(a) presents a typical social annotation system.

So far, we can see that different users are interested in various resources and annotate tags on them due to their specific preferences. Thus, the tags they annotated reflect their own opinions and indicate their interests. Although in the context of social collaborative annotation systems, different users possess broad and diverse annotation preferences, we believe their behaviors are governed by a motivational pattern, or called navigational topics. Driven by these topics, a user may annotate own chosen tags on some specific resources. Particularly, the topic information could be extracted from users' annotation activities via a variety of learning approaches. Furthermore, in a social annotation system, a user may have several types of navigational interests, which are represented by the annotated tags. For example, as illustrated in Figure 1(b), user 1 shows much more interest on Topic 1, 2, while user 2 prefers more on Topic 2, 3 (Figure 1(c)). Leveraging the learned topic information represented by the distinctive tag groups, we are able to capture the user's specific visit preference, and make use it for tag-based recommendation by referring to the most appropriate resources which are closely affiliated to this topic. So, in this paper we utilize the topic of tags as a starting point to capture the users' preference.

On the other hand, there are number of obstacles hindering the use of tags due to the inherent defects in social annotation systems. The first one is the severe sparseness of tagging as a result of the power-law phenomenon of the pairwise relationship between tags and resources and between tags and users. In addition, since tags are the arbitrary words generated by users from an uncontrolled vocabulary, it results in other two problems of ambiguity and redundancy in tagging data. For example, a tag may have several meanings and several tags share the similar meanings or serve for one same topic. To address these problems, some studies [14,3,7] have introduced clustering analysis into social annotation systems. In this paper, we also adopt the tag clustering approach for topic extraction.

Another important issue in tag-based recommender systems is the recommendation scoring mechanism, which is to determine the order of recommendations. A commonly used approach is the similarity computation approach, such as the Cosine function of the vector space model. Though the main advantage of this kind approaches is the simplicity and the clarity of results, the deep correlations between the source and target objects are not fully revealed. From the perspective of optimization, the recommendation based on similarity function only addresses the surface optimization and captures the explicit distance relationships. Graph approach is proposed to tackle the finding of the unseen and implicit dependence amongst the nodes in a connected network [5,16]. Inspired by the outstanding merit of graph approaches, we introduce the tripartite graph into the tagging data model and propose a preference propagation on the graph algorithm, called **T**opic-**O**riented **T**ag-based **R**ecommender **S**ystem (TOAST). Together with the devised weighted tag topic model, this algorithm is able to improve the tag-based recommendation performance.

In particular, in this paper we aim to solve the following questions: (a) how to extract the topic information from tagging data? (b) how to model the user preference over these topics? (c) how to propagate the user’s preference among users, topics, and resources? The contributions of our paper are as follows:

- We address the topic extraction from the bipartite graph between tags and resources via clustering analysis.
- We construct a Topic Oriented Graph (TOG) to incorporate such topic information.
- We devise a preference propagation over the TOG algorithm for recommendation scoring.
- We conduct comparative experiments on two real datasets and investigate the impact of various factor on recommendation.

The remainder of this paper is organized as follows. We introduce the preliminaries in Section 2. The TOG construction and preference propagation on TOG algorithm for recommendation are presented in Sections 3. Experimental evaluation results are reported in section 4. Section 5 reviews the related work, and Section 6 concludes the paper and outlines the future directions.

2 Preliminaries

2.1 Social Annotation System Model

In this paper, our work is to generate Top-N recommendations in social annotation systems. A typical social annotation system has three types of objects, users, tags and resources, which are interrelated with one another. Social tagging data can be viewed as a set of triples [4,5]. Each triple (u, t, r) represents a user u annotate a tag t on a resource r . A social annotation system can be described as a four-tuple, there exists a set of users, U ; a set of tags, T ; a set of resources, R ; and a set of annotations, A . We denote the data in the social

annotation system as D and define it as: $D = \langle U, T, R, A \rangle$. The annotations, A , are represented as a set of triples containing a user, tag and resource defined as: $A \subseteq \{(u, t, r) : u \in U, t \in T, r \in R\}$. Therefore a social annotation system can be viewed as a tripartite hypergraph [11] with nodes represented by users, tags and resources and hyper-edges connecting from users to resources via tags represented by annotations.

2.2 Standard Tag-based Recommendation Model

Standard social annotation system may vary in the ways of handling recommendations. In previous researches, the possible approaches include recency, authority, linkage, popularity or vector space models. In this paper, we conduct our work on the vector space model, which is derived from the information retrieval theory. Under such scheme, each user u , is modeled as a vector over a set of tags (or called user tag profile), i.e., $\mathbf{u} = \langle w(t_1), w(t_2), \dots, w(t_{|T|}) \rangle$, where $w(t_k)$ denotes the relationship weight of tag dimension t_k with this user u . Likewise, each resource r , can be modeled as a vector over the same set of tags (i.e. resource tag profile), $\mathbf{r} = \langle v(t_1), v(t_2), \dots, v(t_{|T|}) \rangle$, which $v(t_k)$ denotes the relationship between this resource v and the tag t_k . To calculate the weight element of the vector, different schemes could be used. For example, the work [14, 8] used the occurrence frequency, $w(t_k) = |\{a_{urt_k} = (u, r, t_k) \in A : u \in U\}|$ as the element. Over the tag expressions, given a user, we can match the user and resource profile via similarity computation for making the personalized recommendation. Here common similarity calculation techniques such as Jaccard similarity coefficient or Cosine similarity are applied to obtain the similarity scores between users or resources from the perspective of the intermediate tags. By sorting the similarity scores in a descending order, we eventually perform the Top-N personalized recommendation.

To obtain the needed tag vectors of user and resource, we intuitively decompose the aforementioned tripartite graph into two bipartite graphs (or sub-matrices) of user-tag and resource-tag along the dimension of resource and user, respectively. The element within each matrix is calculated by the accumulated occurrence frequency of tags on different resources and users as mentioned above, i.e., $w(t_k) = \sum_r a_{urt_k}$ and $v(t_k) = \sum_u a_{urt_k}$, where a_{urt_k} is the annotation of u on r by t_k .

Here we mainly use the matrix of resource-tag for topic extraction. More details with respect to the decomposition are referred to [12].

3 Topic Oriented Tag-Based Recommendation

In this section, we describe the proposed Topic Oriented Graph (TOG) and the recommendation strategy based on TOG (TOAST). The basic idea behind TOAST is: If a user u has used a set of tags (or called topic) frequently, and the set of tags have been annotated on the resource r intensively, then we presume that u exhibits strong preference on r . And the preference score of u on r could be propagated over the graph. We begin with the description of topic extraction.

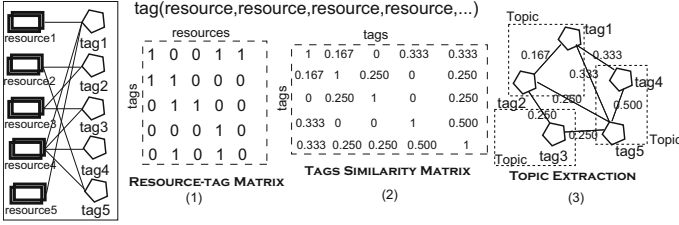


Fig. 2. The illustration on topics generation

3.1 Topic Extraction

Definition 1 (Topic). A Topic is said to be a virtual expression in which the aggregated tags have the similar functioning. Such topic information is extracted by using tag clustering on the bipartite graph between tags and resources.

In this section, we briefly illustrate how to extract the topic information from the tagging data via a working example shown in Figure 2. As discussed above, (1) We start from the tripartite graph to compose a two-dimensional, i.e., the resource-tag matrix by accumulating the frequency of each tag along users. In this expression, each tag is described by a set of resources, to which this tag has been assigned, i.e., $\mathbf{t} = \langle s(r_1), s(r_2), \dots, s(r_{|R|}) \rangle$, where $s(r_j)$ is the accumulated occurrence rate of tag t on resource r_j by all users. (2) After that, we apply the Cosine similarity computation on this matrix and obtain a tag similarity matrix. (3) Upon this similarity matrix, we obtain an affinity graph of tags where the nodes represent the tags and the edge connecting two nodes indicates the similarity between the pair tags. After that, we employ the Hierarchical Agglomerative Clustering (HAC) algorithm [14] to cluster these tags and extract the topics.

As tag clustering is out of the scope of this paper, we will not describe the details of how the HAC algorithm works. Note that the coverage of topics, i.e., the number of tag clusters, is determined by the divisive coefficient of HAC, which likely influences the performance of recommendation. In the later experiment part, we will present empirical results to demonstrate this effect. Thus by this step, we obtain the topic information based on the relationships between tags and resources. Upon the topics (i.e., tag clusters), each user or resource can be re-expressed by a vector over tag clusters i.e. $\mathbf{u} = \langle p(TC_1), p(TC_2), \dots, p(TC_{|TC|}) \rangle$ and $\mathbf{r} = \langle q(TC_1), q(TC_2), \dots, q(TC_{|TC|}) \rangle$. Furthermore we will use the corresponding weights, i.e., p and q , to reflect the affiliated degree between user and topic, and between resource and topic (as shown in Figure 1(b) and (c)).

3.2 Topic Oriented Graph Construction

After the tag clustering, we obtain a number of tag clusters, which here we treat them as topics since these clusters are the groups of tags extracted from

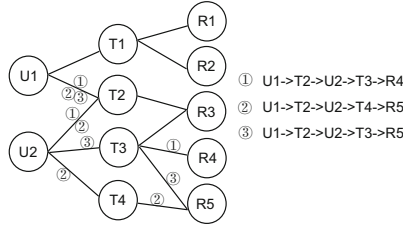


Fig. 3. The illustration on Topic Oriented Graph construction

the bipartite graph of tags and resources. Upon the obtained topic information, we will re-map the preference relationship between users and topics, and affiliation relationship between resources and topics, into two new topic vector forms, which is different from the original pure tag vector expressions. Based on the new topic vector forms, we then construct a Topic Oriented Graph (TOG). Principally, TOG is an undirected graph constituting three kinds of nodes, i.e., users, resources and topics. Topics act as an intermediary bridging the connection between users and resources. The edges connecting the pair nodes are weighted by analogous values, indicating the preference or affiliation scores on various topics. More formally, TOG is defined as follows:

Definition 2 (Topic Oriented Graph). $TOG=G(U, R, T, E, w)$, where U denotes the set of user nodes, R is the set of resource nodes, T is the set of topic nodes, and E is the set the edges connecting the user or resource nodes to topic nodes. $w : E \rightarrow R$ is the weight function for edges.

In particular, the weight w of edge connecting one user u_i node and topic node t_k is determined by the aggregated frequency of the all tags within the topic t_k used by the user u_i over all resources $r \in R$. Likewise, the weight w of edge connecting one resource r_j node and topic node t_k is determined by the aggregated frequency of the all tags within the topic t_k annotated on the resource r_j by all users $u \in U$. In short:

$$w_{ik} = \begin{cases} p(TC_k), i \in U, k \in TC \\ q(TC_k), i \in R, k \in TC \end{cases}$$

Figure 3 illustrates the TOG corresponding to the example shown in Figure 1(c). R_4 and R_5 are two resources which are unknown to u_1 , i.e., no direct edges connecting u_1 to R_4 and R_5 . According to the traditional similarity calculation over the vector space, u_1 does not have preference on R_4 and R_5 . However, from the graph we can see that there are three routes marked by ①, ② and ③ from u_1 to these resources R_4 and R_5 , meaning that u_1 does show some preference on them via the intermediate topics (i.e., preference propagation). In next section, we describe how to propagate the preference over the TOG.

3.3 TOG Based Recommendation

In traditional recommender systems, similarity based strategies such as KNN in Collaborative Filtering are mostly used methods [9]. In contrast, in this paper we

propose a novel recommendation strategy based on preference propagation over the graph. As known in TOG, there are three types of nodes, i.e., user nodes, resource nodes and topic nodes, which are interrelated by the connecting edges. In this case, the preference of one user on a specific resource could be modeled as a traversal path from the user to the resource via one or more topics. More precisely, the recommendation scoring could be considered as a preference propagation process from the starting node to the destination node over the TOG. Bearing this intuition in mind, we intend to make use of the interactions between these three types of nodes and conduct the recommendation via a process called Preference Propagation Algorithm (PPA). Our recommendation strategy works as follows: given a user, we simulate the recommendation scoring as a process to propagate the user's preference through a set of transitional nodes and reach the unknown resource nodes along the shortest connecting path. Here the transitional nodes could be user, topic or even other resource nodes. To make the process clear, below we give some definitions.

Definition 3 (Preference Propagation). *A user u 's preference v_0 can be propagated to the destination node v_n via the intermediate node v_i along the connecting paths in TOG, this process is called Preference Propagation.*

Definition 4 (Propagation Path). *Propagation Path denotes the routes on which preference score passes. Let v_0 be the starting node, and it reaches the destination v_n via v_1, \dots, v_{n-1} . So the Propagation Path is $P(v_0 \rightarrow v_1), \dots, P(v_{n-1} \rightarrow v_n)$, i.e., in short as $P(v_0, v_1, \dots, v_n)$.*

From Definition 4, we know the preference will propagate from the starting node to the destination node step by step along various connecting path. Thus, given a $P(v_0, v_1, \dots, v_n)$ denoting the path from the starting node v_0 to the destination node v_n , the propagated preference information at v_n is defined as:

$$\Theta(v_n) = \prod_{i=1}^{n-1} \theta(v_i, v_{i+1}) \cdot \xi(v_0), \quad v_i \in P(v_1, \dots, v_{n-1}) \quad (1)$$

where $\xi(v_0)$ denotes the preference information of starting node v_0 , which here for simplicity is set be a unity constant, say 1. $\theta(v_i, v_{i+1})$ is a propagation function that measures how the preference score of v_i is propagated to its successive node v_{i+1} . We define it as

$$\theta(v_i, v_{i+1}) = \frac{w(v_i, v_{i+1})}{\sum_{v' \in \text{link}(v_i)} w(v_i, v')} \quad (2)$$

where $\text{link}(v_i) = \{v' \in U \cup R \cup T : e(v_i, v') \in E\}$ indicates the total nodes connected by v_i . From the definition, we also know that node v_{i+1} will receive lower preference propagation if its preceded node v_i has many links to other neighboring nodes.

Given a user and an unknown resource in TOG, there may exist many paths between there two nodes. Recall Figure 3, we can see there are several paths

Algorithm 1. Algorithm of Topic Oriented Tag-based Recommendation

Input: Social Tagging Data, *users*, the set of *Tags*, the set of *Resources*, and *steps***Output:** Recommended Resources for user

- 1 Decompose the social tagging data, and form the resource-tag affinity graph G ;
 - 2 Initialize topic by tag clustering based on bipartite graph between tags and resources and form TOG ;
 - 3 Resources R , a given user u ;
 - 4 **while** *NOT all resources are calculated* **do**
 - 5 Find the shortest path between u and v_n
 - 6 Calculate the propagated preference score by using (1)
 - 7 Go to the next resource node.
 - 8 **end**
 - 9 Sort the propagated preference scores for all resources $v_n \in R$;
 - 10 Return Top-N resources for u ;
-

between user u_1 and resource R_5 . But here we only consider the shortest path between two nodes of u_1 and R_5 . This is due to the consideration that the long step paths contribute less preference information gain in propagation than the shorter paths, and even cause propagation noise, resulting in the ignorance in the Top-N computation [16]. Thus given a user we can obtain a list of propagated preference scores $\Theta(v_n)$, $v_n \in R$, and eventually we sort the scores and return the Top-N resources to the user as recommendations. Algorithm 1 gives the pseudo code of the TOAST algorithm.

4 Experimental Evaluations

To evaluate our approach, we conducted preliminary experiments. We performed the experiments using an Intel Core 2 Duo CPU(2.0GHz) workstation with a 1G memory, running Red Hat Linux 4.1.2-33. All the codes were written in C. We conducted experiments on two real datasets, MedWorm⁴ and MovieLens⁵.

4.1 Experimental Datasets

To get the first dataset, we crawled the article repository in MedWorm system during April 2010 and downloaded the contents into our local experimental environment. After stemming out the entity attributes from the data, four data files, namely user, resource, tags and quads, are obtained as the source datasets of our social annotations. The first three files record the user, tag and document information, whereas the fourth presents the social annotation activities where for each row, it denotes a user u annotates resource r with tag t .

The second dataset is MovieLens, which is provided by GroupLens⁶. It is a movie rating dataset. Users were selected at random for inclusion. All users

⁴ <http://www.medworm.com/>

⁵ <http://www.movielens.org/>

⁶ <http://www.grouplens.org/>

selected had rated at least 20 movies. Unlike previous MovieLens datasets, no demographic information is included. Each user is represented by an ID, and no other information is provided. The data are contained in three files, i.e., movies.dat, ratings.dat and tags.dat. Here we use the tags.dat as the the social annotation activities. The statistical results of these two datasets are listed in Table II.

These two datasets are pre-processed to filter out some noisy and extremely sparse data in order to increase the data quality.

Table 1. Statistics of Experimental Datasets

Property	MedWorm	MovieLens
Number of users	949	4,009
Number of resources	2,61,501	7,601
Number of tags	13,507	16,529
Total entries	15,71,080	95,580
Average tags per user	132	11
Average tags per resource	5	9

4.2 Evaluation Metric and Methodology

We utilized the standard metrics in the area of information retrieval, i.e., *Precision* and *Hit-Ratio*, to evaluate our approach. For each dataset, we divided the whole dataset into two parts by 80% (Training set) and 20% (Test set). We used this 80% of the data to train our topics, upon which we made recommendation. Our recommendation scenario is to use the tags as an intermediary between users and resources to make resource recommendation for each specific user based on our proposed recommendation strategy. The recommended resources can be considered to be closely related to the user interest preference. Due to the random division of training and test sets, we assume that the real annotated resource by each user in the test set should exhibit the similar resource preference distribution to the predicted resource preference. In evaluation, for a given user, we select the Top-N resources as recommendations based on the preference propagation. Meanwhile, we assume the predicted resource list is occurred in the test dataset as well. Thus we look at the real annotated resources in the 20% test set (i.e., the ground truth) and compare it with the Top-N predicted recommendation by our approach to calculate the precision, which measures the proportion of recommended resources that are in the ground truth resources. In particular, we select the Top-N predictions $Pre(N)$ and make these compared with the test set $Data(test)$. The metric $Precision@N$ is defined as follows:

$$Precision@N = \frac{Pre(N) \cap Data(test)}{Pre(N)}$$

In Hit-Ratio evaluation, for each tagging data, we used the “leave-one-out” strategy, i.e., using the user and tag as an input for personalized recommendations and leaving the resource as the ground truth. By comparing the Top-N recommendations with the left ground truth of resource, we can determine whether the

real resource is within the recommended resource list, i.e., given a test element $i, i \in test, hit_i = 1$, if i is in the Top-N recommendation; otherwise $hit_i = 0$. By averaging the sum of hits in the test set, we can calculate the Hit-Ratio:

$$Hit-Ratio = \frac{\sum_{i=1}^N hit_i}{N}$$

where N is the size of test set. From the definitions, we can see that Precision and Hit-Ratio measure the recommendation performance from the overall and Top-N recommendation views.

4.3 Evaluation Results and Discussions

We conduct experiments to evaluate the effectiveness and efficiency of our approach in comparison to the state-of-the-art methods. And we also investigate the impact of topics on the performance of recommendation and efficiency comparisons of four recommendation strategies.

At first, we evaluate our approach against other existing comparative recommendation strategies. Here two comparative methods, i.e., collaborative filtering (CF) and tagging recommender system with clustering only, are used as the baselines. For collaborative filtering approach, we adopt the user based KNN approach [9] (denoted as UserKNN) that does not take any tagging information into account. The recommendation is only dependent on user-based KNN CF approach that referring to the neighboring users' preference. [14] proposed a personalized recommendation approach based on tag clustering where the recommendation scores were determined by Cosine Similarity computation rather than graph based approaches. In that study, they measured the Cosine Similarity of users and resources over the tag clusters to conduct the personalized recommendation. Here we also implement this strategy (denoted as CS) for comparison. As for the proposed preference propagation on graph approach, we implement two approaches with slight difference. One is the TOG that is modeled with a uniform topic weight scheme, i.e., the weight being 1 if a connecting exists between two nodes; otherwise 0. Another implementation is introducing the real weight scheme in TOG, i.e., each connecting edge is weighted by an analogous value instead of a binary value. We denote the former as the Uniform Weight strategy (UW for short) and the latter as Topic Oriented strategy (TO for short).

Figure 4 presents the evaluation results of four strategies on two datasets in terms of precision. From the figure, we can see that, the approaches of CS, UW and TO can both consistently obtain better precision than UserKNN, which justifies the strength of using tags in recommendation that improves the performance of the recommendation. Among these three approaches, TO always achieves the highest precision at various Top-N levels. For example, for Med-Worm dataset, TO is able to improve the precision by around 3% in comparison to CS and UW, and by up to 9% against UserKNN.

Figure 5 gives the hit-ratio results of four strategies on two datasets. In addition to similar outcomes observed, here we would see the more significant improvement of hit-ratio. Again, TO outperforms all other three approaches at four

Top-N settings, increasing the hit-ratio by up to 7.8%, 9% and 14.4% compared to UW, CS, respectively, for MedWorm dataset (shown in Figure 5(a)). Similar conclusions could be found from the results of Movielens (shown in Figure 5(b)). As a consequence, we conclude that the proposed TOAST approach outweighs a big improvement of hit-ratio measure and a small improvement of precision measure over the comparative three approaches. And tagging is empirically verified to enhance the personalized recommendation in overall.

Second, we investigate the efficiency of the proposed approach. We execute the recommendations via various strategies and use the time cost of UserKNN as the baseline, to which the execution duration of other approaches are proportional. The efficiency comparisons are summarized in Table 2. Interestingly, from Table 2, we can see that the execution time of CS is higher than UW and TO, even UserKNN. The main reason to this result is due to the sparseness problem of the tagging data, in which many elements of user-resource matrix appears to be void as a result of no annotation. For matrix based similarity computation, it still needs to calculate the whole similarity matrix, while for graph based approaches, they only need to go across a limited nodes and connecting edges. From such results, we also say that the graph based techniques can behave better than matrix based techniques in terms of efficiency.

Table 2. Execution Efficiency Comparisons

	(Execution time in proportional to User-KNN)			
	UserKNN	CS	UW	TO
MedWorm	1.000000	3.423312	0.873232	0.872613
MovieLens	1.000000	2.332911	0.763782	0.760918

Third, we also study the impact of topic size on recommendation precision. If the topic size is too big, meaning the broad diversity of the topic, it is hard to accurately captures user interest or resource affiliation. On the contrary, too many trivial topics are obtained when topic size is set to small, might harming the accurate recommendation. Table 3 illustrates the recommendation results of MedWorm and MovieLens datasets under different topic size settings. We empirically investigate three topic size settings. In particular, the topic sizes chosen in the experiments (i.e., the divisive coefficients D in HAC) are set to be 0.45(small), 0.6(Medium) and 0.8(Large) for MedWorm and 0.5(small), 0.7(Medium) and 0.8(large) for Movielens, respectively. From the Table, we find that choosing the divisive coefficient in the range between 0.6-0.7 is an appropriate selection, which results in the better precision results.

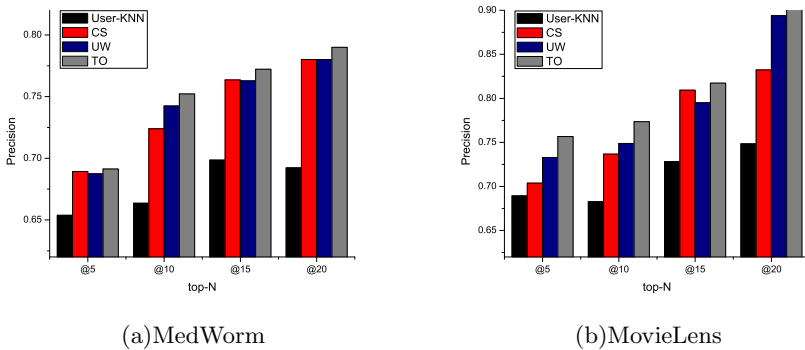
5 Related Work

• Tag-based Recommendation

[2] developed a multi-factorial tag-based recommender system, which took various lexical and social factors of tags into the similarity calculation. [14] proposed

Table 3. Precision at Different Topic Sizes

TopicSize	@5	@10	@15	@20
MedWorm				
Small[D=0.45]	0.615433	0.689823	0.701049	0.709879
Medium[D=0.6]	0.630198	0.70283	0.728912	0.738292
Large[D=0.8]	0.559821	0.612249	0.633002	0.658929
MovieLens				
Small[D=0.5]	0.627738	0.726647	0.762712	0.783321
Medium[D=0.7]	0.691283	0.75221	0.772213	0.789976
Large[D=0.8]	0.677731	0.681021	0.687362	0.692832

**Fig. 4.** Precision Comparisons on Four Different Strategies

a personalized recommendation system by using hierarchical clustering. In this approach, instead of using the pure tag vector expressions, a preprocessing on tag clustering was performed to find out the tag aggregates for personalized recommendation. [17] aimed to integrate the diffusion on user-item-tag tripartite graphs to improve the recommendation of state-of-the-art techniques.

• Information Extraction from Bipartite Graph

We studied several works which are related to our proposal. [15] used the bipartite graph to do neighborhood formation and anomaly detection. In their approach, the bipartite graph was used to reflect the mutual interaction between users and documents in the networked environment. [10] conducted the personalized query recommendation based on the Query-URL bipartite graph, where the queries inputted by users are coupled with the URLs that the users clicked. In social annotation system, because of the triple relationships, it is difficult to deal with the inter-coupled social interactions simultaneously and explicitly, making that we can not apply their strategies to our recommender systems. Therefore, in this paper, we decompose the tripartite graph and extract our topic information based on the bipartite graph of tags-resources.

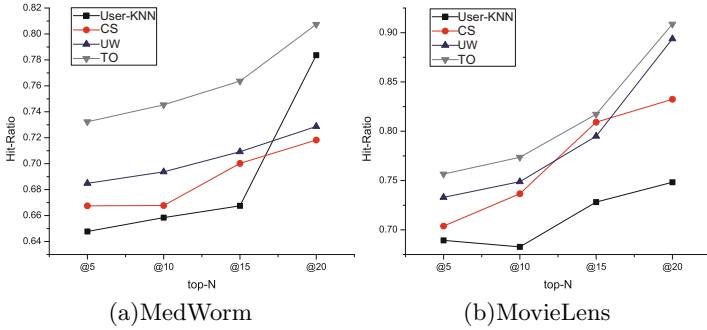


Fig. 5. Hit-Ratio on Four Different Strategies

• Preference Propagation

Preference propagation is an important means in graph-based recommendation. Different from the traditional similarity matching, preference propagation is to reveal the global importance of nodes in the graph via an iterative manner, which was proven to be more robust and accurate. [1] built a video co-view graph, and then adopted the label propagation method in semi-supervised learning to make video recommendation. [13] modeled a random walk model to generate the recommendation. [6] is a topic-sensitive PageRank which introduced the topic vector to make personalized recommendation. Because the topic vector can also be used as a personalized vector to bias users' preference, such models can be applied to make recommendation in different domains. In this paper, we are inspired by such propagation model and propose a topic-oriented propagation strategy to generate the recommendation.

6 Conclusion and Future Work

In this paper, we propose a topic oriented tag-based recommender system. We extract the topic information by clustering tags from the bipartite graph of tags and resources. Each user and resource are re-mapped to an updated vector space, over which the topic oriented graph is constructed. This graph is weighted by the affiliated degrees of users and resources. Then a preference propagation strategy has been devised to make the recommendation scoring. The evaluation results on two real datasets have demonstrated that our strategy can outweigh the other compared strategies in terms of mean precision and hit ratio. In future work, we aim to employ the random walk algorithm into our propagation computation and make more comparisons with more recommendation ranking strategies.

References

1. Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., Ravichandran, D., Aly, M.: Video suggestion and discovery for youtube: taking random walks through the view graph. In: Huai, J., Chen, R., Hon, H.-W., Liu, Y., Ma, W.-Y., Tomkins, A., Zhang, X. (eds.) WWW, pp. 895–904. ACM, New York (2008)

2. Durao, F., Dolog, P.: Extending a hybrid tag-based recommender system with personalization. In: SAC 2010: Proceedings of the 2010 ACM Symposium on Applied Computing, pp. 1723–1727. ACM, New York (2010)
3. Gemmell, J., Shepitsen, A., Mobasher, M., Burke, R.: Personalization in folksonomies based on tag clustering. In: Proceedings of the 6th Workshop on Intelligent Techniques for Web Personalization and Recommender Systems (July 2008)
4. Guan, Z., Bu, J., Mei, Q., Chen, C., Wang, C.: Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. In: Allan, J., Aslam, J.A., Sanderson, M., Zhai, C., Zobel, J. (eds.) SIGIR, pp. 540–547. ACM, New York (2009)
5. Guan, Z., Wang, C., Bu, J., Chen, C., Yang, K., Cai, D., He, X.: Document recommendation in social tagging services. In: Rappa, M., Jones, P., Freire, J., Chakrabarti, S. (eds.) WWW, pp. 391–400. ACM, New York (2010)
6. Haveliwala, T.H.: Topic-sensitive pagerank. In: WWW, pp. 517–526 (2002)
7. Hayes, C., Avesani, P.: Using tags and clustering to identify topic-relevant blogs. In: International Conference on Weblogs and Social Media (March 2007)
8. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: FolkRank: A ranking algorithm for folksonomies. In: Proc. FGIR 2006 (2006)
9. Lathia, N., Hailes, S., Capra, L.: knn cf: a temporal social network. In: Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, pp. 227–234. ACM, New York (2008)
10. Li, L., Yang, Z., Liu, L., Kitsuregawa, M.: Query-url bipartite based approach to personalized query recommendation. In: Fox, D., Gomes, C.P. (eds.) AAAI, pp. 1189–1194. AAAI Press, Menlo Park (2008)
11. Mika, P.: Ontologies are us: A unified model of social networks and semantics. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 522–536. Springer, Heidelberg (2005)
12. Noll, M.G., Meinel, C.: Web search personalization via social bookmarking and tagging. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 367–380. Springer, Heidelberg (2007)
13. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University (1998)
14. Shepitsen, A., Gemmell, J., Mobasher, B., Burke, R.: Personalized recommendation in social tagging systems using hierarchical clustering. In: RecSys 2008: Proceedings of the 2008 ACM Conference on Recommender Systems, pp. 259–266. ACM, New York (2008)
15. Sun, J., Qu, H., Chakrabarti, D., Faloutsos, C.: Neighborhood formation and anomaly detection in bipartite graphs. In: ICDM, pp. 418–425 (2005)
16. Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., Sun, J.: Temporal recommendation on graphs via long-and short-term preference fusion. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 723–732. ACM, New York (2010)
17. Zhang, Z., Zhou, T., Zhang, Y.: Personalized recommendation via integrated diffusion on user-item-tag tripartite graphs. *Physica A: Statistical Mechanics and its Applications* 389(1), 179–186 (2010)

Prefix-Based Node Numbering for Temporal XML

Curtis E. Dyreson and Kalyan G. Mekala

Department of Computer Science, Utah State University, Logan Utah, USA
{curtis.dyreson,kalyan.mekala}@usu.edu

Abstract. Prefix-based numbering (also called Dewey numbering, Dewey level order, or dynamic level numbering) is a popular method for numbering nodes in an XML data model instance. The nodes are numbered so that spatial relationships (e.g., is a node a descendant of another) can be determined just from the numbers. In a temporal XML data collection the spatial relationships change over time as nodes are edited, deleted, and inserted. In this paper we adapt prefix-based numbering to support the concise representation of *items* (elements that share the same identity over time) and *versions* (changes to an item over time). We call our new numbering system time-tunneling dynamic level numbering (TTDLN). We show how to create, store, and update TTDLNs, and how they can be used to efficiently evaluate sequenced and non-sequenced temporal queries.

Keywords: XML, temporal, versioning, prefix-based numbering, Dewey numbering.

1 Introduction

XML is an important language for data representation and exchange, especially in web applications. XML is used to mark-up data adding complex, descriptive structure. For instance, data about a book could be meaningfully marked up in XML with `<title>`, `<author>`, and `<publisher>` elements, each of which describes its enclosed data.

XML data collections change over time as new elements are inserted and existing elements are edited and deleted [13]. Timestamps can be added to XML data to capture this editing history [1],[17]. As an example Fig. 1 shows a fragment of an instance of a temporal XML data model for parts/supplier data. The data in Fig. 1 contains information about parts and suppliers. The time when each datum was put into the data collection is also recorded, i.e., the timestamps capture the *transaction-time* lifetime of each node [18]. The part began on Dec. 23, 2009, and remains current (until *now*). Information about the supplier was entered on Feb. 23, 2010, and the name of the supplier was originally misspelled, and was corrected on Feb 24, 2010.

Efficient evaluation and management of XML data has benefited from the development of numbering schemes for an XML data instance. Among the various numbering systems, *prefix-based* numbering is popular [19][27]. In prefix-based numbering the prefix of each node number is its parent's node number. The node numbers are used to improve the speed of query evaluation since any query language

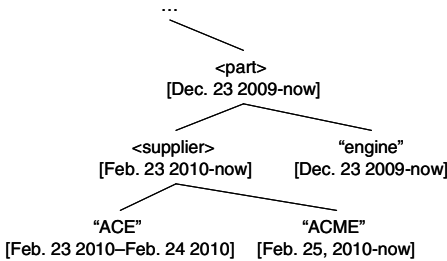


Fig. 1. A temporal XML fragment

```

...
<part>
  <supplier name="ACME">
    <color>red</color>
  </supplier>
  engine
</part>
...

```

Fig. 2. An XML fragment

axis (e.g., preceding-sibling, ancestor, following, etc.) can be evaluated by just comparing numbers.

Prefix-based numbering schemes are currently designed only for non-temporal data model instances which record the state of the data at just a single time point. For temporal data management prefix-based numbering falls short in several ways.

- Prefix-based numbering cannot capture node identity over time. A temporal data collection has to track edits to a node over time and capture whether a node is “new” or previously existed. It is important to observe that prefix-based numbering is incapable of representing this identity over time since a node will not always have the same node number; a node can switch to a new parent or swap position with a sibling, both operations change its number.
- Prefix-based numbering does not represent node versions. Each time a node is “edited” a new version of that node is created. Query operations to find previous or next versions of a node [10] need to be supported by the numbering scheme, but versions are not currently part of a number.
- Prefix-based numbering does not support sequenced evaluation, in which a temporal query is (logically) evaluated at each point in time [14]. The numbers do not have any temporal extent so are incapable of being used for the sequenced evaluation of queries.

The challenge is to construct a numbering system for the items, versions, elements, and other components of a temporal document. The numbering system should have the following properties.

- *Conciseness* – The space cost should be proportional to the size of the document. For a temporal document, it should be linear in the number of items and versions. Said differently it should be linear in the size of an initial slice and the size of changes over time to that slice.
- *Efficient update* – The cost of updating the numbers should be proportional to the size of an edit. Small changes should require few updates.
- *Efficient querying* – The numbers will be used to compute “relationships” between pairs of nodes, ideally in constant time. The non-temporal relationships of interest are parent, ancestor, descendent, etc. Additional temporal relationships of interest are sequenced and non-sequenced versions

of all of the non-temporal relationships, plus relationships about identity over time (is same item) and changes over time (is same/later/earlier version). Ideally all of these relationships can be determined by simply examining or comparing a number for each node.

In this paper, we extend prefix-based numbering to support the full panoply of temporal operations. We focus on dynamic level numbers (DLNs) which are a typical kind of prefix-based number. DLNs are used in eXist [2]. We show how timestamps can be associated with DLNs, how *items* (elements that have identity over time) and *versions* of those items can be represented, how sequenced and non-sequenced queries (e.g., find next version) can be supported, and how all of this can be implemented with minimal changes to an XML DBMS by capturing the temporal history in an XML document [14].

The field of temporal databases has a rich history of research [30], and is impacting industry, for instance Oracle supports some temporal features, in particular, transaction-time, valid-time, and bitemporal tables, current modifications, and automatic support for temporal referential integrity [25]. Concerning the representation of temporal data and documents on the web, Grandi has created a bibliography of research in this area [15]. Chawathe et al. were the first to study time in an XML-like setting [5]. They encoded times in edge labels in a semistructured database and extended the Lorel query language with temporal constructs. Dyreson et al. extended their research with collapsing and coalescing operators [9]. Grandi and Mandreoli presented techniques for adding explicit valid-time timestamps in an XML document [16]. Amagasa et al. next developed a temporal extension of the XML data model [1]. Following that a range of temporal XML topics was investigated, from storage/indexing [6][21][31][32] to querying [10][14][32]. The timestamping of XML documents (or parts thereof) has also been considered in the more general context of versioning of XML documents [21]. Finally, schemes for validating and representing times in XML documents have also been considered [8][17].

Versioning of XML has also been a popular area of research. Marian et al. [20] discuss versioning to track the history of downloaded documents. Chien, Tsotras and Zaniolo [6] have researched techniques for compactly storing multiple versions of an evolving XML document. Buneman et al. [3] provide another means to store a single copy of an element that occurs in many snapshots. This paper differs from all of the above papers since our focus is on node numbering schemes. We cover related work in node numbering in the next section.

2 Dynamic Level Numbering (DLN)

There are many node numbering schemes for XML and good surveys of these schemes are available [24][27]. For a non-temporal document, DLN is a popular node numbering scheme. In DLN each node in a DOM is numbered as follows: a node is numbered $p.k$, where p (the prefix) is the number of its parent and k represents that it is the k^{th} sibling in document order. We explain prefix numbering with an example. Fig. 3 shows an XML data instance for the fragment of a larger XML document that is shown in Fig. 2. We focus just on this fragment in the examples in the remainder of the paper. The instance is represented as part of a larger tree with different kinds of

nodes: *element*, *text*, *attribute*, etc.; whitespace has been stripped. DLNs are assigned as shown in Fig. 4. The DLNs are shown below the elements. The node numbered $p.1$ (`<part>`) represents the first child in document order relative to its parent (which has a DLN of p).

There are strategies for packing DLNs into as few bits as possible, making DLNs relatively concise [19]. DLNs are also very efficient for queries [19]. Given two DLNs we can quickly compute (by comparing the numbers) a specific relationship (child, parent, ancestor, descendant, following sibling, etc.) of one DLN relative to another. For instance $p.1.1.2$ can be compared to $p.1.2$. Since $p.1.1.2$ is neither a prefix nor a suffix of $p.1.2$, it is not a child, parent, ancestor, or descendant. $p.1.1.2$ precedes $p.1.2$ in document order, but is not a preceding sibling since the parent of $p.1.1.2$ ($p.1.1$) is different from that of $p.1.2$ ($p.1$). The efficiency of DLNs in quickly determining these relationships makes them ideal for query processing in XML DBMSs. There also exist schemes for efficient update [33]. Observe that inserting a node high in the tree may force renumbering of an entire subtree, e.g., inserting a node before $p.1.2$ may force the renumbering of $p.1.2.1.3$ to $p.1.3.1.3$. But by using *fractional DLNs* [2], renumbering subtrees can often be avoided, e.g., the node inserted before $p.1.2$ becomes $p.1.1/1$ indicating that it is between $p.1.1$ and $p.1.2$ (in fact, renumbering can be avoided entirely [33]). This paper is orthogonal to fractional DLNs or other renumbering strategies.

3 Numbering a Temporal Document

In this section we extend DLN to number a temporal document or data collection. Extending DLN is complicated by the fact that the DLNs “change” over time, even for nodes that have the same temporal “identity.” We first develop a running example of changes to a document, and then present our new numbering scheme using the example.

3.1 Running Example

There are four basic edit operations: deletion, insertion, update, and subtree move (which can be modeled as a sequence of deletions and insertions). The edits could be specified in an update language like XUpdate, or applied directly in an XML editor. In the running example, we focus on the effects of each kind of edit, treating each edit (or combination of edits) as an individual *snapshot* or *slice* of an evolving temporal document. Assume a second slice shown in Fig. 5, which has one deletion from the previous slice. The new slice is parsed and the (non-temporal) DLNs are assigned as shown in Fig. 5. Note that the DLN for element `<color>` changes from $p.1.1.2$ in Fig. 4 to $p.1.1.1$ in Fig. 5.

Next a third slice is created by combining three edits: moving the `<color>` element subtree from `<supplier>` to `<part>`, updating “engine” to “wing”, and inserting the text “ACME” in the body of `<supplier>`. The third slice and its DLNs are shown in Fig. 6.

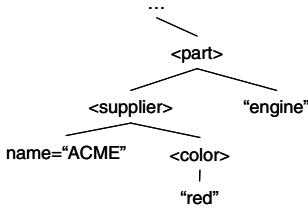


Fig. 3. A non-temporal XML data model instance (a DOM instance)

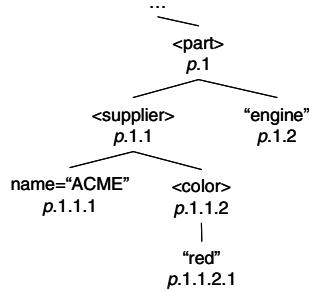


Fig. 4. Node numbers are shown below each node

3.2 Time Tunneling DLN

Now let’s elaborate our new numbering scheme, which we call time-tunneling DLN (TTDLN). We first describe the components of TTDLN and then continue the running example.

In TTDLN, a temporal XML document consists of two separate but related XML documents: a *history document* and an *item document*. (Additionally, the *current* document can be stored as well to improve the speed of queries as of “now” but we do not discuss this optimization further in this paper.) The history document can be thought of as the “coalescing” of all of the slices into a history [11][18]. Each node in the history has the following kinds of numbers.

- DLN_H – The history document is an XML document so each node has a DLN.
- *timestamped DLN (tDLN) list* – A *tDLN* is a DLN together with a timestamp. The timestamp represents the lifetime of the DLN. Since the DLN may change over time, each node stores a list of *tDLNs*.

In addition to the history document a temporal document has to identify *items*, which capture node identity over time, and *versions*, which are changes to the items [8]. In previous research we described how to associate nodes that persist across various slices by *gluing* the nodes [12][29]. When a pair of nodes is glued, an item is created. A version is a change to the item over time.

For simplicity, let’s assume that every element is an item. The items and versions are represented in the *item document*. The item document has the following kinds of numbers.

- *item number* or DLN_I – The item document is an XML document so each node has a DLN.
- *item forward/backward chain number* – The item forward (backward) chain number is the next (previous) DLN_I in the lifetime of the item (only present when an item moves within the history document).

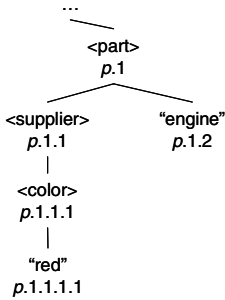


Fig. 5. The second slice (at time 2)

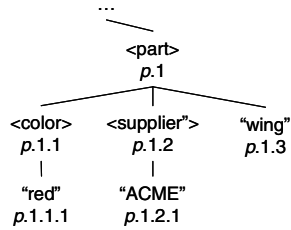


Fig. 6. The third slice (at time 3)

- *version number list* – Each version number in the list is $v [b, e]$ where v is the version number and $[b, e]$ is the lifetime of the version.
- *parent's version number* – A parent's version number is $w:z$ which is a range of versions of the parent to which this node belongs, from version w to version z (* represents an increasing latest version).

Fig. 7 shows a temporal version of the initial slice in Fig. 4. On the left-side of the figure is the item/version document. On the right side is the history document. (To simplify the figure, we have omitted the DLN_{HS} for each node.) We have added pointers to highlight the different kinds of information in each node. The item corresponding to `<supplier>` has item number of “i.1.1”, a version number “1 [1, now]” and a parent's version number of “1:*”. There are no forward/backward chain numbers. Each dashed line represents an annotation or connection between the item document and the history document. The connection means that element `<supplier>` records that it belongs to item “i.1.1”, and item “i.1.1” records that it tracks the element with $tDLN$ “p.1.1 [1,now]”. Note that only elements are items. In general, any type of node could be an item, but in the running example, we assume that the schema designer wants to capture the history of elements rather than text or attribute nodes.

Next we glue the second slice, yielding Fig. 8. Changes are indicated in bold font in the figure. The timestamp in the $tDLN$ of element `name="ACME"` is terminated since it was deleted. A new $tDLN$ is added for `<color>` (and descendents) since it changes from `p.1.1.2` to `p.1.1.1` starting at time 2. In the item document a new version number is added for the item annotating `<supplier>` and the version history of its left child is updated (the right child is in versions 1 and 2, but * represents the latest version so no changes are needed).

Next, a third slice extends the documents as shown in Fig. 9. Changes are shown in bold. New items were created for the moved subtree and item backward/forward chaining is utilized. The changes to `<part>` creates two versions of the element. The first version has children present from in the time interval “[1,2]”. The second version is children present from “[3,now]”.

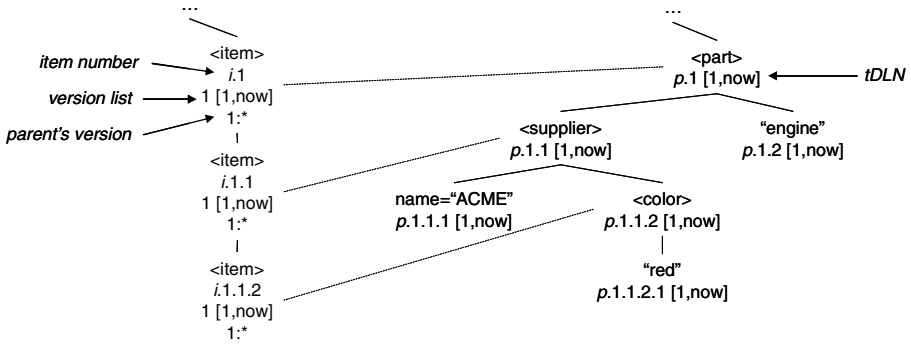


Fig. 7. The first slice as a temporal data model instance, with items and versions represented

There are six important points to observe about our two document approach. First, and most importantly, the history document retains the shape of each slice so extant queries can be directly applied. For example the XPath query “//color[text()=“red”]” will locate color nodes that contain the text “red” across the entire history of the document. Second, the *tDLNs* capture document order and spatial positioning within each slice. Said differently, sequenced evaluation of temporal queries can be supported by simply using the *tDLNs* (and checking timestamp intersection). Third, the item document contains all information necessary to navigate along temporal axes to access past/future versions of elements [10]. Fourth, the history and item documents are append-only. Timestamps and other information are modified but not deleted. Fifth, the history document and the item document are XML documents, and so can be stored using existing native XML DBMSs, with one caveat. Changes to an attribute’s value will create multiple attribute children with the same name. So a small modification to a native XML DBMS is needed to store the history document, e.g., in eXist the HashMap of attributes must be changed to a MultiValueMap, which supports duplicate keys. Sixth and finally, there is a many-to-one relationship from nodes in the item document to nodes in the history document, that is, each node in the item document could be associated to several nodes in the history document. So the item information could be folded into the history document, at the cost of increasing the size of the history document and duplicating the item information. The split into two documents represents a “normalization” of the data.

3.3 Space Cost

In this section we analytically and empirically discuss the space cost. First, the history document must add one node for each element added in the document’s history. So a document with N nodes initially and M insertions will have a history $N + M$ of nodes. But each node also increases in size. Each time a node’s *DLN* changes a new *tDLN* is added. With C changes the size of each node becomes $O(C)$ instead of $O(1)$ (in the non-temporal case). So the history document size is bounded by $O(C(N + M))$.

The item document has one node for every item. There is generally one item for every element in a history document. Nodes in the item document also grow in size over time as new version numbers are added to the version number list. Every edit of an element potentially creates a new version of the item connected to the element, so with C changes the size of each node is bounded by $O(C)$. So the item document size is also bounded by $O(C(N + M))$.

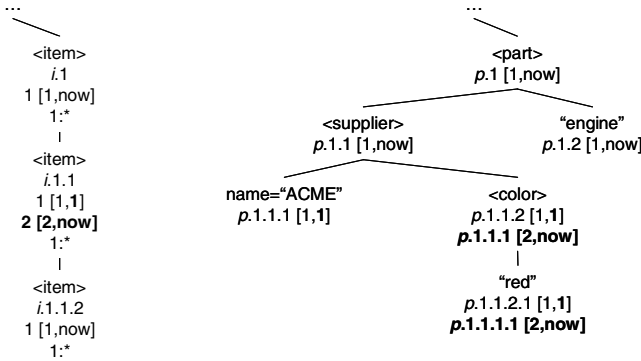


Fig. 8. The second slice added

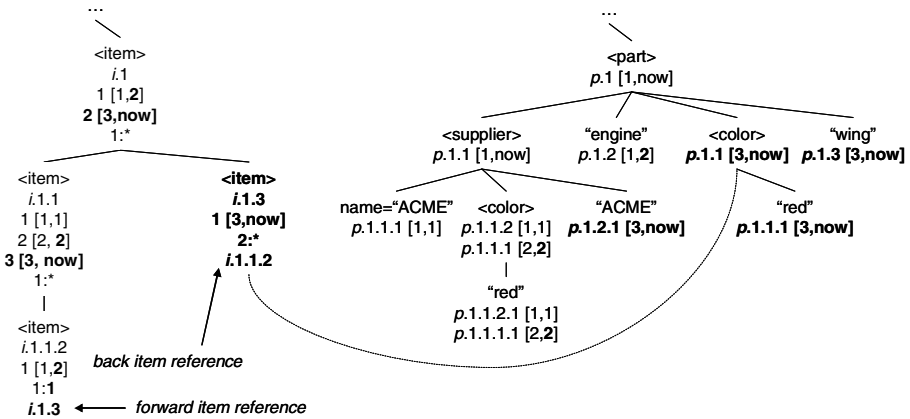


Fig. 9. The third slice added

To get a better feel for the cost, we designed four experiments using the XMark benchmark [28]. We generated an XML document using a benchmark factor of 1, yielding a document approximately 1.1GB in size with approximately two million elements. We then wrote Java programs to randomly edit the document and update the history and item documents. The edit operations insert an element, delete an element, update the string value of an element, and move an element to a new parent.

The operations choose elements at random from the existing document to use in the edit, but only a single element is inserted, deleted, updated, or modified in each edit.

The first experiment tests the cost of insert. We inserted two million elements, essentially doubling the size of the document. Each insertion also adds to the history document, and creates a new item in the item document. Fig. 10 shows the result. The Document line in the figure shows the size of the XMark document (a single slice), which continues to grow as elements are inserted. The History Document line plots the size of the history document, while the Item Document line plots the size of the item document. The history document is slightly larger than the current document since each node in the history document includes *d*DLNs. The item document starts small and grows very slowly.

Experiment two measures the cost of delete. Two million elements were deleted. Fig. 11 shows the result. The current document shrinks to nothing. The other documents stay the same size (but timestamps are updated as elements are logically deleted).

Experiment three measures the cost of update. Two million updates were performed. The results are shown in Fig. 12. The current document stays the same size (the size actually varies but the variation is within a few MBs, invisible in the graph). Each update creates a new text node in the history document, so it essentially doubles in size since text strings are the largest component in the document. No items are created, but new versions are created. The new versions add only a small amount to the item document.

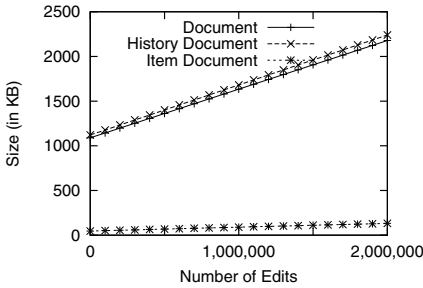


Fig. 10. Element insert

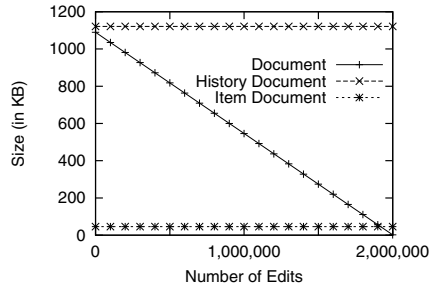


Fig. 11. Element delete

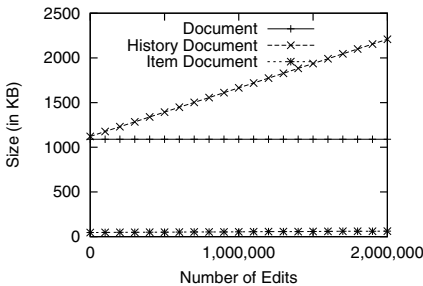


Fig. 12. Value update

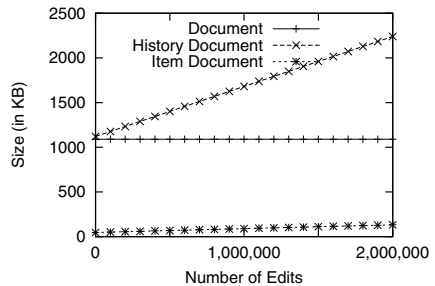


Fig. 13. Element move

The final experiment measures element move. Two million moves were performed. The results are shown in Fig. 13. The moves change the size of the XMark document only by a small amount (not visible in the graph). Each move creates a new item and updates the forward/backward chaining, so the item document doubles in size. The history document also doubles in size as each move is a logical deletion and a physical insertion.

The experiments confirm the analysis that the space cost is linear in the size of the original slice and the number of changes over time.

4 Query Evaluation

The additional space needed for node numbers in the TTDLN scheme pays off in evaluating queries. In this section we consider sequenced versions of DLN operations, as well as non-sequenced extensions that encompass both version and item-related queries. Just as with their DLN counterparts, the temporal operations are implemented by manipulating the numbers without referring to the actual document.

4.1 Sequenced Spatial Relationships

DLNs have predicates to determine whether a specific relationship holds between two nodes. There is one predicates for each axis: parent, child, ancestor, preceding, etc. As canonical examples, we consider only two such predicates.

- $\text{isParent}(X, Y)$ – Is node X a parent of node Y ?
- $\text{isFollowing}(X, Y)$ – Does node X follow node Y (in document order)?

The sequenced version of these predicates determines whether the relationship holds at every point in time for a given time interval from $[b, e]$.

- $\text{isParentSeq}(X, Y, b, e)$ – Is node X a parent of node Y in the history document at each time between time b and e ? We can evaluate this by examining the t DLNs in each node. Let Z be a t DLN in node Y that intersects $[b, e]$. Then for every t DLN, W , in node X such that the timestamp of W intersects that of Z , test whether $\text{isParent}(W, Z)$ holds. Essentially this is the same logic as the non-sequenced case, except for the additional processing of the timestamps.
- $\text{isFollowingSeq}(X, Y, b, e)$ – Does node X follow node Y (in document order) between times b and e ? Evaluate this by utilizing the t DLNs of X and Y . Let Z be a t DLN in Y that intersects $[b, e]$. Then for every t DLN, W , in X such that the timestamp of W intersects that of Z , test whether $\text{isFollowing}(W, Z)$ holds.

Unsurprisingly the other sequenced predicates have a similar form.

4.2 Sequenced Constructors

DLN operations also include constructors that build children or parents relative to a node. The sequenced versions of these constructors are similar to their non-sequenced brethren.

- $\text{getParentSeq}(X, b, e)$ – Get the parent of X in the history document at each time between time b and e . We can quickly find the parent by examining the t DLNs of X . Let Z be a t DLN in node X that intersects $[b,e]$. Then $\text{getParent}(Z)$ is in the list of sequenced parents.
- $\text{getChildren}(X, b, e)$ – Get the children of X in the history document at each time between time b and e . Let Z be a t DLN in X that intersects $[b,e]$. Then add $\text{getChildren}(Z)$ to the list of children returned by the constructor.
- $\text{getSliceDLN}(X, t)$ – Get the DLN in the slice of node X at time t . Extract the DLN from the t DLN in node X at time t .

Other constructors (getAncestors , getDescendants , etc.) are variations of the above constructors.

4.3 Non-sequenced Operations

The item document is used primarily for non-sequenced operations. A non-sequenced operations tunnels through time between different states of a document.

- $\text{nextVersion}(X, v)$ – Extract the time of version $v+1$ of history node X . We can evaluate this by first following the connection to the item for X (node X stores the item number or DLN_i for the item). From the item, we extract the version number list. Next we traverse the list until we locate version $v+1$. If no version $v+1$ exists, then check whether the item has a forward item reference. If not, then there is no version $v+1$ for this item. Otherwise, follow the forward reference to the first version of the next item. Assume that the version starts at time t . Return t . (Alternatively, we could build the text of version $v+1$ by visiting each child of the item to determine whether it is a member of version $v+1$. For the item and its children that are members of the version we traverse the connection back to the corresponding history nodes to obtain the text of the version.)
- $\text{previousVersion}(X, v)$ – Similar to $\text{nextVersion}(X, v)$.

Since the item numbers are essentially DLNs, all of the (non-temporal) DLN operations can be applied to item numbers.

- $\text{parentItem}(X)$ – Return item P , the parent of item X . The item number for P , is a prefix for that of X .
- $\text{childItems}(X)$ – Return the list of items that are the children of item X . Find the smallest k such that there is no item with item number $X.k$, then build a list of items $X.1$ through $X.k-1$.

5 Conclusion

Prefix-based numbering is a node numbering scheme that promotes the efficient querying of XML data. In prefix-based numbering a node is numbered using its parent's number as a prefix. The scheme cannot be applied to a temporal data collection since parent node numbers (the prefixes) change over time. In this paper we present a new node numbering scheme that we call time-tunneling dynamic level

numbering (TTDLN). Prefix-based numbering remains at the core of TTDLN, but the new scheme splits a document into two pieces: a history document, which is the history of the XML document over time, and an item document that keeps track of node identity over time and node versions. In the history document each DLN is transformed into a list of timestamped DLNs. In the item document each DLN becomes an item number, a list of version numbers and timestamps, a range of parent versions, and a forward/backward reference to a new/previous position of the item in the history. We showed how TTDLNs can be used to support sequenced and non-sequenced temporal query operations.

References

- [1] Amagasa, T., Yoshikawa, M., Uemura, S.: A Data Model for Temporal XML Documents. In: Ibrahim, M., Küng, J., Revell, N. (eds.) DEXA 2000. LNCS, vol. 1873, pp. 334–344. Springer, Heidelberg (2000)
- [2] Bohme, T., Rahm, E.: Supporting Efficient Streaming and Insertion of XML data in RDBMS. In: DIWeb 2004, pp. 70–81 (2004)
- [3] Buneman, P., Khanna, S., Tan, W.C.: Why and where: A characterization of data provenance. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, pp. 316–330. Springer, Heidelberg (2000)
- [4] Buneman, P., et al.: Keys for XML. *Computer Networks* 39(5), 473–487 (2002)
- [5] Chawathe, S., Abiteboul, S., Widom, J.: Representing and Querying Changes in Semistructured Data. In: ICDE, pp. 4–13 (1998)
- [6] Chien, S., Tsostras, V., Zaniolo, C.: Efficient schemes for managing multiversion XML documents. *VLDB Journal* 11(4), 332–353 (2002)
- [7] Chomicki, J.: Efficient Checking of Temporal Integrity Constraints Using Bounded History Encoding. *ACM Transactions on Database Systems* 20(2), 149–186 (1995)
- [8] Currim, F., Currim, S., Dyreson, C., Snodgrass, R.T.: A tale of two schemas: Creating a temporal XML schema from a snapshot schema with *tXSchema*. In: Hwang, J., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 348–365. Springer, Heidelberg (2004)
- [9] Dyreson, C., Böhlen, M.H., Jensen, C.S.: “Capturing and Querying Multiple Aspects of Semistructured Data. In: VLDB, pp. 290–301 (1999)
- [10] Dyreson, C.: Observing Transaction time Semantics with TTXPath. In: WISE, pp. 193–202 (2001)
- [11] Dyreson, C.: Temporal Coalescing with Now, Granularity, and Incomplete Information. In: SIGMOD Conference, pp. 169–180 (2003)
- [12] Dyreson, C., Snodgrass, R.T., Currim, F., Currim, S.: Schema-mediated exchange of temporal XML data. In: Embley, D.W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, vol. 4215, pp. 212–227. Springer, Heidelberg (2006)
- [13] Dyreson, C., Grandi, F.: Temporal XML. In: *Encyclopedia of Database Systems*, pp. 3032–3035 (2009)
- [14] Gao, D., Snodgrass, R.T.: “Temporal Slicing in the Evaluation of XML Queries. In: VLDB, Berlin, Germany, pp. 632–643 (September 2003)
- [15] Grandi, F.: An Annotated Bibliography on Temporal and Evolution Aspects in the World Wide Web, TimeCenter Technical Report (2003)

- [16] Grandi, F., Mandreoli, F.: The valid web: An XML/XSL infrastructure for temporal management of web documents. In: Yakhno, T. (ed.) ADVIS 2000. LNCS, vol. 1909, pp. 294–303. Springer, Heidelberg (2000)
- [17] Grandi, F., Mandreoli, F., Tiberio, P.: Temporal Modelling and Management of Normative Documents in XML Format. *Data & Knowledge Engineering* 54(3), 327–354 (2005)
- [18] Jensen, C.S., Dyreson, C.: A Consensus Glossary of Temporal Database Concepts – February 1998 Version. In: Etzion, O., Jaiodia, S., Sripada, S. (eds.) Dagstuhl Seminar 1997. LNCS, vol. 1399, pp. 367–405. Springer, Heidelberg (1998)
- [19] Li, C., Ling, T.W.: An improved prefix labeling scheme: A binary string approach for dynamic ordered XML. In: Zhou, L.-z., Ooi, B.-C., Meng, X. (eds.) DASFAA 2005. LNCS, vol. 3453, pp. 125–137. Springer, Heidelberg (2005)
- [20] Marian, A., et al.: Change-Centric Management of Versions in an XML Warehouse. In: VLDB, Roma, Italy, pp. 581–590 (2001)
- [21] Mitakos, T., Gergatsoulis, M., Stavarakas, Y., Ioannidis, E.V.: Representing Time-Dependent Information in Multidimensional XML. *Journal of Computing and Information Technology* 9(3), 233–238 (2001)
- [22] Navathe, S.B., Ahmed, R.: Temporal Relational Model and a Query Language. *Information Sciences* 49(1), 147–175 (1989)
- [23] Nguyen, B., et al.: Monitoring XML Data on the Web. In: SIGMOD, Santa Barbara, CA, pp. 437–448 (2001)
- [24] O'Connor, M., Roantree, M.: Desirable properties for XML update mechanisms. In: Proceedings of the 2010 EDBT/ICDT Workshops (EDBT 2010), New York, NY, USA, 9 pages (2010)
- [25] Oracle Corporation, Application Developer's Guide – Workspace Manager, 10g Release (December 1, 2003)
- [26] Rizzolo, F., Vaisman, A.A.: Temporal XML: Modeling, Indexing and Query Processing. *VLDB Journal* (2007), doi:10.1007/s00778-007-0058-x
- [27] Sans, V., Laurent, D.: Prefix based numbering schemes for XML: techniques, applications and performances. *Proc. VLDB Endow.* 1(2), 1564–1573 (2008)
- [28] Schmidt, A., Waas, F., Kersten, M.L., Carey, M.J., Manolescu, I., Busse, R.: XMark: A Benchmark for XML Data Management. In: Bressan, S., Chaudhri, A.B., Li Lee, M., Yu, J.X., Lacroix, Z. (eds.) CAiSE 2002 and VLDB 2002. LNCS, vol. 2590, pp. 974–985. Springer, Heidelberg (2003)
- [29] Snodgrass, R., Dyreson, C., Currim, F., Currim, S., Joshi, S.: Validating quicksand: Temporal schema versioning in tauXSchema. *Data Knowl. Eng.* 65(2), 223–242 (2008)
- [30] Tansel, A., Clifford, J., Gadia, S., Jajodia, S., Segev, A., Snodgrass, R.T.: *Temporal Databases: Theory, Design, and Implementation*. Benjamin/Cummins Publishing Company (1993)
- [31] Wang, F.-s., Zaniolo, C.: XBiT: An XML-based bitemporal data model. In: Atzeni, P., Chu, W., Lu, H., Zhou, S., Ling, T.-W. (eds.) ER 2004. LNCS, vol. 3288, pp. 810–824. Springer, Heidelberg (2004)
- [32] Wang, F., Zaniolo, C.: An XML-based Approach to Publishing and Querying the History of Databases. *World Wide Web* 8(3), 233–259 (2005)
- [33] Xu Yu, J., Luo, D., Meng, X., Lu, H.: Dynamically Updating XML Data: Numbering Scheme Revisited. *World Wide Web* 8(1), 5–26 (2005)
- [34] Xyleme, A.: dynamic warehouse for XML Data of the Web. *IEEE Data Engineering Bulletin* 24(2), 40–47 (2001)

Transparent Mobile Querying of Online RDF Sources Using Semantic Indexing and Caching

William Van Woensel¹, Sven Casteleyn², Elien Paret¹, and Olga De Troyer¹

¹ Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussel, Belgium

{William.Van.Woensel, Elien.Paret, Olga.Detroyer}@vub.ac.be

² Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
Sven.Casteleyn@upv.es

Abstract. Due to advancements in mobile technology and connectivity, mobile devices have become fully-fledged web clients. At the same time, more and more Semantic Web data is becoming available, to a point where it becomes usable for various mobile application scenarios. However, most applications are limited to using pre-defined query endpoints in order to access Semantic Web data, which leaves a huge part of the Semantic Web, consisting of online RDF files and semantically annotated websites, inaccessible. In this paper, we present a mobile query service for the efficient and transparent querying of large amounts of small online RDF sources. In order to achieve this, the query service relies on two key components: 1/ a lightweight semantics-based indexing scheme, to identify sources relevant to posed queries, and 2/ a caching mechanism that locally stores frequently-used data.

1 Introduction

The Semantic Web has evolved greatly in its ten-year existence. With the building blocks (i.e. RDF(S), OWL and SPARQL) already in place for some years, and mature and reliable software becoming widespread (e.g. Jena, Sesame, Virtuoso), we are now finally seeing increasing amounts of semantic web data becoming available online. The Linked Data initiative has fostered the deployment and interconnection of large semantic datasets, covers various domains and currently amounts up to 4.5 trillion triples. The so-called lightweight Semantic Web, where existing (X)HTML content is annotated using semantic annotation languages (e.g. RDFa, microformats), is also growing. According to Yahoo! BOSS, currently close to 955 million websites make use of RDFa. These RDFa annotated websites represent semantic sources in their own right, as RDF triples can be extracted from their annotations.

In a parallel evolution, the performance and screen resolution of mobile devices has steadily increased, up to a point where they are capable of supporting common software applications (e.g. organizer, mail client, Web browser). Combined with the widespread availability of wireless networks and affordable high-speed transmission rates for mobile phone networks, these mobile devices have become fully-fledged, (quasi-)permanently connected Web clients. Their use has become prevalent, and it is estimated that mobile Internet access is to surpass desktop access by 2014¹.

¹ http://www.morganstanley.com/institutional/techresearch/pdfs/Internet_Trends_041210.pdf

Therefore, a key factor in the realization of the Semantic Web is realizing efficient *mobile* access to its data. In various mobile application settings, the management, access and integration of local and remote semantic data is already of paramount importance. Examples include context-aware service discovery [1], mobile augmented reality [2], mobile personalization [3], context-aware systems [4] and mobile social applications [5]. Most existing mobile Semantic Web applications gain access to remote RDF(S) datasets via their exposed SPARQL query endpoints, allowing efficient access to online datasets without straining the mobile device. However, it requires a considerable effort to set up a query endpoint, as none of the existing solutions (e.g. OpenLink Virtuoso, Sesame Server) work out-of-the-box and require substantial setup time. In practice, only major data providers make query endpoints available; smaller online RDF sources are mostly put online as semantic documents (i.e. RDF files). Sindice, a lookup index for the Semantic Web, currently indexes around 246 million of such online semantic documents. This means that a huge part of the Semantic Web, consisting of online RDF files and semantically (RDFa) annotated websites, is currently unavailable to mobile clients.

We present a client-side query service that can be employed by mobile applications to transparently and efficiently query large amounts of small online RDF sources. This service is built on top of an existing mobile query engine (such as androjena² or RDF On the Go [6]) to locally query RDF(S)/OWL data. In order to achieve efficient access on mobile devices, with limited processing power and storage space, the query service relies on two key components: 1/ a lightweight indexing scheme, to identify sources relevant for a particular query based on semantic source metadata (i.e. occurring predicates, subject and object types), and 2/ a caching mechanism that locally stores frequently-used data. We evaluate different variants of each component, and discuss their respective advantages and disadvantages. We also compare the performance of the query service to that of the native mobile query engine.

2 General Approach

As mentioned in the introduction, we focus on providing query access to the huge set of online RDF files and semantically annotated websites. Evidently, it is not possible to consider the entire dataset in existence. Instead, we focus on a selection of this data, as typically required by a certain type of mobile applications, such as semantic context-aware systems [4] or mobile social applications [5]. This querying scenario has its own set of challenges for efficient access. One challenge is to identify sources from this dataset that contain information relevant for a posed query, with a high degree of selectivity. This way, sources irrelevant for the current query can be excluded, keeping the final dataset to query smaller and manageable, and therefore the overall query execution time lower. To achieve this, we build and maintain an index containing metadata about datasources. A second challenge is that, once obtained, source data should be cached locally for efficient access and later re-use. The caching of data is paramount in our setting, because of the overhead of obtaining query-relevant data; every datasource that may contain some query-relevant information needs to be downloaded in its entirety. Naturally, the amount of cached data will be limited due to the space restrictions on mobile devices.

² <http://code.google.com/p/androjena/>

Our approach consists of two phases: the indexing phase and the query phase. An overview of these two phases can be found in fig. 1. In the indexing phase, references to new online RDF sources are received from the application (a.1). In our evaluation (see section 5), this application is a context-aware component called SCOUT [4], which identifies datasources related to physical entities in the user’s vicinity (e.g. by reading URLs from RFID tags near the entities). The Source Manager downloads these sources, and extracts metadata on their content (a.2). Our approach focuses on semantic metadata, namely used predicates, subject and object types. This metadata is then added to our index, called the Source Index Model (SIM), along with the URL of its origin source (a.3). Finally, once the source has been indexed, it is passed to the cache component, which locally caches the source data (a.4).

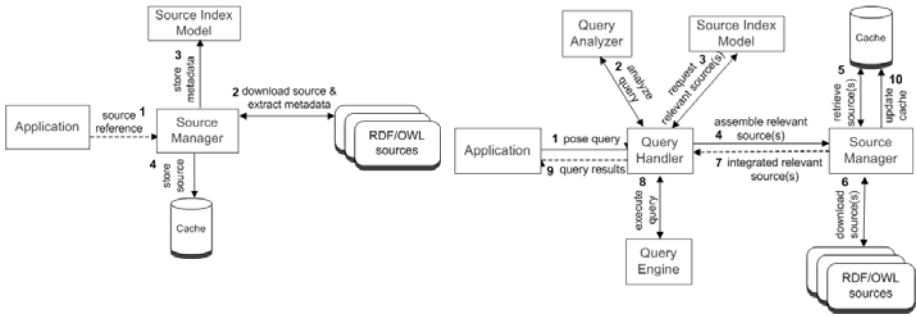


Fig. 1. (a) Indexing phase

(b) Query phase

The query phase is triggered whenever the application poses a new query to the query service (b.1). Firstly, the Query Handler analyzes the query and extracts metadata, i.e. used predicates, subject and object types, from the query (b.2). This query metadata is passed to the SIM, which matches it to the extracted metadata of the indexed sources, and returns references to the sources containing query-relevant information (b.3). Based on these source references, the Source Manager then obtains the required source data (b.4), ideally retrieved from local cache (b.5), or else by re-downloading the corresponding online sources (b.6). Once all source data is obtained, it is combined in one dataset (b.7) on which the query is executed (b.8). After query execution, the results are returned (b.9) to the application. Finally, the cache is updated with the downloaded source data (b.10). In the following sections, we elaborate on the two major parts of our approach, namely indexing and selection of relevant sources, and the local caching of source data.

3 Indexing and Selecting Sources

Our practical experience (confirmed in our experiments, see section 5) shows that executing a query over large sets of datasources is problematic on a mobile device. For one, querying very large datasets slows down query execution time. Also, the entire dataset to query needs to be kept in memory for fast querying, leading to

out-of-memory exceptions for even relatively small amounts of sources (around 400 sources with total size of 67Mb). As a result, we aim to keep the dataset to be queried (called the query dataset) as small as possible. We achieve this by keeping metadata from online datasources in the Source Index Model (SIM); posed queries are analyzed and matched to this source metadata, allowing us to rule out sources irrelevant to the query. Afterwards, the query is executed on the reduced dataset.

Our indexing mechanism needs to comply with certain requirements, related to our mobile, volatile setting. Because we are dealing with mobile devices, we must firstly minimize the required storage space and computational effort for the index. Additionally, in our mobile setting, the query service may receive references to new datasources at any time (e.g. a context-aware system passing data related to the surroundings of the user), meaning the index must be updateable on-the-fly. Therefore, we require a lightweight index that is quick to construct and maintain, and requires minimal space (for optimal performance, the index should fit into volatile memory). At the same time, the index should still guarantee a high selectivity (in other words, filter large amounts of irrelevant sources), in order to reduce the query dataset. We elected for a compact index storing semantic metadata on datasources; namely, the found predicates together with their subject/object types³. This metadata can be efficiently obtained and does not require a lot of storage space (compared to other types of metadata, see related work), making it lightweight. Furthermore, predicates and subject/object type restrictions are often specified in queries, allowing for a high source selectivity (as also shown by our evaluation, see section 5). Therefore, it also complies with the second requirement.

During our evaluation, we compared three variants of the SIM: the first variant (SIM1) stores the found predicates; SIM2 stores the predicates together with their subject types; and SIM3 keeps the found predicates together with their subject and object types. Note that although the two latter variants store additional metadata, they still support queries with unspecified subject and/or object types. In our evaluation section, we compare the different SIMs to study the trade-off between source selectivity on the one hand, and computational and storage overhead on the other.

We can now make some steps in the two general phases of fig. 1 more concrete. In the first phase, the indexing phase, the required metadata is extracted using a predefined SPARQL extraction query (a.2). The metadata is then stored in one of the SIM variants, which are implemented using a collection of hashtables. In the query phase (see fig. 1b), the query is analyzed and the same kind of metadata is extracted from the query (b.2) using the SPARQL Parser library⁴. Finally, RDF triples are extracted from RDFa-annotated websites by using a ported version of java-rdfa⁵.

4 Caching of Source Data

The local caching of source data occurs during the indexing phase, when the source data is passed to the cache after SIM metadata extraction (see fig. 1.a.4), and at the end of the query phase, where the cache may be updated with source data in case of a

³ We currently consider resource types on a per-source level.

⁴ <http://sparql.sourceforge.net/>

⁵ <https://github.com/shellac/java-rdfa>

cache miss (see fig. 1.b.10). Depending on the popularity of the cached data (decided by the used replacement function), some of the cache data will be kept in-memory, while other data will be stored persistently or removed. Since the metadata of removed sources is kept in the SIM, these can still contribute to a posed query; however, they will need to be re-downloaded in case they are required.

In a mobile setting, huge amounts of data can be aggregated after a while. Although it can be argued that modern mid-to high-range mobile devices have ample persistent native storage (e.g. the iPhone is available with 16 and 32 GB of storage), users are probably not keen on spending a large portion to store caching data. By applying a replacement (or removal) function, data is removed from the cache that is not likely to be required in the future. A lot of work has been done concerning replacement strategies specifically meant for location-aware systems (see related work). As our query service does not target one particular type of application, we currently rely on the generic Least Recently Used (LRU) replacement policy; further experimentation with other replacement policies is considered future work.

As was the case for the SIM, the cache needs to comply with certain requirements. Firstly, in order to reduce the query dataset and thereby the overall execution time, a sufficiently fine-grained level of storage and retrieval is needed. Secondly, only a small amount of additional data (e.g. indices such as B+-trees, hashtables) should be kept; ideally, this data should fit in volatile memory (16MB on the Android platform). This is needed to avoid frequent swapping with persistent storage that causes performance loss. Thirdly, it must be possible to add data and update the cache in a quick and efficient way, as data needs to be added on-the-fly on mobile devices with limited computational capabilities. Finally, as for any cache, we must consider the validity of the data in our cache, and ensure data freshness. In the sections below, we discuss the cache organization and cache validity in more detail, and highlight how they comply with the aforementioned requirements.

4.1 Cache Organization

The source data kept in the cache can be organized in different ways, influencing the efficiency and fine-grainedness of cached data retrieval, and the cache construction and update cost. A natural choice for cache organization is to organize the cached triples via their origin source (this organization is called *source-cache*). In this organization, a cache element (or cache unit) corresponds to a downloaded datasource. The cache contains a search index on the source URI of the cached sources, so specific sources can be quickly retrieved. This organization fulfills our second requirement (i.e. minimal additional space overhead) as only one index is kept with a relatively small amount of index entries. Also, the third requirement (i.e. efficient add and update) is met, as the source is simply stored as a single cache unit. However, this organization does not comply with the first requirement, as it does not allow for fine-grained selection of cached source data; it returns all triples from a cached source, instead of only the triples relevant for a given query. Although such coarse-grained retrieval is unavoidable when dealing with online RDF files or RDFa annotated webpages (i.e. they can only be downloaded as a whole), this can be improved upon when dealing with locally stored data.

An alternative choice is to organize the triples according to their shared metadata (called *meta-cache*). This metadata, as was the case for the Source Index Model, includes predicate and subject/object types. For fast retrieval, indices are created on predicates, subject and object types. In this organization, a cache unit contains triples sharing the same metadata (e.g. certain predicate and subject type). This organization allows data to be obtained in a more fine-grained way, as only units with metadata matching the extracted query metadata are retrieved. As such, it complies with our first requirement, i.e. fine-grained storage and retrieval of source data. However, this comes with an additional computational overhead. Firstly, the metadata of the source triples needs to be extracted for each source; secondly, adding the extracted source data to the cache becomes more expensive, as several cache units may need to be updated for a single source (i.e. all units with metadata matching the found predicates, subject and object types). Furthermore, this organization requires three indices with considerable more index entries compared to the source-cache. Finally, each cache unit should also keep the origin source URIs of the triples it contains, to support validity checking (see following section) and to identify sources that need to be re-downloaded, in case this unit is removed and referenced again in the future (i.e. cache miss). Because of these reasons, the fulfillment of the second requirement (i.e. minimal additional storage overhead) and third requirement (i.e. efficient add and update) is tentative. However, meta-cache still has a much lower overhead than the storage and computationally demanding indices traditionally employed to speed up access to RDF data on mobile devices (see related work). We tested both cache organizations in our evaluation, checking how this computational and storage overhead compares to the reduction in data to be retrieved, combined and queried.

We can now again make some of the steps in the general two phases more concrete. For meta-cache, predefined SPARQL extraction queries are employed whenever new source data is added to the cache, in order to extract triples together with their metadata from downloaded sources. When retrieving data from meta-cache (see fig. 1.b.5.), the SPARQL Parser library is again employed to extract metadata from the query, to be matched to the metadata indices. In both source-cache and meta-cache, the indices are implemented as hashtables.

4.2 Cache Validity

Extensive invalidation strategies already exist to efficiently detect invalid data items in traditional client-server architectures and mobile settings. In our setting however, the cached data originates from online RDF files (and annotated websites) stored on general-purpose HTTP webservers, instead of dedicated servers. Therefore, we need to rely the cache support features of HTTP (typically used by proxy caches) to check the validity of cache items. We utilize both the server-specified expiration times (via the `expires` or `Cache-Control: max-age` header fields) and conditional requests for files (using the `Last-Modified` header field) to verify validity of cache units.

To manage the validity of cached data, we need to keep information on the origin source of the cached information, as well as expiration information (i.e. expiration time, or last download time). In source-cache, where all data from a certain source is stored in a single cache unit, we keep this provenance and expiration information per cache unit. For meta-cache, where cache units contain triples from various sources, we also chose

to store origin information per cache unit. To avoid duplicating expiration data, we keep a separate data-structure that stores the expiration information for each source. In case source data requires updating, the source is downloaded and re-indexed, replacing all outdated data from that source in the cache.

5 Experiments

Our experiments have been designed to evaluate the two major parts of our approach. For the first part, the three different SIM variants are compared. In this part, we store all downloaded datasources locally in a cache of unlimited size, to avoid cache interference. For the second part, we employ the best performing SIM variant when comparing the two cache organizations discussed in section 4.1, namely source-cache (i.e. based on origin source) and meta-cache (i.e. based on metadata).

We validate our solution in the application field of context-aware mobile applications using the SCOUT [4] framework. SCOUT gradually discovers physical entities in the user's vicinity, and offers application developers a query-able view of the user's physical environment by combining information from a variety of online RDF(S) sources describing these detected entities. To manage this data and provide efficient query access, SCOUT makes use of the developed query service. In our architecture, SCOUT fulfills the role of the application component (see fig. 1.a), gradually providing new source references to the Source manager over time.

Both SCOUT and the developed query service are based on Android OS 2.2. We employ a fine-tuned version of the androjena library to access, manipulate and query RDF data on the mobile device. The experiments were performed on a Samsung Galaxy S with a 1 GHz processor and 512 MB RAM memory. A total number of 2500 datasources is employed in the experiments⁶, with total size of 477Mb and average size of around 195 Kb. These sources were partly obtained from real-life datasets (e.g. IMDB data), and complemented with generated data, using random resource types and predicates from both selected well-known ontologies (e.g. geo, travel or SUMO) and from proprietary ontologies. In order to reflect a real-world situation, different properties and types reflecting the same concepts (e.g. absolute coordinates) were randomly used. The datasources were distributed across three different web servers with different response times. Finally, we extracted five types of queries from existing mobile SCOUT applications [3, 4] to evaluate the performance of the query service. We give two examples of these queries in listing 1.

<pre>SELECT ?photo ?lat ?long WHERE { ?statue rdf:type region:Statue . ?statue perv-sp:latitude ?lat . ?statue perv-sp:longitude ?long . ?statue dc:description ?photo . ?photo rdf:type dc-type:Image . }</pre>	<pre>SELECT ?rest ?cuisine WHERE { ?rest rdf:type region:Restaurant . ?rest rest:typeOfCuisine ?cuisine . ?cuisine rdf:type rest:ItalianCuisine . }</pre>
--	---

Listing 1. Two employed validation queries

⁶ The used dataset and queries can be found at <http://wise.vub.ac.be/SCOUT/WISE2011/>

5.1 Indexing and Selecting Sources

Firstly, we measure the overhead in size and time for maintaining the SIM during the indexing phase, as the application passes source references to the Source Manager (see fig. 1.a.1). Table 1a shows the total size of the different SIM variants, for increasing portions of the total dataset (625-1250-2500 sources). The table also shows how this size compares to the total size of the indexed sources. The computational overhead to create the SIM is shown in table 1b, and includes the average times of downloading a source, extracting the required metadata, and updating the SIM.

Table 1. (a) Size overhead (Kb)

(b) Computational overhead (ms)

# sources&size	SIM1	SIM2	SIM3		SIM1	SIM2	SIM3
625	210	412	521				
(119Mb)	(0,2%)	(0,3%)	(0,4%)	extract + add	38	140	298
1250	416	798	991	download	372	434	336
(243Mb)	(0,2%)	(0,3%)	(0,4%)				
2500	833	1586	1946	total	410	574	634
(477Mb)	(0,2%)	(0,3%)	(0,4%)				

Table 2a illustrates the selectivity of each SIM variant for our 5 sample queries, by showing the number of identified relevant sources. For brevity, we only show the results for the total dataset of 2500 sources. In table 2b, we show the corresponding overall query execution time over the collected set of datasources. This includes the times required for query analysis, SIM access, data collection, and query execution. Note that SIM1 fails with out-of-memory error for all but the first and third queries; in the case where no SIM is used (i.e. native query engine performance) the same error occurs for any query. Therefore, no entries are available for these cases.

Table 2. (a) Source selectivity(#sources)

(b) Query execution time (s)

	SIM1	SIM2	SIM3		SIM1	SIM2	SIM3	no SIM
Q1	263	46	6	Q1	112,0	14,3	1,9	/
Q2	2025	326	326	Q2	/	21,4	20,9	/
Q3	48	48	4	Q3	8,4	8,3	0,7	/
Q4	1875	83	83	Q4	/	13,1	12,9	/
Q5	2203	328	328	Q5	/	34,1	33,6	/

5.2 Caching of Source Data

Like for the SIM, we first measure the overhead of constructing and maintaining the cache during the indexing phase. For each of the tests, we allow the cache to use 75% of the size of the total dataset as persistent storage space on the device. We also allow 1Mb of volatile memory space for storing frequently-used source data (this does not include extra data such as indices). This limited amount was chosen because Android applications only have 16Mb heap space available, and other components of the query service also consume this memory: for instance, SIM3 takes up about 2Mb for 2500 sources (which will grow as the number of sources increases), while androjena graph objects are also created in-memory.

Table 3 shows the size (in Kb) of in-memory and persistent storage space used by the two different cache organizations, for increasing portions of the dataset (625-1250-2500). We show both the size of in-memory source data and the total size of the in-memory cache, which includes additional data such as index data.

Table 3. (a) size overhead for source-cache (Kb) (b) size overhead for meta-cache (Kb)

# sources	in-memory		persistent	in-memory		persistent
	total	source data		total	source data	
625	1076	1024	3083	258	136	3235
1250	1130	1024	175955	1623	871	171894
2500	1195	1024	365694	2781	1023	331431

In table 4, we show the average computational overhead of adding new source data to the cache during the indexing phase. For meta-cache, this includes extracting the metadata from the source, and updates to existing cache units. These two overheads do not occur in source-cache. For both cache types, overhead also comprises running the replacement function in case the cache becomes full.

Table 4. (a) time overhead for source-cache (ms) (b) time overhead for meta-cache (ms)

extract	add	update	replacement	extract	add	update	replacement
0	1	0	563	673	2	180	50

Tables 5 and 6 contain the overall query execution times for the query phase. The following parts are distinguished: 1) query analysis, 2) cache access, 3) data assembly, and 4) query execution. Query analysis denotes the extraction of metadata from a query. Cache access denotes total access time; tables 7 and 8 show a more detailed view on the constituent parts. Data assembly represents the time needed to combine the retrieved data triples into a single RDF graph, on which the query will be executed. Here, the total number of returned triples is also shown, illustrating the fine-grainedness of the data retrieval. Finally, we show the total amount of time required for query resolving (together with the number of query results in brackets).

Table 5. Query execution times for source-cache (ms)

query	query analysis	cache access	combine data		query execution	total
			# triples	time		
Q1	195	2463	784	81	34 (6)	2773
Q2	46	47383	5460	1541	1105 (409)	50075
Q3	32	1068	550	35	17 (4)	1152
Q4	31	28593	10076	985	37 (4)	29646
Q5	40	61635	15898	1564	515 (214)	63754

Table 6. Query execution times for meta-cache (ms)

query	query analysis	cache access	combine data		query execution	total
			# triples	time		
Q1	29	3090	6	5	7 (6)	3131
Q2	31	39789	1061	464	301 (409)	40585
Q3	30	1654	4	1	10 (4)	1695
Q4	15	17383	371	261	21 (4)	17680
Q5	57	9276	5965	2908	123 (214)	12364

Tables 7 and 8 show the times related to cache access and maintenance. Cache access comprises: 1) SIM access time, 2) time required to retrieve the cache units (the number of returned units is also shown in brackets), and 3) number of cache misses (the resulting amount of sources to be downloaded is shown in brackets), together with the retrieval times for missing cache data. Note that in case of meta-cache, the latter not only includes the download time but also the time required to extract the relevant triples. In case of source-cache, the SIM is employed to identify relevant sources (present in the cache or not); meta-cache does not employ the SIM for cache element identification. Cache maintenance comprises adding new source data and updating the cache (in case missing data was downloaded), and running the replacement strategy if space is needed. As cache maintenance occurs after query execution, it is not included in the cache access times shown in tables 5 and 6.

Table 7. Cache access / update times for source-cache (ms)

query	Cache access				Cache maintenance	
	SIM access	cache retrieval	cache miss		add / update	replacement
			# misses	retrieval		
Q1	4	2087 (5)	1 (1)	372	0	11806
Q2	127	25008 (254)	72 (72)	22248	0	31043
Q3	1	1067 (4)	0 (0)	0	0	318
Q4	7	17282 (59)	24 (24)	11304	1	24818
Q5	75	61560 (328)	0 (0)	0	0	70455

Table 8. Cache access / update times for meta-cache (ms)

query	Cache access				Cache maintenance	
	SIM access	cache retrieval	cache miss		add / update	replacement
			# misses	retrieval		
Q1	0	4 (0)	2 (6)	3086	10	0
Q2	0	1994 (6)	4 (112)	37795	64	572
Q3	0	2 (0)	1 (4)	1652	4	0
Q4	0	477 (16)	21 (40)	16906	219	193
Q5	0	9276 (6)	0 (0)	0	0	6251

5.3 Discussion

First, we discuss the results for the identification and selection of sources. Regarding space overhead (see table 1a), we observe that any of the SIM variants stores only a very small fraction of the total dataset (the largest, SIM3, stores around 0,4%), complying with the requirement for minimal storage space set in section 3. Nevertheless, it should be noted that space overhead amounts to around 2Mb for the largest SIM (SIM3) for 2500 sources. Considering Android applications have a max

heap space of 16Mb (not exclusive for use by the SIM), a sufficiently larger amount of sources would require swapping parts of the SIM to persistent storage, decreasing performance. The computational overhead of extracting and adding data (see table 1b) is reasonable, complying with the requirement for minimal computational overhead. It can therefore be observed that, while the computational and storage overhead rises with the SIM complexity, this overhead appears to be acceptable (making the SIMs indeed lightweight). As expected, the selectivity of the SIM increases with the amount of metadata stored (see table 2a). The selectivity of SIM1 is so poor that an out-of-memory error occurs when assembling the sources for three of the queries (see table 2b); for no SIM (i.e. native query engine performance), an out-of-memory error already occurs for 400 sources. We thus observe that a considerable gain is made in query execution time for SIM2 and SIM3. The difference in selectivity between SIM2 and SIM3 is only visible for queries 1 and 3. This is because these queries constrain the object types of most triple patterns, allowing SIM3 to be more selective. We may thus conclude that the best SIM depends on the posed queries and is application dependent. In our cache experiments, we opted to work with SIM3, as we found the potential increase in selectivity makes up for its (relatively small) overhead.

We now elaborate on the results of caching downloaded source data. First, we consider the cache space and build times (see tables 3 and 4). The additional in-memory storage, taken up by cache index data, is about 0,04% of the total source dataset size for source-cache, while meta-cache requires 0,36%. For meta-cache, this may again lead to memory issues for larger sets, requiring swapping to persistent storage with performance loss. We also observe that, as expected, the total cost of adding new sources is higher for meta-cache, as it requires triple metadata to be extracted. Also, a single source may require updating many units in meta-cache, leading to a higher update time. On the other hand, cache replacement is less costly for meta-cache, because cache units are more fine-grained and thus replacement (i.e. moving units to persistent storage, removing units) is more effective.

With regards to query execution (see tables 5 and 6), our results show that meta-cache retrieves data in a much more fine-grained way than source-cache (i.e. less amount of triples), leading to lower overall data combination and query execution times. Looking in more detail at the cache access times (see tables 7 and 8), we observe that in order to serve a given query, source-cache also requires more cache elements to be retrieved than meta-cache. Indeed, typically a large number of sources contain data relevant to posed queries (e.g. certain predicates). This leads to much higher cache retrieval times for source-cache, and also more cache misses on average. However, although the number of cache misses for meta-cache is smaller, the actual number of downloaded sources is much higher. When a cache unit is removed and later referenced again (cache miss), all sources containing the associated metadata need to be re-downloaded. In case this metadata is contained in a large number of sources (e.g. query 2 and 4), the associated source retrieval time becomes exceedingly high. For more complex queries (requiring more cache units), this higher cache miss overhead can be compensated by the much lower cache retrieval times (query 2 and 4). Regarding cache maintenance, replacement times are again much higher for source-cache than for meta-cache. As larger, more coarse-grained cache units are retrieved and employed to serve a given query (becoming “recently used”), fitting these units into the available cache space takes more time.

To conclude, meta-cache considerably outperforms source-cache for the three more complex queries due to the smaller granularity of retrieval, and is only slightly slower for the two simpler queries. In any case, the cache maintenance overhead is much smaller for meta-cache. This improved execution and cache maintenance time outweighs the extra overhead incurred during the indexing phase. However, cache misses present a problem for meta-cache and greatly reduce performance. More advanced and fine-tuned replacement policies could be investigated to avoid cache misses, or the available cache space may be increased. Also, the architecture could be extended to reduce the amount of downloaded sources (see future work).

6 Related Work

Our solution for transparent, efficient querying of a large set of small, online RDF sources is based on two pillars: indexing and caching. Below, we elaborate on both.

In other fields, indexing is a well-known technique to optimize data access. In the field of query distribution, metadata indices are employed to divide a query into subqueries and distribute them over datasources containing relevant data. Often, additional information to optimize the query distribution plan is also stored. For example, Quilitz et. al [7] use a service description containing information about found predicates, together with statistical information such as the amount of triples using a specific predicate and certain objects (e.g. starting with letters A to D). In [8], characteristics about the data provider are also kept, such as the data production rate. In [9], full-text indices are used to determine which peers contains particular triples. So-called source-index hierarchies are employed in [10], which enable the identification of query endpoints that can handle combinations of query triple patterns (or “paths”), to reduce the number of local joins. Although we share a common goal, namely identifying relevant datasources, these approaches focus on keeping index information to optimize query distribution. In the context of RDF stores, full-resource indices (i.e. indexing found s/p/o resources and potentially combinations thereof) are often employed in RDF stores, to speed up access to the RDF data (e.g. androjena, HexaStore [11], RDF On the Go [6]). However, as noted in [11] and similar to full-text indices [9], such indices are very memory and computationally intensive, with high update and insertion costs. Therefore, the index structures from these fields do not comply with the requirement discussed in section 3; namely, that a source index in a mobile setting should be lightweight to construct and update, and compact in size.

The goal of client-side caching is to exploit the capabilities of client devices, such as storage space and processing power, to increase performance and scalability of a system [12, 13]. Most existing caching approaches are based on client-server architectures, where all necessary data can be obtained from the server. In traditional data-shipping techniques, clients perform queries locally on data obtained from a server; the data can then be cached for later re-use [12]. In case of a cache miss, the missing tuples (or pages) are obtained by sending their identifiers to the server. In our setting, such caching cannot be directly applied, as no single server exists defining such unique identifiers. In query caching, query results are cached and re-used by future queries, by using query folding techniques [14]. When the cached query results

are not sufficient to answer a new query, a remainder query is generated to obtain the missing data from the server in a fine-grained way. In our approach, there is no possibility to obtain specific non-cached data items. Instead, the corresponding full sources need to be downloaded, defeating the purpose of the remainder query. Comparable to query caching, we group triples in the cache according to the semantics of the cached data. However, instead of relying on posed queries to define these semantics, we exploit the inherent semantics of the cached data.

Ample work has been put in the development of cache replacement functions fine-tuned towards mobile environments. Unlike traditional replacement policies (e.g. relying on temporal locality), such functions utilize semantic locality, where general properties and relations of the data items are exploited. For instance, in [12], cached query results associated with physical locations furthest away from the location of the latest query are removed. In the FAR policy [15], cached units not located in the user's movement direction *and* furthest away from the user are removed. As our query service is not targeted to one single application type (e.g. location-aware systems), we currently rely on the generic LRU strategy to replace cache units. Future work consists of investigating alternative replacement strategies (see next section).

7 Conclusions and Future Work

We have presented a query service for the efficient and transparent querying of large numbers of small online sources. In order to achieve this efficient access, we rely on 1/ indexing and selection of query-relevant sources, based on semantic source metadata, and 2/ caching of often-used downloaded source data. For each component we have realized different variants, taking into account the requirements that exist in a mobile, volatile setting. For the indexing component, our evaluation has shown that significant reduction in query execution time can be reached for SIM2 and SIM3. SIMs storing more meta-data perform better due to increased selectivity, but also cause increased overhead; therefore, a trade-off needs to be made. Regarding the caching component, we found that organizing the cached data around their metadata (i.e. predicate and type information) significantly increases the fine-grainedness of cached data retrieval and overall cache performance. At the same time however, cache misses present a serious overhead for this kind of cache organization.

Future work consists of minimizing the effects of cache misses, by exploring more advanced replacement policies (e.g. location-aware) and investigating source-level replacement in meta-cache. Also, we aim to investigate the effectiveness of replacement policies in different mobile scenarios (potentially selecting suitable ones automatically). Finally, to deal with the limited storage of mobile devices, two-level indices that can be efficiently swapped to persistent storage should be investigated.

Acknowledgement. Sven Casteleyn is supported by an EC Marie Curie Intra-European Fellowship (IEF) for Career Development, FP7-PEOPLE-2009-IEF, N° 254383.

References

- [1] Bellavista, P., Corradi, A., Montanari, R., Toninelli, A.: Context-Aware Semantic Discovery for Next Generation Mobile Systems. *IEEE Communications Magazine* 44(9), 62–71 (2006)
- [2] Reynolds, V., Hausenblas, M., Polleres, A.: Exploiting Linked Open Data for Mobile Augmented Reality. In: *Proc. of W3C Workshop: Augmented Reality On the Web, Spain* (2010)
- [3] Casteleyn, S., Woensel, W.V., Troyer, O.D.: Assisting Mobile Web Users: Client-Side Injection of Context-Sensitive Cues into Websites. In: *Proc. of 12th International Conference on Information Integration and Web-based Applications & Services, France*, pp. 443–450 (2010)
- [4] Woensel, W.V., Casteleyn, S., Troyer, O.D.: A framework for decentralized, context-aware mobile applications using semantic web technology. In: Meersman, R., Herrero, P., Dillon, T. (eds.) *OTM 2009 Workshops*. LNCS, vol. 5872, pp. 88–97. Springer, Heidelberg (2009)
- [5] Melinger, D., Bonna, K., Sharon, M., SantRam, M.: Socialight: A Mobile Social Networking System. In: *Proc. of the 6th International Conference on Ubiquitous Computing, England*, pp. 429–436 (2004)
- [6] Le-phuoc, D., Parreira, J.X., Reynolds, V.: RDF On the Go: An RDF Storage and Query Processor for Mobile Devices. In: *Proc. of 9th International Semantic Web Conference (ISWC 2010), China*, (2010)
- [7] Quilitz, B., Leser, U.: Querying Distributed RDF Data Sources with SPARQL. In: *Proc. of 5th European Semantic Web Conference, Spain* (2008)
- [8] Lynden, S., Kojima, I., Matono, A., Tanimura, Y.: Adaptive Integration of Distributed Semantic Web Data. In: Kikuchi, S., Sachdeva, S., Bhalla, S. (eds.) *DNIS 2010*. LNCS, vol. 5999, pp. 174–193. Springer, Heidelberg (2010)
- [9] Kaoudi, Z., Kyzirakos, K., Koubarakis, M.: SPARQL query optimization on top of dHTs. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) *ISWC 2010, Part I*. LNCS, vol. 6496, pp. 418–435. Springer, Heidelberg (2010)
- [10] Stuckenschmidt, H., Vdovjak, R., Houben, G.J., Broekstra, J.: Towards Distributed Processing of RDF Path Queries. *International Journal of Web Engineering and Technology* 2(2/3), 207–230 (2005)
- [11] Weiss, C., Bernstein, A.: Hexastore: Sextuple Indexing for Semantic Web Data Management. *Proc. of VLDB Endowment* 1(1), 1008–1019 (2008)
- [12] Dar, S., Franklin, M.J., Jónsson, B., Srivastava, D., Tan, M.: Semantic Data Caching and Replacement. In: *Proc. of the 22th Int. Conference on Very Large Data Bases, USA*, pp. 330–341 (1996)
- [13] Jónsson, B., Arinbjarnar, M., Börsson, B., Franklin, M.J., Srivastava, D.: Performance and overhead of semantic cache management. *ACM Trans. Int. Technology* 6(3), 302–331 (2006)
- [14] Ren, Q., Dunham, M.H., Kumar, V.: Semantic Caching and Query Processing. *IEEE Trans. on Knowl. and Data Eng.* 15(1), 192–210 (2003)
- [15] Ren, Q., Dunham, M.H.: Using semantic caching to manage location dependent data in mobile computing. In: *MobiCom 2000: Proc. of the 6th Annual Int. Conference on Mobile Computing and Networking, USA*, pp. 210–221 (2000)

An Automation Support for Creating Configurable Process Models*

Wassim Derguech and Sami Bhiri

Digital Enterprise Research Institute (DERI)
National University of Ireland, Galway
firstname.lastname@deri.org
www.deri.org

Abstract. Configurable process models are constructed via the aggregation of several process models. Manual creation of configurable process models is tedious, time consuming and error prone task. We propose in this paper an automation support for creating these models. The contribution of this paper is a merging algorithm for integrating a set of process variants into a single configurable process model. This integrated process model should (i) subsume the behaviours of all original models, (ii) ensure a trace back of the origin of each element and (iii) derive any of the input models by means of configuration and individualization. Existing solutions either fail in respecting all these requirements or allow for merging only pairs of process models. However, our algorithm allows for merging a set of process models at once. This algorithm has been implemented and tested over a set of different process variants.

Keywords: business process modelling, reuse, merging, configuration.

1 Introduction

Reference process models describe proven practices for a specific industry. They are often aligned with emerging industry-specific and cross-industry standards. We refer the reader to [3] for an overview. One of the scenarios of using reference process models is the reference process model customization [9]. It begins with a reference process model that provides configuration facilities. This model can be configured to specific needs of an enterprise e.g., by refining business rules or enabling/disabling some activities. In this scenario, there is an increasingly important need to assist business analysts in creating the reference model (i.e., the configurable model).

Configurable process models are constructed via the aggregation of several variants of a process model [15]. Such models are considered for example when companies become subject of acquisitions and mergers, in case of improvement of existing business processes or simply when different business analysts define

* This work is funded by the Lion II project supported by Science Foundation Ireland under grant number 08/CE/I1380.

their customized process models for achieving the same business goal. We call these business process models *business process variants*. Since they achieve in essence the same business goal, these variants slightly differ from each other in their structure [11,12]. Therefore, managing these variants can be made easier by handling the common parts just once and not for each variant separately.

Manual creation of configurable process models is tedious, time-consuming and error-prone task. Authors in [14] mention that it took a team of five analysts and 600 man-hour to merge 25% of an end-to-end process model. In this paper, we are proposing an automatic merging method that allows for creating a configurable business process model from a collection of process variants. The algorithm that we propose respects the following requirements:

1. The merged model should allow for the behaviour of all the original models. Traditionally, this operation is manually made by business analysts which comes with the risk that some aspects of the original models are accidentally neglected [6]. With the automation support for merging process variants, this risk can be minimized considerably.
2. Each element of the merged process model should be easily traced back to its original model [14]. A business analyst needs to understand what do the process variants share, what are their differences, etc. This can be made possible if he can trace back from which variant does an element originate.
3. Business analysts should be able to derive one of the input models from the merged process model [14].

The contribution of the paper is an algorithm that allows for merging a collection of business process variants given as input and delivers a configurable business process model. The resulting model should subsume the behaviours of all input models and these original models can be derived by means of configuration and individualization. For more information about the configuration and individualization algorithms, we refer to [5,10,15].

Several methods have been proposed to merge business process variants such as [6,8,14], their main weakness resides in the fact that they allow for merging only pairs of process variants. In order to merge a collection of process variants, they should merge a first pair then they add one variant at a time to the configurable process model. In contrast to existing proposals, our algorithm allows for merging all original process variants at once.

The remainder of the paper is organized as follows: Section 2 introduces our running example and the adopted notation which will be used in the rest of the paper. Section 3 presents the merging algorithm. Section 4 reports on the implementation and evaluation of the algorithm. Section 5 discusses the related work and Section 6 concludes the paper and states our future work.

2 Prerequisites

In this section, we introduce our running example which will be used to explain the steps of the proposed merging algorithm. Then, we present our notation which will be used for defining the merging algorithm.

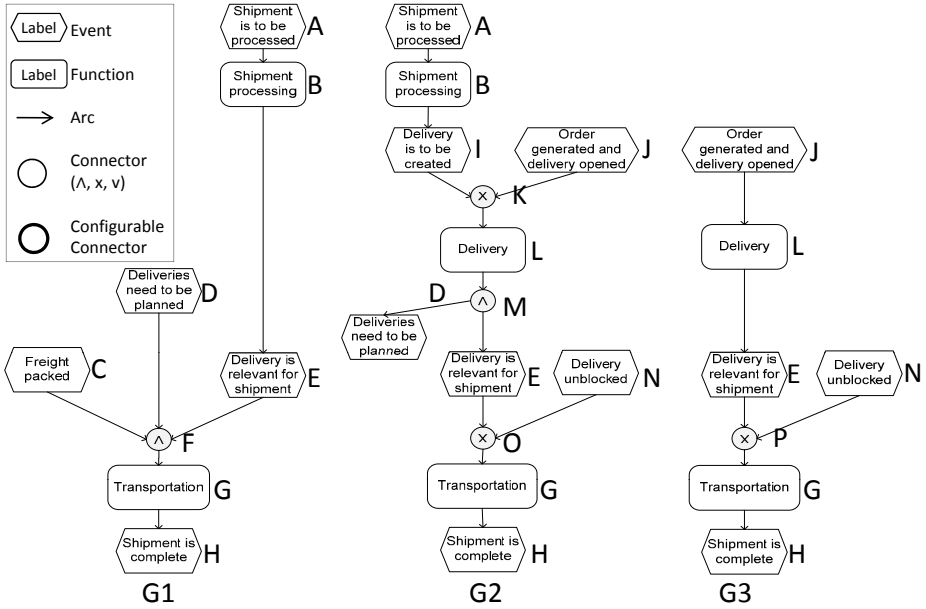


Fig. 1. Three business process variants (adapted from [14])

2.1 Running Example

Several modelling languages have been proposed to represent business process models (e.g., Event-driven Process Chain (EPC), UML Activity Diagrams and Business Process Model and Notation (BPMN)). In this paper we will adopt EPC [13] to illustrate our running example as well as for showing results of the steps of the merging algorithm.

Our running example depicted in Fig. 1 presents three process variants that follow the EPC notation (G1, G2 and G3). These process models describe a shipment process that involve three functions: “*Shipmentprocessing(B)*”, “*Delivery(L)*” and “*Transportation(G)*” (for presentation purposes, each node will be referred by a letter (i.e., A,B,...P) as marked in Fig. 1).

- Function “*B*” appears in G1 and G2 and in both variants it is triggered with the same event (i.e., “*A*”).
- Function “*L*” appears in G2 and G3. In G3 it is triggered with the event “*J*”, however, in G2 it is triggered either by the event “*J*” or “*I*”.
- Function “*G*” appears in all these variants. In G2 and G3 it is triggered either by the event “*E*” or “*N*”, however, in G1 it is triggered by all the events “*C*”, “*D*” and “*E*”.

Next, we will present our notation for describing business process models. This notation will be used for presenting our merging algorithm.

2.2 Notation

Even though we will use EPC for illustrating the results of our merging algorithm, our proposed merger is not exclusively made for this modelling notation. It is still possible to adapt some steps of the algorithm to allow its applicability for other modelling notations. In our work, we represent a business process as a directed graph with annotated nodes and edges. A *business process graph* \mathbf{G} is a triple $\langle WN, RN, E \rangle$ where:

- WN is the set of work nodes: in EPC, work nodes are either function or event nodes. Function nodes are used for representing tasks of the business process. Event nodes are used for presenting preconditions that need to be satisfied to perform a particular function. A work node n of the graph G is a triple $(id_G(n), \lambda_G(n), \tau_G(n))$ where:
 - $id_G(n)$ is a string that represents a unique identifier.
 - $\lambda_G(n)$ is a string that represents the label of the node.
 - $\tau_G(n)$ represents the type of the node (i.e., event or function in EPC).
- RN is the set of routing nodes: in EPC, connectors are the routing nodes. They are used for controlling the flow of tasks that need to be performed for achieving the goal of the business process. A routing node is a tuple $(id_G(n), \lambda_G(n), \tau_G(n), \eta_G(n))$ where:
 - $id_G(n)$ is a string that represents a unique identifier.
 - $\lambda_G(n)$ is a string that indicates if the connector is a join or a split connector.
 - $\tau_G(n)$ is a pair $(PID, Operator)$ where PID represents the process identifier and $Operator$ is its corresponding operator (i.e., \wedge , XOR , \vee). We deliberately mention the process identifier as part of $\tau_G(n)$ because configurable process models are also represented using this notation. Besides, a connector in a configurable process model can have multiple operators, having in mind that each operator can originate from a different process model. This notation will allow for keeping track where a connector originate from. In case of multiple variants of a configurable connector, $\tau_G(n)$ contains a set of pairs $(PID, Operator)$.
 - $\eta_G(n)$ is a flag that indicates whether this node is configurable or not, this applies when G represents a configurable process model.
- E is the set of directed edges which connect nodes from WN and RN . Entries of E are as follows: $(Source, PID, Destination)$ where:
 - $Source \in WN \cup RN$. It represents the source node of the edge.
 - $Destination \in WN \cup RN$. It represents the destination node of the edge.
 - PID represents the set of process identifiers where this edge originates from.

We use $\bullet n = \{m / (m, PID, n) \in G\}$ to denote the set of input nodes of n . We use $n \bullet = \{m / (n, PID, m) \in G\}$ to denote the set of output nodes of n .

3 Merging Business Process Models

For merging business process models, we start by representing them according to our notation. Before merging $G1$, $G2$ and $G3$ of the Fig. 11 we start by a pre-processing step (see Section 3.1). Then we merge and post-process these business process graphs (see Section 3.2). Finally, a reduction step (see Section 3.3) is recommended to ensure that the final result is a reduced configurable process model.

3.1 Pre-processing Business Process Graphs: Resolving Conflicts of Start and End Nodes

So far we have identified conflicts related only to start and end nodes of any process model. In other words, if a node is a start node in a particular model it should remain as a start node in the merged model. This property should be preserved for end nodes as well. We pre-process all the business process graphs that we need to merge before proceeding to the merging step.

The solution that we propose consists of checking all the start/end nodes of all the models that we need to merge and verify if they are start nodes everywhere (i.e., in all the models). If a node appears as a start/end node in some models and not a start/end node in others, then it is considered as a conflict and has to be resolved.

Algorithm 26 resolves the conflict related to start nodes. It starts by detecting nodes that need to be resolved (i.e., line 3) and creates for each one of them a new identifier via the function $RandomNewID()$; (i.e., line 5). Finally, when parsing all the input process graphs, if one of the nodes that need to be resolved appears as a start node in a graph, its identifier is changed by the new one.

Algorithm 1: Detect and resolve conflicts related to start nodes.

Input: ListGraphs LG: List of business process graphs that need to be merged.

```

1 begin
2    $k \leftarrow length(LG)$ ;
3    $NodesToResolve \leftarrow \bigcup_{i=1}^k \{n \in WN_{G_i} / |\bullet n| = 0\} \cap \bigcup_{i=1}^k \{n \in WN_{G_i} / |\bullet n| \geq 1\}$ ;
4   foreach  $Node$  in  $NodesToResolve$  do
5      $NewID \leftarrow RandomNewID()$ ;
6     foreach  $G$  in  $LG$  and  $Node \in WN_G$  and  $|\bullet Node| = 0$  do
7        $id_G(Node) \leftarrow NewID$ ;
8     end
9   end
10 end

```

Let's go back to our running example. We can notice that the work node "Deliveries need to be planned (D)" is a start node in the first process graph (i.e., $G1$ of Fig. 11) and it is not a start node in the second variant (i.e., $G2$ of

Fig. 10. To resolve this conflict, the algorithm operates as follows: it selects all nodes which are start nodes in each of the input models (i.e., $\bigcup_{i=1}^k \{n_{G_i} / |\bullet n_{G_i}| = 0\}$), then selects all nodes which are not start nodes in each of the input models (i.e., $\bigcup_{i=1}^k \{n_{G_i} / |\bullet n_{G_i}| \geq 1\}$) and finally computes their intersection. The result of this intersection is the set of nodes that need to be resolved. In our example the only node that need to be resolved is “*Deliveries need to be planned (D)*”. The resolution of the conflict related to this node consists of generating a new identifier for the node “*D*” that is “*DStart*” and assign this identifier to nodes which appear as start nodes in all the models, in our case only in the first variant.

We proceed similarly to resolve the possible conflicts related to end nodes, to ensure that each end node in a model is an end node in the merged model.

Once the conflicts related to start and end nodes are resolved, we move to the next steps that consist of merging the process graphs and post-processing them which is the aim of the following sections.

3.2 Merging and Post-processing Business Process Graphs

The merging phase consists of a simple set union operation between all the process graphs. The merged process graph (MG) = $\langle WN_{MG}, RN_{MG}, E_{MG} \rangle$ such that (k represents the number of the initial business process graphs):

$$WN_{MG} = \bigcup_{i=1}^k WN_i ; RN_{MG} = \bigcup_{i=1}^k RN_i \text{ and } E_{MG} = \bigcup_{i=1}^k E_i.$$

After merging the business process graphs, we expect obviously that some edges having the same source and destination will appear in E_{MG} . For example, in our running use case, we have in G1 of Fig. 11 an edge between “*Shipment is to be processed (A)*” and “*Shipment processing (B)*” presented by (A,(1),B) in E_{MG} . The same edge appear in the second variant (i.e., G2 of Fig. 11) which is presented by (A,(2),B) in E_{MG} . For this, we need to merge these two edges into a single edge having as *PID* a set of both process identifiers where it originates from (i.e., (A,(1,2),B)).

The resulting *MG* needs some post-processing in order to be well formed according to some requirements imposed by the modelling notation. Recall that in this paper we consider EPC as a modelling notation. A well formed EPC should satisfy several requirements. We assume that the input models are well formed, i.e., respect all the EPC requirements [13]. After the merging operation two requirements are violated. These requirements are:

1. for each $n \in WN$: $|\bullet n| \leq 1$. This requirement means that each work node must have at most one input.
2. for each $n \in WN$: $|n \bullet| \leq 1$. This requirement means that each work node must have at most one output.

Changes that need to be applied to have a well formed model need some additional features that are provided by C-EPC [15,16]. C-EPC stands for Configurable EPC. It is an extended version of EPC where some *connectors* can be

marked as configurable. A configurable connector can be configured by reducing its incoming branches (in the case of a join) or its outgoing branches (in the case of a split) [14]. The result will be a regular connector with a reduced number of incoming or outgoing branches. In addition, the type of a configurable OR can be restricted to a regular XOR or AND.

Next, we will use configurable connectors in order to respect the requirements previously mentioned (i.e., $\forall n \in WN, |\bullet n| \leq 1$ and $\forall n \in WN, |n \bullet| \leq 1$).

Each Work Node Has a Single Entry and a Single Exit. To ensure that each work node has a single entry, Algorithm 2 operates as follows: if a work node has more than one input, we create a configurable connector (XOR-Join) that becomes the new destination of all input edges of that work node (i.e., lines 3 to 12). Finally, it creates a new edge from the new configurable connector to the work node that previously had more than one entry (i.e., line 13).

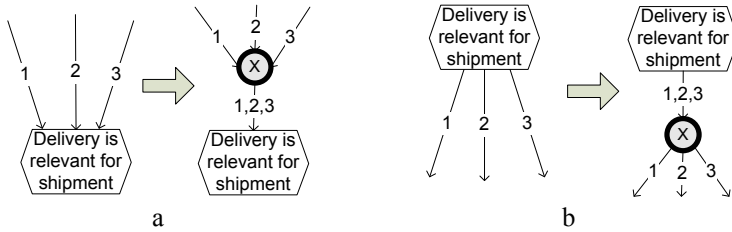


Fig. 2. A work node has a single entry and a single exit

Fig. 2a illustrates an example of this transformation. The left hand side of this figure depicts three input edges for the event “*Delivery is relevant for shipment*” which has been changed in the right hand side of this figure by inserting a configurable XOR connector and an edge from this connector to “*Delivery is relevant for shipment*” that has the label of all previous edges (i.e., “1,2,3”).

We proceed similarly to ensure that each work node has a single exit. Fig. 2b illustrates an example of this transformation. The left hand side of this figure depicts three output edges for the event “*Delivery is relevant for shipment*” which has been changed in the right hand side of this figure by inserting a configurable XOR connector and an edge from “*Delivery is relevant for shipment*” to this connector that has the label of all previous edges (i.e., “1,2,3”).

At this level, the merged process model is completely constructed and Fig. 3a depicts the resulting model. However, during this post-processing step, we have inserted several configurable connectors. This may lead to the appearance of several connector chains. Indeed, in Fig. 3a, the rounded regions represent various connector chains. In the following, we will show how we reduce these connector chains in order to obtain a reduced configurable process model such as depicted in Fig. 3b.

Algorithm 2: Single Entry

Input: Graph G : A graph that represents a configurable process graph.

```

1 begin
2   foreach Node in  $WN_G$  and  $|\bullet Node| > 1$  do
3      $IDs \leftarrow \emptyset$ ;
4     CreateNewCXOR(CXOR);
5      $\lambda(CXOR) \leftarrow join$ ;
6      $\tau(CXOR) \leftarrow (id(G), XOR)$ ;
7      $\eta(CXOR) \leftarrow true$ ;
8     foreach  $\{(Source, PID, Destination) \in E_G / Destination = Node\}$  do
9        $E_G \leftarrow E_G \setminus \{(Source, PID, Destination)\}$ ;
10       $E_G \leftarrow E_G \cup (Source, PID, CXOR)$ ;
11       $IDs \leftarrow IDs \cup \{PID\}$ ;
12    end
13     $E_G \leftarrow E_G \cup \{(CXOR, IDs, Node)\}$ ;
14     $RN_G \leftarrow RN_G \cup \{CXOR\}$ ;
15  end
16 end

```

3.3 Reduction of the Configurable Business Process Graph

In this section, we present two simplification rules that help for reducing connector chains. These rules should reduce the business process graph while preserving its behaviour. Reducing a business process graph consists of deleting some routing nodes which are not mandatory. These reduction rules are: (i) merging consecutive split/join connectors and (ii) removing trivial connectors.

Merge Consecutive Connectors. Algorithm 3, merges any consecutive split/join connectors into a single connector. It starts by parsing all the edges of G . If the source and the destination of an edge are two connectors of the same type (i.e., join or split) (i.e., line 3), then the algorithm fetches all the adjacent connectors having that type in order to create a set of connectors that have to be reduced (i.e., lines 5 to 7). The reduction of this set of connectors is made by creating a new configurable connector that replaces them. If one of the connectors that need to be reduced is either an AND or an OR (i.e., line 11), then the new connector must be a configurable OR keeping trace back to the original operator with the identifier of the process where it originates from (i.e., line 14) otherwise it is a configurable XOR (i.e., line 9). Then, the algorithm continues to parse the remaining edges to detect input/output edges of the current connectors in order to link them to the new connector (i.e., lines 18 to 23). Line 25 of algorithm 3 allows for merging edges having the same source and destination.

In Fig. 4a (as it appears in our use case), connectors F , P , O and R are four consecutive join connectors which are merged into Z in the Fig. 4b. In Fig. 4b there are three edges from U to Z with labels “1”, “2” and “3” which

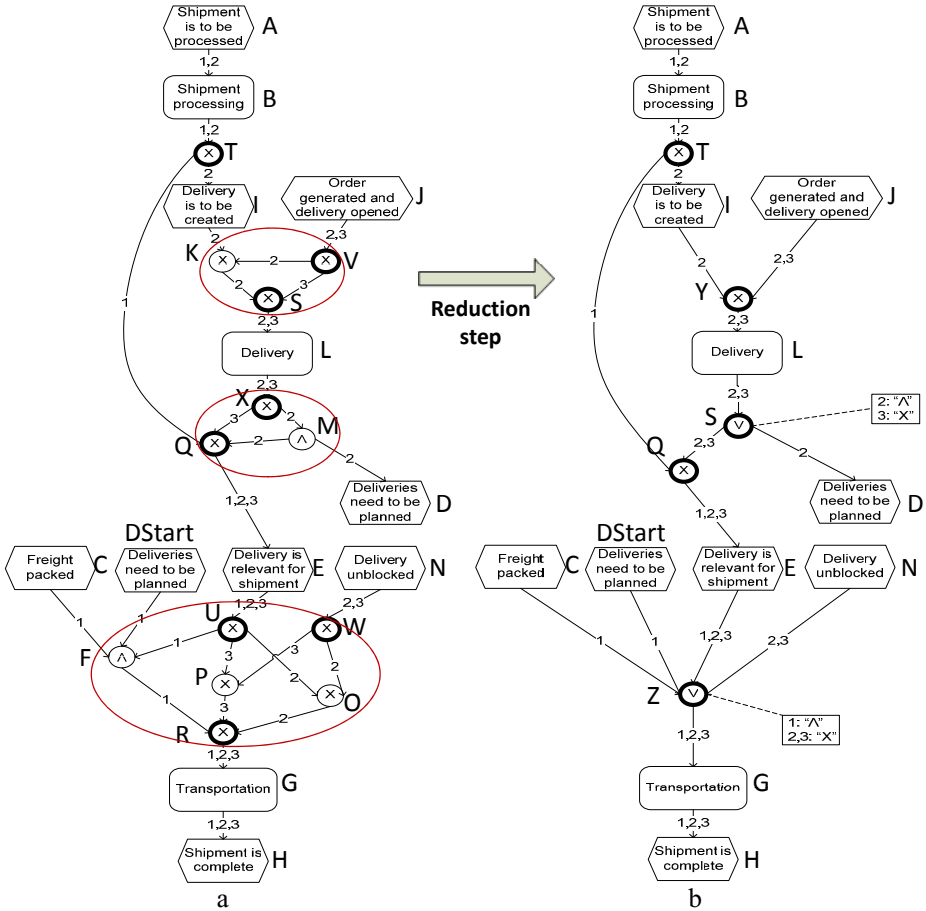


Fig. 3. Configurable business process model before and after the reduction step

respectively originate from (Fig. 4.a) the edge from *U* to *F*, the edge from *U* to *O* and the edge from *U* to *P*. These three edges are merged into a single edge with the label “1,2,3” in the Fig. 4.c.

Remove Trivial Connectors. A *trivial connector* is a connector that has only one input and one output edge. They do not provide any useful routing information. Thus, they can be removed without altering the process behaviour. In our running example, Fig. 4.c depicts two trivial connectors (i.e., *U* and *W*) which are to be removed. Fig. 3.b depicts the resulting optimal configurable process model.

In the worst-case, the complexity of this merging algorithm is $O(|S| * |N|^2)$. Where $|S|$ represents the number input business process models and $|N|$ represents the is the number of nodes of the largest model. This is the complexity of the algorithm that dominates the complexity of the other steps. As the merging

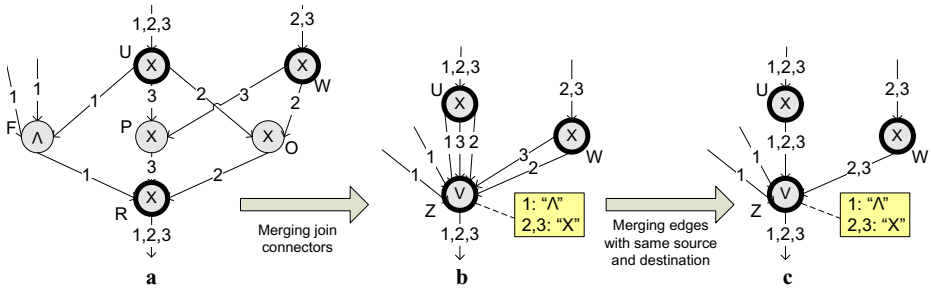


Fig. 4. Reducing a business process graph by merging consecutive join connectors

step is simple union operation that has a linear complexity depending on the number of input business process models. Algorithm 2 has a complexity of $O(|N^2|)$ where N presents the number of all the nodes of the merged model. Algorithm 3 has the same complexity.

The output process model of the merging algorithm subsumes the behaviours of all the input process models. Indeed, all the work nodes of the input models are preserved as our algorithm does not remove any of them. In addition, the reduction step removes only connectors that do not alter the behaviour of the model. A detailed proof of this feature is not in the scope of this paper.

4 Evaluation

We have implemented the approach described in this paper as java program. It merges a set of selected EPCs in four phases as explained in this paper (Pre-processing, Merging, Post-processing and Reduction).

We have manually created a test-bed for evaluating our merger. The process variants that we used in our experiment are those presented in 5. They were subject of a case study in 7 in which techniques for managing configurable process models were extensively tested on a real-world scenario. The process models of this case study are four selected processes out of five most executed registration processes in the civil affairs department of Dutch municipalities 5.

The process variants that we considered in our evaluation are initially available in Protos¹. These models represent four registration processes: (1) Acknowledging an unborn child, (2) Registering a newborn, (3) Marriage and (4) Issuing a death certificate. Each process has five process variants. Consequently, a total of $5 \times 4 = 20$ process models were considered in our work (similar to the case study 7). We have manually translated these models into EPC and used our tool for merging them in order to create configurable process models for each process.

One of the benefits of configurable process models is eliminating redundancy of common practices within a business process. In fact for each process, we can notice that several process steps are duplicated in each variant. Table 11

¹ Protos is part of Pallas Athena's BPM toolset BPM|one.

Algorithm 3: Merge consecutive connectors**Input:** Graph G : A graph that represents a configurable process model.

```

1 begin
2   foreach ( $Src1, PID1, Dest1$ ) in  $E_G / \{Src1, Dest1\} \subset RN_G$  and
3     ( $\lambda(Src1) = \lambda(Dest1)$ ) do
4        $ToBeReduced \leftarrow \{Src1, Dest1\}$ ;
5       foreach ( $Src2, PID2, Dest2$ )  $\in E_G / Src2$  or  $Dest2 \in ToBeReduced$ 
6         and ( $\lambda(Src2) = \lambda(Dest2) = \lambda(Src1)$ ) do
7            $ToBeReduced \leftarrow \{Src2, Dest2\}$ ;
8       end
9       CreateNewConnector(ConfigurableConnector);
10      Operator = "XOR";
11      foreach  $Connector \in ToBeReduced$  do
12        if  $\tau(Connector) = (PID, "AND")$  or  $\tau(Connector) = (PID, "OR")$ 
13          then
14            Operator = "OR";
15          end
16           $\tau(ConfigurableConnector) \leftarrow \tau(Connector)$ ;
17      end
18       $\tau(ConfigurableConnector) \leftarrow \{(id(G), Operator)\}$ ;
19       $\eta(ConfigurableConnector) \leftarrow true$ ;
20      foreach ( $Src, PID, Dest$ )  $\in E_G / Src$  or  $Dest \in ToBeReduced$  do
21         $E_G \leftarrow E_G \setminus (Src, PID, Dest)$ ;
22        if  $Dest \in ToBeReduced$  and  $Src \notin ToBeReduced$  then
23           $E_G \leftarrow (Src, PID, ConfigurableConnector)$ ;
24        end
25      end
26    end
27  end

```

summarizes number of EPC nodes for each process before and after the creation of its corresponding configurable process model.

Table 1 shows the size of the input and output models (size in terms of number of EPC nodes). Column one states the four processes considered here (P1: Acknowledging an unborn child, P2: Registering a newborn, P3: Marriage and P4: Issuing a death certificate). Column two shows the size of the input models, entries of this column present the sum of the number of nodes of each variant as it is mentioned between parenthesis. Columns three and four show the size of the output models before and after the reduction step of our algorithm which represent the size of the constructed configurable process model. The percentage value between parenthesis shows the compression rate gained from the creation

Table 1. Results of merging registration processes of Dutch municipalities

	Input size	Output size before reduction	output size after reduction	Execution Time (ms)
P1	190 (29+56+52+29+24)	131 (31%)	71 (62%)	157
P2	347 (63+84+73+57+70)	276 (20%)	180 (48%)	235
P3	507 (76+127+127+114+63)	298 (41%)	214 (57%)	407
P4	355 (56+111+91+67+30)	266 (25%)	160 (54%)	282

of the configurable process models. Column five shows the execution time in milliseconds needed for merging the input process models.

We can notice that we can gain around 50% in terms of space for storing several process variants. Besides this space gain, we can see that in few milliseconds a set of five process variants can be automatically merged which would take much longer for a business analyst to perform such task manually.

5 Related Work

For merging process variants, [4] proposes a formalism to detect equivalent business process models based on the detection of equivalent fragments contained in these models. This work is presented in the context of detection and resolution of version conflicts. Authors here refer to their existing tool support for model merging in IBM WebSphere Business Modeler [8]. The merge procedure defined here is not intended to be fully automated, it is rather developed for reducing the number of false-positive differences and conflicts in models management.

[6] defines an approach exclusively intended for merging models following the EPC notation. This approach consists first of transforming EPCs into a so called abstraction of EPCs, namely function graphs. The second step is the combination of these function graphs by means of set union operations. Finally, they transform back the combined function graph into an EPC. The object in their approach is not to create a configurable EPC, there are no configurable connectors introduced which would allow for extracting one of the original models.

Both approaches discussed previously [4,6] do not allow for the second nor the third requirements of Section 1. Indeed, the generated merged models do not allow to trace back where an element of the model originates from. As well as they do not provide any possibility to configure the obtained model in order to derive one of the input models. However, the proposed algorithm has a complexity of $O(|N|^3)$ for merging only one pair of process models where $|N|$ is the number of nodes of the largest model.

As we share the same requirements of Section 1, authors of [14] propose a technique that allows for the three of them. The proposed technique starts by computing a similarity measure between nodes of pairs of process models. Then, given a mapping between different elements of the original models, they propose a merge operator that computes maximum common regions (MCR) and then links elements of the second models, which are not in the MCR, to the MCR of

the first model. Similar to our approach, they use arcs' annotations to allow for tracing back the origin of an element.

All the proposed approaches [4,6,14] are intended to merge pairs of process variants. This means that if a business analyst need to merge several process variants, he has to merge a first pair and then add the other variants one by one. However, our approach allows for merging process models at once.

6 Conclusion and Future Work

In this paper, we have presented a novel algorithm that allows for merging a collection of business process models in order to create a configurable process model. Even though we used EPC to illustrate our running example and results of the steps of our algorithm, our approach is not exclusively made for EPCs. Indeed, we represent a process variant as a business process graph according to the notation shown in Section 2.2 which has been used throughout the steps of the merging algorithm.

Our algorithm ensures that the resulting configurable model includes the behaviours of the original business process variant by considering work nodes with identical labels and preserving the status of start and end nodes.

Current approaches, discussed in the related work section, provide algorithms for merging pairs of process models. Unlike these solutions, our algorithm allows for merging a collection of business process variants at once.

A strong assumption that we need to break in the future work concerns the fact that our approach allows for merging work nodes with identical labels whereas the approach of [14] supports approximate matching. As part of our future work, we plan to allow for such feature.

In terms of the evaluation of our algorithm, we have manually checked the correctness of the generated configurable process models by comparing the results to those presented in [5]. We intend to define or apply a methodology of verification that allows for checking if the configuration process can lead to correct process models such as discussed in [12].

Another aspect that we plan to evaluate in our work is the scalability. In fact, the process models that we were merging in our work contain between 29 and 127 nodes. We plan to generate a another test-bed for our work in order to conduct tests with large models to assess the scalability of the merging tool.

References

1. van der Aalst, W.M.P., Dumas, M., Gottschalk, F., ter Hofstede, A.H.M., Rosa, M.L., Mendling, J.: Preserving correctness during business process model configuration. *Formal Asp. Comput.* 22(3-4), 459–482 (2010)
2. van der Aalst, W.M.P., Lohmann, N., Rosa, M.L., Xu, J.: Correctness ensuring process configuration: An approach based on partner synthesis. In: Hull, R., Mendling, J., Tai, S. (eds.) *BPM 2010*. LNCS, vol. 6336, pp. 95–111. Springer, Heidelberg (2010)

3. Fettke, P., Loos, P., Zwicker, J.: Business process reference models: Survey and classification. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 469–483. Springer, Heidelberg (2006)
4. Gerth, C., Luckey, M., Küster, J.M., Engels, G.: Detection of semantically equivalent fragments for business process model change management. In: IEEE SCC, pp. 57–64. IEEE Computer Society, Los Alamitos (2010)
5. Gottschalk, F.: Configurable Process Models. Ph.D. thesis, Eindhoven University of Technology, Eindhoven, Netherlands (December 2009)
6. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H.: Merging event-driven process chains. In: Chung, S. (ed.) OTM 2008, Part I. LNCS, vol. 5331, pp. 418–426. Springer, Heidelberg (2008)
7. Gottschalk, F., Wagemakers, T.A.C., Jansen-Vullers, M.H., van der Aalst, W.M.P., Rosa, M.L.: Configurable process models: Experiences from a municipality case study. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 486–500. Springer, Heidelberg (2009)
8. Küster, J.M., Gerth, C., Förster, A., Engels, G.: A tool for process merging in business-driven development. In: CAiSE Forum. CEUR Workshop Proceedings, vol. 344, pp. 89–92 (2008), CEUR-WS.org
9. Küster, J.M., Koehler, J., Ryndina, K.: Improving business process models with reference models in business-driven development. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 35–44. Springer, Heidelberg (2006)
10. La Rosa, M.: Managing Variability in Process-Aware Information Systems. Ph.D. thesis, Queensland University of Technology, Brisbane, Australia (April 2009)
11. Li, C., Reichert, M., Wombacher, A.: Discovering reference models by mining process variants using a heuristic approach. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 344–362. Springer, Heidelberg (2009)
12. Li, C., Reichert, M., Wombacher, A.: The minadept clustering approach for discovering reference process models out of process variants. *Int. J. Cooperative Inf. Syst.* 19(3-4), 159–203 (2010)
13. Mendling, J., Nüttgens, M.: Epc syntax validation with xml schema languages. In: EPK, GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, pp. 19–30 (2003)
14. Rosa, M.L., Dumas, M., Uba, R., Dijkman, R.M.: Merging business process models. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6426, pp. 96–113. Springer, Heidelberg (2010)
15. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. *Inf. Syst.* 32(1), 1–23 (2007)
16. Weber, B., Mendling, J., Reichert, M.: Flexibility in process-aware information systems (proflex) workshop report. In: WETICE, pp. 269–270. IEEE Computer Society, Los Alamitos (2006)

Graph-Based Named Entity Linking with Wikipedia

Ben Hachey^{1,2}, Will Radford^{2,3}, and James R. Curran^{2,3}

¹ Department of Computing
Macquarie University
NSW 2109, Australia

ben.hachey@mq.edu.au

² Capital Markets CRC
55 Harrington Street
NSW 2000, Australia

³ School of Information Technologies
University of Sydney
NSW 2006, Australia

{wradford, james}@it.usyd.edu.au

Abstract. Named entity linking (NEL) grounds entity mentions to their corresponding Wikipedia article. State-of-the-art supervised NEL systems use features over the rich Wikipedia document and link-graph structure.

Graph-based measures have been effective over WordNet for word sense disambiguation (WSD). We draw parallels between NEL and WSD, motivating our unsupervised NEL approach that exploits the Wikipedia article and category link graphs. Our system achieves 85.5% accuracy on the TAC 2010 shared task — competitive with the best supervised and unsupervised systems.

Keywords: web intelligence, text mining, integration, entity resolution, Wikipedia.

1 Introduction

Named entity linking (NEL) grounds mentions of entities in text to a central knowledge base (KB). Before Wikipedia, open-domain NEL lacked the requisite wide-coverage KB [26]. For example, we can ground: *David Murray recruited Amidu Berry from Positive Black Soul* to Wikipedia articles for David Murray (saxophonist) and Positive Black Soul and mark that *Amidu Berry* does not appear in Wikipedia. NEL systems have begun to exploit Wikipedia’s rich structure: category data [5,6] and infobox data [30,19] have complemented traditional string matching and context similarity approaches; link probabilities conditioned on mentions have been calculated across Wikipedia [23]. State-of-the-art supervised approaches [31,9,18] learn to rank entity candidates.

NEL is very similar to word sense disambiguation (WSD), with Wikipedia articles playing the role of WordNet [11] synsets. Yet the connection with WSD has not been exploited by existing approaches to NEL. Unlike WSD, NEL does not assume the KB is complete, requiring entity mentions without KB entries to be marked as NIL.

Both Wikipedia and WordNet provide links between KB entries: as hyperlinks to other articles and categories in Wikipedia; and as semantic relations, e.g. hypernym and

meronym, in WordNet. [24,25] exploited the graph structure of WordNet for unsupervised WSD rivalling supervised approaches.

In this paper, we adapt these successful approaches for WSD to the NEL task by exploiting Wikipedia’s link graph structure, where articles and categories are nodes. The nature and distribution of Wikipedia and WordNet links are very different, and so there is no guarantee WSD approaches will work for NEL.

We use local subgraphs containing the candidate Wikipedia articles for all entity mentions in the document. We include nodes for any intervening article and/or category on short paths connecting entity mention nodes. We explore subgraph selection parameters including inter-article versus category links, maximum path length, and link directionality. We compare two graph measures: Degree centrality [24] and PageRank [4] for ranking candidates. The graph scores are combined with cosine similarity calculated from the paragraph around the entity mention and the candidate Wikipedia article.

We follow the standard Text Analysis Conference (TAC) NEL evaluation methodology. Our best graph-based model achieves 85.5% accuracy on the TAC 2010 shared task. This is competitive to the top-reported TAC 2010 supervised (86.8%) and unsupervised (85.8%) systems. This demonstrates that NEL is indeed similar to WSD and can be performed using graph-based approaches.

2 Background

NEL is similar to the widely-studied problem of word sense disambiguation (WSD), where the goal is to identify the sense of a word given the context. For example, the word *bank* in the sentence *I put money in the bank* is more likely to refer to a financial institution than a river side. WSD is often performed with respect to WordNet [11], a lexical database that maps words to synonym sets (synsets).

In NEL, the same entity can similarly be referred to using different mentions strings; and a single mention string can ambiguously refer to multiple entities. But, due to the lack of a comparably comprehensive *sense* inventory for entities, most previous work focuses on identifying and characterising entity mentions [14,29], or on clustering mentions within documents [15,28] and across documents [18].

Recently, Wikipedia has emerged as a wide-coverage KB of collective knowledge about notable entities, leading to exploration of NEL over arbitrary named entity types [5,6]. Unfortunately, previous evaluations have focused on final accuracy only [20]. Here, we break NEL into three components for systematic comparison: *extraction* of entities and their contexts; *search* generates candidate entity nodes; and *disambiguation* selects the best candidate or NIL. We explore search and disambiguation independently, holding the other components constant to isolate their effect.

Search for WSD assumes that WordNet is a complete lexical resource and consists of a lexical lookup to find the possible synsets for a given word. The same approach is taken in a number of Wikipedia linking approaches [22,23,17,12]. However, this does not provide a mechanism for dealing with entities that are not present in the database. Search for NEL requires a noisier candidate generation process, often using fuzzy matching to improve recall [30,18]. Finally, NEL does not assume the KB is complete, requiring entity mentions without KB entries to be marked as NIL [20].

Table 1. Comparison of TAC data sets

	TAC 2009 test		TAC 2010 train		TAC 2010 test	
N	3,904		1,500		2,250	
KB	1,675	(43%)	1,074	(72%)	1,020	(45%)
NIL	2,229	(57%)	426	(28%)	1,230	(55%)
PER	627	(16%)	500	(33%)	751	(33%)
ORG	2710	(69%)	500	(33%)	750	(33%)
GPE	567	(15%)	500	(33%)	749	(33%)
News	3904	(100%)	783	(52%)	1500	(67%)
Web	0	(0%)	717	(48%)	750	(33%)

Simple disambiguators based on cosine similarity between mention contexts and article text were highly successful in NEL evaluations [30]. However, other successful systems have exploited Wikipedia’s rich structure: Bunescu and Paşca and Cucerzan use page categories as features [5,6]; Dredze et al. use key-value information from infoboxes [9]; and Milne and Witten calculate probabilities of pages given aliases across Wikipedia [23]. Simple features based on common links between articles have also been used recently [17,10,18,27]. However, the graph-based techniques we explore here have not been used for NEL disambiguation.

Further parallels between NEL and WSD can be drawn in terms of their graph structure. WordNet includes links between synsets that represent semantic relations (e.g., hyponymy, meronymy, antonymy). This graph has been used successfully to incorporate global information into unsupervised WSD rivalling supervised approaches (Section 6). While WordNet and Wikipedia have been found to have similar small-world graph structures [13], the inter-article links in Wikipedia are unconstrained in terms of the relation types they represent.

Based on this comparison, we explore whether successful graph-based approaches from the WSD literature [24,25] are appropriate for NEL with respect to Wikipedia. We also report results for implementations of seminal approaches from the literature [5,6], providing a comparison to more recent approaches for the first time.

3 Evaluation Data and Methodology

The Text Analysis Conference Knowledge Base Population (TAC-KBP) shared tasks have established common datasets that emphasise ambiguous queries, and formalise NIL linking for queries not referring to a KB node [20]. TAC queries consist of an entity mention string (e.g., *Mr. Murray*) and a source document containing it. The gold standard is a reference to a TAC KB node or NIL if there is no corresponding node in the KB. The data sets are summarised in Table 1. There are several notable differences. First, TAC 2010 training data is highly skewed towards non-NIL queries at 72%. Second, the TAC 2009 data is highly skewed towards ORG entities (69%). Third, the TAC 2010 data sets include web documents as well as newswire. We use the TAC 2010 training data as our training set, the TAC 2009 data as our development set (Section 4 and 5), and the TAC 2010 test data as our final test set (Section 7).

Table 2. Notation for searcher analysis measures

N	Number of queries in data set
\mathcal{G}	Gold standard annotations for data set ($ \mathcal{G} = N$)
\mathcal{G}_i	Gold standard for query i (KB ID or NIL)
\mathcal{C}	Candidate sets from system output ($ \mathcal{C} = N$)
\mathcal{C}_i	Candidate set for query i
$\mathcal{C}_{i,j}$	Candidate at rank j for query i (where $\mathcal{C}_i \neq \emptyset$)

The TAC KB used for all experiments is derived from articles in the October 2008 Wikipedia dump that have infoboxes. It includes approximately 200,000 PER nodes, 200,000 GPE nodes, 60,000 ORG nodes and more than 300,000 miscellaneous/non-entity nodes. We also use a more recent Wikipedia dump (29 July 2010) for extracting aliases (e.g., titles of redirect pages, link anchor text) and building graphs. This contains 3,398,404 articles. After disambiguation, any articles that can not be mapped to the subset of articles in the TAC KB are marked as NIL.

We use the following evaluation measures, defined using the notation in Table 2. The first is the official TAC measure for evaluation of end-to-end systems. TAC also reports KB accuracy (A_C) and NIL accuracy (A_\emptyset), which are actually recall scores for non-NIL and NIL respectively. The remaining measures are used here to analyse performance of the different search strategies for generating candidate entities.

Accuracy: percentage of correctly linked queries (i.e., queries where the top-ranked candidate/NIL is the same as the gold standard annotation).

$$A = \frac{|\{\mathcal{C}_{i,0} | \mathcal{C}_{i,0} = \mathcal{G}_i\}|}{N} \quad (1)$$

Candidate Count: mean cardinality of the candidate sets. Fewer candidates mean easier disambiguation.

$$\langle C \rangle = \frac{\sum_i |\mathcal{C}_i|}{N} \quad (2)$$

Candidate Precision: mean percentage of non-empty candidate sets containing the correct entity.

$$P_C = \frac{|\{\mathcal{C}_i | \mathcal{C}_i \neq \emptyset \wedge \mathcal{G}_i \in \mathcal{C}_i\}|}{|\{\mathcal{C}_i | \mathcal{C}_i \neq \emptyset\}|} \quad (3)$$

Candidate Recall: mean percentage of KB queries where the candidate set includes the correct candidate.

$$R_C = \frac{|\{\mathcal{C}_i | \mathcal{G}_i \neq \text{NIL} \wedge \mathcal{G}_i \in \mathcal{C}_i\}|}{|\{\mathcal{G}_i | \mathcal{G}_i \neq \text{NIL}\}|} \quad (4)$$

Nil Precision: mean percentage of NIL queries with *correctly empty* candidate sets.

$$P_\emptyset = \frac{|\{\mathcal{C}_i | \mathcal{C}_i = \emptyset \wedge \mathcal{G}_i = \text{NIL}\}|}{|\{\mathcal{C}_i | \mathcal{C}_i = \emptyset\}|} \quad (5)$$

Table 3. High-precision search (TAC 2009 data).

Alias Source	$\langle C \rangle$	P_C^∞	R_C^∞	P_\emptyset	R_\emptyset
Title+Redirect	30.0	83.8	58.6	76.6	93.9
– Hatnote	18.9	89.0	39.3	68.7	97.5

Nil Recall: mean percentage of NIL queries for which the candidate set is empty. A high NilRecall rate is valuable because it is difficult for disambiguators to determine whether queries are NIL-linked when candidates are returned.

$$R_\emptyset = \frac{|\{\mathcal{C}_i | \mathcal{G}_i = \text{NIL} \wedge \mathcal{C}_i = \emptyset\}|}{|\{\mathcal{G}_i | \mathcal{G}_i = \text{NIL}\}|} \quad (6)$$

Extraction. The extraction component for preprocessing query source documents is held constant for the experiments here. We first extract article text, discarding markup and non-visible content if they are formatted using a markup language. The C&C tagger [7] is used to extract named entity mentions from the text. Finally, coreference chains are formed using an implementation of a high-precision algorithm from Cucerzan [6]. This assumes that longer mentions (e.g., *David Murray*) are generally more specific than shorter mentions (e.g., *Murray*) and will thus be easier to disambiguate. A match is accepted when the short mention is the beginning or end of the long mention and the long mention has no more than three additional tokens. Mentions entirely in uppercase may be acronyms (e.g., *DM*). In this case, a match is accepted when the acronym letters correspond to the token-initial characters of a longer mention.

4 Search Experiments

Table 3 contains analysis of a search strategy that is tuned for the highest possible precision at the expense of recall. The first row corresponds to exact matching against Wikipedia article titles and redirect page titles. The second row uses the same search but removes any results that have hatnote templates — i.e., a Wikipedia link to another article or disambiguation page describing entities with the same name. Removing these pages results in a 5 point increase in candidate precision at the cost of 19 points candidate recall. The imbalanced effect is due to the fact that pages with hatnote templates are more popular than their non-hatnoted counterparts.

This strategy is designed for backoff approaches, where preference is given to high-precision matches before resorting to noisier approaches tuned for recall. It is also useful for identifying minimally ambiguous entities in a document, which we use as reliable context articles in building local networks for graph-based disambiguation. All graph results in the following sections use the hatnote-filtered search for this purpose. This is comparable to the approach in [18], which only uses context entity mentions that are close to the query, have candidate page probability in Wikipedia of ≥ 0.01 given the mention string, and frequency in Wikipedia ≥ 4 .

Analysis of Implemented Searchers. Table 4 contains search results for our implementations of prominent approaches from the literature on the TAC 2009 data. The first row

Table 4. Performance of searchers from the literature

Searcher	$\langle C \rangle$	P_C^∞	R_C^∞	P_\emptyset	R_\emptyset
Bunescu and Paşca	3.60	56.3	77.0	86.6	62.7
Cucerzan	2.42	59.7	79.8	88.4	66.0
Varma et al.	2.99	59.8	81.2	90.9	66.4
Tuned backoff	3.84	58.2	87.3	92.9	57.9

Table 5. Effect of searchers on graph-based reranking

Searcher	A_C	A_\emptyset	A
Cucerzan	72.7	83.4	78.8
Backoff	74.4	80.5	77.9

corresponds to our Bunescu and Paşca [5] searcher, which uses exact matching over Wikipedia page titles, titles of redirect pages and titles of disambiguation pages. The second row corresponds to our Cucerzan [6] searcher, which forms coreference chains (Section 3) creates a query containing all mentions in the chain. The query is searched using an exact-match lookup against article titles, redirect page titles, disambiguation page titles and bold terms from article first paragraphs. The third row corresponds to our Varma et al. [30] searcher, which replaces acronyms with full-forms where possible and employs a backoff search strategy that favours high-precision matching against page titles that map to the KB over alias search. Search includes exact match against article, redirect and disambiguation titles as well as disambiguation hatnotes and bold terms in the first paragraph of an article. Finally, the last row corresponds to a high recall searcher, which is discussed in more detail below.

The implemented Cucerzan and Varma et al. searchers are clearly the winners here. They both achieve candidate precision of approximately 60% at candidate recall of 80%. This suggests that coreference and acronym handling are important. In terms of candidate count, the Cucerzan searcher performs slightly better. It returns a candidate set size that, on average, contains 0.57 fewer items. This corresponds to a reduction in ambiguity of 19% with respect to the Varma et al. searcher.

The last row of Table 4 corresponds to a backoff approach that aims to further increase candidate recall without sacrificing too much in terms of ambiguity. This first performs a hatnote-filtered high-precision match (Table 3). If this produces no results, it resorts to exact match over all Wikipedia alias sources (i.e., article titles, redirect titles, link anchor text, apposition-stripped titles, bold words in article first paragraphs, titles of pages with hatnotes, disambiguation titles, bold disambiguation terms, and titles of redirects pointing to disambiguation pages). Finally, it resorts to character n-gram match. Based on development experiments (TAC 2009 data), n-gram matches are only accepted if they have a Solr score \geq three and an edit distance \geq 0.7 times the number of characters in the term. The last row in Table 4 shows that this increases candidate recall by 6 but also results in the highest ambiguity with a candidate count of 3.84.

Effect of Searchers on Disambiguation. Table 5 contains a comparison of high recall (Backoff) and high precision (Cucerzan) searchers for graph-based reranking of cosine scores on the TAC 2009 data. The high-precision searcher is 1.1 points better in overall

accuracy. In the rest of the paper, we use this searcher to generate candidates for the Cosine, Milne and Witten, Cucerzan and graph-based disambiguation approaches. This serves to focus the remaining experiments on the relative effect of these disambiguators given the same search candidates as input.

5 Existing Approaches to Disambiguation

A number of disambiguators from the literature are used to benchmark our graph-based approaches. Bunescu and Paşca [5] and Cucerzan [6] are the first reported NEL approaches. Milne and Witten [23] is the seminal system from a closely task that treats Wikipedia as a complete lexical resource and formalises linking as a WSD-style task. Since this system does not have a search phase as defined here, we use the Cucerzan searcher for comparability to that system and our graph-based approaches. Finally, Varma et al. [30] and Lehmann et al. [18] are included since they were the top systems at TAC 2009 and TAC 2010 respectively.

The Bunescu and Paşca disambiguator uses a Support Vector Machine (SVM) ranking model, using the SVM^{light} toolkit¹. Two types of features are used. The first is the real-valued cosine similarity between the query context and the text of the candidate entity page. The second is generated by creating a 2-tuple for each combination of candidate categories and context words. Based on development experiments (TAC 2009 data), we use great and great-great grandparent categories. Following Bunescu and Paşca, categories have to occur 200 times or more and context words are those that occur within a 55-token context window of the entity mention.

Cucerzan disambiguates the query mention with respect to document-level vectors derived by summing over candidate entity vectors. Candidate vectors include the article's categories (e.g., American jazz composers²) and its contexts (anchors of links in the first paragraph and of reciprocal links anywhere in the article). Candidate vectors are scored by taking the scalar product with the document-level vector, minus a penalty to subtract the candidate's contribution to the document-level vector. The resulting calculation is equivalent to an un-normalised cosine, which has the effect of giving more weight to candidates with more categories and/or contexts.

The Varma et al. approach ranks candidates based on the textual similarity between the query context and the text of the candidate page, using the cosine measure. Here, the query context is the full paragraph surrounding the query mention, where paragraphs are easily identified by double-newline delimiters in the TAC source documents.

The Milne and Witten disambiguator uses a C4.5 decision tree to combine three sources of information: commonness (probability of an article given an alias), relatedness (set overlap of in-links for the query candidate and the unambiguous articles discovered from the local context of the query document) and context quality (sum of the weights that were previously assigned to each unambiguous article from the local

¹ http://www.cs.cornell.edu/People/tj/svm_light/svm_rank.html

² Following Cucerzan, we exclude categories that we do not deem to be useful semantic types for NEL. We filter categories whose name contains a key word or its plural (article, page, date, year, birth, death, living, century, acronym, stub) a four-digit number (i.e., a year), or was Exclude in print.

context). For the comparison here, we adapted the Wikipedia Miner version 1.1³ disambiguator to take a list of mentions and their candidate sets. The data and models are based on a different version of Wikipedia, but we ensure that the disambiguation considers only candidates that we have passed in to avoid returning a candidate that does not exist in our version of Wikipedia.

The Lehmann et al. disambiguator uses features based on mention and candidate name similarity, as well as context similarity. As in Milne and Witten [23], unambiguous link anchors close to the mention are used as context. High-coverage entity type matching is also used to ensure semantic compatibility between mention and candidate entities. Features indicating which searcher source retrieved the candidate let the classifier learn *which* sources to trust. A heuristic over the features is used to rank the candidates. They propose two NIL classification systems: a supervised binary logistic classifier that classifies the top ranked few candidates and an unsupervised system based on a subset of features and a minimum confidence threshold for non-NIL.

6 Graph-Based Approaches to Disambiguation

In this section, we propose an unsupervised approach that operates over a link graph of Wikipedia articles for document mentions. Wikipedia’s size prohibits building the full graph, instead we construct a subgraph by creating vertices for each unambiguous mention. Following [23], these are mentions for which a searcher (see Section 4) returns exactly one candidate article. We also create one vertex for each candidate article returned for the query mention. The seed subgraph is expanded by tracing length-limited paths between each vertex via other articles and categories – extra vertices, adding them as required. Once built, a graph measure is applied, and the query candidates can then be ranked by their score.

Consider Figure 1, where *Positive Black Soul* has been unambiguously matched but we do not know whether the query ‘David Murray’ refers to the Iron Maiden guitarist or the jazz saxophonist. The sub-graph shows that David Murray (saxophonist) is connected to *Positive Black Soul*, since their pages are both linked to by the article *Fo Deuk Revue*, an album they collaborated on. The Iron Maiden guitarist, however, does not have any connections.

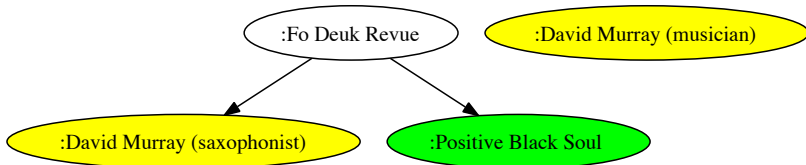


Fig. 1. Graph of ‘David Murray’ candidates with unambiguous match: ‘Positive Black Soul’

³ <http://wikipedia-miner.sourceforge.net/> (code: 04/2009, data: 03/2009)

Table 6. Characteristics of Wikipedia subgraphs

Article	Category	$ Q $	$\langle V \rangle$	$\langle E \rangle$
2	0	1,343	14,271	33,037
0	3	1,344	2,535	23,689

We are also motivated by the comparison to WSD (Section 2), which raises the question of whether graph-based approaches from the WSD field [22] could lead to improvements in NEL. Navigli and Lapata [25] provide a recent comparative survey and demonstrate that unsupervised, graph-based approaches rival supervised WSD. They build graphs based on all candidate groundings for all nouns in a sentence and compare the performance of various graph measures, including degree centrality, PageRank [4], HITS [16], the key-player problem [3] and betweenness centrality. Results show that degree centrality leads to the best results. Degree centrality is also preferable in terms of runtime complexity $O(n)$ compared to $O(n^2)$ or worse for other algorithms.

Based on these results, we explore the two graph measures: degree centrality and PageRank. Degree centrality is simply the number of edges terminating in a given vertex v normalised by the total number of vertices in the graph minus 1:

$$D(v) = \frac{|\{u \mid \langle u, v \rangle \in E\}|}{|V| - 1} \quad (7)$$

Degree treats all edges as being equal. PageRank, by contrast, encodes the notion that edges involving more highly linked nodes are more important. PageRank scores are calculated using a recursive Markov chain algorithm based on the following equation:

$$PR(v) = \frac{(1 - \alpha)}{|V|} + \alpha \sum_{\langle u, v \rangle \in E} \frac{PR(u)}{|\{w \mid \langle u, w \rangle \in E\}|} \quad (8)$$

where α is a damping factor representing the probability at each page a random surfer will get bored and request another random page. It is set to the default of 0.85 here. The denominator in the sum is the number of links going out of vertex u .

The maximum path length for inter-article links is two, a reasonable trade-off between efficiency and detail. For category links, the maximum is three, which allows paths through up to two category pages (e.g., David Murray (saxophonist) \leftarrow American jazz composers \leftarrow American composers \rightarrow Leon “Pee Wee” Whittaker).

We consider two kinds of extra vertices individually and combined: inter-article links in paragraphs and category links. For example, the David Murray (saxophonist) page has links from the pages describing albums to which he contributed, including Fo Deuk Revue. Categories collect articles on similar subjects (e.g., the David Murray page contains a American jazz composers category, which is also present on pages for other musicians including John Coltrane). Categories are not strictly hierarchical, since they can have loops. However, there is a notion of parent (e.g., American composers for American jazz composers).

Table 7. Comparison of systems (TAC 2010 test data). Rows 15-17 are from the literature.

Row	System	A_c	A_\emptyset	A
1	NIL Baseline	0.0	100.0	54.7
2	Title Baseline	37.8	98.4	70.9
3	+ Redirect Baseline	63.7	95.1	80.9
4	Cosine	72.5	88.9	81.5
5	Bunescu and Paşca	69.1	90.8	81.0
6	Milne and Witten	50.9	95.9	75.5
7	Varma et al.	70.4	91.1	81.7
8	Cucerzan	78.5	89.2	84.4
9	Article Graph, Degree	79.8	90.2	85.5
10	Category Graph, Degree	79.0	89.3	84.6
11	Combined, Degree	79.8	89.9	85.3
12	Article Graph, PageRank	78.0	90.0	84.6
13	Category Graph, PageRank	78.5	89.1	84.3
14	Combined, PageRank	78.4	90.0	84.8
15	TAC 2010 median	UNK	UNK	68.4
16	Lehmann unsupervised	79.2	91.2	85.8
17	TAC max (Lehmann)	80.6	92.0	86.8

We use the graph scores to reweight cosine-ranked candidates as follows, e.g.:

$$s_D(c) = \text{cosine}(c) * D(c) \quad (9)$$

$$s_{PR}(c) = \text{cosine}(c) * PR(c) \quad (10)$$

where c is a candidate, D is degree centrality and PR is PageRank. Unlike Navigli and Lapata [25], we use directed edges since Wikipedia does not have necessarily reciprocal links like WordNet. In development experiments (TAC 2009 data), this resulted in a negligible difference for Degree and an *Accuracy* improvement of 0.5 for PageRank.

Table 6 contains some descriptive statistics for the extracted graphs on the development data (TAC 2009). The first and second columns contain the maximum path length for article and category links respectively. The third column ($|Q|$) contains the number of queries for which a graph was extracted (out of 3,904 total). The fourth column ($|V|$) contains the average number of vertices across these queries. And, the final column ($|E|$) contains the average number of edges. The first row corresponds to the article graph. And, the second row corresponds to the category graph, which has far fewer nodes than the article graph, but a substantially higher ratio of edges per node.

7 Linking Experiments

Table 7 contains evaluation results on the held-out TAC 2010 test set. The first three rows correspond to baseline systems that have no disambiguation component. The NIL baseline simple returns NIL for each query and achieves an overall accuracy of 54.7% due to the slightly higher proportion of NIL queries in the TAC 2010 test data. The Title

Table 8. Overall accuracy by genre and entity type

System	News			Web		
	ORG	GPE	PER	ORG	GPE	PER
NIL Baseline	72.6	21.0	91.0	33.2	56.6	33.1
Title Baseline	74.6	52.4	91.0	50.8	75.1	76.5
+ Redirect Baseline	76.8	66.8	98.0	81.2	76.7	86.9
Cosine	81.6	69.8	97.6	84.8	62.2	88.0
Bunescu and Paşca	77.2	64.4	97.2	88.8	72.7	89.6
Milne and Witten	78.0	56.8	97.6	71.6	67.9	74.9
Varma et al.	78.0	67.8	97.2	90.8	70.3	88.0
Cucerzan	79.6	80.8	97.0	82.8	73.1	88.4
Article Graph, Degree	78.2	82.2	97.2	84.0	73.9	88.4
Category Graph, Degree	80.0	80.8	97.4	87.6	75.9	89.6
Combined, Degree	80.0	81.4	97.4	86.4	75.5	88.4

baseline consists of exact match on article titles and the Title+Redirect baseline consists of exact match on both article and redirect page titles. Note that the Title+Redirect baseline is a much stronger benchmark than the median result (Row 15) reported at TAC 2010, achieving an overall accuracy 14 points higher. The fourth row corresponds to a system that uses our implementation of the Cucerzan searcher and a cosine disambiguator. This is used for search and cosine scoring in the graph-based approaches.

The fifth through eighth rows correspond to our implementation of systems from the literature. Among these, the Cucerzan [6] approach is a clear winner. At an overall accuracy of 84.4%, it is four points better than our implementations of Bunescu and Paşca [5] and Varma et al. [30] approaches. And, it is only two points shy of the top result from TAC 2010 (86.8%, row 17). The Milne and Witten approach [23] is not a fair comparison here since it was tuned for a precision-oriented linking task that is more like WSD. However, it does give an idea of what recall is sacrificed to achieve high precision. With respect to our other implementations, the Bunescu and Paşca and Varma et al. approaches lead to conservative linkers (high NIL accuracy and much lower candidate accuracy) while the Cucerzan approach results in a liberal linker. This may be due in part to their respective search strategies, explored in Section 4 above. Bunescu and Paşca and Varma et al. have search strategies with relatively high ambiguity, making it more difficult for the disambiguator to identify the correct candidate. By contrast, Cucerzan has a search strategy that generates the fewest candidates on average, but achieves comparable candidate recall.

Rows 9–14 correspond to our graph-based reranking approaches, parametrised by the source of their link information — articles, categories, or combined — and by the graph connectivity measure used — Degree or PageRank. In terms of overall accuracy (A), graph-based reranking leads to substantial improvements over the cosine input, ranging from 2.8 to 4. Substituting or combining link sources has little effect, though the category graph scores are slightly lower. Comparing measures, Degree achieves higher overall accuracy than PageRank, consistent with results for WSD [25].

At 85.5%, our best graph-based approach represents an improvement of 1.1 in overall accuracy with respect to the previous state of the art in unsupervised linking

(Row 8) and is comparable to the best unsupervised approach at TAC 2010 (Row 16). Our approach uses a simple solution that combines highly efficient cosine scoring with an elegant approach to global disambiguation based on the Wikipedia link graph. Furthermore, our graph-based result is not far off the maximum overall accuracy reported at TAC 2010 (86.8%), a result obtained by a supervised system that relies on annotated training data to incorporate various features (Row 17).

Results by Genre and Entity Type. Table 8 contains overall accuracy broken down by genre (news or web) and entity type (ORG, GPE or PER) on the TAC 2010 test data. The best score in each column is in bold. The first thing to note is that no approach is consistently best across genres and entity types. This suggests two directions for future work: 1) system combination by voting and 2) entity-specific models and tailoring approaches to individual entity types and/or genres. Next, the percentage of NIL queries (as reflected in the NIL baseline scores) varies hugely across genre and entity types. In particular, the NIL percentage in web text is much lower than in news text for ORG and PER entities, but much higher for GPE entities.

In terms of our graph-based systems, it is interesting to note the improvement over cosine is due primarily to large gains for GPE entities, where scores are more than 12 points higher on both news and web text. It is also interesting to note that the article and combined graph approaches are consistently better than or equal to our implementations of previous unsupervised strategies from Varma et al. and Cucerzan. ORG in web text is the exception, where Varma et al.'s backoff search strategy is beneficial.

8 Conclusion

Named entity linking (NEL) is similar to the widely-studied problem of word sense disambiguation (WSD), with Wikipedia articles playing the role of WordNet synsets. Our contribution is to exploit unsupervised, graph-based approaches for NEL that have previously led to results rivalling supervised approaches for WSD [25].

Analysis of NEL searchers from the literature suggests that candidate recall is important. However, increasing recall is detrimental when it results in substantially increased ambiguity, because existing disambiguators do not perform well enough yet to overcome this additional ambiguity. This is highlighted by the fact that our tuned backoff approach, which achieves an 8% increase in candidate recall, still degrades performance on the end-to-end linking task due to the consequent increase in ambiguity.

Our results on the TAC 2010 data show another similarity to the WSD task: simple baselines lead to highly competitive results. Nevertheless, the best graph-based approach leads to a substantial increase of 4% over the cosine baseline.

Our final score of 85.5% is competitive with the best reported supervised approach (86.8%) and the best unsupervised approach (85.8%) to NEL. It incorporates document-wide link graph structure in linear time. This demonstrates that despite the different types of link information, NEL is indeed similar to WSD and can be accurately and efficiently performed using graph-based approaches.

References

1. Bagga, A., Baldwin, B.: Entity-based cross-document coreferencing using the vector space model. In: Proceedings of the 17th International Conference on Computational Linguistics, Montreal, Quebec, Canada, pp. 79–85 (1998)
2. Barzilay, R., Elhadad, M.: Using lexical chains for text summarization. In: Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization, Madrid, Spain, pp. 10–17 (1997)
3. Borgatti, S.: Identifying sets of key players in a network. In: Proceedings of the International Conference on Integration of Knowledge Intensive Multi-Agent Systems, Cambridge, MA, USA, pp. 127–131 (2003)
4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: Proceedings of the 7th International Conference on the World Wide Web, Brisbane, Australia, pp. 107–117 (1998)
5. Bunesco, R., Paşca, M.: Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, pp. 9–16 (2006)
6. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic, pp. 708–716 (2007)
7. Curran, J.R., Clark, S.: Language independent NER using a maximum entropy tagger. In: Proceedings of the Seventh Conference on Natural Language Learning, Edmonton, Canada, pp. 164–167 (2003)
8. de Vries, A.P., Vercoustre, A.M., Thom, J.A., Craswell, N., Lalmas, M.: Overview of the INEX 2007 entity ranking track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) INEX 2007. LNCS, vol. 4862, pp. 245–251. Springer, Heidelberg (2008)
9. Dredze, M., McNamee, P., Rao, D., Gerber, A., Finin, T.: Entity disambiguation for Knowledge Base Population. In: Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, China, pp. 277–285 (2010)
10. Fader, A., Soderland, S., Etzioni, O.: Scaling Wikipedia-based named entity disambiguation to arbitrary web text. In: Proceedings of the IJCAI Workshop on User-contributed Knowledge and Artificial Intelligence: An Evolving Synergy, Pasadena, CA, USA, pp. 21–26 (2009)
11. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
12. Ferragina, P., Scaiella, U.: TAGME: on-the-fly annotation of short text fragments (by Wikipedia entities). In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, Toronto, ON, Canada, pp. 1625–1628 (2010)
13. Garoufi, K., Zesch, T., Gurevych, I.: Graph-theoretic analysis of collaborative knowledge bases in natural language processing. In: Proceedings of the 7th International Semantic Web Conference, Karlsruhe, Germany (2008)
14. Grishman, R., Sundheim, B.: Message Understanding Conference-6: a brief history. In: Proceedings of the 16th Conference on Computational Linguistics, Copenhagen, Denmark, pp. 466–471 (1996)
15. Hobbs, J.R.: Pronoun resolution. Tech. rep., Department of Computer Science, City University of New York (1976)
16. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, CA, USA, pp. 668–677 (1998)
17. Kulkarni, S., Singh, A., Ramakrishnan, G., Chakrabarti, S.: Collective annotation of wikipedia entities in web text. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, pp. 457–466 (2009)

18. Lehmann, J., Monahan, S., Nezda, L., Jung, A., Shi, Y.: LCC approaches to knowledge base population at TAC 2010. In: Proceedings of the Text Analysis Conference, Gaithersburg, MD, USA (2010)
19. McNamee, P.: HLT/COE efforts in entity linking at TAC KBP 2010. In: Proceedings of the Text Analysis Conference, Gaithersburg, MD, USA (2010)
20. McNamee, P., Dang, H.T., Simpson, H., Schone, P., Strassel, S.M.: An evaluation of technologies for knowledge base population. In: Proceedings of the 7th International Conference on Language Resources and Evaluation, Valletta, Malta, pp. 369–372 (2010)
21. Mihalcea, R.: Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Vancouver, BC, Canada, pp. 411–418 (2005)
22. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of the 16th ACM conference on Conference on Information and Knowledge Management, Lisbon, Portugal, pp. 233–242 (2007)
23. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, Napa Valley, CA, USA, pp. 509–518 (2008)
24. Navigli, R., Lapata, M.: Graph connectivity measures for unsupervised word sense disambiguation. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, pp. 1683–1688 (2007)
25. Navigli, R., Lapata, M.: An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32(4), 678–692 (2010)
26. NIST: Task description for knowledge-base population at TAC 2010 (2010), http://nlp.cs.qc.cuny.edu/kbp/2010/KBP2010_TaskDefinition.pdf (accessed August 20, 2010)
27. Radford, W., Hachey, B., Nothman, J., Honnibal, M., Curran, J.R.: Cmcrc at tac10: Document-level entity linking with graph-based reranking. In: Proceedings of the Text Analysis Conference, Gaithersburg, MD, USA (2010)
28. Soon, W.M., Lim, D.C.Y., Ng, H.T.: A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics* 27(4), 521–544 (2001)
29. Tjong Kim Sang, E.F.: Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In: Proceedings of the 6th Conference on Natural Language Learning, Taipei, Taiwan, pp. 1–4 (2002)
30. Varma, V., Bysani, P., Reddy, K., Bharat, V., GSK, S., Kumar, K., Kovelamudi, S., Kiran Kumar, N., Maganti, N.: IIIT Hyderabad at TAC 2009. In: Proceedings of the Text Analysis Conference. Gaithersburg, MD, USA (2009)
31. Zheng, Z., Li, F., Huang, M., Zhu, X.: Learning to link entities with knowledge base. In: *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, CA USA, pp. 483–491 (2010)

Prediction of Age, Sentiment, and Connectivity from Social Media Text

Thin Nguyen*, Dinh Phung, Brett Adams, and Svetha Venkatesh

Curtin University

thin.nguyen@postgrad.curtin.edu.au,
{d.phung,b.adams,s.venkatesh}@curtin.edu.au

Abstract. Social media corpora, including the textual output of blogs, forums, and messaging applications, provide fertile ground for linguistic analysis material diverse in topic and style, and at Web scale. We investigate manifest properties of textual messages, including latent topics, psycholinguistic features, and author mood, of a large corpus of blog posts, to analyze the impact of age, emotion, and social connectivity. These properties are found to be significantly different across the examined cohorts, which suggest discriminative features for a number of useful classification tasks. We build binary classifiers for old versus young bloggers, social versus solo bloggers, and happy versus sad posts with high performance. Analysis of discriminative features shows that age turns upon choice of topic, whereas sentiment orientation is evidenced by linguistic style. Good prediction is achieved for social connectivity using topic and linguistic features, leaving tagged mood a modest role in all classifications.

1 Introduction

Social media genres, such as blogs, social networking sites, and forums, provide venues for diverse people to gather, communicate, make friends, and record experiences and feelings. Textual corpora comprising social media messages can be massive, diverse in topic and style, and enriched by user- and system-supplied metadata, that allow statistically significant analysis of linguistic style, demographic properties (e.g., gender and age), sentiment and social connectivity. Insights gained from such analysis have wide application, ranging from sociology (where the web is viewed as a very large community sensor), search applications, and business and government intelligence.

A number of approaches have been proposed in the literature, especially in psychology, for analyzing the relationship between the textual content of blog posts and demographic properties of the authors. Studies conducted from the perspective of psychology tend to be very small in size, due to the time and cost required to administer them. On the other hand, data-driven analyses tend to be restricted to simplistic methodologies, such as word counting. There is a need for large-scale analyses that make use of, among other things, advances in probabilistic text modelling.

* Corresponding author.

This study examines a large corpus of blog posts to analyze the effect of age, mood, topic, and social connectivity on post text. In particular, we make three hypotheses: old and young bloggers manifest significant differences in mood, the topics they choose to write about, and how they write; happy and sad posts differ in topic and the use of other psycholinguistic features; bloggers with different degrees of social connectivity write about different topics, and exhibit different writing styles and moods. These hypotheses are evaluated by means of classifiers trained with appropriate features.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. Experimental setup is explained in Section 3, including various sub-corpora created for this study. Section 4 presents prediction results for age, social connectivity, and mood orientation. Section 5 presents an analysis of the correlation between linguistic style with age, sentiment, and social connectivity of bloggers, and between topic with age and social connectivity. Section 6 concludes the paper and notes prospects for future work.

2 Background

A number of approaches have been proposed in the literature for analyzing the relationship between textual content of blog posts and other properties. For example, Schler et al. [18] used a unigram model to examine the relationship between post content and author age and gender for blogs; Mihalcea and Liu [9] used n-gram features for classifying happy and sad posts and evaluating the happiness load with respect to time (time of day, and day of week). Lexical approaches have been augmented with other linguistic properties for analysis, including part-of-speech—e.g., Pang et al. [12] consider use adjectives for the purpose of sentiment classification.

Moving toward richer language models, words can be categorized into linguistic or psychological groups. To this end, the Linguistic Inquiry and Word Count (LIWC) package [14] categorizes words into four groups: *linguistic processes*, *psychological processes*, *personal concerns*, and *spoken*. In a recent work, Nguyen et al. [11] found that these psycholinguistic features are powerful predictors for influentials or personality traits of bloggers. LIWC has been used to perform a number of demographic studies: Pennebaker and Stone [15] found that older people use language that is less self-focused, and is more present and future oriented. Focussing on gender, Newman et al. [10] demonstrated that women tend to use more social words and references to others, whereas men use more complex language. For sentiment-laden text, Stirman and Pennebaker [20] and Rude et al. [17] found that depressed and suicidal individuals are more self-focused (first-person singular), express more negative emotion, and occasionally use more death-related words.

Apart from word counting, probabilistic approaches have also been applied in language models to capture a sketch description of the content, e.g., topics. To learn topics discussed in the content, some probabilistic topic modeling schemes, e.g., Probabilistic Latent Semantic Analysis (PLSA) [7] or Latent Dirichlet Allocation (LDA) [1], could be used. We use LDA for learning latent topics mentioned

in the blogosphere. Since exact inference is intractable for LDA, we use Gibbs sampling proposed in [5] for learning blogpost–topic distribution. For each user, those blog posts made by him will be used to learn user–topic proportion.

3 Experimental Setup

We examine how the age, mood, and social connectivity of bloggers correlate with affective, topical, and psycholinguistic features of the texts they author. Specifically, we test the following hypotheses:

Hypothesis 1: Posts written by old vs. young bloggers exhibit significant difference in topic, psycholinguistic features (LIWC), and mood.

Hypothesis 2: Posts written by bloggers of happy vs. sad mood exhibit significant difference in topic and psycholinguistic features (LIWC).

Hypothesis 3: Posts written by bloggers having small vs. large degrees of social connectivity exhibit significant difference in topic, psycholinguistic features (LIWC), and mood.

We test these hypotheses by examining the performance of classifiers built for each of the three dichotomies, and report observations of statistically significant characteristics.

3.1 Dataset

The corpus for these experiments is drawn from blogging site Livejournal. Livejournal is notable for its *current mood* feature, which allows a blogger to attach a mood tag from a vocabulary of 132 predefined moods to a blog post. Examples of positive moods include cheerful, happy, and grateful; negative moods include discontent, sad, and uncomfortable. The original Livejournal corpus is composed of nearly 19 million blog posts having *current mood* tags, made by over 1.6 million bloggers, during the years 2000 to 2010. To evaluate our hypotheses, we construct three sub-corpora.

Old and Young Blogger Sub-corpus. The first corpus targets age difference. Most bloggers are aged between 20 and 60. We presume that difference in age can translate to different choices of topics to discuss, and differing linguistic styles. This sub-corpus is created by selecting users at the extremes of the age spectrum: the *old* category is formed by successively adding bloggers aged 60 and down, to a cut-off of 5,000 bloggers, resulting in an age range 39 to 60; the *young* category is created by adding users aged of 20 and up, to a cut-off of 5,000 bloggers, resulting in an age range 20 to 22. Only users posting not less than 20 posts are included.

Happy and Sad Blog Post Sub-corpus. The second corpus targets sentiment difference. Livejournal’s mood tags provide groundtruth for a user’s mood at time of writing. This sub-corpus is built from 10,000 posts tagged with current mood *happy*, plus 10,000 posts tagged with current mood *sad*.

Table 1. Language groups categorized by LIWC over the corpus of approximately 19 million blog posts studied in this paper. The numbers are the means of the percentages of the features' words used in a blog post, except for *wc* (the mean of the number of words in a post) and *wps* (the mean of the number of words per sentence).

Category	Code	Examples	%	Category	Code	Examples	%
Linguistic processes				Anger	anger	Hate, kill	1.2
Word count	wc		227	Sadness	sad	Crying	0.5
Words/sentence	wps		13.3	Cognitive	cogmech	Cause	15.3
Dictionary words	dic		83.4	Insight	insight	Think	1.9
Words>6 letters	sixltr		12.4	Causation	cause	Hence	1.3
Function words	funct	Since, too	53.4	Discrepancy	discrep	Should	1.5
Total pronouns	pronoun	I, them	17.2	Tentative	tentat	Maybe	2.5
<i>Personal pron.</i>	ppron	I, them, her	11.9	Certainty	certain	Always	1.3
1st pers singular	i	I, me, mine	7.9	Inhibition	inhib	Block	0.4
1st pers plural	we	We, us, our	0.7	Inclusive	incl	And, with	4.5
2nd person	you	You, your	1.6	Exclusive	excl	But	2.8
3rd pers singular	shehe	She, him	1.2	Perceptual	percept	Heard	2.3
3rd pers plural	they	They, their	0.4	See	see	View, saw	0.9
<i>Impers. pron.</i>	ipron	It, those	5.3	Hear	hear	Listen	0.6
Articles	article	A, an, the	4.5	Feel	feel	Touch	0.7
Common verbs	verb	Walk, went	15.8	Biological	bio	Eat, blood	2.6
Auxiliary verbs	auxverb	Am, will	9.3	Body	body	Cheek	0.9
Past tense	past	Went, ran	4.0	Health	health	Flu, pill	0.6
Present tense	present	Is, does	9.8	Sexual	sexual	Horny	0.8
Future tense	future	Will, gonna	1.0	Ingestion	ingest	Dish, eat	0.4
Adverbs	adverb	Very, really	5.8	Relativity	relativ	Bend	13.7
Prepositions	prep	To, with	10.6	Motion	motion	Car, go	2.1
Conjunctions	conj	And, but	6.3	Space	space	Down, in	4.9
Negations	negate	No, never	1.9	Time	time	Until	6.3
Quantifiers	quant	Few, many	2.5	Personal concerns			
Numbers	number	Once, half	0.7	Work	work	Job	1.6
Swear words	swear	Damn, piss	0.7	Achieve	achieve	Hero, win	1.2
Psychological processes				Leisure	leisure	Chat	1.5
Social	social	Mate, they	8.4	Home	home	Kitchen	0.5
Family	family	Son	0.4	Money	money	Cash, owe	0.5
Friends	friend	Buddy	0.3	Religion	relig	God	0.4
Humans	human	Adult, baby	0.8	Death	death	Bury	0.2
Affective	affect	Happy	7.3	Spoken			
<i>Positive</i>	posemo	Love, nice	4.6	Assent	assent	Agree, OK	1.2
<i>Negative</i>	negemo	Hurt, ugly	2.7	Nonfluency	nonflu	Er, hm	0.5
Anxiety	anx	Nervous	0.3	Fillers	filler	Blah	0.1

Social Connectivity Corpora. The last sub-corpora targets social connectivity. Livejournal supports one directed person-person link type. For a given user, we term incoming links *followers*, and outgoing links *friends*. Livejournal also

allows users to join communities that discuss topics of interest. On average each blogger has 10 followers and 23 friends, and joins 7 communities. Three corpora are built from bloggers with extreme numbers of followers, friends, and community membership. For each of the following corpora, bloggers with many friends, followers, or communities, are categorized as *social*; those with few friends, followers, or communities, are termed *solo*.

Based on Number of Followers. For this sub-corpus, the *social* category was created by collecting bloggers having 150 followers and less, to a cut-off of 5000 bloggers. In addition, bloggers in this category had to have been active for less than one year. The resulting range of followers was 78 to 150. The *solo* category was created from 5,000 bloggers with only one or two followers, who had to have been active for more than one year.

Based on Number of Friends. This sub-corpus was created similarly to the followers-based one. The *social* category includes 5,000 bloggers, active for less than one year, having number of friends ranging from 108 to 150. The *solo* category includes 5,000 users, active for more than one year, having between one and three friends.

Based on Number of Communities Joined. For this sub-corpus, the *social* category was created with a cut-off of 5,000 bloggers, active for less than one year, having joined a number of communities ranging from 56 to 150. The *solo* category was created from 5,000 users, active for more than one year, having joined one to two communities.

To avoid spam and noisy data, the upper bound for the number of links is 150 (according to Dunbar's number [3], a quasi-limit on the number of meaningful relationships a person can have). In addition, only users posting not less than 20 posts are included.

3.2 Classification

Three types of feature are used to perform the binary classifications corresponding to the hypotheses.

The first type of feature is drawn from the Linguistic Inquiry and Word Count (LIWC). The LIWC 2007 package and its standard dictionaries [14,21] are used to extract statistics for the 68 language groups. The mean for each group in the entire Livejournal corpus is present in Table 1. Notably, the LIWC dictionaries cover 83.4% of words in the corpus. Also, the percentage of words in *affective* processes in the corpus, 7.3%, is larger than that of any class of text reported in [13], even the emotional writing class, 6.02%. This highlights the usefulness of social media-derived corpora for sentiment analysis.

The second type is based on the topics discussed in blog post content. We use Latent Dirichlet Allocation (LDA) [1] to learn the latent topics from post text. The topic model requires that the number of topics be specified in advance.

A commonly accepted rule of thumb is to choose the number of topics in the scale $c \cdot \log V$ where V is the vocabulary size and c is a constant (often = 10). In addition, the size of the LIWC feature set is 68, which makes 50 a fitting value for this parameter. Gibbs sampling is used to learn 50 topics for each sub-corpus, with 1000 samples for burn-in followed by 5000 iterations, ultimately yielding topic mixtures for each blogger and each blog post.

The third type of feature is mood, as manually annotated by the blog authors. This is used in the classification of age and connectivity.

Classification is performed by Support Vector Machine (SVM) – the Weka implementation of Sequential Minimal Optimization (SMO) [6] – and a logistic regression model (LR), to predict sentiment, age and social connectivity. Ten-fold cross-validation is conducted for each SVM classifier and logistic regression. The average F-measure of predictions are reported.

3.3 Discriminating Features

Feature sets that perform well in classification are potentially useful for defining and interpreting the differences between the categories. Statistical tests are carried out to evaluate the hypotheses. Since all the features in all polar categories are found to be not normally distributed (Kolmogorov–Smirnov tests with Lilliefors significance correction, $ps < .001$), the differences are evaluated using a nonparametric statistical test – Mann-Whitney U. All statistical tests are conducted using IBM SPSS Statistics 17.0.

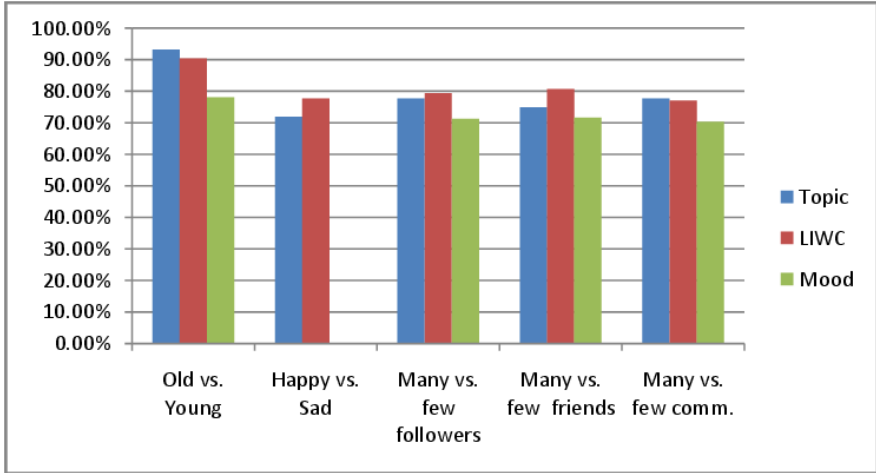
4 Prediction Results

Classification results (in F-measure) for both SVMs and logistic regression are presented in Figure 1. It can be seen that performance was not affected by classifier type. In short, classification of blogger age (old vs. young) achieved an accuracy above 90%. Classifier accuracy for all other categories was close to 80%. Thus all three hypotheses are supported.

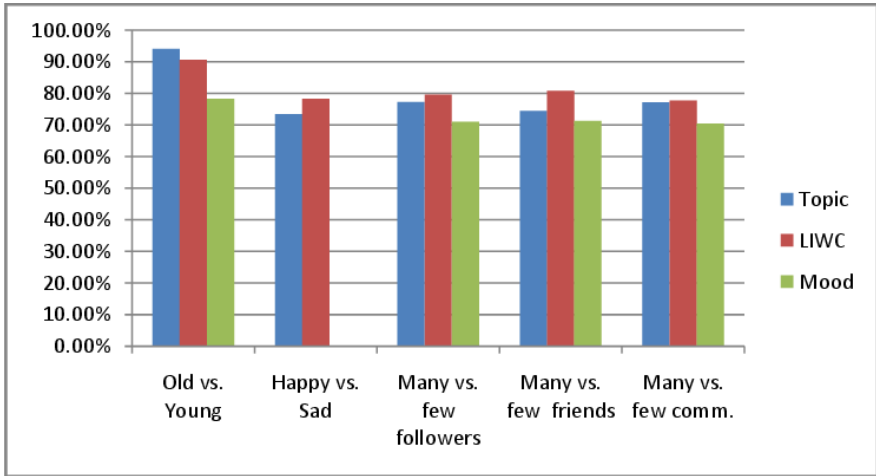
The topics discussed in the content are good features for classifying old and young bloggers, achieving an accuracy of more than 90%. The linguistic styles (through LIWC) are also effective in predicting users by age groups (90%). Using moods as features in age group predictions gains an accuracy less than 80%, worse than topic and LIWC cases.

Contrary to *old* versus *young* prediction, using LIWC groups outperforms using topics as features on sentiment classification. Accuracy ratio for the prediction is nearly 80%.

Similarly to sentiment prediction, *solo* versus *social* bloggers are well predicted using LIWC as features. E.g., prediction of *none* or *many friends* bloggers, using LIWC features alone, achieves an accuracy of 81.7%.



(a) SVM classifiers



(b) Logistic regressions

Fig. 1. Prediction performance (in F-measure)

5 Discussion

As shown in Section 4, topic and linguistic style, as represented by LIWC’s categories, are useful features for discriminating age, sentiment, and social connectivity. Below we further examine differences in these features for the various sub-corpora with reference to U tests.

5.1 Difference of Linguistic Style Across Age, Sentiment, and Social Connectivity

Old versus Young Bloggers. All LIWC features are found to be significantly different between *old* and *young* categories (Mann-Whitney U tests, $ps < .05$ two-tailed, $n_1 = n_2 = 5,000$), barring three exceptions, *future* ($p < .882$), *we* ($p < .393$), and *funct* ($p < .118$).

The differences in LIWC categories can be seen in more detail in Figure 2a. We observe that the elderly favor *greater-than-six-letter* words, which supports the finding of [15] that the aged do not decline in using more complex language, whereas the young prefer to use *spoken* informal language. The young appear to be more *self-focused* (using more *i* words, e.g., I, me, mine), whereas the old favour third person plurals, which also accords with the finding of [15] that the old are less self-focused. The young also use more *affective* and *negation*, and less *article* and *preposition* words, which accords with the finding of [21] that *emotion* words are positively correlated with *negation* use, and negatively correlated with *articles* and *prepositions*.

Happy versus Sad Blog Posts. All LIWC features are found to be significantly different between *happy* and *sad* posts (Mann-Whitney U tests, $ps < .05$ two-tailed, $n_1 = n_2 = 10,000$), barring five exceptions, *percept* ($p < .717$), *space* ($p < .693$), *number* ($p < .584$), *relig* ($p < .516$) and *humans* ($p < .158$). Figure 2b graphs the difference of LIWC categories between happy and sad posts. Not surprisingly, *posemo* and *leisure* words are used more often in *happy* posts, while *negemo*, *sad*, and *death* words dominate *sad* ones.

Social versus Solo Bloggers. Many LIWC features are found to be significantly different for the different social connectivity categories. Figures 3a, 3b, & 3c graph the distinctive use of the LIWC categories for the three social connectivity sub-categories.

First we make some observations that apply to the overarching *social* vs. *solo* blogger categories. Social bloggers make more use of words of greater than six letters, which is referred to as complex language, and a marker of presidential language [19]. Social bloggers use fewer words per sentence (*wps*) than solo bloggers (a difference that is characteristic of extraverts vs. introverts in [8]). In contrast, solo bloggers tend to use the *spoken* category (i.e., assent, nonfluency, and fillers). In addition, solo bloggers prefer to use *self-focused*, *affective*, and *swear* words, which together have implications for how acceptable their language is to others.

Social bloggers use more *articles* than solo bloggers. Normally, an article is followed by a concrete noun. Social bloggers also use less *pronouns* and *verbs*. If we assume social bloggers are likely *extraverts* and solo bloggers are likely *introverts*¹, this finding is partly in contrast with the that of [2] in oral languages that *extroverts* use more *pronouns* and *verbs*; and fewer *nouns* and *prepositions* than do *introverts*.

¹ As defined in [4], *introverts* are those who tend to withdraw from social contacts, whereas *extraverts* tend to make social contacts.

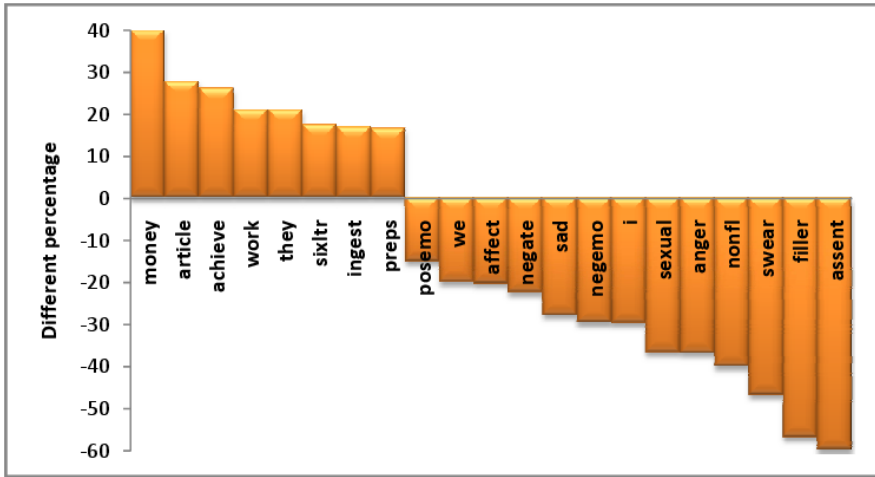
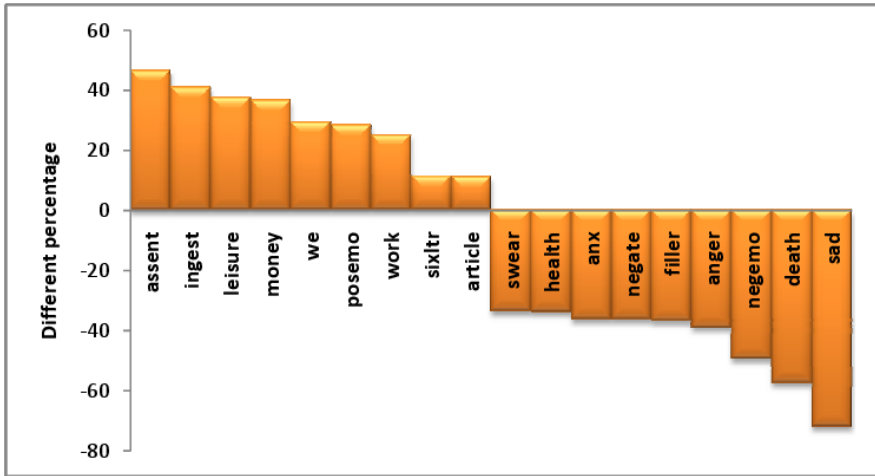
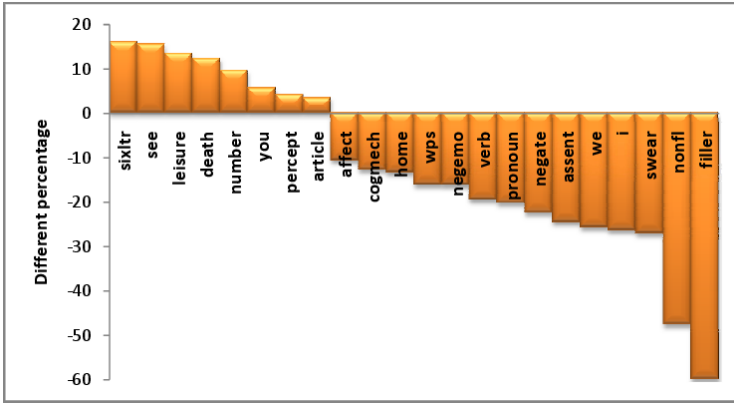
(a) *Old versus young.*(b) *Happy versus sad.*

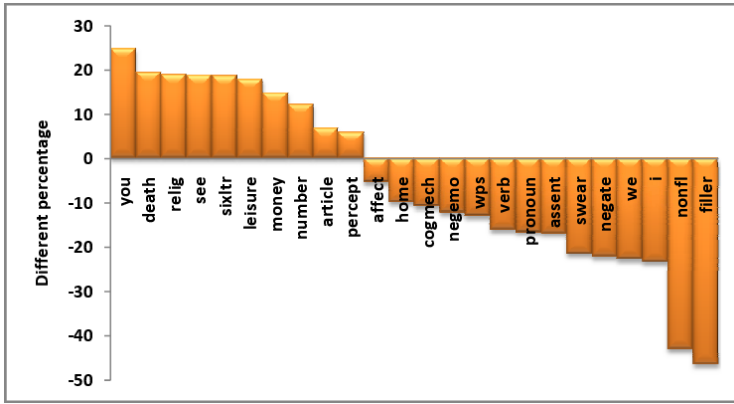
Fig. 2. The LIWC features above zero line are in favor of the *old* and *happy*; otherwise, they are in favor of the *young* and *sad* ($ps < 0.001$)

Bloggers having many friends use more “good” words—e.g., *leisure* or *achievement*, whereas friendless bloggers use more words of *negative* emotion. Similarly, users with few followers use more “bad” words, including *sad*, *anger*, and *anxious* groups.

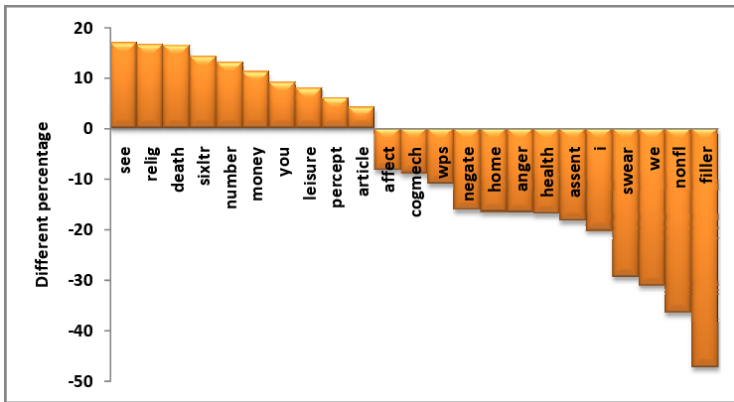
Bloggers who join many communities favour using more *personal concerns* words than those who join few communities, including *work*, *leisure*, *religion*, and *death*. Those joining many communities favour *perceptual* words (e.g., hear, see), whereas the less social bloggers prefer *cognitive* words (e.g., think, because).



(a) Many followers versus few.



(b) Many friends versus few.



(c) Many communities versus few.

Fig. 3. The LIWC features above zero line are in favor of the *social*; otherwise, they are in favor of the *solo* ($ps < 0.001$)

Table 2. Differences in topic consideration among old and young bloggers ($ps<0.001$)

Topics in favor of the old		Topics in favor of the young	
house stuff bit job car years place	working big couple room bed started left money start weeks dinner 10 saturday	favorite sex color hair nope crush book movie	music blue comment food song may related to phone screen of use
bush president war state american	country states government vote gay world america church law iraq court rights political herry pope	anime japanese manga chan character to watch	eat show old 277 part in 32 eating japan nation in 1st
movie book story film character	read series characters books episode star season fic movie game writing favorite some reading	school mom stuff house guess cool cause died	tummy left class awesome okay wanted earned mile march today watched get
chicken cheese chocolate sugar	sauce cup bread cream butter water flavoured lemon cake garlic eggs salt milk rice acid pain		
world away mind years read end	kind heart point care fact wrong course place real words feeling hell far use		
hearts			

Table 3. Differences in topic consideration between those who have many followers and those who have few followers ($ps<0.001$)

Topics in favor of many-follower bloggers		Of few-follower bloggers	
post read movie character song	icon episode story icons favorite book fic watch film okay pants characters season reading serie	favorite color sex hair sleep nope book	current movie blue crush number food song chocolate milk white school wear to
scored higher test	gender school	school house mom friday awesome	cool left watched game saturday party hung stuff funny birthday played 3d movie sunday gifts
song ayumi hamasaki japanese	hugs version pink crack bonnie ni album artist language kinty music mr. d of orange pop	class school	classroom science teacher lesson homework other
anime chan manga japanese game character country	eat games play to the music story teach hand with thing you	guess school away cause miss	wanted stuff mom wish girl care anymore job wait house heaven hell sleep start soon
bush president american country state	government war gay america vote rights political states iraq election george 000 pope herry		
world years mind end point away fact far read	kind course sense different places words sat live heart death real		

5.2 Difference of Topic Across Age and Social Connectivity

Figure 1 reports that topic features enable classification of blogger age (old vs. young) above 90%. Table 2 includes “term clouds” of topics that are used with significantly different frequency between the two groups.

The elderly write more about political matters (e.g., Bush and the world), confirming the finding of [16] that political participation is comparatively lower for young people. The remaining topics make an interesting snapshot into the contrast: for the old, house “stuff”, cooking, movies and books; and for the young, love, school, and anime.

Interestingly, topics noted above as being characteristic of the old are also discriminatory for social bloggers, and those of interest to the young correlate

Table 4. Differences in topic consideration between those who have many friends and those who have few friends ($ps < 0.001$)

Topics in favor of many-friend bloggers				Of few-friend bloggers			
movie	character	read	post story				
icons	icon	episode	fic	jack	characters	book	
season	film	series	favorite	paris	memes	star	reading
scored	created	test	quizzilla	higher			
managemen	level	garcia	quibiam	do	quibus	du	consequat
ipsum	du	consequat	du	consequat	du	consequat	du
anime	japanese	manga	chan	character	gackt		
japan	song	marcus	compley	series	at	harmony	colla
cup	sugar	eggs	sauce	oil			
chicken	butter	bread	dennis	egg	pan		
pepper	wes	add	eat	recipe	soup	cheese	
bush	world	state	war	children	country		
gay	american	america	president	government			
women	men	law	years	michael	use	church	public
eyes	hand	head	face	hands	away	room	door
voice	body	door	mouth	felt	the	tuned	hair
class	school	classes	semester	college	homework	test	
class	school	classes	semester	college	homework	test	
away	world	mind	guess	heart			
feeling	care	wish	end	years	wrong	past	
wanted	girl	kind	hell	reason	anymore	live	matter
school	mom	house	guess	stuff			
cause	cool	miss	kinda	anyways	car	wait	left
awesome	dad	wanted	friday	stayed	tummy	sleep	

with solo bloggers. From the term clouds in Tables 3 and 4 it can be seen that social bloggers write more about entertainment (e.g., books, songs, movies, and animation) whereas solo bloggers focus particularly on school-related topics. In addition, those having more followers and friends write more about politics than do solo bloggers, which is intuitive when the social component of political participation is considered.

6 Conclusion and Future Work

We have investigated the potential for using textual social media to infer properties of their authors, including age, mood, and social connectivity. We find significant differences in among these cohorts when characterized by latent topics of discussion, psycholinguistic features, and tagged mood.

Latent topics are found to have greater predictive power than linguistic features when classifying bloggers as either *old* or *young*, whereas linguistic features outperform for sentiment classification. Degree of blogger social connectivity is effectively predicted by both content-based feature sets, with linguistic features marginally outperforming topic. Manually tagged mood enables modest prediction for age and social connectivity of bloggers.

The results presented here have application to personalized information retrieval, which relies on knowledge of user attributes such as age and personality traits to re-rank results. Online advertising can also make use of estimated user profile attributes to further target advertisements based on age, connectivity, and user mood. We note that the state of the art in automatic mood classification achieves good performance, and thus the application of this analysis is not limited by the absence of manual mood groundtruth.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. *Journal of Machine Learning Research* 3, 993–1022 (2003)
2. Dewaele, J.M., Furnham, A.: Personality and speech production: a pilot study of second language learners. *Personality and Individual Differences* 28(2), 355–365 (2000)
3. Dunbar, R.I.M.: Coevolution of neocortical size, group size and language in humans. *Behavioral and Brain Sciences* 16(4), 681–735 (1993)
4. Freyd, M.: Introverts and extroverts. *Psychological Review* 31(1), 74–87 (1924)
5. Griffiths, T.L., Steyvers, M.: Finding scientific topics. *Proceedings of the National Academy of Sciences* 101(90001), 5228–5235 (2004)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter* 11(1), 10–18 (2009)
7. Hofmann, T.: Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning* 42(1), 177–196 (2001)
8. Mairesse, F., Walker, M.A., Mehl, M.R., Moore, R.K.: Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research* 30(1), 457–500 (2007)
9. Mihalcea, R., Liu, H.: A corpus-based approach to finding happiness. In: *Proceedings of the AAI Spring Symposium on Computational Approaches to Weblogs* (2006)
10. Newman, M.L., Groom, C.J., Handelman, L.D., Pennebaker, J.W.: Gender differences in language use: An analysis of 14,000 text samples. *Discourse Processes* 45, 211–236 (2008)
11. Nguyen, T., Phung, D., Adams, B., Venkatesh, S.: Towards discovery of influence and personality traits through social link prediction. In: *Procs. of the Int. AAI Conference on Weblogs and Social Media, ICWSM* (2011)
12. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: *Proceedings of the ACL 2002 Conference on Empirical Methods in Natural Language Processing*, vol. 10, pp. 79–86. Association for Computational Linguistics (2002)
13. Pennebaker, J.W., Chung, C.K., Ireland, M., Gonzales, A., Booth, R.J.: The development and psychometric properties of LIWC 2007. LIWC Inc., Austin (2007)
14. Pennebaker, J.W., Francis, M.E., Booth, R.J.: Linguistic inquiry and word count (LIWC) [computer software]. LIWC Inc., Austin (2007)
15. Pennebaker, J.W., Stone, L.D.: Words of wisdom: Language use over the life span. *Journal of Personality and Social Psychology* 85(2), 291–301 (2003)
16. Quintelier, E.: Differences in political participation between young and old people. *Contemporary Politics* 13(2), 165 (2007)
17. Rude, S., Gortner, E.M., Pennebaker, J.: Language use of depressed and depression-vulnerable college students. *Cognition & Emotion* 18(8), 1121–1133 (2004)
18. Schler, J., Koppel, M., Argamon, S., Pennebaker, J.: Effects of age and gender on blogging. In: *2006 AAI Spring Symposium on Computational Approaches for Analyzing Weblogs* (2006)

19. Slatcher, R.B., Chung, C.K., Pennebaker, J.W., Stone, L.D.: Winning words: Individual differences in linguistic style among us presidential and vice presidential candidates. *Journal of Research in Personality* 41(1), 63–75 (2007)
20. Stirman, S.W., Pennebaker, J.W.: Word use in the poetry of suicidal and nonsuicidal poets. *Psychosomatic Medicine* 63(4), 517 (2001)
21. Tausczik, Y.R., Pennebaker, J.W.: The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of Language and Social Psychology* 29(1), 24 (2010)

Word Sense Disambiguation for Automatic Taxonomy Construction from Text-Based Web Corpora

Jeroen de Knijff, Kevin Meijer,
Flavius FrasinCAR, and Frederik Hogenboom

Erasmus University Rotterdam
PO Box 1738, NL-3000 DR
Rotterdam, The Netherlands
{312470jk,312177km}@student.eur.nl,
{frasincar,fhogenboom}@ese.eur.nl

Abstract. In this paper, we propose the Automatic Taxonomy Construction from Text (ATCT) framework for building taxonomies from text-based Web corpora. The framework is composed of multiple processing steps. Firstly, domain terms are extracted using a filtering method. Subsequently, Word Sense Disambiguation (WSD) is optionally applied in order to determine the senses of these terms. Then, by means of a subsumption technique, the resulting concepts are arranged in a hierarchy. We construct taxonomies with and without WSD and we investigate the effect of WSD on the quality of concept type-of relations using an evaluation framework that uses a golden taxonomy. We find that WSD improves the quality of the built taxonomy in terms of the taxonomic F-Measure.

1 Introduction

Nowadays, an ever increasing amount of documents is digitally stored and readily available on the Web. A common issue with managing these documents is that many of these are organized in an unstructured manner. Organizing these documents in a structured way by creating a taxonomy representation to clarify documents can be beneficial, as it enhances the overview of available documents. A taxonomy is defined as a specific form of an ontology, which is a formal, explicit specification of a shared conceptualization [5] that provides users with insight into the type relations between (domain) concepts.

Currently, many taxonomies are manually created. While manually constructing taxonomies is usually more accurate because of the involvement of domain experts, automatically generating taxonomies is often less costly and time consuming. Taxonomy construction on the Web is particularly of interest, as it enables inter-operability between Web sites, tools, etc., due to the knowledge aggregation into shared taxonomies. The Web fosters an incredible large amount

of information and knowledge that up until now is mostly not linked and even remains virtually invisible because of the lack of structure. Within the last decade, there has been a trend of connecting related data that has not been previously linked before. The well-known Linked Open Data cloud diagram¹, which aims to show the inter-connectivity of today's Web sites and their knowledge bases, grows by the year. However, this cloud could grow at an even higher rate and become increasingly more complex when widely applying automatic taxonomy construction. Hence, due to the need for structured information, as well as the complexity and considerable effort for manually creating taxonomies, automatic taxonomy construction is an interesting field to explore.

Although there is a substantial body of literature on automatic taxonomy construction, the amount of literature focussing on applying Word Sense Disambiguation (WSD) is limited, even though WSD is proven to be able to improve the results of clustering. Hence, we evaluate the influence of a sophisticated WSD algorithm on an existing concept subsumption method. We propose a framework for Automatic Taxonomy Construction from Text (ATCT), which we use for the evaluation of the added value of WSD in taxonomy construction.

The main contribution of this paper is four-fold. Firstly, we analyze the influence of WSD on taxonomy construction. Secondly, we investigate the optimal parameters for the methods that comprise the ATCT framework. Thirdly, we modify the subsumption algorithm introduced in [11] to take the position of the ancestors with respect to the current node into account. Finally, we present an implementation of this approach for the domain of economics and management, as well as the medical domain. To our knowledge, taxonomy construction has been applied to other domains, e.g., finance and tourism [3], but not to these domains before.

The organization of our paper is as follows. Section 2 includes a review of related work in the area of taxonomy construction. Section 3 describes in detail the ATCT framework that generates the taxonomies. Finally, the framework and its implementation are evaluated in Sect. 4 and conclusions are drawn in Sect. 5.

2 Related Work

Extracting terms from text corpora can be done by means of linguistic methods, statistical methods, and hybrid methods. Linguistic methods generally use Natural Language Processing (NLP) techniques, such as Part-Of-Speech (POS) tagging, morphological analysis, and lexico-syntactic patterns [6]. Linguistic methods are very well capable of defining the function of a word in a sentence, but they do not take into consideration the importance of a term. Statistical methods only use statistical techniques to extract terms from text. A problem with statistical methods is that they can filter out less frequently occurring important terms, as linguistic functions are ignored. This resulted in the development of hybrid methods, using chi-square measures, term lengths, etc. [12, 13].

¹ The diagram is based on data from the W3C SWEO Linking Open Data Community Project and is regularly updated at <http://richard.cyganiak.de/2007/10/lod/>

Several similarity measures exist for Word Sense Disambiguation (WSD), such as the fast (yet possibly inaccurate) Resnik’s similarity [2], which calculates similarity values between terms by analyzing the degree of information they share. Jiang and Conrath’s similarity measure [7] is more accurate, as it takes into account the information content of the lowest common subsumer as well as of the terms themselves.

Multiple techniques can be applied to construct hierarchical relations between terms. For example, one could employ (hierarchical) clustering techniques using various similarity measures, e.g., window-methods and co-occurrences. Labeling clusters can be done by selecting the centroid of the cluster or the lowest hypernym of terms in a cluster as a label [3]. Another approach to taxonomy construction is the usage of a classification method. One could combine a domain corpus, a general corpus, and a named-entity tagger to extract and arrange terms in a taxonomy using additional sources that provide more information about these terms in a tree-ascending or tree-descending way [13]. Classification methods can provide accurate results, but require a large training set, making this method difficult to use when a large training set is not available. A final approach to hierarchical relation creation is the usage of lexical-syntactic patterns, by creating taxonomic relations through pattern matching [6]. In the subsumption method, based on co-occurrences, a term subsumes another term if they co-occur frequently. This method is simple and it does not have any labeling issues, but it is weak in arranging terms that do not occur frequently in documents [11].

3 ATCT Framework

In our four-step ATCT framework for automatically constructing a domain-specific taxonomy, first the terms are extracted from the documents, which subsequently get processed in our term filtering step. Here, terms are filtered on lexical cohesion and domain pertinence, after which the most relevant terms are selected on the basis of a score, which is determined by domain pertinence, domain consensus, and structural relevance. The selected terms are processed into concepts as concept labels. The next optional step is *Word Sense Disambiguation* (WSD), which is used to derive the sense of a concept term and to find synonyms of the disambiguated term. Lastly, a *concept hierarchy* is created by constructing the type relations between concepts. The resulting hierarchy is represented in SKOS [1], a commonly used domain taxonomy representation format.

3.1 Term Extraction

Within the ATCT framework, terms are extracted from a set of documents by tagging all the words that appear in these documents and extracting those terms that are tagged as a noun. The choice for nouns is motivated by the fact that concepts are usually represented by nouns rather than other types of words, which is also the case for our utilized golden taxonomies.

3.2 Term Filtering

Extracted terms are filtered on multiple criteria, i.e., their domain pertinence and their lexical cohesion value. The most relevant terms are selected by calculating a score, which is based on the domain pertinence, domain consensus and structural relevance of the term [12].

Domain pertinence measures whether a term is relevant for the target domain. The more frequently a term appears in the domain corpus and the less frequently the term appears in the contrastive corpus, the higher the domain pertinence. Lexical cohesion measures cohesion among words in a term and is only used for compound nouns. Based on initial experiments, we find that for optimal performance for both measures the 30% terms with the lowest values should be filtered out. Domain consensus checks whether a term is important, i.e., it appears in several documents, and structural relevance is used for measuring importance of terms in relation to their document position appearance (title or body). Finally, the filters are combined and a weighted score is calculated. The terms with the highest scores are selected as concept labels that appear in the constructed domain taxonomy.

3.3 Word Sense Disambiguation

The optional WSD procedure is used for deriving the sense of a concept term and for finding the synonyms of the concept term. In order to find the sense of a term, we follow an approach that is based on the SSI algorithm [10]. First, we retrieve the possible senses that are associated to the term. For this, we employ a semantic lexicon, which is a large lexical database that contains many words with their synsets (collections of synonyms). When a term is recognized as a noun in the semantic lexicon, we retrieve the possible synsets for that particular term. One of those synsets is selected as the sense for a term by following a number of steps. The best suited sense is determined as follows:

$$sense_t = \max_{s_i \in S_t} \sum_{c_j \in C_t} sim(s_i, c_j) \quad (1)$$

where S_t is the set of possible senses for term t , C_t is the set of context senses, and $sim(s_i, c_j)$ is the similarity between possible term sense s_i and context sense c_j . The similarity measure we use is the one proposed by Jiang and Conrath [7], as this has proven to be an accurate and fast similarity measure [2].

3.4 Concept Hierarchy Creation

In order to build the actual concept hierarchy, the subsumption algorithm discussed in [11] is employed. In order to improve the quality of this algorithm we take the position of the ancestors with respect to the current node into account. For each concept, the potential parent concepts (subsumers) are determined. If $P(x | y) \geq t, P(y | x) < t$, where t is a co-occurrence threshold, x potentially subsumes y . If x appears in at least the proportion t of all documents in which y

appears and if y appears in less than the proportion t of all documents in which x appears, x is a potential parent of y . When the potential parents are found, the parent is determined by calculating a score for each potential parent:

$$\text{score}(p, x) = P(p | x) + \sum_{a \in A_p} w(a, x) \cdot P(a | x), \quad (2)$$

where p is the potential parent node of x , A_p represents the list of ancestors of p , and $w(a, x)$ denotes a weight value with which each co-occurrence probability of ancestor a and x is multiplied. This weight is influenced by the amount of layers between ancestor a and node x :

$$w(a, x) = \frac{1}{d(a, x)}, \quad (3)$$

where $d(a, x)$ is the length of the path between node x and ancestor a . When the scores for all the potential parent concepts are calculated, the potential parent with the highest score is chosen as the parent of x .

3.5 ATCT Implementation

We implemented the ATCT framework as a Java-based tool, which parses nouns from texts by means of the Stanford [8] parser. For presenting RDF representations we employed the Jena framework [9]. Our domain taxonomies are exported as RDF files using a SKOS vocabulary [1], so that we can compare our taxonomy with related SKOS taxonomies.

4 Evaluation

For evaluation purposes, we compare two taxonomies generated with and without applying WSD by our implementation, each consisting of 2,000 distinct concepts. The used corpus contains 25,000 abstracts extracted from RePub (<http://repub.eur.nl/>) and RePEc (<http://repec.org/>). The golden taxonomy is preprocessed (translated and pruned) version of STW Thesaurus for Economics (<http://zbw.eu/stw/>), which is a manually created taxonomy in the field of economics and business economics, containing thousands of terms more than our automatically generated taxonomy. Additionally, we construct two taxonomies for the domain of medicine and health, using the large MeSH ontology (<http://onto.eva.mpg.de/obo/mesh.owl>) for arranging medical subject headings as a reference. The built taxonomy consists of 1,000 concepts and is constructed from a total of 10,000 RePub abstracts.

4.1 Experimental Setup

We evaluate the built taxonomies on two levels, using measures from literature [4]. We use the lexical precision (LP) and lexical recall (LR) to evaluate

to what degree the concepts of our constructed taxonomy are lexically shared with the golden taxonomy. We also measure the quality of the type-of relations by using the common semantic cotopy (*csc*), which is the collection of a concept and the concept's sub- and super-concepts that are shared between a core ontology and a reference ontology. Two measures that apply the *csc* to measure the quality of type-of relations of an ontology are the global taxonomic precision (*TP*) and the global taxonomic recall (*TR*), where *TP* reflects how similar the relations in the intersection of both ontologies are with respect to the core ontology, while *TR* reflects how similar the relations in the intersection of the ontologies are with respect to the reference ontology. Last, we apply the taxonomic F-Measure (*TF*), which is the harmonic mean of *TP* and *TR*, to compute the overall quality of the concept type-of relations.

4.2 Experimental Results

When comparing the generated taxonomies with the golden standards, we obtain relatively low lexical precision and recall. For the domain of economics and management, lexical precision and recall are merely 0.1769 and 0.0926, respectively. This can be explained by analyzing the reference taxonomy we used. The STW taxonomy mainly uses the categories (abstract) in the economics and management domain, while our taxonomy uses specific terms in this domain. For the domain of medicine and health, lexical precision is somewhat higher, i.e., 0.2388, while the lexical recall of 0.0156 is very low. The latter low value is explained by the fact that the MeSH ontology consists of 15,337 concepts, which is much higher than the size of our built taxonomy (1,000 concepts).

In order to be able to properly evaluate the quality of taxonomies that are built by applying WSD we use the semantically shared concepts of the core ontologies and reference ontologies rather than the lexically shared concepts to retrieve the quality of the type-of relations as concepts are now disambiguated.

We have applied a WSD approach specific for existing taxonomies on the golden taxonomies to disambiguate its concepts in order to be able to retrieve the semantically shared concepts, which is similar to the one we used for disambiguating concepts from text corpora. The difference is that we now use the surrounding taxonomy concepts (concept neighborhood) of a concept to disambiguate a concept rather than using text surroundings. The concept neighborhood consists of the concepts that are ancestors of a concept, and the concepts that are descendants of a concept. In case none of the concepts surrounding a taxonomy concept can be disambiguated, the most common sense is selected.

We investigated what percentage of concepts were disambiguated correctly for the ontologies for different amounts of ancestor and descendant layers. We found that for disambiguating a concept the best results are obtained when using a concept neighborhood that consists of two layers of ancestor concepts and two layers of descendant concepts. Further increasing the size of the concept neighborhood does not improve the results, while decreasing the concept neighborhood size lowers the percentage of concepts disambiguated correctly.

Table 1. Quality measurements for resulting taxonomies with and without the use of WSD, within the domain of economics and management (E&M), as well as within the domain of medicine and health (M&H)

Domain	Taxonomy	TP	TR	TF
E&M	T_1 (without WSD)	0.7382	0.5082	0.6023
	T_2 (with WSD)	0.8056	0.5813	0.6753
M&H	T_1 (without WSD)	0.5681	0.6051	0.5860
	T_2 (with WSD)	0.5907	0.6016	0.5961

Table 1 shows several quality measurements for the type-of relations. We distinguish between the two domains and between applying the framework with and without WSD. The table shows the TP , TR , and TF . Here, T_1 denotes the taxonomy without WSD and T_2 denotes the taxonomy with WSD. For the domain of economics and management, both taxonomies have high TF values, implying that the taxonomic relations between concepts in both taxonomies are good. The taxonomy with WSD seems to perform better than the taxonomy without WSD on all three quality measures. WSD improves TF by approximately 12.12% for the economics and management domain. WSD thus improves the quality of the concept type-of relations in our experiment. For the medicine and health domain, TF is again higher with WSD, but the increase is not as high as for economics and management. Nevertheless, for all built taxonomies WSD had a positive effect on the overall quality of the type-of relations.

5 Conclusions

We have presented the ATCT framework for the automatic generation of a domain taxonomy from text. The framework extracts potential taxonomy terms from a large corpus, resulting in a number of the most relevant terms, after having filtered the potential terms for domain pertinence, domain consensus, lexical cohesion, and structural relevance. When disambiguating term senses, a sense and alternative labels (synonyms) are added to the concept. Concepts with the same senses are removed from the concept list. Subsequently, taxonomic relations are created by means of a subsumption method, which arranges concepts in a taxonomy according to their co-occurrence in documents. After implementation, we found in our experiments that the usage of WSD in automatic taxonomy construction improves the performance measured in terms of the TF -value by 12.12% with respect to the method without WSD.

For future research it would be interesting to benchmark our method against other taxonomy creation methods, such as hierarchical clustering or classification methods. Furthermore, we would like to investigate the impact of WSD on these other methods. Exploring other term extraction methods such as lexico-syntactic patterns in combination with our framework is also a research direction that we would like to pursue. Finally, we would like to investigate the application of our approach to other domains, e.g., law, chemistry, physics, or history.

Acknowledgements. The authors are partially sponsored by the Dutch Organization for Scientific Research (NWO) Physical Sciences Free Competition project 612.001.009: Financial Events Recognition in News for Algorithmic Trading (FERNAT).

References

- [1] Bechhofer, S., Miles, A.: SKOS Simple Knowledge Organization System Reference - W3C Recommendation, August 18 (2009), <http://www.w3.org/TR/2009/REC-skos-reference-20090818/>
- [2] Budanitsky, A., Hirst, G.: Semantic Distance in WordNet: An Experimental, Application-Oriented Evaluation of Five Measures. In: Workshop on WordNet and Other Lexical Resources, 2nd Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2001), pp. 29–34. Association for Computational Linguistics (2001)
- [3] Cimiano, P., Hotho, A., Staab, S.: Learning Concept Hierarchies from Text Corpora Using Formal Concept Analysis. *Journal of Artificial Intelligence Research* 24(1), 305–339 (2005)
- [4] Dellschaft, K., Staab, S.: On How to Perform a Gold Standard Based Evaluation of Ontology Learning. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 228–241. Springer, Heidelberg (2006)
- [5] Gruber, T.R.: A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition* 5(2), 199–221 (1993)
- [6] Hearst, M.A.: Automatic Acquisition of Hyponyms from Large Text Corpora. In: 14th Conf. on Computational Linguistics (COLING 1992), vol. 2, pp. 539–545 (1992)
- [7] Jian, J.J., Conrath, D.W.: Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In: 10th Republic of China Computational Linguistics Conf. on Research in Computational Linguistics, The Association for Computational Linguistics and Chinese Language Processing (ROCLING 1997), pp. 19–33 (1997)
- [8] Klein, D., Manning, C.D.: Fast Exact Inference with a Factored Model for Natural Language Processing. In: 16th Annual Conf. on Neural Information Processing Systems (NIPS 2002). *Advances in Neural Information Processing Systems*, vol. 15, pp. 3–10. MIT Press, Cambridge (2002)
- [9] McBride, B.: Jena: Semantic Web Toolkit. *IEEE Internet Computing* 6(6), 55–59 (2002)
- [10] Navigli, R., Lapata, M.: Graph Connectivity Measures for Unsupervised Word Sense Disambiguation. In: Veloso, M.M. (ed.) 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007), pp. 1683–1688. AAAI Press, Menlo Park (2007)
- [11] Sanderson, M., Croft, B.: Deriving Concept Hierarchies from Text. In: 22nd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR 1999), pp. 206–213. ACM, New York (1999)
- [12] Sclano, F., Velardi, P.: TermExtractor: a Web Application to Learn the Shared Terminology of Emergent Web Communities. In: 7th Conf. on Terminology and Artificial Intelligence (TIA 2007). Presses Universitaires de Grenoble (2007)
- [13] Weber, N., Buitelaar, P.: Web-based Ontology Learning with ISOLDE. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 428–444. Springer, Heidelberg (2006), <http://www.dfki.de/dfkibib/publications/docs/ISWC06.WebContentMining.pdf>

A Composite Self-organisation Mechanism in an Agent Network

Dayong Ye¹, Minjie Zhang¹ and Quan Bai²

¹University of Wollongong, Wollongong, Australia

²Auckland University of Technology, Auckland, New Zealand
dy721@uowmail.edu.au, minjie@uow.edu.au, quan.bai@aut.ac.nz

Abstract. Self-organisation provides a suitable paradigm for developing autonomic web-based applications, e.g., e-commerce. Towards this end, in this paper, a composite self-organisation mechanism in an agent network is proposed. Based on self-organisation principles, this mechanism enables agents to dynamically adapt relations with other agents, i.e., change the underlying network structure, to achieve efficient task allocation. The proposed mechanism integrates a trust model to assist agents in reasoning with whom to adapt relations and employs a multi-agent Q-learning algorithm for agents to learn how to adapt relations. Moreover, in this mechanism, it is considered that the agents are connected by weighted relations, instead of crisp relations.

1 Introduction

Nowadays, more and more web-based applications emerge, e.g., e-commerce. These applications have high desirability to be autonomic which are capable of self-management, because self-management applications can save labour time of human managers, are able to adapt to environmental changes and ensure their own survivability. Within this context, De Wolf and Holvoet [1] recommended that agent-based modeling is best suited to build such autonomic applications. Thus, based on De Wolf and Holvoet's recommendation, we consider that self-organising multi-agent systems are good choices for developing such autonomic web-based applications, as the self-organising applications can continuously arrange and rearrange their organisational structures autonomously to adapt to environmental changes without external control. In addition, the adaptation process should be performed in a decentralised manner, so that the autonomic applications could be robust against failures of any nodes in the environments. Self-organisation, which is defined as “*the mechanism or the process enabling the system to change its organisation without explicit external command during its execution time* (Serugendo et al. [8])”, can be employed in agent networks to improve the cooperative behaviours of agents. Mathieu et al. [7] provided three principles for self-organisation agent networks design, which include (1) creation of new specific relations between agents in order to remove the middle-agents, (2) exchange of skills between agents to increase autonomy, and (3) creation of new agents to reduce overloading. In this paper, our contribution focuses on the

first principle, i.e., adaptation of existing relations between agents to achieve a better allocation of tasks in distributed environments.

Currently, research on self-organisation mechanisms in multi-agent and agent-based complex systems has produced results of significance. Some works are centralised in nature and have the potential of the single point of failure, e.g., [4]. Self-organisation mechanisms focusing on network structural adaptation (i.e., adapting relations among agents) have also been investigated by several researchers, such as [3]. However, these network structural adaptation methods assumed that only one type of relation exists in the network and the number of neighbours possessed by an agent has no effect on its local load. These assumptions are impractical in some cases where multiple relations exist among agents in a network and agents have to expend resources to manage their relations with other agents. To overcome this disadvantage, Kota et al. [6] devised a network structural adaptation mechanism, which took multiple relations and relation management load into account. The relation adaptation algorithm adopted in their mechanism lets agents take actions which can maximise the utility at each step. Nevertheless, as stated by Kaelbling et al. [5], this kind of algorithm, which always takes the highest utility action, overlooked the tradeoff between exploitation and exploration and may finally converge to a sub-optimal state with a self-organisation process continuing.

Besides the disadvantages mentioned above, the common limitation of current related research is that candidate selection for self-organisation, i.e., relation adaptation, is simplified. For candidate selection, current related works have agents use only their own experience. In addition, current self-organisation mechanisms consider only crisp relations between agents which might be another limitation. Here, crisp relation means that between two agents there is either a relation or no relation. To overcome this limitation, weighted relation is introduced in this paper, which means that between two agents, there is a relation strength, ranged in $[0, 1]$, to indicate how strong the relation is between the two agents. The introduction of weighted relation into self-organisation is reasonable, because, in the real world, the relation change between two persons usually occurs gradually rather than suddenly. Thus, weighted relations should be more flexible and more suitable in agent networks than crisp relations.

Against this background, in this paper, we propose a composite self-organisation mechanism. This self-organisation mechanism consists of three elements, which are claimed as a three-fold contribution of this paper. (1) For candidate selection, we integrate a trust model which lets agents use not only their own experience but also other agents' opinions to select candidates. (2) For adapting multiple relations, we develop a multi-agent Q-learning algorithm which enables two agents to independently evaluate their rewards about changing relations and balances exploitation and exploration. Consequently, our mechanism could overcome the aforementioned flaws of Kota et al.'s mechanism. (3) We also introduce weighted relations into our self-organisation mechanism. The introduction of weighted relations can improve the performance of our self-organisation mechanism and make the mechanism more suitable in dynamic environments.

2 The Agent Network Model

In this section, an agent network model is presented in which we will develop our self-organisation mechanism. The aim of the agent network is to allocate tasks to agents such that the communication cost among agents is minimised and the benefit obtained by completing tasks is maximised. Each task, Φ , is composed of a set of subtasks, i.e., $\Phi = \{\varphi_1, \dots, \varphi_m\}$. Each subtask, $\varphi_i \in \Phi$, requires a particular resource and a specific amount of computation capacity to fulfill. In addition, each subtask has a relevant benefit paid to the agent which successfully completes the subtask. Each subtask has a preset deadline as well, which should be satisfied. Otherwise, the benefit will be decreased gradually with time elapse till 0. A subtask φ_i is modeled as a token Δ_i , which can be passed in the network to find a suitable agent to complete. Each token consists of not only the information about resource and computation requirement of the corresponding subtask, but also the token traveling path which is composed of those agents that the token has passed.

In our model, an agent network comprises a set of collaborative agents, i.e., $A = \{a_1, \dots, a_n\}$, situated in a distributed task allocation environment. In the agent network, instead of crisp relations, two weighted relations are defined which are *peer-to-peer* relation and *subordinate-superior* relation. A *peer-to-peer* relation, denoted as “ $\overset{\mu}{\sim}$ ” ($\overset{\mu}{\sim} \subseteq A \times A$), is a *Compatible Relation*, which is reflexive and symmetric, such that $\forall a_i \in A : a_i \overset{\mu_{ii}}{\sim} a_i$ and $\forall a_i, a_j \in A : a_i \overset{\mu_{ij}}{\sim} a_j \Rightarrow a_j \overset{\mu_{ji}}{\sim} a_i$, where $\mu_{ij} = \mu_{ji}$. A *subordinate-superior* relation, written as “ $\overset{\mu}{\prec}$ ” ($\overset{\mu}{\prec} \subseteq A \times A$), is a *Strict Partial Ordering Relation*, which is irreflexive, asymmetric and transitive, such that $\forall a_i \in A : \neg(a_i \overset{\mu_{ii}}{\prec} a_i)$, $\forall a_i, a_j \in A : a_i \overset{\mu_{ij}}{\prec} a_j \Rightarrow \neg(a_j \overset{\mu_{ji}}{\prec} a_i)$ and $\forall a_i, a_j, a_k \in A : a_i \overset{\mu_{ij}}{\prec} a_j \wedge a_j \overset{\mu_{jk}}{\prec} a_k \Rightarrow a_i \overset{\mu_{ik}}{\prec} a_k$. The notation, μ_{ij} , is *relation strength*, which indicates how strong the relation is between agents a_i and a_j . μ_{ij} is ranged from $[0, 1]$, where higher value means a stronger relation and 0 demonstrates no relation between two agents. *Relation strength* affects the task allocation process, as agents usually prefer to allocate tasks to those agents which have high *relation strength* with them. Initially, the *relation strength* between any two neighbouring agents is set to 0.5 by default, but it might be adapted during the succeeding task allocation process.

In the agent network, each agent is of the form $a_i = \langle Res_i, Comp_i \rangle$, where Res_i is the set of resources possessed by a_i and $Comp_i$ is the computation capacity of a_i for completing tasks. Moreover, the knowledge of each agent is modeled as a tuple $\langle Neig_i(t), Act_i(t), Tokens_i(t) \rangle$. The first element, $Neig_i(t)$, is the neighbour set of agent a_i at time t . $Neig_i(t)$ can be further divided into three subsets, $Neig_i^{\sim}(t)$, $Neig_i^{\prec}(t)$ and $Neig_i^{\succ}(t)$. $Neig_i^{\sim}(t)$ contains the *peers* of a_i , $Neig_i^{\prec}(t)$ consists of the direct *superiors* of a_i , and $Neig_i^{\succ}(t)$ comprises of the direct *subordinates* of a_i . The second element in the agent knowledge tuple, $Act_i(t)$, is the action set of agent a_i at time t . An action set, $Act_i(t)$, is defined as a set of available actions for agent a_i at time t , while an *action* is defined as a decision made by an agent to adapt the relation with another agent. There are seven different atomic actions defined in our model, which are enh_{\sim} , enh_{\prec} ,

$enh_>$, $wkn_ \sim$, $wkn_ <$, $wkn_ >$ and no_action . It should be noted that the meanings of actions $enh_$ and $wkn_$ imply not only *enhance* and *weaken* but also *form* and *dissolve*, respectively. The atomic actions can also be combined together. The meanings of combination actions can be easily deduced from the meanings of atomic actions.

It should be noticed that an agent at different time steps might possess different available actions. The possible choices of actions available to agents in different situations are illustrated as follows. (1) There is no relation between agents a_i and a_j . The possible choices of actions include $enh_ \sim$, $enh_ <$, $enh_ >$ and no_action . (2) a_i is a *peer* of a_j , i.e., $a_i \sim a_j$. The possible actions involve $wkn_ \sim$, $wkn_ \sim + enh_ <$, $wkn_ \sim + enh_ >$ and no_action . (3) a_i is a *subordinate* of a_j , i.e., $a_i < a_j$. The possible actions include $wkn_ <$, $wkn_ < + enh_ \sim$, $wkn_ < + enh_ >$ and no_action . These actions are based on a_i 's perspective, while, in a_j 's view, a_j needs to reverse these actions. (4) a_i is a *superior* of a_j , i.e., $a_i > a_j$. This situation is the reverse condition of $a_i < a_j$.

The last element in the agent knowledge tuple, $Tokens_i(t)$, stores not only the tokens agent a_i currently holds at time t but also the previous tokens incoming and outgoing through a_i . With time elapse, old tokens will be automatically deleted from the set $Tokens_i(t)$. Furthermore, an agent possesses information about the resources it provides, the resources its *peers* could provide, and the resources all of its *subordinates* and its direct *superior* could provide, although the agent might have no idea exactly which *subordinate* owns which resource. During the allocation of a subtask φ , an agent a_i always tries to execute the subtask by itself if it has adequate resources and computation capacity. Otherwise, a_i will generate a token for the subtask and pass the token to one of its *subordinates* which contains the expected resource. Since a_i does not know which *subordinate* has which *resource*, the token might be passed several steps in the agent network forming a delegation chain. If a_i finds no suitable *subordinate* (i.e., that no *subordinate* contains the expected resource), it will try to pass the token to its *peers*. In the case that no *peer* is capable of the subtask, a_i will pass the token back to one of its *superiors* which will attempt to find some other *subordinates* or *peers* for delegation. When more than one agent is able to accept the token, a_i passes the token to the agent which has higher *relation strength* with a_i .

Apparently, the structure of the agent network will influence the task allocation process. In the next section, we will describe the composite self-organisation mechanism used to adapt the structure of the agent network, involving an evaluation method to measure the profit of the network.

3 The Composite Self-organisation Mechanism

Before devising the self-organisation mechanism, it is necessary to introduce an evaluation method to estimate the profit of the agent network. Towards this goal, we illustrate the concept of evaluation criteria, which includes cost, benefit, profit and reward of an agent and the agent network.

3.1 Network Performance Evaluation

The cost, benefit and profit of the network are calculated after a predefined number of time steps. The cost of the agent network, $Cost_{NET}$, consists of four attributes, i.e., communication cost, computation cost consumed by agents to complete assigned subtasks, management cost for maintaining subtasks and management cost for keeping neighbourhood relations with other agents. Due to the page limitation, the detailed calculation of each attribute of $Cost_{Net}$ cannot be presented.

The benefit of the network, $Benefit_{NET}$, is the sum of benefits obtained by all the agents in the network. The benefit of each agent depends on how many subtasks are completed by that agent. As depicted in Section 2, each task Φ contains several subtasks, $\varphi_1, \varphi_2, \dots$, represented as tokens $\Delta_1, \Delta_2, \dots$. When a subtask φ_i is successfully completed by an agent, that agent would obtain the corresponding benefit.

Finally, the profit of the entire network, $Profit_{NET}$, is:

$$Profit_{NET} = Benefit_{NET} - Cost_{NET} \tag{1}$$

3.2 Self-organisation Mechanism Design

As described in Section 1, when an agent, a_i , wants to adapt relations with other agents, there are two problems which have to be faced by a_i . The first problem is determining with whom a_i should adapt relations, and the second one is determining how to adapt relations with those selected agents. For the first problem, the selection process of each agent is based not only on its own former task allocation process but also on the integrated trust model. Through the trust model, an agent can get opinions from other agents about candidate selection. For the second problem, i.e., how to adapt relations, a multi-agent Q-learning approach is employed. The reason for choosing the Q-learning approach is that it provides a simple and suitable methodology for representing our mechanism in terms of actions and rewards. The self-organisation mechanism is now illustrated in the following subsections.

Candidate Selection. To assist each agent to select the most valuable candidates to adapt relations, we present a trust model based on Dezert-Smarandache theory (DSmT) [9]. Other trust models may also be available for our problem, but, through our investigation, Dezert-Smarandache theory is more suitable for our requirements, because the theory has good expressiveness and low computational complexity for trust representation, and is easy to implement.

We now introduce the key concepts of Dezert-Smarandache theory. Let T mean that the given agent considers a given correspondent to be trustworthy and $\theta = \{T, \neg T\}$ be the general framework of discernment. The hyper-power set of θ is represented as $H^\theta = \{\emptyset, \{T\}, \{\neg T\}, \{T \cap \neg T\}, \theta\}$. There is a general basic belief assignment which is a function $m : H^\theta \rightarrow [0, 1]$ where

$$\begin{cases} m(\emptyset) = 0, \\ \sum_{B \subseteq H^\theta} m(B) = 1. \end{cases} \tag{2}$$

Thus, $m(\{T\}) + m(\{\neg T\}) + m(\{\emptyset\}) + m(\{T \cap \neg T\}) = 1$, as $m(\emptyset) = 0$.

The trust model is defined as a tuple, $T_j^i = \langle m_j^i(\{T\}), m_j^i(\{-T\}), m_j^i(\{\Theta\}), m_j^i(\{T \cap -T\}) \rangle$, where i and j represent two different agents a_i and a_j , separately. Each element in T_j^i is described as follows. (1) $m_j^i(\{T\})$ means the degree of a_i trusting a_j ; (2) $m_j^i(\{-T\})$ indicates the degree of a_i distrusting a_j ; (3) $m_j^i(\{\Theta\})$ demonstrates the degree of uncertainty about the trustworthiness a_i to a_j . This case happens when a_i lacks evidence regarding a_j . If a_i has no evidence at all, $m_j^i(\{\Theta\}) = 1$; otherwise, $m_j^i(\{\Theta\}) < 1$; (4) $m_j^i(\{T \cap -T\})$ depicts the degree of contradiction with regard to the trustworthiness a_i to a_j . This case is caused by the situation, for example, that a_i trusts a_j but other agents, who provide a_i their opinions, distrust a_j . Thereby, a_i gets into contradiction.

Thus, initially, trust between any two agents is $T_j^i = \langle m_j^i(\{T\}) = 0, m_j^i(\{-T\}) = 0, m_j^i(\{\Theta\}) = 1, m_j^i(\{T \cap -T\}) = 0 \rangle$. Trust is, then, acquired through task allocation between agents. For example, if a_j completed many tasks for a_i on time, then the value of $m_j^i(\{T\})$ may increase. In addition, a_i might ask one of its neighbouring agents, say a_k , for trust evaluation to a_j , and then combines a_k 's opinion with a_i 's own view to a_j . This is trust evaluation combination which can be computed as

$$T_j^i = T_j^i \oplus T_j^k, \tag{3}$$

where $m_j^i(B_1) = m_j^i(B_2) \oplus m_j^k(B_3) = \sum_{(B_1, B_2, B_3 \subseteq H^\Theta) \wedge (B_2 \cap B_3 = B_1)} m_j^i(B_2) m_j^k(B_3)$.

Furthermore, there might be another case. a_i asks a_k 's opinion to a_j , but a_k may have no idea about a_j . a_k then inquires one of its neighbours, a_l 's, opinion to a_j . This is trust transitivity which can be calculated as

$$T_j^k = T_l^k \otimes T_j^l, \tag{4}$$

where $m_j^k(\{T\}) = (m_l^k(\{T\})) + \beta m_l^k(\{T \cap -T\}) \cdot m_j^l(\{T\})$
 $m_j^k(\{-T\}) = (m_l^k(\{-T\})) + \beta m_l^k(\{T \cap -T\}) \cdot m_j^l(\{-T\})$
 $m_j^k(\{T \cap -T\}) = (m_l^k(\{T \cap -T\})) + \beta m_l^k(\{T \cap -T\}) \cdot m_j^l(\{T \cap -T\})$
 $m_j^k(\{\Theta\}) = 1 - m_j^k(\{T\}) - m_j^k(\{-T\}) - m_j^k(\{T \cap -T\})$,
 and β is a constant which is in the range $(0, 1)$.

The candidate selection process can be simply summarised as follows. After a period, each agent, say a_i , first evaluates the trust of those agents ($TempCands_i$) which completed many or few tasks for it during the last period. Then, a_i asks the opinions of other agents against those agents in $TempCands_i$ and, afterwards, adjusts the trust values based on those opinions. Finally, a_i filters the agents in $TempCands_i$ and makes a new group of agents called *Candidates* with which a_i will adapt relations.

Relation Adaptation. Before describing our relation adaptation algorithm, we consider a simple scenario with two agents, a_i and a_j , and three available actions for each agent. The reward matrix of the two agents is displayed in Table **II**.

Each cell $(r_i^{x,y}, r_j^{x,y})$ in Table **II** represents the reward received by the row agent (a_i) and the column agent (a_j), respectively, if the row agent a_i plays action x and the column agent a_j plays action y . The reward of each agent, r_i , is based on how much load could be reduced on agent a_i and on the intermediate agents, and how much potential benefit might be obtained by agent a_i in the future. Here, an intermediate agent is an agent which resides on a token path,

Table 1. Reward Matrix of a_i and a_j

$a_i \backslash a_j$	$enh_- \prec$	$enh_- \sim$	$enh_- \succ$
$enh_- \succ$	$r_{i,1}^{1,1}, r_{j,1}^{1,1}$	$r_{i,2}^{1,2}, r_{j,2}^{1,2}$	$r_{i,3}^{1,3}, r_{j,3}^{1,3}$
$enh_- \sim$	$r_{i,1}^{2,1}, r_{j,1}^{2,1}$	$r_{i,2}^{2,2}, r_{j,2}^{2,2}$	$r_{i,3}^{2,3}, r_{j,3}^{2,3}$
$enh_- \prec$	$r_{i,1}^{3,1}, r_{j,1}^{3,1}$	$r_{i,2}^{3,2}, r_{j,2}^{3,2}$	$r_{i,3}^{3,3}, r_{j,3}^{3,3}$

written as $\Delta.path$. For example, agent a_i has no relation with agent a_j , but a_j completed many subtasks for a_i during former task allocation processes. a_i , then, might want to establish a relation with a_j . If a_i would like to form the *superior-subordinate* relation with a_j , i.e. performing the action $enh_- \succ$ (for a_j , performing the reverse action $enh_- \prec$), the management cost on both a_i and a_j will rise because both a_i and a_j have to maintain a new neighbour. Nevertheless, those agents, which are in the $\Delta.path$, could save communication cost (which are considered as a_j 's reward), since they do not need to pass tokens between a_i and a_j any more. Here, Δ refers to the tokens that are held by a_j and sent by a_i . In addition, a_i could save management cost for maintaining subtasks, as a_i can directly pass tokens to a_j without waiting for intermediate agents to pass tokens and, hence, a_i 's subtasks could be allocated in less time steps. The potential benefit which would be obtained by a_j is evaluated on the basis of the benefit a_j gained for completing the subtasks assigned by a_i , while the potential benefit of a_i is calculated in an analytical way. We suppose that the action $enh_- \prec$, with a_i as the *superior* and a_j as the *subordinate*, can make a_j potentially receive more subtasks from a_i and then get more benefits. **Algorithm 1** demonstrates our relation adaptation approach in pseudocode form.

After selecting candidates, a_i and a_j , firstly, estimate which actions are available at the current state (Lines 2 and 3) as described in Section 2. Then, a_i and a_j learn the Q-value of each available action, separately (Lines 4-11). In Line 5, the Q-value of each action is initialised arbitrarily, while, in Line 7, π_{ix} indicates the probability regarding agent a_i taking the action x . To calculate π_{ix} , we employ the ϵ -greedy exploration method devised by Gomes and Kowalczyk [2] shown in Equation 5, where $0 < \epsilon < 1$ is a small positive number and n is the number of available actions of a_i .

$$\pi_{ix} = \begin{cases} (1 - \epsilon) + (\epsilon/n), & \text{if } Q_{ix} \text{ is the highest} \\ \epsilon/n, & \text{otherwise} \end{cases} \quad (5)$$

In Line 8, Q_{ix} is the Q-value of action x taken by agent a_i . In Lines 8 and 9, $0 < \alpha < 1$ is the learning rate. When finishing learning Q-values, a_i and a_j (Line 12) cooperate to find the optimal actions. In Line 12, $match(x, y)$ is a function which is used to test whether the actions x and y that are taken by a_i and a_j , respectively, are matched. An action is only matched by its reverse action described in Section 2. Therefore, a_i and a_j have to cooperate to find the actions, which can be matched together and maximise the sum of their Q-values. Finally, a_i and a_j adjust their *relation strength* (Lines 14-17). The adjustment depends on how many subtasks that a_j completed for a_i in the last time steps,

Algorithm 1: Relation adaptation according to a_i

```

1 for each  $a_j \in Candidates$  do
2    $Act_i \leftarrow available\_actions(a_i, a_j)$ ;
3    $Act_j \leftarrow available\_actions(a_i, a_j)$ ;
4   for each  $x \in Act_i, y \in Act_j$  do
5     Initialise  $Q_{ix}$  and  $Q_{jy}$  arbitrarily;
6     for  $k = 0$  to a predefined integer do;
7       calculate  $\pi_{ix}(k)$  and  $\pi_{jy}(k)$ ;
8        $Q_{ix}(k+1) = Q_{ix}(k) +$ 
           $\pi_{ix}(k)\alpha(\sum_y r_i^{x,y}\pi_{jy}(k) - Q_{ix}(k))$ ;
9        $Q_{jy}(k+1) = Q_{jy}(k) +$ 
           $\pi_{jy}(k)\alpha(\sum_x r_j^{x,y}\pi_{ix}(k) - Q_{jy}(k))$ ;
10    end for
11  end for
12   $\langle x_{opti}, y_{opti} \rangle \leftarrow argMax_{match(x,y)}(Q_{ix} + Q_{jy})$ ;
13   $a_i, a_j$  take actions  $x_{opti}$  and  $y_{opti}$ , respectively;
14   $\mu_{ij} \leftarrow \mu_{ij} + (N_j^i/\rho - 1)$ ;
15  if  $\mu_{ij} > 1$  then  $\mu_{ij} \leftarrow 1$ ;
16  if  $\mu_{ij} < 0$  then  $\mu_{ij} \leftarrow 0$ ;
17   $\mu_{ji} \leftarrow \mu_{ij}$ ;
18 end for

```

where the more subtasks are completed by a_j , the higher *relation strength* value is achieved. In Line 14, N_j^i means the number of a_i 's subtasks completed by a_j and ρ is a threshold which is an integer.

Due to the page limitation, the experimental results cannot be presented here.

References

1. DeWolf, T., Holvoet, T.: Towards autonomic computing: agent-based modelling, dynamical systems analysis, and decentralised control. In: The First Intern. Workshop on Auton. Comput. Princip. and Archit., Banff, Canada, pp. 10–20 (2003)
2. Gomes, E.R., Kowalczyk, R.: Dynamic analysis of multiagent q-learning with e-greedy exploration. In: ICML 2009, Montreal, Canada, pp. 369–376 (June 2009)
3. Griffiths, N., Luck, M.: Changing neighbours: Improving tag-based cooperation. In: AAMAS 2010, Toronto, Canada, pp. 249–256 (May 2010)
4. Hermoso, R., Billhardt, H., Ossowski, S.: Role evolution in open mas as an information source for trust. In: AAMAS 2010, Canada, pp. 217–224 (May 2010)
5. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4, 237–285 (1996)
6. Kota, R., Gibbins, N., Jennings, N.R.: Self-organising agent organisations. In: AAMAS 2009, Budapest, Hungary, pp. 797–804 (May 2009)
7. Mathieu, P., Routier, J.C., Secq, Y.: Principles for dynamic multi-agent organizations. In: Kuwabara, K., Lee, J. (eds.) PRIMA 2002. LNCS (LNAI), vol. 2413, pp. 109–122. Springer, Heidelberg (2002)
8. Serugendo, G.D.M., Gleizes, M.P., K., A.: Self-organization in multi-agent systems. *The Knowl. Engin. Review* 20(2), 165–189 (2005)
9. Smarandache, F., Dezert, J.: *Advances and Applications of DSMT for information Fusion*. America Research (2004)

From Interface Mockups to Web Application Models

José Matías Rivero^{1,2}, Gustavo Rossi^{1,2}, Julián Grigera¹,
Esteban Robles Luna^{1,3}, and Antonio Navarro⁴

¹ LIFIA, Facultad de Informática, UNLP, La Plata, Argentina
{mrivero,gustavo,julian.grigera,
esteban.robles}@lifia.info.unlp.edu.ar

² Also at Conicet

³ Also at CICPBA

⁴ Dpto. de Ingeniería del Software e Inteligencia Artificial, Universidad
Complutense de Madrid
anavarro@fdi.ucm.es

Abstract. The process of modeling and implementing Web applications has been successfully improved by the use of Model-Driven Web Engineering (MDWE) methodologies. However, because of their traditional process models, these methodologies delay the generation of application prototypes until all design aspects (e.g. domain and navigation) are completed. These aspects are crucial for developers but not for customers, who are interested in viewing parts of the application running as early as possible. In this paper we introduce a novel model driven approach that starts from user interface specifications, using mockups to derive concrete presentation implementations – we call it Mockup-Driven Development or just *MockupDD*. Then, by using lightweight enrichments and applying heuristics over these models, we show how we obtain navigation and content specifications in the context of different MDWE methods.

Keywords: Mockups, User-Interface, Agile, Web Engineering, MDD.

1 Introduction

The development of web applications is a challenging activity involving complex aspects. Many methodologies [1, 2] use models to specify these aspects at a high level of abstraction, and then derive running implementations from such models. These Model-Driven Web Engineering (MDWE) methodologies tend to be “designer-centric”, as they start the web application specification around design concepts like classes, associations, etc. As a consequence, customers and final users cannot really get involved in the development process. In order to encourage this collaboration, agile methodologies¹ propose to integrate customers in the day-to-day work, letting the processes emerge naturally from daily practice. In this context, the agile modeling approach [3] pursues a fast construction of models by following a *just barely good enough* strategy, i.e. models should be as simple as possible and must be constructed

¹ Principles behind the Agile Manifesto - <http://agilemanifesto.org/principles.html>

in small increments, using the simplest tools. User Interface (UI) mockups are one of this kind of *agile models* that can be used to represent requirements in a language that is understandable by both customers and developers. Also, the recent appearances of mockup tools like Balsamiq² or Axure³ suggest that UI prototypes are becoming a popular way to represent, communicate and specify requirements.

In this work we propose a model-driven development approach that follows the agile modeling principles. It starts by creating UI mockups with direct participation of customers through digital mockup tools. Mockups are then processed to obtain technology independent UI models (called Structural UI models or just SUI) that can be easily mapped to concrete web implementations. Then, these models are enriched to specify navigation, content, content management and data flow. According to our experience this approach: (i) supports customers integration in the development process from the first stages; (ii) supports continuous derivation of application prototypes; (iii) encourages an iterative and lightweight modeling strategy that eases the introduction of new features in short releases; and (iv) it can be also integrated with different MDWE approaches just by adjusting the model transformation process. Since it strongly relies in UI mockups, we call it Mockup-Driven Development (*MockupDD*).

The paper is structured as follows: in Section 2 we discuss some related work. Section 3 introduces how our process is structured in detail, including some examples to show how it works. Section 4 briefly comments our tool support for the approach and some initial feedback that we obtained when applying and evaluating it. Finally, in Section 5 we will conclude summarizing our approach and commenting some further work we are pursuing.

2 Related Work

Many mockup tools have appeared in the last years, showing an increasing interest for UI mockups as requirements artifacts; many cases of successful use of mockups in agile development approaches in industry have been observed as well [4, 5]. In addition, statistical studies have been conducted showing that the use of user interface mockups effectively increases and eases software comprehension with a reduced cost in the development process [6]. Mockup tools have been also successfully integrated to bug tracking, project management and wiki environments.

User interface mockups are usually linked to other requirements specifications artifacts like User Stories [7]. Also, a common practice with mockups is to informally annotate them in order to specify requirements that cannot be expressed directly through user interface in plain text [8]. In [9], enriched mockups have been used in the context of the OO-Method model-driven approach to derive structural task models. The idea of combining abstract UI prototypes with a requirement specification language to specify human-computer interaction in the form of *storyboards* is shown in [10]. Additionally, the use of structured UI prototypes to define detailed content specifications in the form of Entity-Relationship models has been observed elsewhere [11].

² Balsamiq Mockups – <http://balsamiq.com/products/mockups>

³ Axure – <http://axure.com>

In our previous work [12] we defined how mockups constructed with different mockup tools can be translated to a common technology independent UI metamodel and then derived to different concrete web technologies. We also introduced the idea of applying enrichments to this model to derive navigation models for a concrete MDWE methodology and then inferring content models through a set of heuristics [13]. This paper focuses on SUI model level, using a refined version of these enrichments to define basic navigation and content specification and then inferring advanced features like content management and data workflow through heuristics. The resulting approach represents an agile model-driven methodology to specify heterogeneous features of web applications that can be translated into code or MDWE models.

3 The Overall Approach in a Nutshell

Mockups allow specifying user interface features in a simple way, facilitating the fast and *just barely good enough* modeling approach. They eliminate the burden that development environments usually entail for specifying user interfaces, like detailing containment structure, layout, data sources, etc. We propose to reuse mockups by translating them into our technology-independent SUI models, which can be enriched with navigation and content specifications and, using a set of heuristics, derive almost complete web models and functional code. Though in our previous work [13] we comment how heuristics can be applied over MDWE models generated from enriched UI specifications to obtain different features, here we introduce heuristics directly over enriched SUI models to generate them, making it more suitable in the context of non model-driven environments.

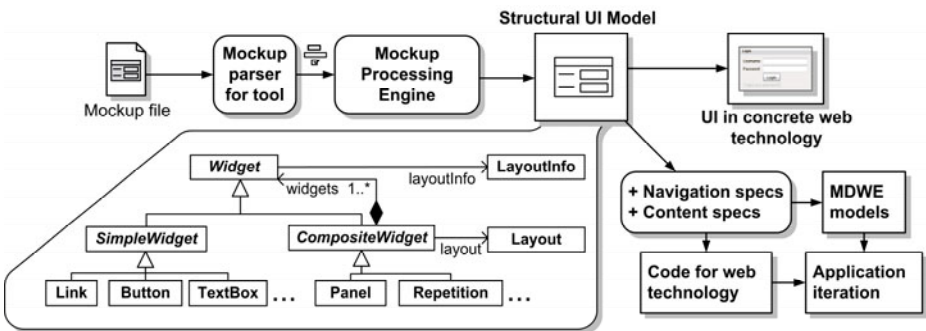


Fig. 1. Workflow of our approach

While most MDWE methodologies do not allow generating code directly from presentation models, we provide a code generation tool that facilitates the generation of running prototypes used to validate requirements with customers. Consequently, as we outline in Figure 1, our approach allows using SUI models enriched with navigation and content specs, generating different kinds of models or following a code-based methodology, in which running prototypes of the application are incrementally generated and enriched.

3.1 From Mockups to Structural UI Models

The first step of our process is to transform mockups into a technology independent UI model, formed out of instances of the meta-model classes described in Figure 1. The key component of this transformation process is the Mockup Processing Engine (MPE) that uses tool-dependent parsers to process mockup files and obtain final SUI models through a series of internal processors. Developers can parameterize algorithms and heuristics applied by the MPE to assure the consistency and adequacy of the obtained SUI model. For the sake of conciseness we do not include a complete description of this step, which can be read in a previous work [12].

3.2 The Tagging Approach

Whereas SUI models represent a structural view of the application (just like any other UI model), no behavior or data can be obtained directly from them, so we define a set of *tags* that can be applied iteratively over the SUI models providing hints to derive several aspects of web application. A *tag* is a lightweight specification formed by a name and zero or more textual parameters that can be applied only over a particular widget type (and its subtypes). In the rest of this section we introduce how we can enrich SUI models with navigation and content aspects using tags.

We will start showing our approach with the specification of navigation, a fundamental concern in web applications. Navigation is mainly described by two elements: navigational nodes (pages) and links between those nodes. We define two tags to represent these concepts:

- `Node(<nodeId>)` is applicable over `Pages` and assigns an unique id to the `Page` in order to be referenced by `Links` (and also for other purposes like naming during code generation).
- `Link(<nodeId>)` is applicable over SUI `Links` and `Buttons`, and defines a navigation to the `Page` tagged as `Node(<nodeId>)` as the default action for the widget to which it is applied.

Using the information provided by navigation tags, we can derive simple and generic navigation models for common MDWE methodologies. In order to add content specifications, we introduce a tag named `Data(<elementType>)` that specifies an association between the underlying `Widget` and the `<elementType>` class, i.e. it shows and/or allows creating or updating instances of that class. An example of the use of the introduced tags can be observed in Figure 2.

3.3 Inferring Features through Heuristics

At this point, navigation and content tags allow inferring pages, links and classes respectively, but through the combination of both tag sets semantics and the application of heuristics, we can infer content flow, content management and associations between classes in the content model.

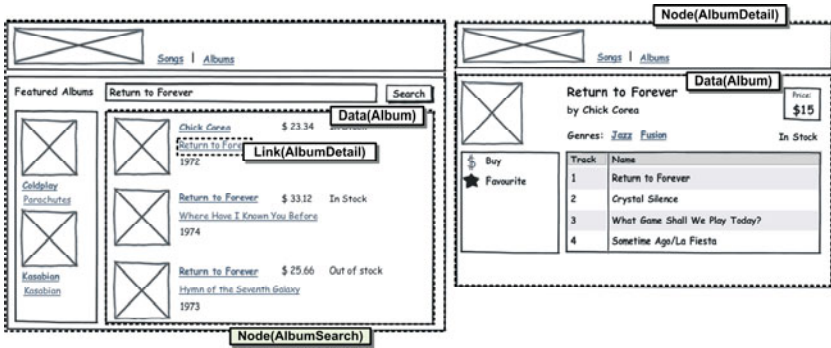


Fig. 2. Two mockups enriched with Node, Link and Data tags

Though our approach can be used in the context of any code-based or MDWE methodology, we have chosen WebML [1] to show the semantics of the inferred features because its navigation metamodel is rich, concise and expresses data flow and content management in a compact and natural way. Basically, WebML allows describing hypertext structures through basic elements called *units*. The language defines several types of units; the main unit types are *content* and *operation units* that allow displaying content and specifying operations like object creation or deletion respectively. Units are connected by links that may represent both navigation and data flow, and can be grouped together into pages and modules. The language also allows defining content models through a metamodel containing classic content specifications like classes/entities, relationships, attributes, etc. Hypertext structures rely on these models to correctly specify and further derive the whole web application.

The aforementioned inferred features from the tagged mockup of Figure 2 can be depicted in the WebML model diagram of Figure 3.a. Web models are obtained through a transformation process in which SUI models enriched with tags are analyzed and further WebML model elements are generated.

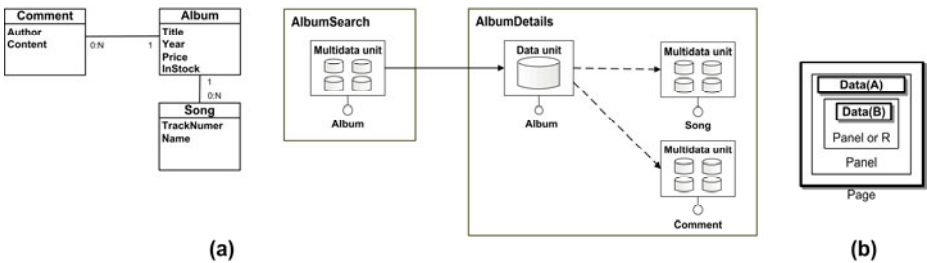


Fig. 3. Inferred WebML models from tagged mockup of Figure 2 and *Containment to Association* heuristic structure

As we can notice in the figure, some heuristics have been implicitly used to infer extra features (e.g., links between *units*) in WebML models. One of this heuristics is called *Containment to Association* (depicted in Figure 3.b), that infers an association

between classes/entities when both a `Widget` and its parent are tagged with `Data` tags. This heuristic is used to infer the association between `Album` and `Comment` classes among others in the model present in Figure 3.a.

While heuristics help to derive approximate models that are effective in the most cases, in less common contexts they can infer imprecise or wrong features. Because of this, a final slight refactoring phase (as common in agile methodologies) must be carried out to obtain the final models.

4 Tool Support and Experience

Our approach uses interface mockups as key artifacts in the development process. Since building mockups is usually simple, it can be done together with customers. To effectively add agility to our modeling process, tool support is fundamental. We have developed a tool (see Figure 4) that has been easily integrated into environments like Eclipse or Visual Studio allowing processing mockups files, drawing the obtained SUI models, tagging them and then watching the generated prototype in action.

We have used `MockupDD` to assist the development of several applications. We appreciated more customer enthusiasm and integration to the development endeavour using mockups and tags in comparison with projects in which we used common textual requirements. We are currently conducting detailed quantitative and qualitative studies to measure the real impact of our approach in the development process.

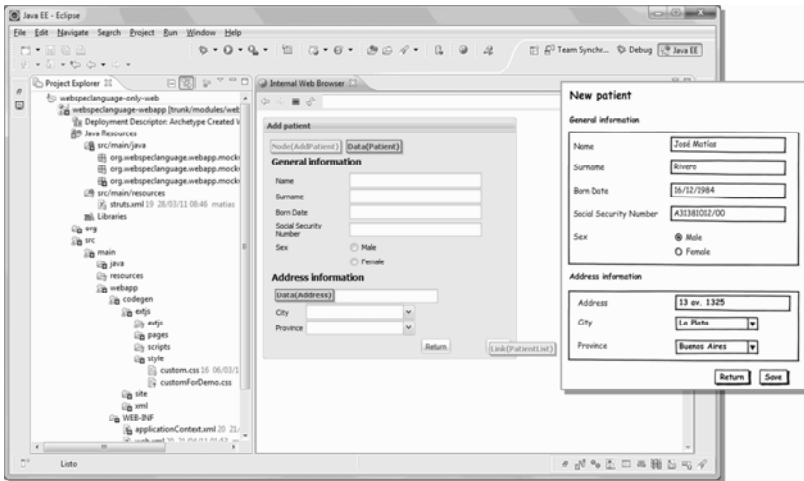


Fig. 4. Screenshot of our tagging tool running on Eclipse IDE. A SUI is being tagged, and the original Balsamiq mockup is shown on the right.

5 Concluding Remarks and Further Work

In this paper we have presented an agile model-driven development approach that aims to enhance customer integration in the development process and, at the same

time, allow modeling in small increments, using UI concepts as a common language between developers and customers. This approach presents a disruption from traditional agile ones since it incorporates mockups to the process, and uses them to create informal models that grow into accurate specifications, combining advantages of both agile and model-driven methodologies.

Considering that we start the development from presentation specifications, we can constantly make use of code and model generation to rapidly obtain application prototypes at all stages of the development process. This particular feature of our approach facilitates constant interaction with customers by discussing on the generated prototypes. We showed how SUI models could be easily enriched with tags, *lightweight* specifications that provide useful hints to code and/or model generators. We have also shown the semantic power of such simple resource.

Regarding future work, the definition of a pattern and heuristics catalogue represents a very important issue. In our opinion, with a complete list of patterns and inference rules, the most of common content management and data flow cases can be tackled and derived to models or code.

Though we have used WebML as the target MDWE methodology along this paper, we are working on generating presentation, navigation, content and behavior models for other MDWE approaches like UWE [2]. Direct code generation to other modern web technologies like GWT and ASP.NET is another interesting field in the context of our approach. We are also interested in refining general tags, or adding methodology/platform-dependent tags to improve code generation.

Acknowledgments. Work partially supported by MICINN (TIN2009-14317-C03-01) and ANPCYT Project PICT 4187.6.

References

1. Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): A Modeling Language for Designing Web Sites. *Computer Networks and ISDN Systems* 33(1-6), 137–157 (2000)
2. Koch, N., Knapp, A., Zhang, G., Baumeister, H.: UML-Based Web Engineering, an Approach Based on Standards. In: *Web Engineering, Modelling and Implementing Web Applications*, pp. 157–191. Springer, Heidelberg (2008)
3. Ambler, S.: *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process*. J. Wiley, Chichester (2002)
4. Ferreira, J., Noble, J., Biddle, R.: Agile Development Iterations and UI Design. In: *AGILE 2007 Conference*, pp. 50–58. IEEE Computer Society, Washington, DC (2007)
5. Ton, H.: A Strategy for Balancing Business Value and Story Size. In: *Agile 2007 Conference*, pp. 279–284. IEEE Computer Society, Washington, DC (2007)
6. Ricca, F., Scanniello, G., Torchiano, M., Reggio, G., Astesiano, E.: *On the Effectiveness of Screen Mockups in Requirements Engineering*. ACM Press, New York (2010)
7. Cohn, M.: *User Stories Applied: for Agile Software Development*. Addison-Wesley, Reading (2004)
8. Moore, J.M.: Communicating Requirements Using End-User GUI Constructions with Argumentation. In: *18th IEEE International Conference on Automated Software Engineering*, pp. 360–363. IEEE Computer Society, Los Alamitos (2003)

9. Panach, J.I., España, S., Pederiva, I., Pastor, O.: Capturing Interaction Re-quirements in a Model Transformation Technology Based on MDA. *J. UCS* 14, 1480–1495 (2008)
10. Mukasa, K.S., Kaindl, H.: An Integration of Requirements and User Interface Specifications. In: 6th IEEE International Requirements Engineering Conference, pp. 327–328. IEEE Computer Society, Los Alamitos (2008)
11. Ramdoyal, R., Cleve, A., Hainaut, J.-L.: Reverse Engineering User Interfaces for Interactive Database Conceptual Analysis. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 332–347. Springer, Heidelberg (2010)
12. Rivero, J.M., Rossi, G., Grigera, J., Burella, J., Robles Luna, E., Gordillo, S.E.: From Mockups to User Interface Models: An Extensible Model Driven Approach. In: Daniel, F., Facca, F.M. (eds.) ICWE 2010. LNCS, vol. 6385, pp. 13–24. Springer, Heidelberg (2010)
13. Rivero, J.M., Grigera, J., Rossi, G., Luna, E.R., Koch, N.: Improving Agility in Model-Driven Web Engineering. In: CAiSE 2011 Forum Proceedings, pp. 163–170. CEUR, London (2011)

Automatic Web Information Extraction Based on Rules*

Fanghuai Hu, Tong Ruan, Zhiqing Shao, and Jun Ding

Department of Computer Science and Engineering,
East China University of Science and Technology

xiaohuqi@126.com, {ruantong,zshao}@ecust.edu.cn, dingjun1989@hotmail.com

Abstract. Web Information Extraction is the initial step of effective web mining. In this article a few heuristic rules which describe the characteristics of the main content of web pages are summarized. The rules are constructed by some pre-defined terms and metrics, which can be considered as reusable and extensible for different kinds of HTML pages. Afterwards, a probabilistic model which utilizes the rules and metrics is suggested and the corresponding algorithm is implemented. The algorithm is tested on 1000 randomly selected web pages. The experiment shows that the algorithm is more precise and more applicable to the diverse structure of different web sites than other algorithms.

Keywords: Web Information Extraction, Probabilistic Model, Rule.

1 Introduction

In order to find useful and high quality information from the Internet, various methods such as search engines, web mining [1], and some NLP techniques have been proposed and applied. The process of extracting the main content (MC) and removing noisy information from a web page is called web information extraction (webIE). Here webIE is different from traditional information extraction (IE) specified in MUC-6 [13] and MUC-7 [14], which deals with extracting specific events or relations from a document collection.

The major difficulty of the task is that HTML pages are full of layout tags and have hardly any semantics. After a careful study of HTML pages from different web sites, we find that the MC block has several statistical characteristics which differ from those of other blocks. Based on this observation, an automatic webIE algorithm is developed.

2 Related Works

Former studies can be roughly divided into three categories.

* This research is supported by the National Science and Technology Pillar Program of China under grant number 2009BAH46B03 and National Nature Science Foundation of China under grant number 61003126.

- Wrapper based methods [2], [3]. Early researchers extracted data from web pages using wrappers. “A wrapper is a program that wraps an information source such that the information integration system can access that information source without changing its core query answering mechanism” [4]. The limitation of wrapper based methods is that they rely heavily on human interaction either to encode wrappers or to create patterns for semi-automatic wrapper generation.
- Template based methods [5], [6], [7]. For template based methods, it is assumed that target web pages were generated by a common template, and there should be more than one web page as input to induce templates. As a result, if we want to handle a page collection in which pages are picked from different web sites, the template based methods will not work.
- Statistic based methods [8], [9], [10]. These methods extract information via different statistical analyses of the content of web pages. [8] assumed that main contents of pages are included in <table> tags, and defined the “entropy” of a <table> tag; they then extract information by analyzing the “entropy” of each <table> tag. [10] found that content area has more text than other blocks and suggested a metric called “Text-to-Tag Ratio”; they calculated the ratio of every line first, and then used several clustering algorithms to distinguish the content area from other areas. These methods are somewhat similar to our approach in that special characteristics of MC are identified. However, for “entropy” based methods, the hypothesis that web pages are organized by <table> tags is no longer valid today, since DIV and CSS are suggested as the best practice to lay out HTML pages. Also, the “Text-to-Tag Ratio” only discovers one characteristic of the MC, and it needs complex clustering algorithms to analyze the statistical result.

3 Term Definitions

This section introduces a number of terms which are reusable and extensible for analyzing web page content; we will use them throughout the paper.

1. HTML tag. HTML tags are the basic elements of HTML pages. The sorts of tags and their usage in HTML are defined in the HTML specification.
 $HT = \{ht \mid ht \text{ is a kind of HTML tag}\}.$ $HT_S = \{\langle br / \rangle, \langle p \rangle\}$
 $HT_L = \{\langle a / \rangle\}$ $HT_N = HT - (HT_S \cup HT_L)$
2. Container HTML tag. HTML tags such as <td> and <div> are called container HTML tags, which usually contain the MC of a web page.
3. Information block. An information block refers to an information area in an HTML page, which consists of an HTML tag and the contents enclosed by the tag. Specially, ib_{MC} stands for MC block, and the first block whose text is the same as the text in <title> tag is called title block, marked as ib_T .
4. Container information block. An information block enclosed by a container HTML tag is called a container information block.
5. HTML document. An HTML document refers to a web page; it consists of a number of information blocks.

6. DOM [11] tree. Every valid HTML document can be parsed to a unique DOM tree. Intuitively, an information block is a node of the DOM tree.
7. Location. The location of an information block is the sequence number of the information block in the DOM tree where depth-first search traversing is applied. Specially, the locations of ib_T and ib_{MC} are denoted as LOC_T and LOC_{MC} respectively.

From the definitions above it can be concluded that an HTML document consists of many information blocks, each of which is in turn composed of an HTML tag and enclosed contents. An HTML page also has one exact corresponding DOM tree, and each information block is a node of the DOM tree.

4 Metric Definitions

The section defines metrics which are the foundation of the rules. We use IB as the set of all information blocks in an HTML document; ib is an element of IB , and i and j are two unequal integers between 1 and the size of IB .

Definition 1. *Content Similarity (CS).* Information blocks are represented as vectors, for example, $ib_i = (w_{1,i}, w_{2,i}, \dots, w_{t,i})$, and each dimension corresponds to a separate term. If a term occurs in the information block, its value in the vector is non-zero; the value used here is the *tf-idf* [12]. The CS of two information blocks (ib_i and ib_j) is defined by the vector space model [12]:

$$CS(ib_i, ib_j) = \frac{\sum_{k=1}^N w_{k,i} * w_{k,j}}{\sqrt{\sum_{k=1}^N w_{k,i}^2} * \sqrt{\sum_{k=1}^N w_{k,j}^2}} \quad (1)$$

Definition 2. *Location Weight (LW).* The LW of two information blocks (ib_i and ib_j) can be defined as:

$$LW(ib_i, ib_j) = \begin{cases} \left(\frac{LOC_i}{LOC_j}\right)^{K_P} & (LOC_i \leq LOC_j) \\ \left(\frac{PL-LOC_i}{PL-LOC_j}\right)^{K_P} & (LOC_i > LOC_j) \end{cases} \quad (2)$$

where PL is the total number of DOM tree nodes, and K_P is an empirical parameter, normally we use $K_P = 5$.

Definition 3. *Word Quantity (WQ).* The WQ of an information block is the total number of words, except for tag words, defined as:

$$WQ(ib_i) = \text{number of all words in } ib_i - \text{number of all tag words in } ib_i \quad (3)$$

Definition 4. *Tag Quantity (TQ).* The TQ of an information block is the total number of HTML tags, defined as:

$$TQ(ib_i, TAGTYPE) = \text{number of special types of tags in } ib_i \quad (4)$$

where $TAGTYPE$ can be HT_S , HT_L , HT_N , or HT .

Definition 5. *Tag Factor (TF).* The TF of an information block illustrates the effect of the HTML tag type in determining whether the block is the MC block. It can be defined as:

$$TF(ib_i) = \begin{cases} 1 + \frac{1}{TTQ(ib_i)} & (TTQ(ib_i) > 1) \\ 2 & (-1 \leq TTQ(ib_i) \leq 1) \\ 1 + |TTQ(ib_i)| & (TTQ(ib_i) < -1) \end{cases} \quad (5)$$

where TTQ is computed by the following formula, and K_L and K_S are also empirical parameters. Normally we use $K_L = 2$ and $K_S = 3$.

$$TTQ(ib_i) = TQ(ib_i, HT_N) + K_L * TQ(ib_i, HT_L) - K_S * TQ(ib_i, HT_S) \quad (6)$$

Definition 6. *Probability (P).* $P(e)$ means the probability of the occurrence of event e , whose value is between 0 and 1.

5 Heuristic Rules

We find that an MC block has several statistical characteristics which differ from those of other blocks. We also find that the MCs are all contained by container HTML tags, and our algorithm takes `<td>` and `<div>` as the container HTML tags, but we do not limit the types of tags which can be container HTML tags. Another thing we need to mention is that from our inspection, there are no MC blocks that contain more than one layer of container HTML tags. Therefore, we only analyze the container information blocks which contain no more than one layer of container HTML tags, and it can be set as more layers if required.

Rule 1: The MC should be more relevant to the title than other blocks. The title should outline the meaning of content, since the title is a hint to readers and search engine crawlers. Certain search engines even punish web sites for irrelevant conditions. The rule is described formally as follows:

$$(\forall ib_i, ib_j)(CS(ib_i, ib_T) > CS(ib_j, ib_T) \rightarrow P(ib_i = ib_{MC}) > P(ib_j = ib_{MC}))$$

Rule 2: The MC is usually close to the title. Although there may be a few illustrative contents such as author, information source, and date of publication between the MC and the title, in general the MC is close to the title. Formally,

$$(\forall ib_i, ib_j)(LW(ib_i, ib_T) > LW(ib_j, ib_T) \rightarrow P(ib_i = ib_{MC}) > P(ib_j = ib_{MC}))$$

Rule 3: The word quantity of the MC is bigger than that of other blocks. The MC contains information which is the authors' main intention and the readers' objective. As a result there should be many words in the MC and few hyperlinks. On the contrary, noisy information is used for navigation or short descriptions; they have few words but many hyperlinks. Formally:

$$(\forall ib_i, ib_j)(WQ(ib_i) > WQ(ib_j) \rightarrow P(ib_i = ib_{MC}) > P(ib_j = ib_{MC}))$$

Rule 4: There are few tags in the MC, except for the tags in HT_S . It is expected that MC has fewer tags compared to other blocks. That is the reason why [10] uses the text-to-tag rate as a metric to identify the content area. However this is not usually true in reality. Tags such as `
` and `<p>` are frequently used to modify the content, and furthermore some may include a few tags such as

$\langle b \rangle$, $\langle i \rangle$, and hyperlinks for the purpose of Search Engine Optimization (SEO). The rule can be measured by TF, formally,
 $(\forall ib_i, ib_j)(TF(ib_i) > TF(ib_j) \rightarrow P(ib_i = ib_{MC}) > P(ib_j = ib_{MC}))$

6 Model and Algorithm Based on Rules

6.1 Model

The set of container information blocks in an HTML document is denoted as CIB . For $ib \in CIB$, we can calculate the $CS(ib, ib_T)$, $LW(ib, ib_T)$, $WQ(ib)$, and $TF(ib)$. Considering “ ib is ib_{MC} ” as an outcome ω , all of the outcomes make up a sample space Ω . The object of extracting the MC can be modeled to calculate the conditional probability as follows (for convenience, we use CS , LW , WQ , and TF instead of $CS(ib_i, ib_T)$, $LW(ib_i, ib_T)$, $WQ(ib_i)$, and $TF(ib_i)$, respectively, and ev stands for the event $ib = ib_{MC}$):

$$P(ev|CS, LW, WQ, TF) \quad (7)$$

That is, given the value of each metric of the ib , how probable is it that ib is the ib_{MC} ? To calculate Formula 7, we make two independent hypotheses here:

- Metrics are independent from each other, that is:
 $P(CS, LW, WQ, TF) = P(CS) * P(LW) * P(WQ) * P(TF)$
- Metrics are independent from each other when given the ib 's value:
 $P(CS, LW, WQ, TF|ev) = P(CS|ev) * P(LW|ev) * P(WQ|ev) * P(TF|ev)$

Now we can use Bayes' theorem [15] to deduce Formula 7:

$$\begin{aligned} & P(ev|CS, LW, WQ, TF) \\ &= \frac{P(CS, LW, WQ, TF|ev) * P(ev)}{P(CS, LW, WQ, TF)} \quad (a) \\ &= \frac{((P(CS|ev) * P(LW|ev) * P(WQ|ev) * P(TF|ev)) * P(ev))}{P(CS, LW, WQ, TF)} \quad (b) \\ &= \frac{\frac{P(ev|CS) * P(CS)}{P(ev)} * \frac{P(ev|LW) * P(LW)}{P(ev)} * \frac{P(ev|WQ) * P(WQ)}{P(ev)} * \frac{P(ev|TF) * P(TF)}{P(ev)} * P(ev)}{P(CS, LW, WQ, TF)} \quad (c) \\ &= \frac{P(ev|CS) * P(ev|LW) * P(ev|WQ) * P(ev|TF)}{P(ev) * P(ev) * P(ev)} \quad (d) \end{aligned} \quad (8)$$

Where step (a) and step (c) are deduced by Bayes' theorem, and step (b) and step (d) use the independence hypothesis. Since $p(ev)$ is the prior probability, which is the same for all information blocks, $P(ev|CS, LW, WQ, TF)$ is in direct proportion to $P(ev|CS) * P(ev|LOC) * P(ev|WQ) * P(ev|TF)$. For an arbitrary ω_i , its conditional probability for each metric is computed by the following formula (take CS for example):

$$P(ev|CS)_i = \frac{v_i}{\sum_{i=1}^n v_i} \quad (9)$$

Where v_i is the value of ib_i in a particular metric. It is self-evident that from Formula 9, $P(ev|CS)_i \in [0, 1]$ and $\sum_{i=1}^n P(ev|CS)_i = 1$.

6.2 Algorithm Description

The algorithm has four main steps: (1) Filtering the input HTML Document and parsing it into DOM tree, and then get all container information blocks. (2) Calculating metrics and the final probability values of each container information block. (3) Analyzing the results and finding the MC block, the progress is: Choose the two blocks which have the maximum value, and if the value of the first one is obviously bigger (here we choose 10 times bigger) than the second one, we take the block corresponding to the first one as the MC block; Otherwise, we need to judge whether the two blocks are nested; If nested we select the outer block as our result; otherwise, we select the block with the biggest value. (4) Extracting content from the MC block and filtering some useless content.

7 Experiments

To show that our algorithm can adapt to different kinds of web pages, we take 1000 randomly selected web pages as experimental data from different kinds of sites, 100 pages from each site. The sources cover public information portals, vertical industry portals, enterprise information portals, and government websites. There are a few pages that mainly consist of pictures or videos, and we remove these pages from the precision calculation as many IE algorithms do [10].

7.1 Experimental Results of a Single Page

Table 1 is the result of a web page whose url is “http://ca.news.yahoo.com/blogs/dailybrew/japan-earthquake-shifted-balance-planet-20110314-065608-417.html”; we show only the five maximum blocks. The P of the first block is much larger than the others (more than 10 times), so we choose it as the MC block.

In Table 1, every metric of the first block is larger than the others. However, this is not true in all cases. For a web page with short MC block but long comments, it will have other blocks with bigger WQ than the MC block, but the probabilistic value of the MC block will still be the biggest under the influence of other three metrics.

7.2 Experimental Results of Different Sites

The results of the randomly chosen 1000 pages are shown in Table 2. For 6 out of 10 web sites, the precision reaches 100%; the worst result comes from “http://forex.hexun.com/hotnews/”, with an precision of 95%, but this is still better than most results reported in literatures. For the 11 pages that have not been correctly extracted, we have analyzed them carefully. Some of them are not well structured, they lead to bad DOM trees, which in turn lead to incorrect analysis and metrics calculations. Another cause is that the MC is too short, and there are other blocks have longer text near the title.

Table 1. Result of an HTML document

<i>No.</i>	<i>CS</i>	<i>LW</i>	<i>WQ</i>	<i>TF</i>	<i>P</i>
1	0.57951	0.42927	0.55390	0.13169	0.01814
2	0.07335	0.23564	0.00387	0.03292	2.20E-6
3	0.11867	0.00273	0.19893	0.00412	2.65E-7
4	0.01681	0.01638	0.03216	0.00329	2.92E-8
5	0.07908	0.00514	0.02621	0.00329	4.25E-17

Table 2. Results of different websites

Websites	Total	Correct	Precision(%)
http://news.yahoo.com/sports/	100	100	100
http://www.foxnews.com/sports/index.html	100	100	100
http://roll.mil.news.sina.com.cn/col/gjjq/index.shtml	99	99	100
http://news.163.com/special/00013C00/guojibjtj.html	100	100	100
http://www.xinhuanet.com/world/tt.htm	96	94	97.92
http://forex.hexun.com/hotnews/	100	95	95
http://www.cnforex.com/global/yanghang/	97	94	96.91
http://www.gov.cn/yjgl/tfsj.htm	100	100	100
http://www.gdemo.gov.cn/gzyw/gj/	100	100	100
http://world.huanqiu.com/well_read/	97	96	98.97

7.3 Comparison with Other webIE Algorithms

We compare our algorithm with the two statistic based algorithms, including the “Text-to-Tag Ratio” (TTR) algorithm [10] and the algorithm based on the “entropy” of <table> [8]. First, the precision of our algorithm is 0.989, which is better than 0.9419 of the TTR algorithm and 0.956 of the entropy based algorithm. Second, like the TTR algorithm, our algorithm makes no assumptions, but the entropy based algorithm assumes that web pages are organized by <table> tags, which is not valid today. Finally, in the results analysis process, our algorithm only needs simple calculations such as float comparisons; in contrast, the TTR based method needs an additional complicated clustering step, and it is hard for the entropy based method to automatically choose the optimum entropy which is different for various pages.

8 Conclusions and Future Works

In this paper we proposed a rule-based webIE algorithm based on the characteristics of the page content and a probabilistic model. The algorithm demonstrated the highest precision among the algorithms in the literature. It was also demonstrated that the algorithm need no pre-assumptions and is adaptable to different sorts of web sites without any manual work.

It makes sense to reuse the same idea behind the metrics, rules, and probabilistic model to extract contents from other web sources such as forums, micro

blogs, and social networks. To achieve this we need to find more metrics and rules suitable for these resources, although the idea should be the same. We are also trying to develop a universal HTML text processing language which can be used to extract information from diverse information resources.

References

1. Chakrabarti, S.: Mining the Web: Discovering Knowledge from Hypertext Data. Morgan Kaufmann, San Francisco (2002)
2. Muslea, I., Minton, S., Knoblock, C.: Hierarchical Wrapper Induction for Semistructured Information Sources. *J. Auton. Agent. Multi-Ag.* 4, 93–114 (2001)
3. Pan, A., Raposo, J., Alvarez, M., Hidalgo, J., Vina, A.: Semi-Automatic Wrapper Generation for Commercial Web Sources. In: IFIP Working Conference on Engineering Information Systems in the Internet Context, pp. 265–283. Kluwer Academic Publishers, Norwell (2002)
4. Chang, C.H., Kayed, M., Girgis, M.R., Shaalan, K.: A Survey of Web Information Extraction Systems. *J. IEEE T. Knowl. Data En.* 18, 1411–1428 (2006)
5. Ji, X.W., Zeng, J.P., Shang, S.Y., Wu, C.R.: Tag Tree Template for Web Information and Schema Extraction. *J. Expert Syst. Appl.* 37, 8492–8498 (2010)
6. Crescenzi, V., Mecca, G.: Automatic Information Extraction from Large Websites. *J. ACM* 51, 731–779 (2004)
7. Wang, J.Y., Lochovsky, F.H.: Data-Rich Section Extraction from HTML Pages. In: Proceedings of the 3rd International Conference on Web Information Systems Engineering, pp. 313–322. IEEE Computer Soc., Los Alamitos (2002)
8. Lin, S.H., Ho, J.M.: Discovering Informative Content Blocks from Web Documents. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York (2002)
9. Mengel, S., Jing, Y.: Extracting Structured Data from Web Pages with Maximum Entropy Segmental Markov Model. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) WISE 2009. LNCS, vol. 5802, pp. 219–226. Springer, Heidelberg (2009)
10. Weninger, T., Hsu, W.H.: Text Extraction from the Web via Text-to-Tag Ratio. In: 19th International Workshop on Database and Expert Systems Application, pp. 23–28. IEEE Computer Soc., Los Alamitos (2008)
11. Document Object Model, <http://www.w3.org/DOM/>
12. Salton, G., Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. *J. Inf. Process. Manag.* 24(5), 513–523 (1988)
13. Grishman, R., Sundheim, B.: Message Understanding Conference 6: A Brief History. In: Proceedings of the 16th Conference on Computational Linguistics. Association for Computational Linguistics, Stroudsburg (1996)
14. MUC-7 Information Extraction Task Definition, http://www-nlpir.nist.gov/related_projects/muc/proceedings/ie_task.html
15. Lewis, D.D.: Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 4–15. Springer, Heidelberg (1998)

An Artifact-Centric View-Based Approach to Modeling Inter-organizational Business Processes

Sira Yongchareon¹, Chengfei Liu¹, and Xiaohui Zhao²

¹ Faculty of Information and Communication Technologies
Swinburne University of Technology, Victoria, Australia
{syongchareon, cliu}@swin.edu.au

² Department of Computing, Faculty of Creative Industries and Business
Unitec Institute of Technology, Mount Albert, New Zealand
xzhao@unitec.ac.nz

Abstract. Over past several years, there have been increasing needs for more efficient approaches to the design and implementation of inter-organizational business processes. The process collaboration spanning organizational boundaries is deemed to keep the organization autonomy, which means each organization owns its freedom of modifying internal operations to meet their private goals while satisfying the mutual objective with its partners. To achieve these, we propose an artifact-centric view-based framework for inter-organizational process modeling consisting of an *artifact-centric collaboration model*, and a *conformance* mechanism between public view and private view to support the participating organization's customization and changes of their internal operations while ensuring the correctness of collaboration process.

1 Introduction

Over past years, service-oriented architecture (SOA) has been serving an increasing demand of IT for businesses to meet the challenges of the ever-changing market [7]. SOA particularly enables the business collaboration across organizations by the means of web service composition to achieve the mutual goal of collaboration as well as the individual goal of each participant. Service choreography is used to specify the interaction between business parties whereby providing the agreed behavioral contract between collaboration participants. Although service choreography technology intends to provide a global view of the collaboration and allow each party to design and implement its own portion, the existing choreography modeling approaches and languages mainly describe the collaboration from the procedural perspective, and focus on control-flow, message sequencing, etc. With this limited focus, the current service choreography approaches cannot well support the three major requirements of the collaboration: *compliance*, *flexibility*, and *autonomy*. The *compliance* refers to the expectation that all parties must provide the services as they have agreed in the contract. The *flexibility* means that each party has the freedom to change and implement its own part in the collaboration while remaining *autonomous*.

In recent years, a new approach for modeling business processes has emerged, namely *artifact-centric process* modeling, providing a higher level of robustness and flexibility for describing process specification than traditional process-centric

approaches [1, 2, 3, 4, 12]. In the process coordination, we observed that at a particular state a message exchanged between organizations reflects the attribute changes of the artifacts used in an organization’s internal process. Some states of the artifacts of each individual party link with the global constraints in the collaboration. Based on this observation and the three requirements, we propose a new paradigm of inter-organizational process modeling namely *artifact-centric collaboration model* together with the notion of *conformance* between public and private views to tackle the compliance, autonomy, and flexibility issues of business collaboration.

The remainder of this paper is organized as follows. Section 2 introduces an artifact-centric approach for modeling inter-organizational business processes. Section 3 discusses the construction of public and private views. Section 4 reviews the related works. Finally, the conclusion and future work are given in Section 5.

2 Modeling Inter-organizational Business Processes

In this section, we take supply chain collaboration as an example to illustrate and motivate the artifact-centric approach to modeling inter-organizational business collaboration, as shown Fig. 1. We initiate the discussion of this example by identifying the types of involved business artifacts and describing how they are modeled for this business collaboration. Here, we classify artifacts into two types: (1) *local artifact* and (2) *shared artifact*. A *local artifact* is used internally within one organization and can be used for supporting the coordination between intra-organizational processes and inter-organizational processes. A *shared artifact* serves as a contract between involved organizations, and it is used to indicate the agreed business stages towards the completion of the collaborative process. Note that in the implementation, shared artifacts act as messages sent and received between the organizations. In Fig. 1 (a), the Supplier will ship the order only if the state of the *Purchase Order (PO)* is in the *confirmed* state and the designated logistic company is arranged by using the *Shipping List (SL)* and *Shipping Order (SO)* artifacts. We draw the interrelation between artifacts as *synchronization dependency* (dashed-line).

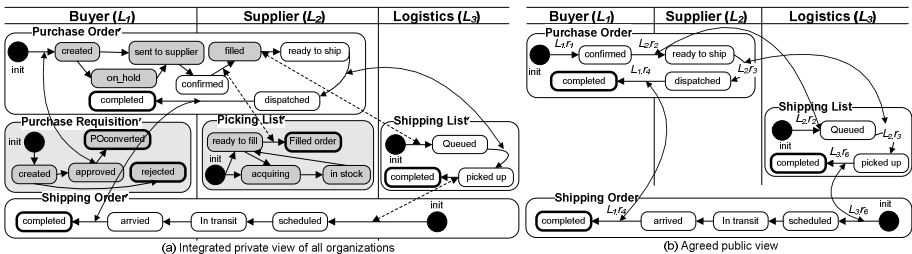


Fig. 1. Overall collaboration and its public view of supply chain business processes

In the view-based approach to modeling inter-organizational processes, e.g., [7, 10, 11], a *public view* has to be constructed at the first stage based on the integration of individual local process of each organization in the collaboration. For

our artifact-centric model, only shared artifacts and their necessary (goal) states that are required to be globally visible to other parties appear in the public view. After the public view is built, the involved organizations should also have freedom to change their own parts (i.e., their responsible parts of shared artifacts and their local artifacts) later. As such, it is important to guarantee that such changes in the local process, or *private view*, do not influence the correctness of the overall collaboration. We propose the notion of *view conformance* as the validation mechanism of such changes in the private view w.r.t. the public view. Due to page limitation, we do not detail how public view is constructed in this paper; however, it can be achieved by adapting artifact-centric process abstraction presented in our previous work [4].

Now, we define *Artifact-Centric Collaboration Model (ACC model)* extended from the artifact-centric process model (*ACP model*) [3, 4, 16] to capture inter-organizational processes. It consists of four core constructs: *roles*, *artifacts*, *services*, and *business rules*. An *artifact* is either a *local-typed* or a *shared-typed* business entity or an object involved in inter-organizational processes. An artifact class C is a tuple (A, S, S^f) where A is set of a name-value pair attribute of scalar-typed value or nested attribute structure (array-typed), S is a set of states, and $S^f \subseteq S \setminus \{s^{init}\}$ is a set of *final* states where s^{init} denotes *initial* state. A *service* is a task owned by one organization in the collaboration and is used to perform (read/write) operations on artifact(s). A *business rule* is defined by a *condition-action* style, which describes what condition a service is invoked, and what (post) conditions attributes and state of artifacts must satisfy after the invocation. Business rule r is triple (λ, β, ν) where λ and β are *pre-condition* and *post-condition*, respectively, containing two types of propositions: state proposition and attribute proposition; and ν is a service to be performed. A business rule can be used to synchronize two or more artifacts, called *synchronization dependency*. We denote D for its set, e.g., in Fig. 1, $D(PO, SO) = \{L_1.r_4\}$ and $D(PO, SL) = \{L_2.r_2, L_2.r_3\}$. Note that the synchronization dependency is transitive.

Definition 1: (Artifact-Centric Collaboration Model or ACC model). Let Π denote an *ACC model*, and it is tuple (Z, V, R, L, ZL, VL) where,

- Z is a set of artifact classes, V is a set of services, and R is a set of business rules
- L is a set of organization roles participating in the collaboration
- $ZL \subseteq Z \times 2^L$ is a set of *artifact-role* relations between artifacts and their corresponding set of organization roles. Relation $(C_i, \{l_1, l_2, \dots, l_x\}) \in ZL$ means that organization roles l_1, l_2, \dots, l_x involve in the changes of states of artifact C_i
- $VL \subseteq V \times L$ is a set of *service-role* relations between services and their corresponding organization roles. Relation $(v_i, l_x) \in VL$ means that service v_i is provided by organization role l_x .

Given ACC model Π , a *lifecycle model* of an artifact can be generated by deriving corresponding business rules that are used to induce state transitions of such artifact. A *lifecycle* of artifact class C_i , denoted as \mathcal{L}_{C_i} , can be defined as tuple (C_i, T) , where $T \subseteq C_i.S \times \Pi.R \times (C_i.S \cup C_i.S^f)$ is a 3-ary transition relation. T^* is a reflexive transitive closure of T . Next, we define the *lifecycle occurrence (L-occurrence)* for a particular sequence of states occurring in the lifecycle. Based on *L-occurrences*, we define a *well-formed* behavior property of an individual lifecycle in which is used

later for the discussion of a *soundness* property of the ACC model. A *L-occurrence* of lifecycle \mathcal{L}_{C_i} is denoted as $\sigma = (s^{init}, \dots, s_f)$ such that for every state s in σ , there exists final state $s_f \in C_i.S^f$ and s_f can be reached from s^{init} through s by a particular firing sequence of some business rules in R . We also denote $\delta_{C_i} = \{\sigma_1, \sigma_2, \dots, \sigma_x\}$ for a pairwise disjoint set of all possible *L-occurrences* in \mathcal{L}_{C_i} . Now, given ACC model Π , Lifecycle \mathcal{L}_{C_i} of $C_i \in \Pi.Z$ is *well-formed* iff (1) there exists business rule $r \in \Pi.R$ such that r induce one and only one transition in \mathcal{L}_{C_i} ; and (2) for every non-final state $s \in S_i$, there exists s in some *L-occurrence* of \mathcal{L}_{C_i} ; and (3) for every final state $s_f \in S_i^f$, there exists *L-occurrence* $\sigma \in \delta_{C_i}$ such that s_f is a last state of the sequence in σ .

Definition 2: (Public and Private views). Given ACC model Π , a *public view* of Π , denoted as $p(\Pi)$, is the abstraction of Π , such that it represents only agreed (abstracted) lifecycles of shared artifacts in Π . *Private view* Π^l is the local process of Π for organization role $l \in \Pi.L$ such that it represents only lifecycles of its own local artifacts and the lifecycles of shared artifacts in $p(\Pi)$.

As already discussed, when an agreed public view is constructed, involved organizations can have their own freedom to modify their own private view. Here, we define two behavioral-based modification operations, namely *lifecycle modification*: (1) *refine* (responsible parts of) the lifecycle of shared artifacts, and (2) *extend* shared artifacts with local artifact(s) that are required to coordinate with such shared artifacts. Note that refining existing local artifacts is also considered as extending them to the public view. To refine a shared artifact in a private view, we define *lifecycle fragment* representing a partial lifecycle that is embedded into its existing lifecycle.

Definition 3: (Lifecycle Fragment or L-fragment). Given ACC model Π , artifact $C_i \in \Pi.Z$, and a set of states $S = C_i.S \cup C_i.S^f$, we denote lf^{C_i} for a *L-fragment* which is a nonempty connected sub-lifecycle of lifecycle \mathcal{L}_{C_i} . Let lf^{C_i} be a 4-tuple $(S', T', T_{in}, T_{out})$ where $S' \subseteq C_i.S \setminus \{s^{init}\}$, $T' \subseteq S' \times \Pi.R \times S' \subseteq \mathcal{L}_{C_i}.T$, a set of entry transitions $T_{in} = \mathcal{L}_{C_i}.T \cap ((S \setminus S') \times \Pi.R \times S')$, and a set of exit transitions $T_{out} = \mathcal{L}_{C_i}.T \cap (S' \times \Pi.R \times (S \setminus S'))$ such that for every state $s \in S'$, there exist s in *L-occurrences* δ of lf^{C_i} .

Next, we define *lifecycle modification* in a private view for the refinement (by *L-fragments*) of shared artifact and the extension of local artifacts.

Definition 4: (Lifecycle modification). Given public view $p(\Pi)$, private view Π^l can be modified based on $p(\Pi)$ with a *lifecycle modification*, denoted as Σ^l , and it is tuple (LM^l, LF^l) , where LM^l is a set of extended lifecycles (of local artifacts) and LF^l is a set of *L-fragments* $\bigcup_{i=1}^{|Z^l|} \{lf_j^{C_i^l}\}$, $lf_j^{C_i^l} = \{lf_1^{C_i^l}, lf_2^{C_i^l}, \dots, lf_k^{C_i^l}\}$, where $lf_n^{C_i^l}$ ($1 < n < k$) is a *L-fragment* of $C_i^l \in \Pi^l.Z$ refining the shared lifecycle of its C_i in $p(\Pi)$.

Fig. 2 shows an example of public and a private view of organization L_1 in the collaboration where private view Π^{L_1} modify its public view with a lifecycle modification consisting of extended local lifecycles of artifacts $\{C_1, C_2, C_3\}$, refined *L-fragments* $\{lf_1^{C_A}, lf_2^{C_A}\}$ on artifact C_A , and *L-fragment* $lf_3^{C_B}$ on artifact C_B . The

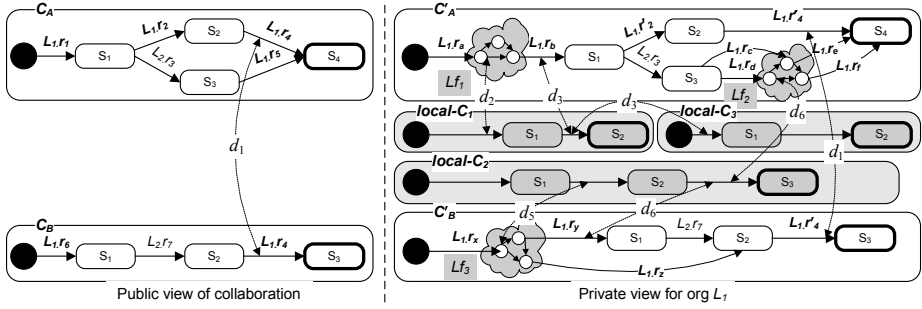


Fig. 2. Public view and its private view with lifecycle modification

organization also adds or modifies business rules used to synchronize local artifacts with shared artifacts (within the refined L -fragments) to complete the overall process.

3 Validating Changes to Private View of a Collaboration Model

In this section, we discuss about the validation of changes made to private views caused by *lifecycle modification*, i.e., how to ensure that the local changes to the private view of one organization do not interfere with the whole collaboration. Note that we confine our discussion only to the behavioural aspect of collaboration not the validation of artifacts' data. However, formal approaches to data verification can be found in [3]. Here, we use state transition system to describe the behavioral aspect of the ACC model, which can be generated by a composition technique presented in [6].

A state transition system of ACC model Π , namely *ACC machine* and denoted as M_Π , is tuple (S, s^{init}, T, S^f) , where S and s^{init} are a set of ACC states and the initial state, respectively. $T \subseteq S \cup \{s^{init}\} \times \Pi.R \times G \times S$ is a set of transitions and G is a set of guards (state propositions). $S^f \subseteq S$ is a set of final states such that for every state s in S^f , s contains a combination of final state of every artifact in Π . M_Π is generated by iteratively composing a lifecycle of each artifact in Π .

Definition 5: (Soundness). Given ACC model Π and its ACC machine M_Π , Π is *sound* iff (1) for every artifact $C_i \in \Pi.Z$, the lifecycle of C_i is *well-formed*; and (2) for every artifact $C_i \in \Pi.Z$, there exists a synchronization dependency between C_i and some other artifact; and (3) M_Π is *well-formed* and for every transition $t \in M_\Pi.T$, guard g of t contains no state referencing of any other artifact.

Condition (2) of the *soundness* property guarantees that the ACC model will consist of only desired artifacts that are used in the collaboration, and the condition (3) ensures the *reachability of goal states* of the collaboration and it also implies that a domain of artifacts and their states referenced in the model is bounded. Next, we define the *view conformance* between public and private views by checking whether the lifecycle of the later covers the former. We generalize *lifecycle coverage* for both artifact and ACC machine by reusing the L -occurrences definition of an individual artifact for a machine. Given two machines $m_x = (S_x, s_x^{init}, T_x, S_x^f)$ and

$m_y = (S_y, S_y^{init}, T_y, S_y^f)$ and a set of states that exist in both $m_x.S$ and $m_y.S$, $S_{x \cap y} = S_x \cap S_y$, L -occurrences of m_x is covered by L -occurrences of m_y iff $\forall s_i, s_j \in S_{x \cap y}, \exists (s_i, r, s_j) \in T_x, \forall s_k \in S_y \setminus S_{x \cap y}, s_i T_y^* s_k \wedge s_k T_y^* s_j$. Fig. 3 shows that lifecycle (a) is not covered by (b) as in the occurrences of (b) state a can reach state c without passing state b ; in contrast, lifecycle (a) is covered by (c).

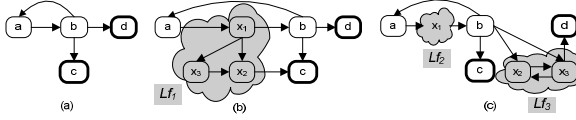


Fig. 3. Example of different refinement of L -fragments

Definition 6: (View conformance). Given private view Π^l modified based on public view $p(\Pi)$, Π^l conforms to $p(\Pi)$ iff L -occurrences of the ACC machine of $p(\Pi)$ is covered by L -occurrences of the ACC machine of Π^l .

Although the *view conformance* is defined based on the overall behavior of the model, it is important to discuss how we can validate the local modification. If such modification is valid and correctly applied, then we can guarantee that it will preserve the consistent behavior between the public and private views. Specifically, we apply the lifecycle coverage checking between public and private views of a shared artifact refined by L -fragments, and then we validate the whole modification with the extended local artifacts in the private view. Now, given private view Π^l modified based on *sound* public view $p(\Pi)$ with *lifecycle modification* Σ^l , Σ^l is said to be *valid* iff Σ^l can make Π^l *sound* and Π^l conforms to $p(\Pi)$. Let the lifecycle of C_i^l in private view Π^l refine its public lifecycle of corresponding artifact C_i in public view $p(\Pi)$ with a set of L -fragments $LF^{C_i^l}$. L -occurrences of C_i is covered by L -occurrences of C_i^l iff, for every $lf_j \in LF^{C_i^l}$, lf_j has a single entry state and a single exit state. We name this lf_j as *SESE L -fragment*. For example, in Fig. 3, lifecycle (c) consists of *SESE L -fragments* lf_2 and lf_3 refining lifecycle (a); thus, lifecycle (c) covers lifecycle (a).

Next, we discuss how the synchronization is taken into account for the validation of private views. First, we define the *locally-bound* property of the lifecycle of local artifact that synchronizes with a shared artifact. Then, we use this property to induce the coverage of all related artifacts in the private view. Let private view Π^l be modified based on *sound* public view $p(\Pi)$. For any local artifact $C_x^l \in \Pi^l.Z$ and any shared artifact $C_y^l \in \Pi^l.Z \cap p(\Pi).Z$ which is refined by a set of L -fragments $LF^{C_y^l}$, if every synchronization dependency between lifecycle $\mathcal{L}_{C_x^l}^l$ of C_x^l and lifecycle $\mathcal{L}_{C_y^l}^l$ of C_y^l occurs within one and only one fragment $lf_j \in LF^{C_y^l}$ such that lf_j is *SESE L -fragment*, then $\mathcal{L}_{C_x^l}^l$ is *locally-bound* and $\mathcal{L}_{C_x^l}^l$ preserves the *soundness* of Π^l . The *locally-bound* property is transitive from one to another local artifact if a *locally-bound* artifact synchronizes with another local artifact such that they do not synchronize with any shared artifact, then such artifact is transitively *locally-bound*.

The transitivity is invalid if such artifact synchronizes with a *non locally-bound* artifact. For example in Fig. 2, lifecycle of C_1 is *locally-bound* since synchronizations d_2 and d_3 occur within *SESE fragment* lf_1 of C'_A , and the lifecycle of C_3 is transitively *locally-bound* via d_3 . In contrast, with C_2 , we can see that synchronizations d_5 and d_6 occur in non-*SESE L-fragment* lf_3 of artifact C'_B (even though d_6 synchronizes with *SESE L-fragment* lf_2); thus, the lifecycle of C_2 is not *locally-bound*. Now, if organizations want to modify their private views using lifecycle modification based on their public view, they need to ensure that their local modifications do not violate the view conformance. Here, we show the conditions that restrict the *valid* modification. *Lifecycle modification* $\Sigma^l = (LM^l, LF^l)$ is *valid* for Π^l based on *sound* $p(\Pi)$, if, for every *L-fragment* $lf_j \in LF^l$, lf_j is *SESE L-fragment* and for every lifecycle $\mathcal{L}_{C_i}^l \in LM^l$, $\mathcal{L}_{C_i}^l$ is *locally-bound*, then *L-occurrences* of ACC machine of Π^l is *covered* by *L-occurrences* of ACC machine of $p(\Pi)$ (i.e., Π^l conforms to $p(\Pi)$) and Π^l is *sound*. As the result, we can assert the *soundness* property with the *conformance* by only checking the local lifecycle modification. Whatever changes made on a private view will not impact on the behavior of collaboration if they preserve *valid* modification.

4 Related Work and Discussion

In the area of artifact-centric process modeling, Nigam and Caswell [2] introduced the concept of modelling business artifacts with lifecycles. Bhattacharya et al. [3] used services to model activities and a set of business rules to capture and represent a process model with the study of necessary properties such as reachability of goal states, absence of deadlocks, and redundancy of data. Fritz et al. [12] studied problems of goal-directed artifact-centric workflow construction. Hull et al. [1] proposed an approach to interoperation of organizations hubs based on business artifacts providing a centralized and computerized rendezvous point. Kuster et al. [8] presented a notion of compliance of a business process model with object lifecycles and a technique for generating the model from such set of lifecycles. In [4], we proposed an artifact-centric process view framework to allow role-based process abstraction that can be used to construct public views. All the above works serve as a basis for this work. In the area of object-oriented design, Schrefl and Stumptner [5] studied the consistency criteria of the inheritance of object life cycles. While their work focused on the inheritance of single object, we adopt and extend their study of consistent refinement to multiple lifecycles where the multiple inheritances of artifacts may interact with each other in the collaboration.

In the area of cross-organizational workflow management, several approaches presented in [10, 11, 13, 14, 15] were proposed to define, construct, and validate public and private process views based on consistency rules and correctness-preserving constraints. Van der Aalst et al. [7] proposed a process-oriented contract agreed in the public view with a criterion for accordance between public and private views. Their work is done based on process-centric setting while our work is on artifact-centric setting. Lohmann and Wolf [9] presented an approach for artifact-centric choreographies with the concept of agents and location-aware artifacts using Petri-net model. They proposed a mechanism for automatic generation of an

interaction model that serves as a contract between the agents ensuring that specified global goal states on the involved artifacts can be reached. Compared to our work, their work only focused on the interaction model and did not consider the model of local processes, while our approach presents the artifact-centric collaboration model together with the validation mechanism for local model modification where preserving global correctness.

5 Conclusions

This paper presents an artifact-centric view-based approach to modeling inter-organizational business processes. We define the ACC model with the notion of *conformance* of public and private views to address the choreography requirements. The view conformance checking serves as the main resolving mechanism. We also show that a modification on the private view of local organization can be theoretically validated against the *soundness* properties for both public and private views.

References

1. Hull, R., Narendra, N.C., Nigam, A.: Facilitating Workflow Interoperation Using Artifact-Centric Hubs. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 1–18. Springer, Heidelberg (2009)
2. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal 42(3), 428–445 (2003)
3. Bhattacharya, K., Gerede, C., Hull, R.: Towards Formal Analysis of Artifact-Centric Business Process Models. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 288–304. Springer, Heidelberg (2007)
4. Yongchareon, S., Liu, C.: Process View Framework for Artifact-Centric Business Processes. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6426, pp. 26–43. Springer, Heidelberg (2010)
5. Schrefl, M., Stumptner, M.: Behavior-Consistent Specialization of Object Life Cycles. ACM TOSEM 11(1), 92–148 (2002)
6. Lind-Nielsen, J., Andersen, H., Hulgaard, H., Behrmann, G., Kristoffersen, K., Larsen, K.: Verification of Large State/Event Systems Using Compositionality and Dependency Analysis. Formal Methods in System Design 18(1), 5–23 (2001)
7. Van Der Aalst, W.M.P., Lohmann, N., Masuthe, P., Stahl, C., Wolf, K.: Multipart Contrats: Agreeing and Implementing Interorganizational Processes. The Computer Journal 53(1), 90–106 (2010)
8. Küster, J.M., Ryndina, K., Gall, H.C.: Generation of Business Process Models for Object Life Cycle Compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 165–181. Springer, Heidelberg (2007)
9. Lohmann, N., Wolf, K.: Artifact-centric Choreographies. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6470, pp. 32–46. Springer, Heidelberg (2010)
10. Chebbi, I., Dustdar, S., Tata, S.: The view-based approach to dynamic inter-organizational workflow cooperation. DKE 56, 139–173 (2006)

11. Van Der Aalst, W.M.P., Lohmann, N., Massuthe, P., Stahl, C., Wolf, K.: From Public Views to Private Views – Correctness-by-Design for Services. In: Dumas, M., Heckel, R. (eds.) WS-FM 2007. LNCS, vol. 4937, pp. 139–153. Springer, Heidelberg (2008)
12. Fritz, C., Hull, R., Su, J.: Automatic construction of simple artifact-based business processes. In: ICDT 2009, vol. 361, pp. 225–238. ACM, New York (2009)
13. Zhao, X., Liu, C., Sadiq, W., Kowalkiewicz, M., Yongchareon, S.: Implementing Process Views in the Web Service Environment. WWWJ 14(1), 27–52 (2011)
14. Zhao, X., Liu, C., Sadiq, W., Kowalkiewicz, M.: Process View Derivation and Composition in a Dynamic Collaboration Environment. In: Chung, S. (ed.) OTM 2008, Part I. LNCS, vol. 5331, pp. 82–99. Springer, Heidelberg (2008)
15. Zhao, X., Liu, C., Yang, Y., Sadiq, W.: Aligning Collaborative Business Processes - An Organisation-oriented Perspective. IEEE TSMC 39(6), 1152–1164 (2009)
16. Yongchareon, S., Liu, C., Zhao, X., Xu, J.: An Artifact-centric Approach to Generating Web-based Business Process Driven User Interfaces. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 419–427. Springer, Heidelberg (2010)

Enhance Web Pages Genre Identification Using Neighboring Pages

Jia Zhu¹, Xiaofang Zhou², and Gabriel Fung³

¹ School of ITEE, The University of Queensland, Australia
jiazhu@itee.uq.edu.au

² School of ITEE, The University of Queensland, Australia
zxf@itee.uq.edu.au

³ IINGAB Lab, Hong Kong
gabriel@ingab.com

Abstract. Recently web pages genre identification attracts more attentions because of its importance in web searching. Most of existing works used the features extracted from web pages and applied machine learning approaches like SVM as classifier to identify the genre of web pages. However, in the case where web pages do not contain enough information, such an approach may not work well. In this paper, we consider to tackle genre identification in such situations. We propose a link-based graph model that taking into account neighboring pages but greatly reducing the noisy information by selecting an appropriate subset of neighboring pages. We evaluated this neighboring pages based classifier with other classifiers. The experiments conducted on two known corpora, and the favorable results indicated that our proposed approach is feasible.

1 Introduction

As the World Wide Web continues to grow exponentially, web pages classification becomes extremely important in web searching. Web pages genre identification, as a sub category of web pages classification, focuses on functional purposes and discovering groups of texts compared to topic identification. For example, a website like www.news.com.au, has many topics in the pages, e.g. sport news and finance news, but the genre of these pages in this website is *news*.

The most important component to identify web genre is to select an appropriate feature set. As the word *genre*, genre identification focuses on the properties in the pages include textural information [12], e.g. term frequency, and structural information [2], e.g. HTML tags, because textural information often represents the content of a web page while structural information is about how the content is presented. However, these features are sometimes missing or unrecognizable for various reasons. For example, those web pages contain large images or video objects but little textual content(see Figure 1). In such case, it is difficult for a classifier to correctly identify the genre of a web page. Our approach uses additional information from web pages' neighboring pages because more information could provide more evidence for classifiers to learn the classification model [3].

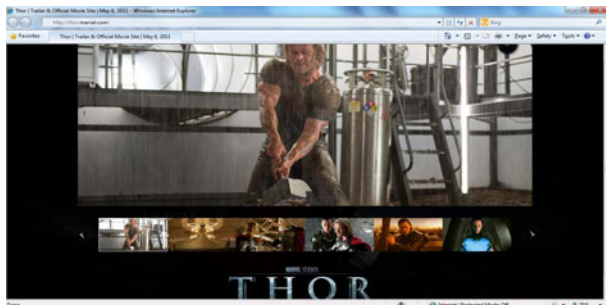


Fig. 1. A web page with few information

We define those pages from a page's backward and forward links as neighboring pages, and the radius is 2 which means the farthest pages we consider are 2 hops from the original page according to our observation and computation cost. The problem is one page might have many neighboring pages and they are likely to belong to different web genres.

We propose a link-based graph model to select the most related neighboring pages and extract those information which are useful for the task of web pages genre identification. Comparing with the existing methods, our approach takes into account as many neighboring pages as possible while greatly reducing the noisy information by selecting an appropriate subset of neighboring pages.

The proposed approach is evaluated using two benchmark corpora for web pages genre identification. We compare our approach to existing works from different points of views. The experimental results show that our approach is feasible. The rest of this paper is organized as follows. In Section 2, we discuss related works in web page genre identification. In Section 3, we describe the main challenges of genre identification and the details of our approach. In Section 4, we present our experiments, evaluation metrics, and results. We also conclude this study in Section 5.

2 Related Work

The previous studies of web pages genre identification differ significantly with respect to two factors: 1) the feature set they use to represent the content and structure of web pages, 2) the method to classify web pages based on the chosen feature set. The following review of some previous works is presented in chronological order.

Meyer and Stein [15] provided a corpus of eight genres following a user study on genre usefulness with a support vector machine classifier. Moreover, they examined various feature sets attempting to combine different kinds of information including word frequencies, text Statistics and part-of-speech frequencies.

Kennedy and Shepherd [7] emphasized on a specific genre and its sub-genres. Using a neural network they attempted to discriminate between home pages and

non home pages. Their feature set comprises features about the content (e.g. common words and meta tags), form (e.g. number of images), and functionality (e.g. number of links and use of scripts). The best reported results were for personal home pages.

Kanaris and Stamatatos [6] purpose low-level feature sets of variable-length character n-grams and combine this representation with information about the most frequent html-tags. Based on two benchmark corpora, their SVM based classification approach can improve the results in both cases.

Lei [4] examined the effect of various attributes on four different web genres, FAQ, News, E-Shopping and Personal Home Pages using the Naive Bayes classifier and the Information Gain Measure for feature selection. The results indicated that fewer features produce better precision but more features produce better recall, and that attributes in combinations will always perform better than single attributes.

3 Proposed Approach

The aim of our approach is to identify web pages genre based on a list of features. As described above in Section 2, researchers usually use On-Page Features [12] for web pages genre identification include textural information written in HTML, e.g. the text in the heading, and structural information [4], such as HTML tags, because they can be accessed and gained easily.

Additionally, we retrieve additional information from neighboring pages and propose a link-based graph model to select a subset of neighboring pages that is the same genre as the original page. Details are given in this section.

3.1 On-Page Features

Like most of other existing works, we extract terms from page URL, title, keywords, headings and anchors and apply TFIDF [13] to these textural information. Eventually, we have a vector that is a list of TFIDF score represented terms we extract from each page.

For structural information, we use features that have been examined in existing works as discussed in Section 2. We select number of images, links, emails, forms, tables and div tags which represent the form and function of a page respectively. However, our preliminary study and evaluations show that structural information use independently cannot get sound results if the dataset contains various genres of web pages. Therefore, we combine structural information with textual information together to perform genre identification.

3.2 Neighboring Pages Selection

We collect neighboring pages from web pages' backward and forward links and design a link-based similarity measure called GenreSim based on the PageSim

similarity measure [10]. Compared to other popular link-based similarity measures like SimRank [5] and Co-citation [13], PageSim can measure similarity between any two pages without the need of intermediate pages as required in SimRank. Additionally, PageSim can consider neighboring pages in multiple hops in contrast to Co-citation only considers direct neighboring pages.

We model all the neighboring pages and the original page as a directed graph $G = (V, E)$ with vertices V representing web pages $v_i (i = 0, 1, 2, \dots, n)$ and directed edges E representing hyperlinks among the web pages. Page v_0 is the original page. Below is a list of definitions for our model:

Definition 1. *Path:* We call $p(u, v)$ is a path donates a sequence of vertices from vertex u to vertex v through a set of edges $e_i (i = 1, 2, \dots, n), e \in E$ and $E \in G$.

Definition 2. *Path Set:* We call $PATH(u, v)$ is the set of all possible paths from vertex u to v .

Definition 3. *Page Score:* We call $Score(v)$ is the score of a page v which represents the importance of v in the web graph according to its backward and forward links and it is query independent.

Definition 4. *Backward and Forward Links:* We call $B(v)$ is the set of backward link neighboring pages vertices of vertex v and $F(v)$ is the set of forward link neighboring pages vertices of vertex v .

In GenreSim, we not only use forward links as the original PageSim similarity measure but also use backward links because the number of backward links of a page can give us a general estimate of its prominence on the web graph. We also implement a weighted method to compute the page score to make it suitable for genre identification.

Let $Score(u, v)$ denotes the recommendation score of page u propagate to v through $PATH(u, v)$ because a hyperlink from a page u to v can be considered as the recommendation of page v by u while u and v should have some kinds of similarity [1]. Naturally, the recommendation are decreased along with the links. We then define:

$$Score(u, v) = \begin{cases} \frac{\sum_{p \in PATH(u, v)} d \cdot Score(u)}{\prod_{x \in p, x \neq v} (|F(x)| + |B(x)|)}, & v \neq u \\ Score(u), & v = u \end{cases}$$

where $d \in (0, 1]$ is a decay factor and $u, v \in V$.

Regards to the score of a page in above equation, we extend the HITS algorithm to calculate the score because our web graph is based on neighboring pages only. Hypertext-induced topic selection (HITS) is a algorithm developed by Kleinberg [8] originally. Web pages that have many links pointing to them are called *authorities*, while web pages that have many outgoing links are called

hubs. Let $H(p)$ and $A(p)$ be the hub and authority score of page p . These scores are defined such that the following equations are satisfied for all pages p :

$$H(p) = \sum_{u \in S|p \rightarrow u} A(u), A(p) = \sum_{v \in S|v \rightarrow p} H(v) \quad (1)$$

where $H(p)$ and $A(p)$ are normalized for all web pages and calculated by number of links. We use the sum of authority and hub score as the score of a page because pages with high authority score are expected to have relevant content, whereas pages with high hub scores are expected to contain links to relevant content. However, the problem of this computation method is for some pages with high score which only have few backward links but large number of forward links, they highly likely are spam pages and point to many irrelevant pages. To avoid this issue, we modify above equation as:

$$H(p) = \sum_{u \in S|p \rightarrow u} \omega(p) \cdot A(u), A(p) = \sum_{v \in S|v \rightarrow p} \omega(p) \cdot H(v) \quad (2)$$

where $\omega(p)$ is the weighted parameters and calculated by the relevance value of a neighboring page to the original page. Since our purpose is to find neighboring pages that are the same genre as the original page, if two pages in the same domain are the same genre, they are likely have similar amount of backward links because they are expected to have relevant content. We define:

$$\omega(p) = \frac{1}{|N - N(p)| + 1} \quad (3)$$

where N is the number of backward links of the original page and $N(p)$ is the number of backward links of neighboring page p . By this way, the closer number of backward links to the original page, the higher weight the page will be given.

Then we have the similarity score $Sim(u, v)$ donates the similarity between u and v which represents their importance against other pages and the recommendation they propagate to other pages are similar in the graph. Since the graph is built from neighboring pages which means most of pages are from the same domain. Therefore, it is likely that u and v belong to the same web genre. For instance, Michael Jordon' Wiki page has a number of backward and forward links and most of links are NBA related while its neighboring page Kobe Byrant' Wiki page has similar recommendations to other pages in the web graph according to the linking information. The similarity score is defined as below by adopting the Jaccard measure, which is commonly used in information retrieval to measure the similarity:

$$Sim(u, v) = \frac{\sum_{i=1}^n \min(Score(v_i, u), Score(v_i, v))}{\sum_{i=1}^n \max(Score(v_i, u), Score(v_i, v))} \quad (4)$$

where $u, v \in V$.

We then select the top 10 neighboring pages with the highest similarity score to extract textual information and construct feature set similar to the On-Page features we discussed in Section 3.1.

4 Evaluations

4.1 Corpora and Data Preparation

In our experiment, we use two popular corpora: KI-04 corpus and 7-Web collection [14]. Each web page is associated with a specific source URL address and belongs to a single genre class. We preprocess all of web pages in the dataset before use classifiers to train. We select and tokenize text into words and remove numbers, non-letter characters, stop words and special characters. We stem select terms using the Lovins stemmer and calculate TFIDF value for each page.

4.2 Evaluation Results

We evaluated our approach based on metrics adopted by some previous works [9,11]. Pairwise F1 is defined as the harmonic mean of pairwise precision and pairwise recall, where pairwise precision is the number of true positives divided by the total number of elements labeled as belonging to the positive class. Recall in this context is defined as the number of true positives divided by the total number of elements that actually belong to the positive class.

Since our data is multi-class, the evaluation is done in two binary classifications, one-class-against-the-rest classification and pairwise classification. Both classifications are measured by Macro-averaging, which means precision and recall are first evaluated "locally" for each class then "globally" by averaging over the results of different classes. The Macro-averaging of precision $Precision_M$ and recall $Recall_M$ are represented as $Precision_M = \frac{\sum_{i=1}^{|C|} Precision_i}{|C|}$, $Recall_M = \frac{\sum_{i=1}^{|C|} Recall_i}{|C|}$, where $|C|$ is the total number of the classes.

Evaluations of Neighboring Pages Selection. We first evaluated our neighboring pages selection model by apply different similarity measures in both KI-04 and 7-Web corpora. We select the top 10 neighboring pages with highest similarity score generated by different similarity measures, and manually check if these pages are the same genre as the original page. In this testing, GenreSim achieves better results than others as shown at Table 1.

Comparisons of Individual Classifiers. This section is to discuss the comparisons of individual classifiers. All classifiers are implemented by SVM RBFK-kernel in Weka API¹. We split the dataset for classifiers training and testing, the proportion is 80% and 20%. All classifiers are trained by use 10-fold cross validation.

In addition, in order to evaluate the performance of our model to select the most related neighboring pages for genre identification, we also use the Google Similar Pages function² to return the first 10 pages as neighboring pages. Table 2 shows that Macro-averaging of four classifiers for the pages in the test

¹ <http://www.cs.waikato.ac.nz/ml/weka/>

² <http://www.google.com/help/features.html>

Table 1. Comparisons of Neighboring Pages Selection.

Model	KI-04(%)	7-Web(%)
Co-citation	77.3	75.6
SimRank	81.7	79.5
PageSim	88.5	86.8
GenreSim	94.5	92.6

Table 2. Macro-averaging in Pairwise Classification

KI-04	Precision(%)	Recall(%)	F1(%)
SVM On-Page Features	42.2	38.3	40.2
SVM Neighboring Pages	48.2	49.6	48.9
SVM Google Similar Pages	44.5	46.4	45.4
7-Web	Precision(%)	Recall(%)	F1(%)
SVM On-Page Features	58.1	48.3	52.7
SVM Neighboring Pages	43.2	59.7	50.1
SVM Google Similar Pages	41.2	51.8	45.9

Table 3. Macro-averaging in One-Class-Against-the-Rest Classification

KI-04	Precision(%)	Recall(%)	F1(%)
SVM On-Page Features	78.2	73.3	75.8
SVM Neighboring Pages	88.2	88.3	88.2
SVM Google Similar Pages	78.3	84.2	81.1
7-Web	Precision(%)	Recall(%)	F1(%)
SVM On-Page Features	78.1	86.5	82.1
SVM Neighboring Pages	91.2	89.7	90.4
SVM Google Similar Pages	82.7	83.3	82.9

dataset KI-04 and 7-Web for pairwise classification whilst Table 3 shows that Macro-averaging of four classifiers for the pages in the test dataset KI-04 and 7-Web for one-class-against-the-rest classification.

As the results shown in both tables, the classifier based on the feature of neighboring pages generated from our algorithm achieves the best results in all of three corpora though in some cases SVM On-Page Features classifier performs better due to lack of text information in neighboring pages. In addition, the classifier based on the neighboring pages generated by our model outperforms the one generated by Google Similar Pages function in all cases.

5 Conclusions and Future Work

In this paper, we propose a link-based graph model for genre identification of web pages. Our approach uses additional information from neighboring pages generated by our selection method to identify the genre of web pages.

The experiments indicated that our proposed approach is feasible. Next, we will focus on investigating dynamic weighting for individual classifier to further improve the ensemble performance of multiple classifiers.

References

1. Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A., Raghavan, S.: Searching the web. *ACM Transactions on Internet Technology*, 2–43 (2001)
2. Boese, E., Howe, A.: Effects of web document evolution on genre classification. In: *Proc. of the ACM 14th Conference on Information and Knowledge Management (2005)*
3. Chen, G., Choi, B.: Web page genre classification. In: *Proc. of 2008 ACM symposium on Applied computing*, pp. 2353–2357 (2008)
4. Dong, L., Watters, C., Duffy, J., Shepherd, M.: An examination of genre attributes for web page classification. In: *Proc. of the 41th Annual Hawaii International Conference on System Sciences*, pp. 129–138 (2008)
5. Jeh, G., Widom, J.: Simrank: a measure of structural-context similarity. In: *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 538–543 (2002)
6. Kanaris, I., Stamatatos, E.: Web page genre identification using variable-length character n-grams. In: *19th IEEE International Conference on Tools with Artificial Intelligence*, vol. 7(1), pp. 3–10 (2007)
7. Kennedy, A., Shepherd, M.: Automatic identification of home pages on the web. In: *Proc. of the 38th Annual Hawaii International Conference on System Sciences*, pp. 99–108 (2005)
8. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. *Journal of the ACM* 46(5), 604–632 (1999)
9. Laender, A.H.F., Goncalves, M.A., Cota, R.G., Ferreira, A.A., Santos, R.L.T., Silva, A.J.C.: Keeping a digital library clean: New solutions to old problems. In: *Proc. of the 8th ACM Symposium on Document Engineering*, pp. 257–262 (2008)
10. Lin, Z., King, I., Ly, M.R.: Pagesim: A novel link-based similarity measure for the world wide web. In: *Proc. of the 5th International Conference on Web Intelligence*, pp. 687–693 (2006)
11. Pereira, D.A., Ribeiro, B.N., Ziviani, N., Alberto, H.F., Goncalves, A.M., Ferreira, A.A.: Using web information for author name disambiguation. In: *Proc. of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 49–58 (2009)
12. Qi, X., Davison, B.D.: Web page classification: Features and algorithms. *ACM Comput. Surv.* 41(2), 1–31 (2009)
13. Salton, G., McGill, M.J.: *Introduction to modern information retrieval* (1986)
14. Santini, M.: Automatic genre identification: Towards a flexible classification scheme. In: *BCS IRSG Symposium: Future Directions in Information Access (2007)*
15. Meyer zu Eissen, S., Stein, B.: Genre classification of web pages. In: Biundo, S., Frühwirth, T., Palm, G. (eds.) *KI 2004. LNCS (LNAI)*, vol. 3238, pp. 256–269. Springer, Heidelberg (2004)

Stepwise and Asynchronous Runtime Optimization of Web Service Compositions^{*}

Philipp Leitner, Waldemar Hummer, Benjamin Satzger, and Schahram Dustdar

Distributed Systems Group, Vienna University of Technology
Argentinierstrasse 8/184-1, A-1040, Vienna, Austria
lastname@infosys.tuwien.ac.at

Abstract. Existing research work considers runtime adaptation of service compositions as a viable tool to prevent violations of service level agreements. In previous work we have formalized the optimization problem of identifying the most suitable adaptations to prevent a predicted set of violations, and presented suitable algorithms to solve this problem. Here, we introduce the idea of stepwise optimization as a solution to the problem of how to deal with situations when the optimization result is not available in time, i.e., when decisions need to be taken before the optimization problem can be fully solved.

1 Introduction

In information systems based on the concept of Service-Oriented Architecture (SOA), business processes are implemented as higher-level compositions of Web services (service compositions [1]). Providers of service compositions often guarantee certain quality characteristics using service level agreements (SLAs). Basically, SLAs are collections of target qualities (service level objectives, SLOs) and monetary penalties that go into effect if the promised target quality cannot be achieved. Hence, providers of service compositions have strong incentives to prevent cases of SLA violation. One promising approach to achieve this is predicting violations at runtime, before they have actually occurred, and using adaptation to prevent these violations [2, 3]. Evidently, an important part of runtime adaptation is deciding which adaptations to apply. We argue that this decision should be based on both, the costs of violation (the penalties associated with SLOs), and the costs of adaptation. We have presented a formalization of this decision process as an optimization problem as part of our work on the PREVENT (Event-Based Prediction and Prevention of SLA Violations) project [4]. However, a limitation of the PREVENT approach so far is that it is inherently assumed that the optimization problem can be solved in time, before the first adaptation has to be applied. Even using fast meta-heuristics this is not guaranteed, especially so for shorter service compositions.

^{*} The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement 215483 (S-Cube) and grant agreement 257483 (Indenica).

In this paper, we improve on this by proposing an asynchronous and stepwise optimization model, in which we do not generate a decision for all adaptations at once. Instead, we run the optimization in parallel to the execution of the composition. At so-called decision points we make a decision for only those adaptations that absolutely need to be decided at that moment, update the optimization problem according to the made decisions, and continue optimizing for all remaining possibilities for adaptation. Note that the contribution presented here is not specific to the approach presented in [4]. Much more, the same ideas are applicable for other runtime adaptation approaches facing similar problems as well, e.g., [3].

2 Runtime Optimization of Service Compositions

The decision which combination of runtime adaptations is best suited to prevent one or more predicted SLA violations in an instance of a service composition can be formulated as an optimization problem. For this paper the concrete structure of this optimization problem is not essential. However, for ease of understanding we briefly summarize the formalization used in PREVENT, which is the basis of the remaining discussions. This model has originally been presented in [4].

We assume the following inputs to the optimization. Let I be the set of all possible instances of the service composition, and let $i \in I$ be one instance that we need to optimize. Furthermore, let the relevant SLA be given as a set of SLOs $S = \{s_1, s_2, \dots, s_k\}$. Every SLO s has an associated penalty function, which governs the payment that the composite service provider has to pay based on a measured SLO value m_s . Penalty functions are defined as $p_s : \mathbb{R} \rightarrow \mathbb{R}$, $s \in S$. We refer to the collection of all penalty functions as $P = \{p_{s_1}, p_{s_2}, \dots, p_{s_k}\}$. Moreover, let $A = \{a_1, a_2, \dots, a_l\}$ be the set of all possible adaptations, and $A^* \in \mathcal{P}(A)$ one concrete subset of adaptations. Applying adaptations transforms composition instances, which we capture with the \circ operator ($\circ : I \times \mathcal{P}(A) \rightarrow I$). For simplicity, we assume that all adaptations have constant costs, defined as $c : A \rightarrow \mathbb{R}$. However, adaptations are not necessarily independent, i.e., there can be constraints on which adaptations can be selected at the same time. We use a simple penalty term to express that two adaptations are mutually exclusive ($v(A^*) = \infty$ if A^* contains at least one constraint violation, $v(A^*) = 0$ otherwise).

With these definitions, we can describe the total costs (TC) of a composite service provider for one instance of the business process as $TC(A^*) = v(A^*) + \sum_{s_x \in S} p_{s_x}(i \circ A^*) + \sum_{a_x \in A^*} c(a_x) \rightarrow \min!$ In this definition, the first term is the potential penalty for constraint violations in A^* . The second term represents the costs accrued via penalty payments for SLA violations. Finally, the third term represents the costs of adaptation. Naturally, the goal of the provider is to select A^* so that TC is minimal for a given i .

Evidently, the exact penalties that have to be paid ($p_{s_x}(i \circ A^*)$) are unknown at runtime for any combination of adaptations (even if we do not apply any adaptations, we still do not know for sure which SLOs are going to be violated). However, we assume that it is possible to predict the penalty before and after

adaptation with a reasonably small estimation error, using a set of estimation functions $e_s : I \rightarrow \mathbb{R}, s \in S$. We can then replace the penalty payments with their estimation, leading to the following estimation: $TC(A^*) \approx v(A^*) + \sum_{s_x \in S} e_{s_x}(i \circ A^*) + \sum_{a_x \in A^*} c(a_x) \rightarrow \min!$

In practice, estimation can be implemented for instance use machine learning based regression, as presented in [5, 6]. Using this technique one can estimate monitorable SLO values m_s in advance, and use these estimated values to calculate what the penalty will be after applying a given set of adaptations. This is the approach that we have chosen to follow in the PREVENT framework, but in principle our model is not restricted to these machine learning based estimation functions.

3 Stepwise Service Composition Optimization

The problem presented in Section 2 can be solved at runtime, however, generating a good solution may be time-consuming. One promising approach that we have utilized in PREVENT with good results are genetic algorithms [7] (GA). In the remainder of the paper, we assume that the runtime optimization of A^* is implemented using GA, however, the general principles presented here still apply if other means are used. However, even using GA, optimization is still time-consuming. Therefore, in order not to delay the execution of the composition, it is desirable to execute the optimization asynchronously, i.e., in parallel to the service composition. However, in this case we need to keep timing aspects of the optimization and the composition in mind.

Before explaining optimization timing, we need to concretize what adaptation actually means in the scope of this paper. In general, we assume that adaptation can either be implemented via service rebinding, i.e., exchanging a service in the composition for another, or via structural adaptation of the composition, i.e., freely adding, removing or modifying activities in the composition. Figure 1 exemplifies these types of adaptation. This adaptation model is in line with related work, as other approaches to self-adapting compositions usually assume similar possibilities for adaptation [8, 9]. The excerpt in Figure 1 is a small part of an assembling case study presented in [4]. We will use this example in the remainder of the paper. Note that even though we present our work on a simple sequential process for simplicity, the same ideas can be used for arbitrarily complex composition graphs, as long as they are circle-free.

For any adaptation of the types discussed above we can identify the affected region in the service composition, i.e., the activities in the composition which are affected by the adaptation. For rebinding, this is exactly one activity. For structural adaptation the affected region may be arbitrarily large, but is still always clearly defined. In the remainder of this paper, we use the term “beginning of the affected region” (t_a^x) as the time that the first activity affected by adaptation x starts to execute. In Figure 1, the beginning of the affected region is indicated by **X**.

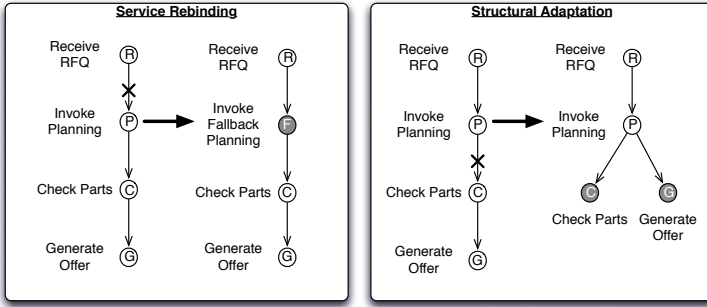


Fig. 1. Types of Adaptation

3.1 Timely Optimization and Stale Results

Figure 2 showcases the timing of asynchronous optimization. For an instance of the composition, an optimization is triggered at time t_0 (e.g., because SLO estimation mechanisms have predicted that this instance is going to violate its SLA). A meta-heuristic optimization algorithm starts searching the solution space. Meanwhile, the service composition continues executing. Firstly, assume that at time t_1 the optimization has converged and delivers the result that two adaptations have to be applied. For both actions the affected region has not yet been reached, i.e., the optimization was timely and the result is useful. However, if we assume now that the algorithm takes more time and delivers its result at time t_2 . Now, the activity “Invoke Planning” (P) has already been executed, and part of the result (the decision to adapt P) came too late.

Intuitively, for every adaptation x with a defined affected region there is also a decision point t_d^x , the latest time in the execution of the composition when a decision needs to be made. Assuming that we know t_d^x , and the time that the application of the adaptation technically takes (d_x), we can define the decision

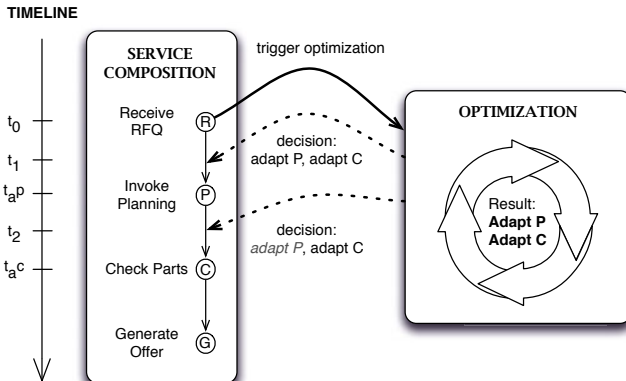


Fig. 2. Optimization Timing

point of an adaptation as $t_d^x = t_a^x - d_x$. We refer to an optimization result A^* produced at time t as stale if $\exists a \in A^* : t_d^a < t$. Stale optimization results cannot be applied in full anymore when they are available, and evidently should be avoided.

3.2 Stepwise Optimization

Two approaches can be used to handle the problem of stale results. Firstly, one can decide not to deal with the problem at all, ignoring stale adaptations and applying only what is still possible when the result becomes available. This approach is very simple, and even in the worst case this is at least never worse than not doing optimization to begin with, even if the result may be suboptimal in the presence of stale results. Secondly, one can drop the idea of asynchronous optimization and halt the service composition while the optimization is running. This trivially prevents stale results, but severely degrades the performance of the service composition. It is well possible (if the optimization takes more time than what can be gained using adaptation) that using this approach is actually worse than not doing any optimization at all. It is easy to see that both of these ideas are not optimal.

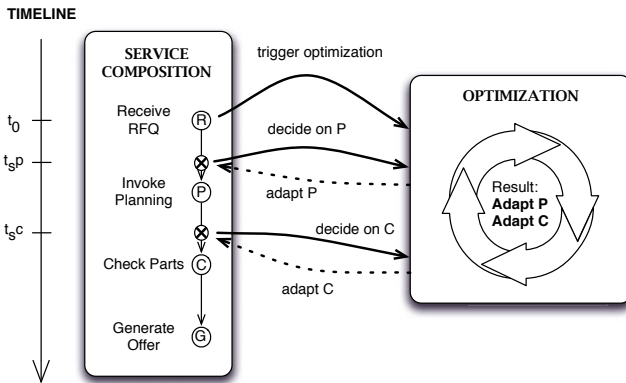


Fig. 3. Stepwise Asynchronous Optimization

Hence, we now introduce stepwise asynchronous optimization as an alternative principle to prevent stale results. The general approach is sketched in Figure 3. First of all, we order all adaptations according to their decision points. Actions with identical decision points ($t_d^x = t_d^y$) are collected in decision sets (D_i). Let t_s^x be the decision point of a decision set D_x , defined as the decision point of all adaptations contained in the set ($\forall a \in D_i : t_d^a = t_s^i$). In the figure, two decision sets with decision points t_s^p and t_s^c exist. The first decision set contains only the action “Invoke Planning” (P), the second contains only the action “Check Parts” (C).

As before, an optimization is triggered at t_0 . However, results are now delivered differently as in the naïve approach sketched before. Instead of waiting for

```

1  decide(DecisionSet ds, Optimization opt):
2
3      suspend_process()
4
5      foreach(Adaptation a in ds):
6          decision = decide_on_adaptation(a, opt)
7          if(decision == true)
8              apply_adaptation(a)
9          adapt_target_function(a, decision, opt)
10
11     resume_process()

```

Fig. 4. Decision Procedure for Stepwise Optimization

the optimization to converge, we now trigger the decision procedure sketched in the algorithm in Figure 4 when t_s^p and t_s^c are passed.

For every decision point associated with a decision set, we briefly halt the composition and decide on all adaptations associated with this set. If it is decided to apply one or more adaptations, we do so. Finally, we resume the composition. This way, instead of producing the solution for the optimization problem in one big bang (and risking that the result arrives too late), we use a stepwise, constructive approach which defers all decisions about adaptations to the latest possible time, but never later. Hence, results are never stale, and the service composition needs to be halted only briefly. Note that it is generally advantageous to wait as long as possible before making a decision (per definition, that means waiting until t_d^x), under the assumption that the quality of a decision is monotonically increasing with optimization time. This is true for most implementations of optimization algorithms, including GA (if elitism is used).

The decision algorithm in Figure 4 contains three separate challenges. Firstly, one needs to be able to apply adaptations at runtime (Line 8). We do not discuss solutions for this problem here, and refer the reader to existing work instead [4]. Secondly, we need to be able to actually make a decision on a single adaptation based on a still ongoing optimization (Line 6). One simple, yet promising, strategy is to base the decision on the best currently known intermediary result, i.e., decide to apply an adaptation if and only if the currently best known solution applies the adaptation. We refer to this strategy as current-optimum based decision. Thirdly, after deciding on an adaptation, the target function of the optimization problem needs to be modified. This is to reflect the fact that whenever a decision is made (and a given adaptation is applied or rejected) the underlying problem of the ongoing optimization has in fact changed. If the adaptation has been applied, all solutions that do not use this adaptation are invalid. Similarly, if the adaptation has not been applied, all solutions using the adaptation are invalid. This change is easily represented using an additional penalty term v_x in the target function. For instance, if a_x is applied, a new penalty term v_x defined as $v_x(A^*) = \infty$ if $a_x \notin A^*$, and $v_x(A^*) = 0$ otherwise, is added to the

target function. Hence, in Line 9 of the algorithm, we pause the optimization algorithm, add an additional constraint for each adaptation in the decision set and resume optimizing.

4 Related Work

The general idea of optimizing running composition instances is related to the larger research field of QoS-aware service composition. QoS-aware service composition is usually a static process, which aims at finding the best instantiation of an abstract composition before or during deployment. The optimization problem is finding the most suitable combination of concrete services. The seminal work that introduced this idea already dates back to 2004 [10], however, newer research is still able to provide new insights. For instance, [11] defined a domain-specific language from which dynamic QoS-optimized compositions are generated. [12] improved on the methods of optimization that are used in QoS-aware composition, and proposed to use a combination of local selection and global optimization. The approach presented in this paper differs from all these contributions in that we do not consider optimization statically. Indeed, traditional QoS-aware composition does not face the problem of timeliness at all, as optimization is done once and is not repeated for every problematic instance, as it is the case in our approach. The research work most closely related to this paper is work on runtime adaptation of service compositions. First ideas on self-adaptive compositions can be found in [13], even if this work is more closely related to adaptive workflows than service compositions. Recently, work in this area seems to have gravitated towards using the aspect-oriented programming (AOP) paradigm to technically implement adaptation, as exemplified by [9,14]. Other approaches use pure service rebinding [8] or parametrization of compositions [15]. The PREVENT framework [2] supports adaptation on many different levels, and forms the basis of the research work presented in this paper. Other research of note in the area of adaptation include [3], which uses machine learning techniques similar to the mechanisms used in PREVENT to trigger adaptation. All of these approaches have a similar base premise (optimization of the performance of a service composition through monitoring and runtime adaptation), but none discusses timing aspects explicitly. We argue that our work is complementary to all these approaches, and similar ideas as discussed here may be worthwhile to incorporate in any framework that aims at optimizing running composition instances.

5 Conclusions

In this paper we have introduced the problem of stale results in the optimization of service compositions, and proposed stepwise optimization as a possible solution to tackle this issue. As future work, we plan to investigate how appropriate stepwise optimization is for usage with different optimization algorithms, e.g., Ant Colony Optimization or Simulated Annealing, and compare the stepwise optimization approach with quick construction heuristics.

References

1. Dustdar, S., Schreiner, W.: A Survey on Web Services Composition. *International Journal of Web and Grid Services* 1, 1–30 (2005)
2. Leitner, P., Michlmayr, A., Rosenberg, F., Dustdar, S.: Monitoring, Prediction and Prevention of SLA Violations in Composite Services. In: *ICWS 2010*, pp. 369–376 (2010)
3. Kazhamiakin, R., Wetzstein, B., Karastoyanova, D., Pistore, M., Leymann, F.: Adaptation of Service-Based Applications Based on Process Quality Factor Analysis. In: *MONA+*, pp. 395–404 (2009)
4. Leitner, P., Hummer, W., Dustdar, S.: Cost-Based Optimization of Service Compositions. Technical Report TUV-1841-2011-1, Vienna University of Technology, Austria (2011)
5. Leitner, P., Wetzstein, B., Rosenberg, F., Michlmayr, A., Dustdar, S., Leymann, F.: Runtime Prediction of Service Level Agreement Violations for Composite Services. In: *NFPSLAM-SOC 2009*, pp. 176–186 (2009)
6. Zeng, L., Lingenfelder, C., Lei, H., Chang, H.: Event-Driven Quality of Service Prediction. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.) *ICSOC 2008*. LNCS, vol. 5364, pp. 147–161. Springer, Heidelberg (2008)
7. Goldberg, D.E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, Reading (1989)
8. Ardagna, D., Comuzzi, M., Mussi, E., Pernici, B., Plebani, P.: PAWS: A Framework for Executing Adaptive Web-Service Processes. *IEEE Software* 24, 39–46 (2007)
9. Charfi, A., Mezini, M.: AO4BPEL: An Aspect-oriented Extension to BPEL. *World Wide Web* 10, 309–344 (2007)
10. Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., Chang, H.: QoS-Aware Middleware for Web Services Composition. *IEEE Transactions on Software Engineering* 30, 311–327 (2004)
11. Rosenberg, F., Celikovic, P., Michlmayr, A., Leitner, P., Dustdar, S.: An End-to-End Approach for QoS-Aware Service Composition. In: *EDOC 2009*, pp. 151–160 (2009)
12. Alrifai, M., Risse, T.: Combining Global Optimization With Local Selection for Efficient QoS-Aware Service Composition. In: *Proceedings of the 18th International Conference on World Wide Web (WWW 2009)*, pp. 881–890 (2009)
13. Casati, F., Ilnicki, S., Jin, L., Krishnamoorthy, V., Shan, M.: Adaptive and Dynamic Service Composition in eFlow. In: Wangler, B., Bergman, L.D. (eds.) *CAiSE 2000*. LNCS, vol. 1789, pp. 13–31. Springer, Heidelberg (2000)
14. Leitner, P., Wetzstein, B., Karastoyanova, D., Hummer, W., Dustdar, S., Leymann, F.: Preventing SLA Violations in Service Compositions Using Aspect-Based Fragment Substitution. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) *ICSOC 2010*. LNCS, vol. 6470, pp. 365–380. Springer, Heidelberg (2010)
15. Karastoyanova, D., Leymann, F., Nitzsche, J., Wetzstein, B., Wutke, D.: Parameterized BPEL Processes: Concepts and Implementation. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) *BPM 2006*. LNCS, vol. 4102, pp. 471–476. Springer, Heidelberg (2006)

Supporting Sharing of Browsing Information and Search Results in Mobile Collaborative Searches

Daisuke Kotani, Satoshi Nakamura, and Katsumi Tanaka

Graduate School of Informatics, Kyoto University,
Yoshida-Honmachi, Sakyo, Kyoto 606-8501 Japan
kotani@net.ist.i.kyoto-u.ac.jp
{nakamura, tanaka}@dl.kuis.kyoto-u.ac.jp

Abstract. We propose and implement methods for making collaborative searches more efficient in mobile computing environments. In mobile collaborative searches, the small screens of mobile devices make it difficult to share information such as discovered Web pages, evaluations of these, and information related to their search topics. We propose and implement a system supporting mobile collaborative searches, which provides a function for informing others about Web pages by inserting them into others' search results using simple actions; a function of displaying information related to others' search topics; and a function of showing the most positively evaluated Web pages at the top of the list of discovered Web pages. We also propose a method for the system to create a group of mobile devices easily. The results of evaluation tests show these methods support the sharing of information and improve the users' satisfaction.

Keywords: Mobile Computing, User Interface, Collaborative Search.

1 Introduction

Today, almost everyone has at least one mobile phone [6] and most mobile phone have a feature for accessing the Web [4]. These facts suggest that people can obtain any information they need at any time and anywhere by searching the Web using mobile phones.

In the near future, it may be common to search the Web collaboratively in mobile computing environments using a "Mobile Collaborative Search", if all those involved have one or more mobile phones. Examples of possible mobile collaborative search situations include when people are deciding where to go after a party, or when people would like to find how to get home in a time of disaster.

We distinguish three kind of information exchanged with others in mobile collaborative searches: "What are you searching for?", "How do you think of this Web page?", and "Which Web pages do you think good?". Without using any system to help sharing such information, the users' satisfaction with their decisions from the task may be low and it may take much time to make decisions because it is not easy to share such information using a mobile phone's small display. Furthermore, conversations to share such information interrupt their search processes and reduce the efficiency of Web searches. In addition, it is hard to compare and analyze many valuable Web pages and it is also hard for users who do not actively join the conversation to contribute to their decisions.

There are many researches about collaborative Web searches where one or more large displays exist [1][2][3], but these are inapplicable directly to the mobile computing environments because these does not consider the limitations of a situation in which only mobile devices exists, such as the small display. Komaki et al [5] proposed a system to support the comparison and analysis of Web content, but our goal is to support the entire process of mobile collaborative searches.

We propose three methods for supporting users to share, compare, and analyze information to improve users' satisfaction with their decisions in mobile collaborative searches. The first is to enable a user to inform others about a Web page that he or she thinks valuable by adding the Web page into the currently displayed search results on the others' mobile phones, so that they can browse such Web pages using their mobile phones at a convenient time. The second method is to show information related to their search spaces, such as keywords featuring their search topics, queries they have been input, and the Web pages that they have already visited. This enables them to search the Web while considering others' search spaces. The third method is to show the most positively evaluated Web pages at the top of the list of discovered Web pages based on the users' feedback on such Web pages. This method enables them to compare and analyze Web pages and make their decisions more efficiently. We also propose a method for the system to recognize a group of mobile devices based on times and distance so that users can start collaborative searches smoothly.

We implemented these methods in our prototype system and showed that these methods could improve the ease of the sharing of information and the users' satisfaction with their decisions after each search task by user-based evaluation tests.

2 Methodology

In a collaborative Web search session, users first search the Web individually. When a user finds a Web page valuable, he or she may ask others to evaluate it. Then they discuss it, and return to search the Web. A user would change the query while searching the Web. At this time, he or she may ask others for their search spaces in order to search for different topics from others. When they have searched the Web to some extent, they compare and analyze the gathered Web pages. Whenever they come to a conclusion, they would finish the session. We focus on three problems in this communication.

The first problem is caused by the small displays of mobile devices. It is hard for users to see details on the small display at the same time. It takes much time for them to pass around and operate a mobile device one by one. We will solve this by allowing users to see necessary information in their mobile devices. The second problem is caused by interruptions to their search processes by having conversations with others. Users who are spoken to have to stop browsing in their mobile devices. Then they join in the conversation and return to browse in their mobile devices. Constant interruptions reduce the efficiency of searching the Web and thus that of mobile collaborative searches. We address this by reducing their conversations and by allowing them to stop their search processes at a convenient time. The third problem is that it is difficult to compare and analyze all the valuable Web pages discovered by users, and thus to make their decisions. We address this by ranking these Web pages based on their feedback.

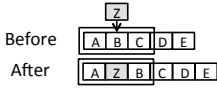


Fig. 1. The insertion of the Web page Z

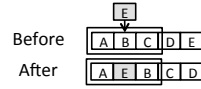


Fig. 2. The insertion of the Web page E

Later, we propose three methods to address these problems and a method to making a group of mobile devices easily on the system, and show screens of our prototype system, which runs on iPhones and iPod touches with iOS 4.2.

2.1 Insert an Item of a Web Page to Search Results

The first method is to enable a user to inform others about a Web page that he or she wants them to see. Search results form a set of candidate Web pages which a user starts browsing the Web. We propose the insertion of search result items from Web pages that are passed on by others into the list of search results. This enables users to access such Web pages at a convenient time on their own mobile devices.

It is important that users can recognize the search result items that have been inserted by others. We would ensure that by inserting these items into the currently displayed parts of search results and by changing background color of them.

The details of the method are as follows. (1) A user tells the system a Web page which he or she wants them to evaluate. In our system, pressing “Notify” in a Web browser corresponds to this. (2) The system sends the data on the Web page specified 1 to other mobile devices. (3) The system on the mobile devices inserts the item of the Web page received into the currently displayed part of the search results with changing its background color (Fig. 1). If the items pointing to the same Web page already exist, the system removes them from the search results, then inserts the item so that users do not bother to consider multiple items pointing to the same Web page (Fig. 2).

Images illustrating this method is given in Fig. 1 and 2. The search results include the Web pages A, B, C, D and E in that order, and A, B and C are displayed. A screen in our system is given in Fig. 3. Items with a background color of red are inserted ones.

Having many items inserted by others may prevent them from browsing the search results. They can skip these items by scrolling the list, but that operation alone may not be sufficient in some cases. We provide a way to return to show the original search results by simple actions, for example by shaking the mobile device in our system.

2.2 Showing Information Related to Other Group Members’ Search Spaces

Users would be able to know about others’ search spaces without interrupting their search processes if the system records and shows their search activities, such as queries they have input and visited Web pages. The reason for users asking for information about their search spaces could be that they wish to avoid overlapping search topics and to search for a wider range of topics in a group. Therefore, we propose the following: marking the search result items of Web pages which have been visited by the users so that they can avoid accessing the same Web pages as others during their browsing of

search results (Fig. 3 items with black circle); showing inputted queries and keywords featuring their search topics so that they can know what they are searching for whenever they want (Fig. 5). In the second case, showing the list of visited Web pages would be unsuitable for a small display as a large display and many scrolling operations would be required to browse the list.

As users search the Web for similar topics but not the same, two kinds of terms would exist in their visited Web pages: terms which often appear in the Web pages visited by most of them, and terms which often appear only in the Web pages visited by a certain user. To find out what each user is searching for, the latter kind of terms would be useful. We can extract such terms from visited Web pages using the *tf/idf* method.

2.3 Using Inserted Web Pages in the Comparing and Analyzing Phase

Web pages that a user asks others to evaluate may be included in the Web pages to compare and analyze as he or she thinks these are valuable. Therefore, it is helpful for users to see a list of such Web pages that are inserted in their search results like Fig. 4.

It would be efficient to compare and analyze the most positively evaluated Web pages. Therefore, the system lists such Web pages at the top so that users can refer to such Web pages quickly. To do this, the system enables users to give positive or negative feedback for Web pages suggested by others in a Web browser (The buttons “+1” (Positive), “0” (Cancel), “-1” (Negative) exist in our system). The system lists such Web pages in descending order by the supportive degree, “the number of positive ratings - the number of negative ratings”.

2.4 Creating a Group Collaboratively Using Time and Spatial Closeness of Users

We propose a method for users to create a group of their mobile devices on the system by simple operations using data on when and where the operations were carried out so that they can start collaborative searches smoothly. A screen to create a group appears when a user touches “Group” in Fig. 3 in our system. The procedure is as follows:

When users carry out a specific operation, their mobile devices send a group creation request (called request 1) to the server. N seconds later, the mobile devices send a request for the result of the request 1 (called request 2) to the server. The response to the request 2 includes data on the clients which satisfy the following conditions: a client sent a request 1 within the previous $N + \alpha$ seconds and within a radius of β from the location of the client which sent the request 2. If clients that satisfy the conditions have already created the group, the server sends data on that client’s group.

3 Evaluation Experiment and Discussions

We carried out evaluation tests in a room in our lab to confirm that our proposed methods can support information sharing in mobile collaborative searches and increase the users’ satisfaction with their decisions. We recruited 16 subjects who were male, aged twenties, students at our university and majored in computer science. We divided them into four groups (G1, G2, G3 and G4). The members of each group often talk.



Fig. 3. Search results



Fig. 4. Inserted items List

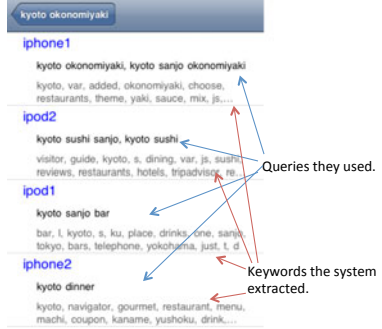


Fig. 5. Others' queries and keywords

To compare the subjects' search activities and feelings between the tasks using and not using our proposed methods, we asked them to do two tasks using the comparison system for one task and our system for the other. The comparison system had almost the same interfaces as our system, but no information related to others' search activities is displayed to imitate Web searches using normal Web browsers in mobile devices.

At the beginning of the experiment, we explained the functions and the usage of two systems and asked the subjects to use each system for 10 minutes. Then we provided two tasks for each group, and asked them to search the Web and to make decisions within 10 minutes. We allowed around a five-minute break between tasks. We recorded logs of their operations and videos of their actions. Finally, we asked them to write comments and to fill out questionnaires individually, using a five-point Likert scale.

G1 and G3 used the comparison system and our system in that order, while G2 and G4 used in the reverse order. G1 and G2 searched for restaurants in two locations, assuming they were travelling. The tasks of G3 and G4 were to plan a trip to two locations, assuming that they were attending a conference and that they wanted to go sightseeing.

3.1 Results of Experiments

The subjects made collaborative searches for around 11 minutes. We analyzed the logs and the videos recorded for 10 minutes from the start of the task, and the questionnaires.

We asked the subjects about their satisfaction with their decisions (5:satisfy). The averages of the scores were 3.9($\sigma = 0.8$) for tasks using the comparison system and 4.5($\sigma = 0.5$) for our system. The scores for how well they could share information during their tasks (5:well) increased in our system: 3.0($\sigma = 0.8$) and 4.4($\sigma = 0.6$) in each system. In terms of how much trouble they had sharing information (5:easy), they scored 2.5($\sigma = 0.8$) for the comparison system and 4.3($\sigma = 0.7$) for our system on average. One subject commented he strongly felt he had comprehensively searched various kinds of Web pages with others in our system.

The average scores for their satisfaction with their Web searches (5:satisfy) were 3.3($\sigma = 0.7$) in the comparison system and 3.6($\sigma = 0.6$) in our system. We also asked the subjects about the efficiency of the Web search (5:efficient) and the scores were

Table 1. Utterances and uses of functions

Comparison: Utterances	106.3
Prototype: Utterances	106.3
Prototype: Inserted items	21.5
Prototype: List of inserted items	11.8
Prototype: List of queries, etc	6.8

Table 2. Queries and other users' activities

	Comparison	Prototype
Queries (Unique queries)	19.0 (16.8)	19.0 (16.8)
Visited total Web pages	34.0	36.5
Visited unique Web pages	31.5	24.0
Accessing inserted Web pages		15.0

3.0($\sigma = 0.7$) and 3.4($\sigma = 0.9$) in each system. In this question, the answers in G1, G2 and G3 are increased in our system: 2.9 and 3.7 in each system, but that in G4 decreased: 3.0 and 2.8. The answers to the question on how far their opinion was reflected by their decisions (5:reflected) were 3.4($\sigma = 1.2$) and 4.0($\sigma = 1.2$) in each system on average. Tendencies also differed between G1, G2, G3 and G4 in this question: 3.5 and 4.3 in each system for G1, G2 and G3, but 3.3 and 3.0 for G4. The averages of the per-group sums of figures related to the search activities are shown in Table 2. Comments included that it took longer to see Web pages that had been inserted than to see Web pages in the original search results as more items were inserted.

The average of the per-group sums of utterances and that of uses of the functions only in our system are shown in Table 1 and these counted per minute are given in Fig 6 and 7. The standard deviations of utterances per subject for G1, G2, G3 and G4 were 6.5, 10.2, 4.6, 23.1 each in the comparison system and 9.1, 10.7, 7.9, 18.2 each in our system. We saw that the subjects shared information through conversations while searching the Web. Sometimes they asked others to look at their display, and more than one looked at it simultaneously in the comparison system. In our system, some of them showed others their display, but we didn't see anyone ask others to look at their display.

We also asked them to choose the most useful function and to evaluate the usefulness of each function (5:useful, Table 3). For inserting Web page items into others' search results, we distinguished two kinds of comments: positive ones such as "It is useful as we can be aware of others' search topics" and "I can browse inserted Web pages when I can pause my search activities", and negative ones such as "It is hard to see lots of inserted items". A list of inserted Web pages was mainly displayed in the latter half of the task (Fig 7). There were many utterances during the first and last few minutes (Fig 6). We saw the subjects talked about how they divided their search space into individual spaces and knowledge they already had at the beginning. Their talks at the end seemed focused on findings they had obtained while browsing the Web, things to compare and analyze in discovered Web pages, and things to help in making their decisions. Some subjects commented that it was impossible to give feedback on all the Web pages inserted, and that it was easier to have conversations. Comments on displaying others' queries and keywords included "I did not use it since I could see others' search spaces by observing inserted items in search results", "It was inconvenient because it took a effort to display", and "I want to see others' queries at any time".

3.2 Discussion

Firstly, we discuss information sharing between users. There was little difference in the utterances between tasks using the two systems (Table 1, Fig 6). However, the subjects

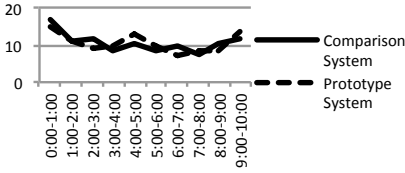


Fig. 6. Utterances per minute

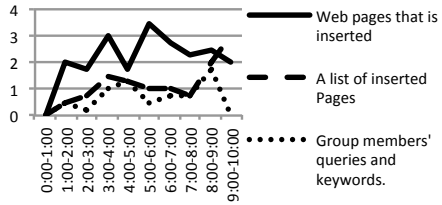


Fig. 7. Uses of functions per minute

Table 3. Evaluations for functions (One subject did not choose the most useful function)

	Insert Web page items	List of inserted Web pages	Show queries & keywords
The most necessary	10	1	4
Usefulness	4.0($\sigma = 1.1$)	3.1($\sigma = 1.0$)	3.8($\sigma = 1.3$)

felt that they could share more information while doing the tasks in our system than in the comparison system. That would be because it was easy to exchange information on what they were searching without having conversations by using the function in our system. In fact, these functions were used throughout the tasks (Table 1, Fig 7).

In terms of the satisfaction for Web searches, there were few differences in the satisfaction and efficiency of their Web searches between the two systems. There were no great differences in Table 2 except for visited unique Web pages, which was smaller in our system than in the comparison system. The difference between total and unique visited Web pages in our system was relatively close to access the inserted Web pages. In our system, the total number of visited Web pages increased when users accessed inserted Web pages, but the number of visited unique Web pages remained the same. That would cause a difference. The closeness between the number of accessing the inserted Web pages and that difference suggests this. In addition, the time limitation caused the subjects to spend less time browsing Web pages other than inserted Web pages in search results, and that affected the difference. This may suggest that the speed of widening the search space is slower in our system than in the comparison system. On the other hand, as a result of sharing more information, the subjects may be able to take others' opinions into account while searching the Web, thus they could find the Web pages suited to their needs faster. We need more tests to verify these possibilities.

In the point of the satisfaction for the decisions, it is interesting that the answers to the question on how far subjects' opinions were reflected by their decisions differed between G1, G2, G3 and G4. The averages in G1, G2 and G3 increased in our system. That is because our system encourages them to share more information, and thus they can discuss the task more smoothly and search for more suitable information for their decisions. Let us consider the differences between G3 and G4 since they did the same type of tasks. The standard deviations of utterances per user differed greatly. This means some of them spoke very much, but others did not in G4. We think the reason for this difference was that subjects who did not speak a lot could not give their opinions to the others, and scored a degree of the agreement with their decisions.

In the following we discuss about our proposed methods. The subjects evaluated that inserting Web pages into others' search results was the most useful, followed by displaying their queries and keywords, and lastly displaying a list of inserted Web pages.

The method of inserting a Web page into others' search results as a search results item was useful, according to the comments, Table 2 and 3. However, the results suggest that displaying much information from others may reduce the efficiency of Web searches. A list of inserted Web pages was mainly displayed when the subjects compared and analyzed discovered Web pages at the end of the task (Fig. 7). Some commented that it was hard to give feedback on all the inserted Web pages. We would need to improve a way to support the comparison and analysis of Web pages. There were some comments on displaying information related to search spaces like useful and no need. Some subjects commented that they would like the system to display other users' queries and Web pages they were viewing at any time. We need to support such needs.

4 Conclusion and Future Work

We proposed and implemented three methods to address the problems of sharing, comparison and analysis of information between users in mobile collaborative searches. Evaluation tests confirmed that our prototype system increased the ease of sharing information, that the subjects shared more information while searching the Web in our prototype system, and that satisfaction with the decisions from the tasks was increased.

Future work could include improving the method of extracting keywords, suggesting spaces which no one has searched, and showing what others are searching for by inserting items into the search results selectively.

Acknowledgement. This work was partly supported by Grant-in-Aid for challenging Exploratory Research (# 22650018) and Grant-in-Aid for Young Scientists (A) (# 23680006) from JSPS.

References

1. Morris, M.R., Horvitz, E.: SearchTogether: an interface for collaborative web search. In: UIST 2007, pp. 3–12 (2007)
2. Amershi, S., Morris, M.R.: CoSearch: A System for Co-located Collaborative Web Search. In: CHI 2008, pp. 1647–1656 (2008)
3. Morris, M.R., Lombardo, J., Wigdor, D.: WeSearch: supporting collaborative search and sensemaking on a tabletop display. In: CSCW 2010, pp. 401–410 (2010)
4. Telecommunications Carriers Association.: Number of subscribers by carriers, <http://www.tca.or.jp/english/database/index.html>
5. Komaki, D., Oku, A., Arase, Y., Hara, T., Uemukai, T., Hattori, G., Nishio, S.: Content Comparison Functions for Mobile Co-located Collaborative Web Search. *Journal of Ambient Intelligence and Humanized Computing* 3(3), 1–10
6. Ministry of Internal Affairs and Communications: Information & communications statistics database, <http://www.soumu.go.jp/johotsusintokei/english/>
7. Morris, M.R.: A survey of collaborative web search practices. In: CHI 2008, pp. 1657–1660 (2008)

GTrust: An Innovated Trust Model for Group Services Selection in Web-Based Service-Oriented Environments

Xing Su¹, Minjie Zhang¹, Yi Mu¹, and Quan Bai²

¹ School of Computer Science & Software Engineering, University of Wollongong,
Wollongong, NSW 2522, Australia

xs702@uowmail.edu.au, {minjie,ymu}@uow.edu.au

² School of Computing & Mathematical Sciences, Auckland University of Technology,
Auckland, New Zealand
quan.bai@aut.ac.nz

Abstract. In past twenty years, the multi-agent technology has been widely employed for the development of web-based systems. Currently, agent-based service-oriented applications have been widely applied in many complex domains such as web-based e-markets, web-based grid computing, e-government and service-oriented software systems, cross Internet and organizations. In this kind of service-oriented systems, service provider (agents) and service consumer (agents) are autonomous entities and can enter and leave the environment freely. How to select the most suitable service providers according to the requested services from consumers in such an open environment is a very challenge issue. In this paper, we propose an innovated trust model-the GTrust model for group services selection in a general service-oriented environment. In our model, the trust evaluation for a group service is based on (1) the coverage rate of the requested functionalities from a group service, (2) the dependency relationships among individual services in a group, (3) reference reports from third parties for each provider of individual services in a group and (4) the similarity measurement about to what extent the reference reports can reflect the new service request in terms of priority distributions on attributes of the service. The experimental results demonstrate the good performance of the GTrust model in terms of satisfaction degree in group service selections.

1 Introduction

Multi-Agent Systems (MASs) have attracted a lot of attention from researchers in computer science, information technology, engineering, as well as other disciplines due to their abilities of autonomous decision making, collaborative problem solving, learning and adaptation abilities under open and distributed environments. In past decade, agent and multi-agent technologies have been widely employed for developing web-based service-oriented systems such as Internet-based grid systems [7], e-market [2], as well as pervasive computing systems.

The web-based service-oriented environment is an open environment where most service consumer (agents) and service provider (agents) have only local views about their partners and the environment and can also join and leave the environment freely at any time. In such a dynamic environment, how to select a trustworthy service provider to fulfill a requested service from a consumer is a very challenging problem for most service-oriented applications.

‘Agent Trust’ is one of important research issues and many researchers in MASs had made significant effects on trust and reputations models such as probabilistic theory-based models [4], the certified reputation model [1] and evidential trust models [5]. In past decade, some trust models have been developed in service-oriented domains to help consumers evaluate the trust values of potential service providers based on different considerations. In 2000, Zacharia et. al. proposed a reputation-based trust evaluation model based on the historical performance of a provider, called the SPORAS [6], for service provider selection. In 2006, Huynh et. al. introduced a famous trust model, called the Certified Reputation (CR), to evaluate provider’s trust through the third party references [1]. In 2010, Su et. al. did further work based on the CR model and developed a priority-based trust model to evaluate a trust value of a potential service provider based on the third party references, its historical performance and the priority distribution on the attributes of the requested service [3]. The most current models evaluate the trust values for individual providers. However, in recent years, many complex service requests from consumers cannot be handled by a single service and a group of services from different providers are needed to satisfy these service requests. Therefore, trust models focusing on the trust evaluations for single service providers cannot be directly used for the group trust evaluation and how to choose a group of services for a consumer has become a new challenge issue.

The trust evaluation for a group of service providers is different from that of for a single service provider, because there are more factors may affect the trust values of group services. The main factors include (1) the coverage rate of the requested functionalities from a service group, which determines whether the service group can satisfy all of the attribute of a service requested by a consumer, (2) the relationships among individual services in a group, (3) the performance of the individual services and reputations of individual providers in a service group, and (4) the suitability measure for the group fitting the new service based on the priority distribution on service attributes requested from a consumer.

The GTrust model has the following merits: (1) we use the ‘functionality coverage’ value to represent the functionalities which a potential service group can provide corresponding to the request from the consumer; (2) we introduce the concept of the ‘dependency degree’ to represent relationships among services in a service group; (3) we use the concept of the ‘third party reference’ from the PBTrust model [3] to represent the performance of individual services in the same group; and (4) we use the concept of ‘similarity’ to measure the similarity in terms of priority distributions on attributes between historical services of group members and requested services.

The rest of paper is organized as follows. Section 2 is the problem description. The basic components of the GTrust model is briefly introduced in Section 3. The detail descriptions of each components in the GTrust are introduced in detail Section 4. Finally, the paper is concluded and future work is outlined in Section 5.

2 Problem Description and Definitions

In general, a service can be described by a number of attributes such as price, time, quality etc. For different requests, the priority distribution on each attribute for the same service can be different. In order to describe the attributes and their corresponding priority values of a service, we make a service description in a formal way.

Suppose that a requested service includes n attributes and each attribute has a priority value to describe the request for the service. A service can be represented by n attributes and their corresponding priority values as follows.

Definition 1. A service description $SDes$ is defined by a $2 \times n$ matrix.

$$SDes = \begin{pmatrix} A_1 & A_2 & A_3 & \dots & A_i \\ P_1 & P_2 & P_3 & \dots & P_i \end{pmatrix} \quad (1)$$

where i indicates the number of attributes in requested service, A_i indicates the i^{th} attribute of the requested service, P_i is priority value of the A_i and $\sum_{i=1}^n P_i = 1$.

Definition 2. A reference report Rf is defined as a 2-tuple, $Rf = \langle SDes, Ratings \rangle$, where $SDes$ is the service description of the service requested by the pervious consumer (referee) and $Ratings$ is defined as a vector, $Ratings = \langle R_1, R_2, \dots, R_i \rangle$, where R_i represents the performance rating value of the provider on i^{th} attribute of the requested service and R_i is a value between $[0, 1]$, where 0 and 1 represents the worst and best performance of a provider, respectively.

To deal with a complex request, a number of individual services need to form a group with certain workflows and dependency relationships among individual services in the group. Even if two groups have the same individual services, if the workflows and dependency relationships of the individual services in the two groups are different, the two groups may have different performance on the requested service. For example, two groups have the same individual services $S1, S2, S3, S4$ and $S5$, but different workflows and dependency relationships as follows.

In Fig. 1, Group 1 has a sequential workflow to process from $S1$ to $S5$, i.e. the later service depends on the former service. However, the workflow is different in Group 2, $S1, S2, S3$ and $S4$ can work at the same time and $S5$ can only be conducted when the former four services is finished. We can see that there are no dependency relationships among $S1$ to $S4$ but 4 dependency relationships exist between $S5$ with other 4 services. In another word, $S5$ depends on $S1, S2, S3$ and $S4$, respectively. In order to identify relationships among services in a group, we define the dependency degree as follows.



Fig. 1. Workflows and dependency relationships of services in Two Groups

Definition 3. A dependency degree λ is defined as a value in-between $[0, 1]$, where 0 represents an independency relationship between two services and 1 denotes the strongest dependency relationship between two services.

We also use a matrix to describe the workflow of a group by using the following definition.

Definition 4. A workflow description $WDes$ of a group is represented by a $n \times n$ matrix, where 'n' is the number of individual services in the group. The $WDes$ is defined by Equation 2 as follows.

$$WDes = \begin{pmatrix} \lambda_{11}, \lambda_{12}, \lambda_{13}, \dots, \lambda_{1n} \\ \lambda_{21}, \lambda_{22}, \lambda_{23}, \dots, \lambda_{2n} \\ \dots, \dots, \dots, \dots, \dots \\ \lambda_{n1}, \lambda_{n2}, \lambda_{n3}, \dots, \lambda_{nn} \end{pmatrix} \quad (2)$$

where λ_{ij} represents the value of dependency degree between service i and service j . $\lambda_{ij} = 0$ represents there is no dependency relationship between service i and the service j . If $\lambda > 0$, there exists a dependency relationship between service i and service j and service j depends on service i .

Definition 5. A service reply SR is defined as a 2-tuple, $SR = \langle WDes, RfSet \rangle$, where $WDes$ is the workflow description of a group and $RfSet$ is the set of reference reports of each services in the group.

3 Basic Modules of the GTrust Model

The GTrust model consists of three modules which are the Request Module, the Reply Module and the Priority-based Group Trust Calculation Module. The working procedure of the GTrust model can be described as follows. When a consumer requests a complex service (1) the request module will generate the service requirements and broadcast it to potential providers; (2) potential service groups with requested services will reply the service request by using the reply module; (3) the consumer will evaluate the trust value for each potential service group using the priority-based group trust calculation module and choose the best service group based on the trust value of the group; With the reference report, the members of the service group can dynamically update their reference report.

4 The Principle of the GTrust Model

In this section, four major modules of the GTrust model are introduced in detail in the following four subsections, respectively.

4.1 The Request Module

The objective of the Request Module is to create a *service request* based on the request from a consumer. For example, consumer C in an e-market environment requests a complex service described by 5 attributes, i.e. cost, speed, quality, color and warranty with corresponding priority values for each attribute as 0.1, 0.4, 0.2, 0.1 and 0.2, respectively. Based on the service request, the Request Module will generate a service request in the format of *service description*, (recall Definition 1) as follows:

$$SDes = \begin{pmatrix} Cost & Speed & Quality & Color & Warranty \\ 0.1 & 0.4 & 0.2 & 0.1 & 0.2 \end{pmatrix}$$

Then, the service request will be broadcasted to the system to discover potential service providers.

The above example will be used for the explanation in rest modules.

4.2 The Reply Module

The function of the reply module is to generate a service reply to describe a service group and the individual services in the group. For example, if a Service Group (SG) intends to offer the requested service, the reply module of SG will collect the following information: the group description of SG , and reference reports of each individual services in SG . Each individual service in SG will present its best reference report. The reply module will create a service reply, (recall Definition 3) for SG in the following format including the *workflow description* of SG and a set of reference reports for five members, respectively, $SR = \langle GDes, RfSet \rangle$.

4.3 The Priority-Based Group Trust Calculation Module

The main purpose of this module is to evaluate the trust value for each potential service group based on the *service reply* SR and service request.

Because a service group is composed of different individual services owned by different providers, the group ability to handle a new service depends on the abilities of individual members. We use a *group service description* to formally describe a group ability by extracting useful information from reference reports provided by group members.

Definition 6. A *group service description* $GSDes$ is represented by a $m \times n$ matrix. m is the number of the individual services in a group and n is the number of attributes in service request. $GSDes$ is defined by the following matrix.

$$GSDes = \begin{pmatrix} A_1 & A_2 & \dots & A_n \\ P_{11} & P_{12} & \dots & P_{1n} \\ P_{21} & P_{22} & \dots & P_{2n} \\ \dots & \dots & \dots & \dots \\ P_{m1} & P_{m2} & \dots & P_{mn} \end{pmatrix} \quad (3)$$

where A_i indicates the i^{th} attribute of the requested service. The i^{th} row (excluding the first row) in the matrix represents the priority distribution on a pervious service completed by the corresponding group member and P_{ij} represents the priority value on the j^{th} attribute of the requested service on that service, where $P^{ij} = m$, if the pervious service dose not contain the j^{th} attribute; otherwise P_{ij} is in-between $[0,1]$, where 0 and 1 represent the highest and lowest priority values, respectively. By using Equation 3, the comprehensive ability of a service group can be described.

Functionality Coverage Calculation. The purpose of functionality coverage calculation is to measure whether the functionalities offered by a potential service group can cover all the attributes in the service request. A functionality coverage is defined by the following definition.

Definition 7. A functionality coverage $FCov$ is defined as a vector, $FCov = \langle ACov_1, ACov_2, ACov_3, \dots, ACov_i \rangle$, where $ACov_i$ is a value in-between $[0, 1]$, which represents the functionality coverage value of a service group on i^{th} attribute in the service request.

$ACov_i$ can be calculated based on the information in $GSDes$ (recall Definition 6) as follows.

$$ACov_i = \frac{m - MS_i}{m} \quad (4)$$

where $ACov_i$ represents the functionality coverage value of a service group on i^{th} attribute of the requested service, m represents the number of the individual services in a group and MS_i represent the number of ‘m’ (i.e. how many members cannot cover the i^{th} attributes) in the i^{th} column of the matrix $GSDes$. If the functionality coverage on i^{th} attribute is ‘0’, we can say that this service group is not suitable to conduct the requested service.

Group Similarity Calculation. The objective of the group similarity calculation is to measure the similarity of the priority distribution between a group service and the requested service. In the GTrust model, the priority distribution of a service is represented by a vector. To compare the similarity between two vectors, we can use the concept ‘dot product’ of the two vectors. To calculate the similarity of priority distribution between a group service and the requested, we can use a vector $GPV = \langle GP_1, GP_2, GP_3, \dots, GP_n \rangle$ to represent the priority distribution in a group of services extracted from reference reports, where GR_i is the priority value on the i^{th} attribute in a group service. GP_i is calculated by the following formula.

$$GP_i = \sum_{i=1}^m P_{ij} \tag{5}$$

where, P_{ij} is the priority value of the i^{th} individual service in the group on j^{th} attribute of the requested service.

We can calculate each element in Vector GPV , then normalize two vectors if necessary before using the dot product. The group similarity calculation can be obtained by the following formula.

$$GSim = \frac{\sum_{i=1}^n NGP_i \cdot NP_i}{\sqrt{(\sum_{i=1}^n (NGP_i)^2) \cdot (\sum_{k=1}^n (NP_i)^2)}} \tag{6}$$

where $GSim$ is the similarity between the priority distribution of the requested service and a service group, NP_i and NGP_i represent the normalized priority values of the i^{th} element of priority distribution vector in the requested service and the priority distribution vector in the service group, respectively.

Group Rating Calculation. The purpose of group rating calculation is to predict the performance of a service group on each attribute of the requested service based on the reference reports. The rating for the group’s potential performance in j^{th} attribute is calculated as follows.

$$GRating_j = \frac{\sum_{i=1}^m FRating_{ij}}{m} \tag{7}$$

where ‘m’ is the number of individual services in the service group and $FRating_{ij}$ represents the final rating of the i^{th} individual service, after considering the dependency degrees with other services in the group, on the j^{th} attribute in the group service. $FRating_{ij}$ is calculated by the following formula.

$$FRating_{ij} = Rating_{ij} - \frac{\sum_{k=1}^n \lambda_{ki} \cdot (1 - FRating_{kj})}{n} \tag{8}$$

where n represents the number of the individual services which the i^{th} service depends on, $Rating_{ij}$ is the rating of the i^{th} individual service on j^{th} attribute shown in the reference report and $FRating_{kj}$ is the the final performance rating of the k^{th} dependency service on j^{th} attributes, and λ_{ki} is the dependency degree of the i^{th} individual service depending on the k^{th} dependency service.

Final Trust Calculation. After functionality coverage calculation, similarity calculation and group rating calculation, we can calculate the final trust value $Trust$ for a service group by the following formula.

$$Trust = GSim \cdot \sum_{i=1}^n P_i \cdot ACov_i \cdot GRating_i \tag{9}$$

where $GSim$ is the similarity value, P_i is the priority value of the i^{th} attribute in the requested service, $ACov_i$ is the functionality coverage value of a service

group on the i^{th} attribute of the requested service and $GRating_i$ represents the group rating after considering the dependency relationships and workflows of services in the group.

5 Conclusion and Future Work

In this paper, we proposed the GTrust model for group services selection in web-based service-oriented environments. In current stage, we use the group performance evaluated by the consumer as the reference report for each members of the group during updating their historical records without considering the different roles of each individual services. In the future, we will employ agent learning approach to our trust model to analyze the performance of each member in a group.

References

1. Huynh, T., Jennings, N., Shadbolt, N.: Certified reputation: How an agent can trust a stranger. In: AAMAS 2006: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 1217–1224 (2006)
2. Ren, F., Zhang, M., Sim, K.: Adaptive conceding strategies for automated trading agents in dynamic, open markets. *Decision Support Systems* 48(2), 331–341 (2009)
3. Su, X., Zhang, M., Mu, Y., Sim, K.: Pbtrust: A priority-based trust model for service selection in general service-oriented environments. In: 2010 IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC), pp. 841–848 (December 2010)
4. Teacy, W., Patel, J., Jennings, N., Luck, M.: Travos: Trust and reputation in the context of inaccurate information sources. *Journal of Autofocus Agent and Multiagent Systems (JAAMAS)* 12, 183–198 (2006)
5. Wang, J., Sun, H.: A new evidential trust model for open communities. *Comput. Stand. Interfaces* 31, 994–1001 (2009), <http://portal.acm.org/citation.cfm?id=1555011.1555187>
6. Zacharia, G., Maes, P.: Trust management through reputation mechanisms. *Applied Artificial Intelligence* 14(9), 881–907 (2000)
7. Zhang, M., Tang, J., Fulcher, J.: Agent Based Grid Computing. In: *Advanced Computational Intelligence Paradigms: Theory and Applications*. SCI, vol. 115, pp. 439–483. Springer, Heidelberg (2008)

Trust as a Service: A Framework for Trust Management in Cloud Environments

Talal H. Noor and Quan Z. Sheng

School of Computer Science,
The University of Adelaide, Adelaide SA 5005, Australia
{talal,qsheng}@cs.adelaide.edu.au

Abstract. Trust is one of the most concerned obstacles for the adoption and growth of cloud computing. Although several solutions have been proposed recently in managing trust feedbacks in cloud environments, how to determine the credibility of trust feedbacks is mostly neglected. In addition, managing trust feedbacks in cloud environments is a difficult problem due to unpredictable number of cloud service consumers and highly dynamic nature of cloud environments. In this paper, we propose the “Trust as a Service” (TaaS) framework to improve ways on trust management in cloud environments. In particular, we introduce an adaptive credibility model that distinguishes between credible trust feedbacks and malicious feedbacks by considering cloud service consumers’ capability and majority consensus of their feedbacks. The approaches have been validated by the prototype system and experimental results.

Keywords: Trust Management, Cloud Computing, Distributed Computing, Credibility Model.

1 Introduction

Over the past few years, cloud computing is gaining a considerable momentum as a new computing paradigm for providing flexible services, platforms, and infrastructures on demand [13]. For instance, it only took 24 hours, at the cost of merely \$240, for the New York Times to archive its 11 million articles (1851-1980) using Amazon Web Services¹.

Given the quick adoption of cloud computing in the industry, there is a significant challenge in managing trust among cloud service providers and cloud service consumers [13,8]. Recently, the significance of trust management is highly recognized and several solutions are proposed to assess and manage trust feedbacks collected from participants [8,5]. However, one particular problem has been mostly neglected: to what extent can these trust feedbacks be credible. Trust management systems usually experience malicious behaviors from its users. On the other hand, the quality of trust feedbacks differs from one person to another, depending on how experienced she is. This paper focuses on the cloud service

¹ <http://open.blogs.nytimes.com/2007/11/01/self-service-prorated-super-computing-fun/>

consumers perspective (i.e., cloud service consumers assess the trust of cloud services). In particular, we distinguish several key issues of the trust management in cloud environments including i) *Trust Results Accuracy*: determining the credibility of trust feedbacks is a significant challenge due to the overlapping interactions between cloud service consumers and cloud service providers. It is difficult to know how experienced a cloud consumer is and from whom malicious trust feedbacks are expected that requires extensive probabilistic computations [17,9]; ii) *Trust Feedback Assessment and Storage*: the trust assessment of a service in existing techniques is usually centralized, whereas the trust feedbacks come from distributed trust participants. Trust models that use centralized architectures are prone to scalability and security issues [7].

In this paper, we overview the design and implementation of the *Trust as a Service* (TaaS) framework. This framework helps distinguish between the credible and the malicious trust feedbacks through a credibility model. In a nutshell, the salient features of the TaaS framework are i) *A Credibility Model*: we develop a credibility model that not only distinguishes between trust feedbacks from experienced cloud service consumers and from amateur cloud service consumers, but also considers the *majority consensus* of feedbacks; ii) *Distributed Trust Feedback Assessment and Storage*: to avoid the drawbacks of centralized architectures, our trust management service allows trust feedback assessment and storage to be managed distributively.

The remainder of the paper is organized as follows. The design of the TaaS framework is presented in Section 2. Details of the trust management service (TMS) including the distributed trust feedback collection and assessment are described. Section 3 describes the credibility model. Section 4 reports the implementation and several experimental evaluations. Finally, Section 5 discusses the related work and provides some concluding remarks.

2 The Framework

We propose a framework using the Service Oriented Architecture (SOA) to deliver trust as a service. SOA and Web services are one of the most important enabling technologies for cloud computing in the sense that resources (e.g., software, infrastructures, and platforms) are exposed in clouds as services [6,16]. In particular, our framework uses Web services to span several distributed TMS nodes that expose interfaces so that trust participants (i.e., the cloud service consumers) can give their trust feedbacks or inquire about the trust results based on SOAP or REST [15] messages. Figure 1 depicts the framework, which consists of three different layers, namely the *Cloud Service Provider Layer*, the *Trust Management Service Layer*, and the *Cloud Service Consumer Layer*.

- The Cloud Service Provider Layer. This layer consists of different cloud service providers who provide cloud services. The minimum indicative feature that every cloud service provider should have is to provide the infrastructure as a service (i.e., the cloud provider should have a data center that provides the storage, the process, and the communication).

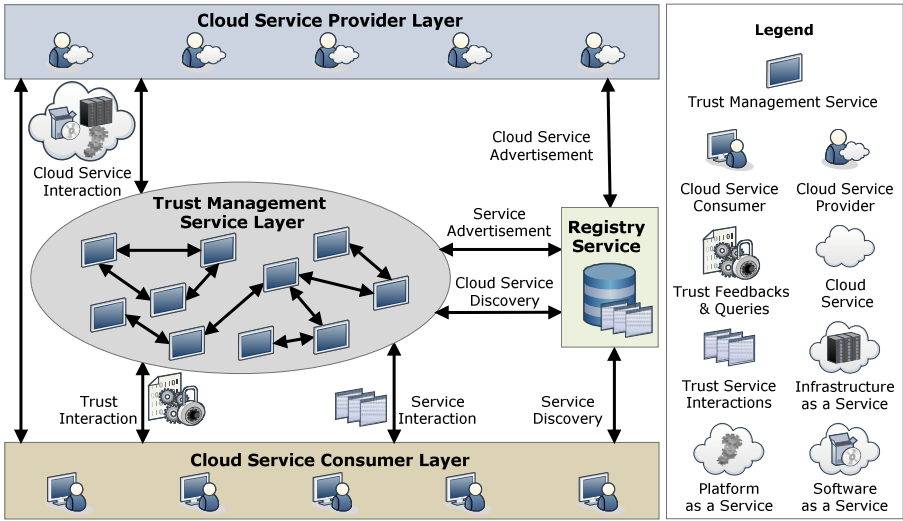


Fig. 1. Architecture of the Trust as a Service Framework

- The Trust Management Service Layer. This layer consists of several distributed TMS nodes that expose interfaces so that cloud service consumers can give their trust feedbacks or inquire about the trust results represents.
- The Cloud Service Consumer Layer. Finally, this layer consists of different cloud service consumers who consume cloud services. For example, a new startup that has limited funding can consume cloud services (e.g., hosting their services in Amazon S3). A cloud service consumer can give trust feedbacks of a particular cloud service by invoking the TMS (see Section 2.1).

Our framework also contains a *Registry Service* (see Figure 1) that has several responsibilities including i) *Service Advertisement*: both cloud service providers and the TMS are able to advertise their services through the *Service Registry*; ii) *Service Discovery*: the TMS and cloud service consumers are able to access the *Service Registry* to discover services.

2.1 Trust Feedback Collection and Assessment

In our framework, the cloud service trust behavior is represented by a collection of invocation history records denoted as \mathcal{H} . Each cloud service consumer c holds her point of view regarding the trustworthiness of a specific cloud service s which is managed by the assigned TMS. \mathcal{H} is represented in a tuple that consists of the cloud consumer primary identity \mathcal{C} , the cloud service identity \mathcal{S} , a set of trust feedbacks \mathcal{F} and the aggregated trust feedbacks weighted by the credibility \mathcal{F}_c ,

i.e., $\mathcal{H} = (\mathcal{C}, \mathcal{S}, \mathcal{F}, \mathcal{F}_c)$. Each trust feedback in \mathcal{F} is represented in numerical form in which the range of the normalized feedback is $[0, 1]$, where 0, +1, and 0.5 means negative, positive, and neutral respectively. Whenever a cloud consumer inquires the TMS about the trustworthiness of a cloud service s , the trust result ($Tr(s)$), is calculated using:

$$Tr(s) = \frac{\sum_{l=1}^{|\mathcal{V}(s)|} \mathcal{F}_c(l, s)}{|\mathcal{V}(s)|} \quad (1)$$

where $\mathcal{V}(s)$ is all trust feedbacks given to the cloud service s and $|\mathcal{V}(s)|$ represents the length of the $\mathcal{V}(s)$. $\mathcal{F}_c(l, s)$ are trust feedbacks from the l^{th} cloud consumer weighted by the credibility.

The TMS distinguishes between credible trust feedbacks and malicious trust feedbacks through assigning the *Cloud Consumer's Experience* aggregated weights $Exp(l)$ to trust feedbacks $\mathcal{F}(l, s)$ as shown in Equation 2, where the result $\mathcal{F}_c(l, s)$ is held in the invocation history record h and updated in the assigned TMS.

$$\mathcal{F}_c(l, s) = \mathcal{F}(l, s) * Exp(l) \quad (2)$$

3 Credibility Model

There is a considerable possibility that the TMS receives *inaccurate* or even *malicious* trust feedbacks from amateur cloud service consumers (e.g., who lack experience) or vicious cloud service consumers (e.g., who submit lots of negative feedbacks to disadvantage a particular cloud service). To overcome these issues, we propose a *credibility model*, which is centered on the *cloud consumer's experience*. To differentiate between expert and amateur cloud service consumers, we consider the *Majority Consensus* and the *Cloud Consumer's Capability*.

Majority Consensus. It is well-known that the majority of people usually agree with experts' judgments about what is good [4]. Similarly, we believe that the majority of cloud consumers agree with *Expert* cloud service consumers' judgments. In other words, any cloud service consumer whose trust feedback is close to the majority of trust feedbacks is considered an *Expert Cloud Service Consumer* (ECSC), or an *Amateur Cloud Service Consumer* (ACSC) otherwise. In order to measure how close the cloud service consumer's trust feedbacks to the majority (i.e., the *Majority Consensus* ($\mathcal{J}(c)$) which is calculated as follows:

$$\mathcal{J}(c) = 1 - \sqrt{\frac{\sum_{h \in \mathcal{V}_c(c)} \left(\sum_{k=1}^{|\mathcal{V}_c(c,k)|} \left(\frac{\mathcal{F}(c,k)}{|\mathcal{V}_c(c,k)|} - \left(\frac{\sum_{l \neq c, l=1}^{|\mathcal{V}_c(l,k)|} \mathcal{F}(l,k)}{|\mathcal{V}(k)| - |\mathcal{V}_c(c,k)|} \right) \right) \right)^2}{|\mathcal{V}_c(c)|}} \quad (3)$$

where the first part of the numerator represents the mean of the cloud service consumer c 's trust feedbacks $\mathcal{F}(c, k)$ for the k^{th} cloud service. The second part of

the numerator represents the mean of the majority trust feedbacks given by other cloud service consumers ($\mathcal{F}(l, k)$) (i.e., the l^{th} cloud service consumer, except the cloud service consumer c) to the k^{th} cloud service.

Cloud Service Consumer's Capability. It is a common sense that older people are likely to be more experienced in judging things than younger people [14]. However, this is only true if the older people have experienced considerable number of judging practices. As a result, we believe that "older" cloud service consumers who have many judging practices are likely to be more experienced and capable. A cloud service consumer's capability (\mathcal{B}) is measured as follows:

$$\mathcal{B}(c) = \begin{cases} 1 + \frac{|\mathcal{V}c(c)|}{\mathcal{A}g(c)} & \text{if } |\mathcal{V}c(c)| \leq \mathcal{A}g(c) \\ 2 & \text{otherwise} \end{cases} \quad (4)$$

where $\mathcal{V}c(c)$ represents all good feedbacks (i.e., feedbacks which are close to the majority) given by the cloud service consumer c . $\mathcal{A}g(c)$ denotes the virtual *Age* of a certain cloud service consumer, measured in days since the registration in the TMS. The idea behind adding the number 1 to this ratio is to increase the value of a cloud service consumer experience based on $\mathcal{B}(c)$ result. In other words, we use $\mathcal{B}(c)$ as a *reward* factor. The higher $\mathcal{B}(c)$ is, the more experienced a cloud service consumer is. It should be noted that even if a malicious cloud service consumer attempts to manipulate the capability result, the capability result will not exceed 2.

Based on the specified cloud service consumer's experience factors (i.e., $\mathcal{B}(c)$ and $\mathcal{J}(c)$), the TMS distinguishes between ECSC and ACSC through assigning the cloud service consumer's *Experience* aggregated weights $Exp(c)$ to each of the cloud consumers' trust feedbacks as shown in Equation 2. $Exp(c)$ is calculated as follows:

$$Exp(c) = \frac{\beta * \mathcal{B}(c) + \mu * \mathcal{J}(c)}{\lambda} \quad (5)$$

where β and $\mathcal{B}(c)$ denote the *cloud service consumer's Capability* factor's normalized weight and the factor's value respectively. The second part of the equation represents the *Majority Consensus* factor where μ denotes the factor's normalized weight and $\mathcal{J}(c)$ denotes the factor's value. λ represents the number of factors used to calculate $Exp(c)$ (e.g., if we only consider cloud service consumer's capability, $\lambda = 1$; if we consider both cloud service consumer's capability and majority consensus, $\lambda = 2$).

We use $\mathcal{J}(c)$ as a *penalty* factor (i.e., because $\mathcal{J}(c)$ ranges [0,1] as described in equation 3). The lower $\mathcal{J}(c)$ is, the lower the experience of the cloud service consumer c is. However, $\mathcal{B}(c)$ is used as a *reward* factor (i.e., because $\mathcal{B}(c)$ ranges [1, 2] as described in equation 4). Higher $\mathcal{B}(c)$ means more experienced of a cloud service consumer. It is worth mentioning that our credibility is dynamic and is able to detect behavior changes. For example, if a cloud service consumer behaves good for a period of time (e.g., to gain credibility) and then starts misbehaving, $\mathcal{J}(c)$ can detect such behavior through applying the standard deviation.

4 Implementation and Experimental Evaluation

Our implementation and experiments were developed based on the NetLogo platform², which was used to simulate the cloud environments. We particularly focused on validating and studying the performance of the proposed credibility model (see Section 3). In our experiments, we used real-life trust data set, Epinions³ rating data set which was collected by Massa and Avesani [13]. We choose to use Epinions data set because its data structure is similar (i.e., consumers opinions and reviews on specific products and services) to our cloud service consumer trust feedbacks. The data set has 49,290 users, 139,738 items, and 664,824 trust feedbacks.

Table 1. Experiment Factors and Parameters Setup

Experiment Design	β	μ	λ	$Exp(c)$
With Credibility Factors	1	1	2	
Without Credibility Factors				1
Cloud Service Consumer's Capability Factor	1	0	1	
Majority Consensus Factor	0	1	1	

We evaluate our credibility model using both *analytical analysis* and *empirical analysis*. The analytical analysis focuses on measuring the trust result accuracy when using the credibility model and without using the credibility model (i.e., we turn the $Exp(c)$ to 1 to exclude the credibility factor). The empirical analysis focuses on measuring the trust result accuracy for each factor in our credibility model (i.e., $\mathcal{B}(c)$ and $\mathcal{J}(c)$). The parameters setup for each corresponding experiment are depicted in Table 1.

Figure 2(a) depicts the analytical analysis of the trust results for a particular cloud service. We note that the trust results are oscillating more significantly when calculating the trust without considering the credibility factors than when calculating the trust with credibility factors. In other words, even if the TMS receives inaccurate or malicious trust feedbacks, it is difficult to manipulate the trust results by using our credibility model.

Figure 2(b) shows the empirical analysis of the same cloud service. We note that trust results obtained by only considering $\mathcal{B}(c)$ are higher than the trust results by only considering $\mathcal{J}(c)$. This is true, because we use $\mathcal{B}(c)$ as a reward factor and the $\mathcal{J}(c)$ as a penalty factor. This reflects how adaptive our credibility model is where the credibility factors can easily be tweaked according to the TMS's needs. For instance, for optimistic situations where only a few cloud service consumers have high values of capability, increasing the cloud service consumer's capability factor (i.e., β) will help the TMS to distinguish between experienced cloud consumers and inexperienced ones. On the other hand, for pessimistic situations where many cloud consumers have high values of capability, the majority consensus factor (i.e., μ) needs to be increased.

² <http://ccl.northwestern.edu/netlogo/>

³ http://www.trustlet.org/wiki/Downloaded_Epinions_dataset

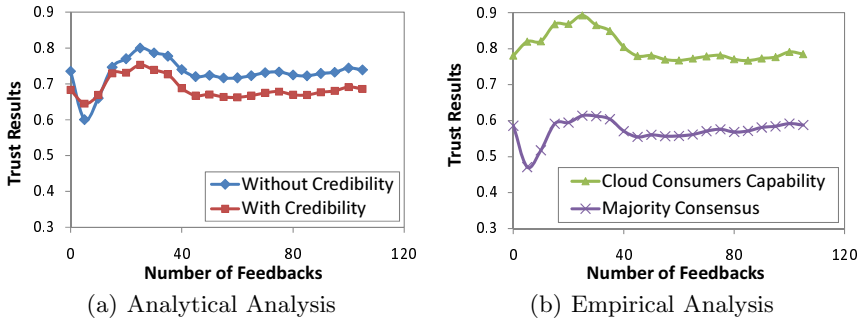


Fig. 2. Experimental Evaluation

5 Discussions and Conclusion

Trust management is one of the critical issues in cloud computing and a very active research area [10,12,8,2]. For instance, Hwang et al. [8] proposed a security-aware cloud architecture where trust negotiation and data coloring techniques are used to support the cloud service provider perspective. The cloud service consumer’s perspective is supported using the trust-overlay networks to deploy a reputation-based trust management. Brandic et al. [2] proposed a centralized approach for compliance management in cloud environments that supports the cloud service consumer’s perspective using compliant management to help the cloud service consumers in selecting proper cloud services. Unlike previous works that use centralized architecture, we present a credibility model supporting distributed trust feedback assessment and storage. This credibility model also distinguishes between trustworthy and malicious trust feedbacks.

Conner et al. [5] proposed a decentralized trust management framework for SOA that supports the service provider’s perspective. This framework offers multiple trust evaluation metrics to allow customized evaluation and assessment of service consumers. Malik and Bouguettaya [11] proposed decentralized reputation assessment techniques based on the existing quality of service (QoS) parameters. The proposed framework supports different assessment metrics such as rater credibility, past rating history, etc. Unlike previous works that require trust participants’ collaboration by rating trust feedbacks, our credibility model distinguishes between trustworthy and malicious trust feedbacks without such technique. We were inspired by Xiong and Liu who differentiate between the credibility of a peer and the credibility of the feedback [18]. However, this approach is inappropriate in cloud environments because peers give and receive services and they are evaluated on that base. In other words trust results are used to distinguish between credible and malicious feedbacks.

In this paper, we have presented a “Trust as a Service” framework to manage trust in cloud environments. We introduced an adaptive credibility model that assesses cloud services’ trustworthiness and distinguishes between credible and malicious trust feedbacks. We particularly introduced the cloud service consumer’s *Capability* and the *Majority Consensus* factors in calculating the

trust of a cloud service. In addition, our TMS allows trust feedback assessment and storage to be managed in a distributed way. In the future, we plan to deal with more challenging problems such as the *Sybil* attack and the *Whitewashing* attack. Performance optimization of TMS is another focused work.

References

1. Armbrust, M., et al.: A View of Cloud Computing. *Communiacion of the ACM* 53(4), 50–58 (2010)
2. Bradic, I., Dustdar, S., Anstett, T., Schumm, D., Leymann, F., Konrad, R.: Compliant Cloud Computing (C3): Architecture and Language Support for User-Driven Compliance Management in Clouds. In: *Proc. of IEEE CLOUD 2010*, Miami, Florida, USA (July 2010)
3. Buyya, R., Yeo, C., Venugopal, S.: Market-oriented Cloud Computing: Vision, Hype, and Reality for Delivering it Services as Computing Utilities. In: *Proc. of IEEE HPCC 2008*, Dalian, China (September 2008)
4. Child, I.: The Psychological Meaning of Aesthetic Judgments. *Visual Arts Research* 9(2(18)), 51–59 (1983)
5. Conner, W., Iyengar, A., Mikalsen, T., Rouvellou, I., Nahrstedt, K.: A Trust Management Framework for Service-Oriented Environments. In: *Proc. of WWW 2009*, Madrid, Spain (April 2009)
6. Dillon, T., Wu, C., Chang, E.: Cloud Computing: Issues and Challenges. In: *Proc. of AINA 2010*, Perth, Australia (April 2010)
7. Hoffman, K., Zage, D., Nita-Rotaru, C.: A Survey of Attack and Defense Techniques for Reputation Systems. *ACM Computing Surveys* 42(1), 1–31 (2009)
8. Hwang, K., Li, D.: Trusted Cloud Computing with Secure Resources and Data Coloring. *IEEE Internet Computing* 14(5), 14–22 (2010)
9. Jøsang, A., Quattrociochi, W.: Advanced Features in Bayesian Reputation Systems. In: Fischer-Hübner, S., Lambrinouidakis, C., Pernul, G. (eds.) *TrustBus 2009*. LNCS, vol. 5695, pp. 105–114. Springer, Heidelberg (2009)
10. Krautheim, F., Phatak, D., Sherman, A.: Introducing the Trusted Virtual Environment Module: A New Mechanism for Rooting Trust in Cloud Computing. In: Acquisti, A., Smith, S.W., Sadeghi, A.-R. (eds.) *TRUST 2010*. LNCS, vol. 6101, pp. 211–227. Springer, Heidelberg (2010)
11. Malik, Z., Bouguettaya, A.: RATEWeb: Reputation Assessment for Trust Establishment Among Web services. *The VLDB Journal* 18(4), 885–911 (2009)
12. Manuel, P., Thamarai Selvi, S., Barr, M.E.: Trust Management System for Grid and Cloud Resources. In: *Proc. of ICAC 2009*, Chennai, India (December 2009)
13. Massa, P., Avesani, P.: Trust Metrics in Recommender Systems. In: *Computing with Social Trust*. Human-Computer Interaction Series. Springer, Heidelberg (2009)
14. Roosevelt, E.: Facing the problems of youth. *The P.T.A. magazine: National Parent-Teacher Magazine* 29(30), 1–6 (1935)
15. Sheth, A.P., Gomadam, K., Lathem, J.: SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups. *IEEE Internet Computing* 11(6), 84–87 (2007)
16. Wei, Y., Blake, M.B.: Service-oriented Computing and Cloud Computing: Challenges and Opportunities. *IEEE Internet Computing* 14(6), 72–75 (2010)
17. Weng, J., Miao, C., Goh, A.: Protecting Online Rating Systems from Unfair Ratings. In: Katsikas, S.K., López, J., Pernul, G. (eds.) *TrustBus 2005*. LNCS, vol. 3592, pp. 50–59. Springer, Heidelberg (2005)
18. Xiong, L., Liu, L.: Peertrust: Supporting Reputation-based Trust for Peer-to-Peer Electronic Communities. *IEEE TKDE* 16(7), 843–857 (2004)

Quality Driven Web Services Replication Using Directed Acyclic Graph Coding

An Liu^{1,2,3}, Qing Li^{2,3}, and Liusheng Huang^{1,3}

¹ University of Science and Technology of China, Hefei, China

² City University of Hong Kong, Hong Kong, China

³ CityU-USTC Advanced Research Institute, Suzhou, China

Abstract. Web services cannot be always available as they are typically deployed in a dynamic environment. As an effective approach to improving Web services availability, replication has received much attention recently. How to design an optimal replication scheme with the best QoS, however, remains an open problem due to its inherent computational hardness. In this paper, we propose an efficient approach to designing a near-optimal replication scheme. We adopt directed acyclic graph (DAG) as the modeling tool for replication scheme and then utilize DAG coding for performance optimization. Simulation results show our approach can generate a near-optimal replication scheme with acceptable computation overheads.

1 Introduction

Web Services (WSs) are a kind of software built upon a set of widely accepted specifications. As typically deployed on the dynamic Internet, WSs are very likely to be temporally unavailable due to a variety of events like power outage, hardware breakdown, and software failure. Recent studies (e.g., [1,2,4]) have shown that replication is an effective approach to realizing highly available WSs. Specifically, in a replication scheme, several WSs with the same functionality are used together to fulfil a task, so if one WS fails, the others can continue providing the required service.

It is common that WSs with the same functionality have distinguishable QoS, so different replication schemes usually have a remarkable QoS difference. To design a replication scheme with the best QoS is of great interest, but is also an extremely challenging problem due to its inherent computational hardness: the number of replication schemes is growing exponentially with the number of WSs used for replication. Up to now, only a few well-known schemes have been thoroughly investigated [5,8]. However, a large number of unknown schemes still need to be investigated if we want to achieve optimal WS replication.

In this paper, we propose an approach to designing a near-optimal replication scheme. The main contributions of our work is two-fold. On one hand, we adopt Directed Acyclic Graph (DAG) as the modeling tool for replication scheme. The rich expressiveness of DAG endows us with the ability to explore all possible replication schemes in the problem solving phase. On the other hand, we use

Genetic Algorithm (GA) to generate a near-optimal replication scheme. Though GA is a mature optimization technique, we devise a new kind of chromosome based on DAG coding to improve the performance of GA.

The remainder of the paper is organized as follows. Section 2 provides basic concepts of WS replication. Section 3 describes a numerical example that gives the motivation of our work. The approach to designing a near-optimal replication scheme is presented in Section 4, and experimental results are reported in Section 5. Finally, Section 6 concludes the paper.

2 Preliminaries

WSs provide their functionalities through WSDL operations. In this paper, we assume for simplicity that every WS has only one request-response operation. That is, a message comes to the WS, and the WS produces a message in response. In addition, we introduce the following three types of QoS attributes that are the primary concern of researchers in the area of WS replication:

- *Latency*: The time between sending a request and receiving the response.
- *Availability*: The probability that a WS is available.
- *Cost*: The financial remuneration that a client has to pay for invoking a WS.

Here, we assume that WS latency is deterministic for two reasons. Firstly, this assumption can greatly simplify replication QoS computation. As will be discussed later, the fitness function of the proposed GA depends on replication QoS, so this assumption will introduce some errors in the QoS computation, but it gives us more space to present the idea of DAG coding. Secondly, if we decide to consider uncertain WS latency, we only need to conduct probability distribution calculation to revise the fitness function, and other components of the proposed approach can be directly applied without any modification.

In order to describe all replication schemes, we present here a graph model. Specifically, we express a replication scheme as a labeled DAG $G \equiv (V, E)$ where the vertex set V represents a set of WSs used for replication, and the directed edge set E defines the invocation order over these WSs.

Given the DAG of a replication scheme, a WS is invoked as long as two conditions hold: 1) no response has been received; 2) its in-neighborhood is null or its direct predecessors are all unavailable. Once the scheme receives the first response from invoked WSs, it forwards this response to the client and discards other later responses. If all WSs are unavailable and the scheme cannot receive any response, it will inform the client of the failure.

For example, in Fig. 1(a), s_1 is first invoked as its in-neighborhood is null and no response has been received yet. If s_1 is available, its response is sent to the client. Otherwise, s_2 is invoked as all its direct predecessors are unavailable. Similarly, s_3 is invoked if s_2 is unavailable, and s_4 is invoked if s_3 is unavailable. Therefore, the DAG in Fig. 1(a) represents *passive* replication. Similarly, we can learn that Fig. 1(b) describes *active* replication of four WSs. Other DAGs in Fig. 1 depict complicated replication schemes, including *active-passive* and *passive-active*, whose details can be found in [8].

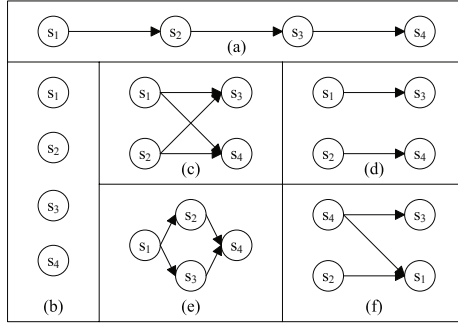


Fig. 1. DAG of different replication schemes of 4 WSs: (a)passive; (b)active; (c)active-passive; (d)passive-active; (e)hybrid-1; (f)hybrid-2

3 Motivating Example and Problem Statement

Suppose the four WSs in Table 1 are used for replication. It is clear from Table 1 that the schemes shown in Fig. 1 need different latency and cost to achieve the same availability. To determine which one is the best, we adopt the *simple additive weighting* method [7] to assign a score to a scheme by the formula $\omega_1c + \omega_2l$, where c and l are its cost and latency, respectively. The parameters ω_1 and ω_2 are domain-dependent and thus can be adjusted dynamically, but here are both set to 0.5 for computation. From Table 1, replication scheme hybrid-2 is clearly the optimal. Unfortunately, to the best of our knowledge, hybrid-2 is not considered by any existing work, which shows the deficiency of current research on optimal WS replication. To overcome this shortcoming, we adopt DAG as the modeling tool for replication schemes. Next, we give a formal definition of our target problem.

Assume G is the set of DAG with n vertices. Then, our target problem - quality driven WS replication - can be formulated as follows:

$$\min_{g \in G} \{ \omega_1c_g + \omega_2l_g \} \tag{1}$$

where c_g and l_g are the cost and latency of a replication scheme described by a DAG $g \in G$, respectively.

Table 1. QoS of four WSs and six replication schemes. a : availability

WS	a	c	l	scheme	a	c	l	score
s_1	0.9	4	3	passive	0.998	4.148	3.244	3.701
s_2	0.8	1	2	active	0.998	10.00	1.366	5.683
s_3	0.7	2	1	active-passive	0.998	5.100	2.238	3.669
s_4	0.6	3	4	passive-active	0.998	5.640	2.232	3.936
				hybrid-1	0.998	4.318	3.154	3.736
				hybrid-2	0.998	4.256	2.522	3.404

According to [3], we have $|G| = n!2^{\binom{n}{2}}M^{-1}p^{-n}$ where $p = 1.488$ and $M = 0.474$, which means the number of replication schemes is growing exponentially with the number of WSs used for replication. Therefore, we decide to adopt and adapt GA to devise a general approach that sacrifices the theoretical optimality but can produce near-optimal (even optimal) solutions in practice.

4 Approach

4.1 QoS of Hybrid Replication

We first describe how to estimate the expected cost and latency of a replication scheme. The expected cost obviously equals to the sum of the expected cost of every involved WS. Suppose WS i is invoked at the probability p_i , then we have:

$$c_g = \sum_{i=1}^n p_i c_i \tag{2}$$

To determine p_i , we introduce two notations: st_i (expected start time) is the time that WS i is expected to be invoked; ft_i (expected finish time) is the time that WS i is expected to be finished. Note that, "finish" here has two meanings: 1) the response of WS i has been received; 2) the response of WS i is not received and WS i is then considered to be unavailable. Then, a WS j is said to be a *temporal predecessor* of a WS i if and only if $ft_j \leq st_i$.

Recall that WSs whose in-neighborhood is null must be invoked. For other WSs, say, i , it needs to be invoked if all its temporal predecessors are unavailable, so its invocation probability p_i can be calculated by:

$$p_i = \prod_{j \in TP(i)} (1 - a_j) \tag{3}$$

where $TP(i)$ is the set of temporal predecessors of WS i .

To calculate the expected latency of a replication scheme, we first arrange the expected finish time of n WSs in an ascending order ft_1, \dots, ft_m ($m \leq n$). Then, these n WSs can be divided into m sets. Each set $S_k = \{i_{k1}, \dots, i_{k|S_k|}\}$ ($1 \leq k \leq m$) contains $|S_k|$ WSs whose expected finish time is ft_k . Suppose event e_i ($1 \leq i \leq m$) meets the condition that "at least one WS in S_i is available and all WSs in S_j ($j \leq i - 1$) are unavailable" and event e_{m+1} is "all WSs in S_j ($1 \leq j \leq m$) are unavailable". If event e_i ($1 \leq i \leq m$) occurs, the expected latency equals to the expected finish time of WSs in S_i . If event e_{m+1} occurs, the expected latency equals to the expected finish time (the second meaning of "finish") of WSs in S_m . As these events are mutually exclusive, we have:

$$l_g = \sum_{i=1}^m p(e_i)ft_i + p(e_{m+1})ft_m \tag{4}$$

$$p(e_i) = (1 - \prod_{j \in S_i} (1 - a_j)) \prod_{k=1}^{i-1} \prod_{r \in S_k} (1 - a_r) (1 \leq i \leq m) \tag{5}$$

$$p(e_{m+1}) = \prod_{k=1}^m \prod_{r \in S_k} (1 - a_r) = \prod_{i=1}^n (1 - a_i) \tag{6}$$

4.2 DAG Coding Based GA

To apply GA in our problem, we first need to represent a replication scheme by a chromosome. Instead of using adjacency matrix, we adopt and adapt DAG code [6] for chromosome design.

Definition 1. Let $D = \{A_1, \dots, A_{n-1}\}$, where $A_i \subset V = 1, \dots, n$ for $i = 1, \dots, n - 1$. D is the code of a DAG with the vertex set V iff $|\bigcup_{i=1}^k A_i| \leq k$.

For example, $\{\{2\}, \emptyset, \{1\}\}$ and $\{\{1\}, \{1\}, \{2, 3\}\}$ are the code of the DAG shown in Fig. 1(d) and (e), respectively.

Definition 2. Suppose $S = \{1, \dots, n\}$. The lexicographic order \prec_{lex} is defined as a binary relation over 2^S , such that if $T_1 \in 2^S$ and $T_2 \in 2^S$, then $T_1 \prec_{lex} T_2$ iff either $T_1(1) < T_2(1)$ or $\exists k, \forall i < k, T_1(i) = T_2(i)$ and $T_1(k) < T_2(k)$ where $T(i)$ is the i th element of T (the elements in T is in an ascending order).

Definition 3. Suppose $S = \{1, \dots, n\}$. The first-size-then-lexicographic order \prec_{sl} is defined as a binary relation over 2^S , such that if $T_1 \in 2^S$ and $T_2 \in 2^S$, then $T_1 \prec_{sl} T_2$ if and only if either $|T_1| < |T_2|$ or $|T_1| = |T_2|$, and $T_1 \prec_{lex} T_2$.

For example, if $S = \{1, 2, 3\}$, then we have:

$$\emptyset \prec_{lex} \{1\} \prec_{lex} \{1, 2\} \prec_{lex} \{1, 2, 3\} \prec_{lex} \{1, 3\} \prec_{lex} \{2\} \prec_{lex} \{2, 3\} \prec_{lex} \{3\}$$

$$\emptyset \prec_{sl} \{1\} \prec_{sl} \{2\} \prec_{sl} \{3\} \prec_{sl} \{1, 2\} \prec_{sl} \{1, 3\} \prec_{sl} \{2, 3\} \prec_{sl} \{1, 2, 3\}$$

Definition 4. Suppose $S = \{1, \dots, n\}$. The first-size-then-lexicographic rank (or rank for short) of a subset T of S is defined as

$$rank(T, n) = |\mathbb{T}| \text{ where } \mathbb{T} = \{T_i | T_i \prec_{sl} T\}$$

For example, if $S = \{1, 2, 3\}$, we have $rank(\{2\}) = 2$, and $rank(\{1, 2, 3\}) = 7$.

We are now in the position to present our chromosome design. For a replication scheme containing n WSs, assume $\{A_1, \dots, A_{n-1}\}$ is the code of its DAG, then its corresponding chromosome is defined as a tuple

$$[rank(A_1, n), \dots, rank(A_{n-1}, n)]$$

For example, the chromosomes of the replication schemes described in Fig. 1(d) and (e) are $[2, 0, 1]$ and $[1, 1, 8]$, respectively.

The above chromosome design not only facilitates the implementation but also can improve the efficiency of basic GA operations, which is explained as follows:

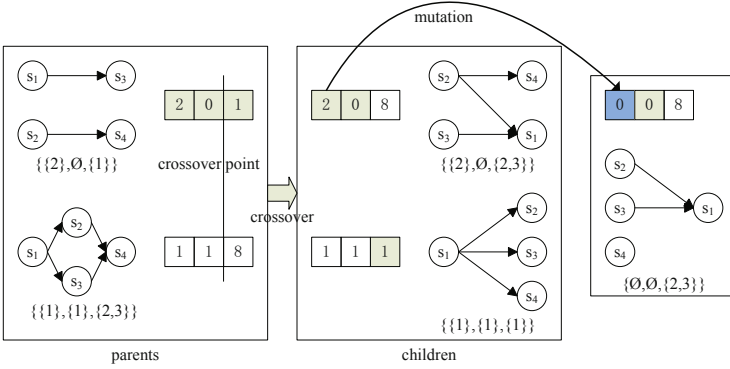


Fig. 2. Crossover and mutation of chromosome built on DAG code

- *Crossover.* Using DAG coding, a chromosome can be represented by integer arrays. In Fig. 2, two new chromosomes $[2, 0, 8]$ and $[1, 1, 1]$ are generated after a one-point crossover between $[2, 0, 1]$ and $[1, 1, 8]$. Note that crossover cannot always generate valid chromosomes, but an invalid chromosome can be detected easily based on the definition of DAG code. For an invalid chromosome, its fitness is set to 0, so it will eliminate in the evolution process.
- *Mutation.* Recall that a DAG code requires $|\bigcup_{i=1}^k A_i| \leq k$, and the allele (i.e., the specific value of a gene) equals to the first-size-then-lexicographic rank of a set, so we have $0 \leq \text{rank}(A_k) \leq \sum_{i=0}^k \binom{n}{i}$. Consequently, mutation can be implemented by simply generating a random number within this range.
- *Initialization.* This becomes extremely simple as the DAG code also gives a unique rank to every DAG. Therefore, to generate an initial population with size of m , we only need to generate m random integers. For example, when $n = 4$, there are 543 DAGs. An initial population with 20 chromosomes is actually 20 random integers between 0 and 542.

Another key element of GA is fitness function, which is defined as follows:

$$\phi(ch) = \begin{cases} \frac{1}{\omega_1 c_g + \omega_2 t_g}, & \text{if } ch \text{ is a valid chromosome} \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

where $ch = [\text{rank}(A_1, n), \dots, \text{rank}(A_{n-1}, n)]$ is said to be a valid chromosome if $\{A_1, \dots, A_{n-1}\}$ is a DAG code.

5 Experiment

In this section, we study the performance of the proposed approach. To facilitate later discussion, the replication scheme generated by our approach is called DAR. Our study includes two parts: 1) quality comparison of DAR and other existing schemes; 2) effectiveness of DAG coding in GA. All experiments are performed on a PC with 2.13GHz Intel Core i3 CPU and 2GB of RAM.

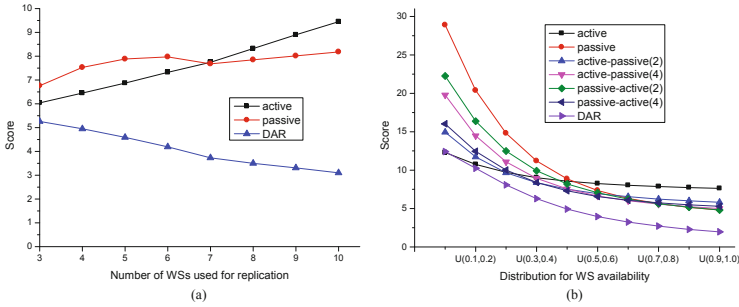


Fig. 3. Quality comparison: DAR vs. some well-known replication schemes

Table 2. Performance comparison: DAG code vs. adjacency matrix

Performance	Chromosome	Number of WSs used for replication							
		3	4	5	6	7	8	9	10
Time (ms)	DAG code	492	796	1257	1780	2322	2915	3554	4004
	Adjacency matrix	1435	3061	N/A	N/A	N/A	N/A	N/A	N/A
Score	DAG code	5.26	4.95	4.58	4.19	3.72	3.49	3.32	3.12
	Adjacency matrix	5.30	5.02	N/A	N/A	N/A	N/A	N/A	N/A

In the first part of experiments, we first compare DAR with two basic schemes: active and passive. Given n WSs used for replication, we first generate their QoS according to the following settings: 1) the cost of a WS is uniformly distributed over $[0,3]$; 2) the latency of a WS is uniformly distributed over $[1,15]$; 3) the availability of a WS is uniformly distributed over $(0,1)$. Then, we compute for every scheme its score defined in formula 1 where $\omega_1 = \omega_2 = 0.5$. 100 groups of n WSs are generated randomly and the average scores are reported.

Fig. 3(a) shows the scores of DAG and basic schemes where we vary n from 3 to 10. Clearly, DAR performs the best, and the score difference between DAG and basic schemes becomes larger as n increases, which indicates that DAR is especially suitable when there are lots of WSs used for replication.

Next, we compare DAR with some complicated schemes by setting the number of WSs used for replication to be 8. Four specific schemes are considered here: active-passive(2), active-passive(4), passive-active(2), and passive-active(4) (Detail of active-passive and passive-active schemes can be found in [8], but are omitted here due to space limitation). In addition, we use the same experimental setting discussed above, except that the availability distribution varies from $U(0,0.1)$ to $U(0.9,1)$, instead of a fixed uniform distribution over $(0,1)$, as we want to examine how WS availability affects the performance of replication. The experimental result is shown in Fig. 3(b), where we see it again that DAR performs the best compared with any other replication schemes.

In the second part of experiments, we implemented two types of GA: one adopts DAG code, and the other adopts adjacency matrix for chromosome

design. From Table 2, we can see that DAG code based GA works well in all cases, but adjacency matrix based GA can only give answers (with worse quality and larger computation time) when $n \leq 4$. Hence, DAG code is a key factor in ensuring the success of GA in the problem of quality driven WS replication.

6 Conclusion

Replication can effectively improve WS availability. In this paper, we have proposed a GA based approach to efficiently designing a near-optimal replication. By adopting DAG as the modeling tool for replication schemes, we are capable of exploring all possible schemes in the problem solving phase and utilizing DAG coding to improve the performance of GA. Concerning future work, we plan to make our approach more practical by adopting probability distribution to describe the uncertain WS latency and deducing a more accurate estimation of replication QoS. In addition, we plan to apply our approach to real WSs to further demonstrate its feasibility.

Acknowledgments. The work is supported by the National Natural Science Foundation of China under Grant No. 61003044, a strategic research grant from City University of Hong Kong (project no. 7002606), the Natural Science Foundation of Jiangsu Province under Grant No. BK2010257, and State Key Laboratory of Software Engineering (SKLSE).

References

1. Guo, H., Huai, J., Li, H., Deng, T., Li, Y., Du, Z.: ANGEL: Optimal Configuration for High Available Service Composition. In: ICWS 2007, pp. 280–287. IEEE Computer Society, Los Alamitos (2007)
2. Maamar, Z., Sheng, Q.Z., Benslimane, D.: Sustaining Web Services High-Availability Using Communities. In: ARES 2008, pp. 834–841. IEEE Computer Society, Los Alamitos (2008)
3. McKay, B.D., Oggier, F.E., Royle, G.F., Sloane, N.J.A., Wanless, I.M., Wilf, H.S.: Acyclic Digraphs and Eigenvalues of (0,1)-Matrices. *Journal of Integer Sequence* 7, article 04.3.3 (2004)
4. Salas, J., Perez-Sorrosal, F., Patino-Martinez, M., Jimenez-Peris, R.: WS-Replication: A Framework for Highly Available Web Services. In: WWW 2006, pp. 357–366. ACM, New York (2006)
5. Stein, S., Payne, T.R., Jennings, N.R.: Flexible Provisioning of Web Services Workflows. *ACM Transactions on Internet Technology* 9(1) (February 2009)
6. Steinsky, B.: Efficient Coding of labeled Directed Acyclic Graphs. *Soft Computing* 7(5), 350–356 (2003)
7. Yoon, K.P., Hwang, C.-L.: Multiple Attribute Decision Making: An Introduction (Sage University Paper Series on Quantitative Applications in the Social Sciences, 07-104). Thousand Oaks, CA (1995)
8. Zheng, Z., Lyu, M.R.: A Distributed Replication Strategy Evaluation and Selection Framework for Fault Tolerant Web Services. In: ICWS 2008, pp. 145–152. IEEE Computer Society, Los Alamitos (2008)

OntoAssist: Leveraging Crowds for Ontology-Based Search

Winston H. Lin and Joseph Davis

School of Information Technologies, The University of Sydney
Sydney, Australia
{huai ren.lin, joseph.davis}@sydney.edu.au

Abstract. In this demonstration we present OntoAssist, a semantic navigation support system designed to address the ontology evolution problem and thus improve the quality of ontology-based search. OntoAssist integrates the computational features of an existing search engine with the human computation provided by the crowd of users to find useful search results.

Keywords: crowdsourcing, semantic search, ontology evolution.

1 Introduction

As the internet and the World Wide Web (WWW) continue to evolve, the current Web 2.0 technology leaves much to be desired in terms of speed, innovation and support. Also, as the amount of user-generated content increases, exploration and retrieval of useful and relevant information poses challenges.

Developments in semantic web technologies offer us a new approach to managing information online and overcoming the semantic problems. The use of ontology in Semantic Web allows users to more precisely express their queries and thus improve the search precision and recall. However, ontologies need to be frequently updated to compile the knowledge emerging from the daily experiences of online users. At this point, ontology development is primarily based on the manual efforts of skilled experts and professionals. Many significant challenges have to be overcome before we can achieve greater maturity in semantic search.

In this demonstration, we develop a crowdsourcing system for blending ontology evolution tasks seamlessly with public users' daily search activities. With this design, our proposed crowdsourcing approach can get actual users involved without necessarily offering a monetary reward. Our design allows users to refine their searches on web repositories by choosing relationships between query keywords and relevant terms in the search results. With a few simple clicks, an online user helps the initial ontology to evolve as well as provide better search results for that particular query.

2 OntoAssist

In this section, we describe OntoAssist, a prototype implementation of a crowdsourcing application. It is a semantic navigation support system designed to

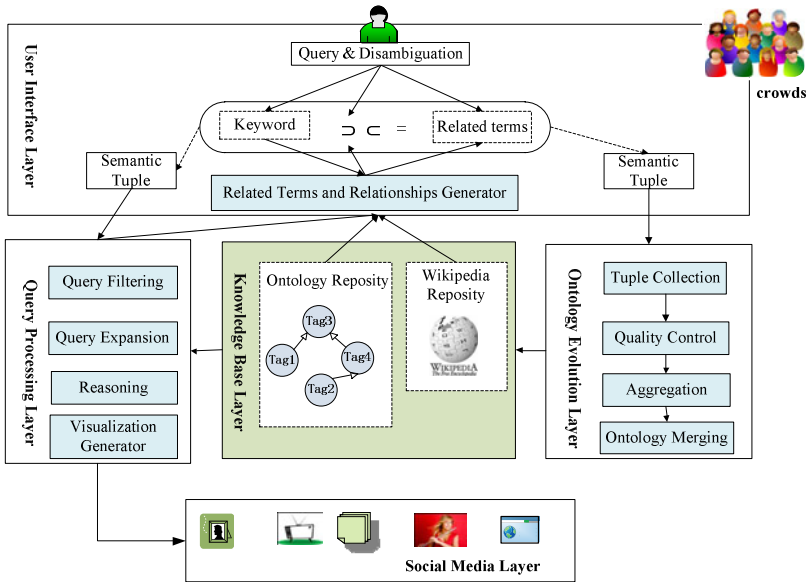


Fig. 1. OntoAssist System Architecture

address the ontology evolution problem. See figure 1 for the general architecture of OntoAssist. We focus on three key features of the system:

Semantic Navigation User Interface: The design of the semantic navigation component in OntoAssist is based on general search assist tools most search engines provide. We attempt to collect both these related terms and their relationships for ontology evolution purpose. The semantic navigation component allows the users to express their search intent as a tuple (*keyword, relation, related term*). For instance, a user can refine an original query *python* by the tuple: (*python, is a kind of, programming language*).

Crowdsourcing Based Ontology Evolution: The power of semantic navigation comes from the underlying ontology. The improved semantic navigation experience is linked closely to the evolution of that ontology. OntoAssist aggregates a large amount of user inputs collected from the semantic navigation interface to evolve the base ontology.

Ontology-Based Search: By assigning relationship between the query keyword and one of the related terms, a user is able to express his/her query intent in a machine understandable format. Thus, the search result can be improved by performing an semantic search with the meaning expressed.

As an example, the OntoAssist tool was implemented based on Yahoo BOSS and released at www.hahia.com. It offers the ability to perform semantically enriched exploration and search of the resources on the web.

MashSheet: Mashups in Your Spreadsheet

Dat Dac Hoang, Hye-Young Paik, and Wei Dong

School of Computer Science & Engineering,
The University of New South Wales
Sydney NSW 2052, Australia
{ddhoang, hpaik, wdon049}@cse.unsw.edu.au

Abstract. We demonstrate MashSheet, a spreadsheet-based, generic purpose mashup development framework that allows users to create applications by using spreadsheet-like formulas. The key innovation of MashSheet is a collection of operators that supports orchestrating Web services, manipulating and visualising data created by the services. MashSheet applications are incrementally built and data in each intermediary step is only visualised when needed. It makes the mashup application concise and easy to follow, as well as keeping the computation logic separate from presentation. In the demo, we will show executions of mashup operators, along with use case applications from real-world scenarios.

1 Introduction

In Web 2.0, “mashups” is considered as a key enabler in self-service application development. With a mashup tool, individuals rapidly build applications without any in-depth technical skill requirements. Typically, mashup tools possess user-friendly interfaces (e.g., visual, browser-based) with simple user interaction mechanisms (e.g., drag-and-drop).

Spreadsheets are popular productivity tools and are useful in easy manipulation, analysis, visualisation and reporting of data. In [3], Scaffidi et al. estimate that there will be 55 million spreadsheet’s users in America alone in 2012, which provides a compelling argument for creating a mashup environment customised for spreadsheets.

In this paper, we demonstrate MashSheet. We extend conventional spreadsheet paradigms to facilitate Web services “mashup” in a spreadsheet environment. The users, who are familiar with spreadsheet concepts, can use pre-defined operators as spreadsheet formulas to orchestrate Web services, manipulate/visualise data to create mashup applications. The key innovation of MashSheet is a collection of operators that supports orchestrating Web services, manipulating and visualising data created by the services.

In recent years, there have been a few attempts to create mashup development environments in spreadsheets with varying degree of success [1, 2]). However, our tool is unique in that (i) the syntax of the operators is based on a paradigm that many users are already familiar with, (ii) the operators support basic control and data flow (rather than simple sequences), (iii) there is a separation of data and presentation for flexibility and (iv) reuse of already-built mashup applications.

In what follows, we present a system overview and demo scenarios.

¹ Our survey identifies the systems like StrikeIron, AMICO-CALC, Mashroom, SpreadMash in detail.

2 System Overview

MashSheet Operators. MashSheet operators are classified into four main categories: *lifecycle* (e.g., `bind`), *process* (e.g., `invoke`), *data* (e.g., `merge`), and *visualisation* (e.g., `grid`, `chart`). The life cycle operators are used for managing the representations of Web services within the framework. The process operators underpin orchestration of Web services. The data operators are used for manipulating the data generated by Web services' invocations. The goal of the visualisation operators is to separate the presentation layer from data layer so that users can visualise the same data using different layouts, without having to re-invoke services.

Application model. A MashSheet application is a worksheet that contains a collection of cells organized in a tabular grid. Each cell is identified by its address, value and formula. An address uses absolute coordinates of the cell in the grid (e.g., A1). A formula may contain a value (e.g., a number), a reference to another cell address (e.g., A1) or MashSheet operators (e.g., `invoke()`). A cell's value is obtained by evaluating the cell's formula. A value can be either a simple data type (e.g., number, string) or complex data type. We model the complex type as XML-based data type and name it MashSheet Object Type (MOT). Using MOT, we build two wrapper components for representing a Web service and its output data.

Implementation. MashSheet is implemented as a plug-in of Microsoft Excel program. It extends the work presented in [2], which is mainly designed for importing data services into a spreadsheet. The language of choice for implementation is C# using Visual Studio Tools for Office toolkit.

3 Demo Scenario

In this demonstration, we first present an overview of the MashSheet framework. We, then, demonstrate how to use MashSheet through three use cases. We will present three scenarios, each showing different aspects of the tool. In the first scenario, we use a News feed service to show data import, filtering and multiple visualisations of mashup results. In the second scenario, we present a simple Web service composition using Driving Guide and Google Maps services. Lastly, we show a complex composition in MashSheet and also demonstrate how reuse is supported through templates. The MashSheet introduction, screen casts, and video demos are also available at: <http://mashsheet.cse.unsw.edu.au:8080/Demo/>.

References

1. Hoang, D.D., Paik, H.-y., Benatallah, B.: An analysis of spreadsheet-based services mashup. In: ADC 2010, Brisbane, Australia (2010)
2. Saint-Paul, R., Benatallah, B., Vayssière, J.: Data services in your spreadsheet. In: EDBT 2008, Nantes, France (2008)
3. Scaffidi, C., Shaw, M., Myers, B.: Estimating the numbers of end users and end user programmers. In: VL/HCC 2005, Dallas, USA (2005)

SOAC Engine: A System to Manage Composite Web Service Authorization

Haiyang Sun, Weiliang Zhao, Jian Yang, and Guizhi Shi

Department of Computing, Macquarie University, Australia
 {haiyang.sun,weiliang.zhao,jian.yang,Guizhi.shi}@mq.edu.au

1 Introduction

The authorization of composite web services is different from traditional authorization in a close system due to the dynamic and complex relationships among service consumers and resources (component services). This demonstration is to show the functionality of a system named Service Oriented Authorization Control Engine (*SOAC Engine*).

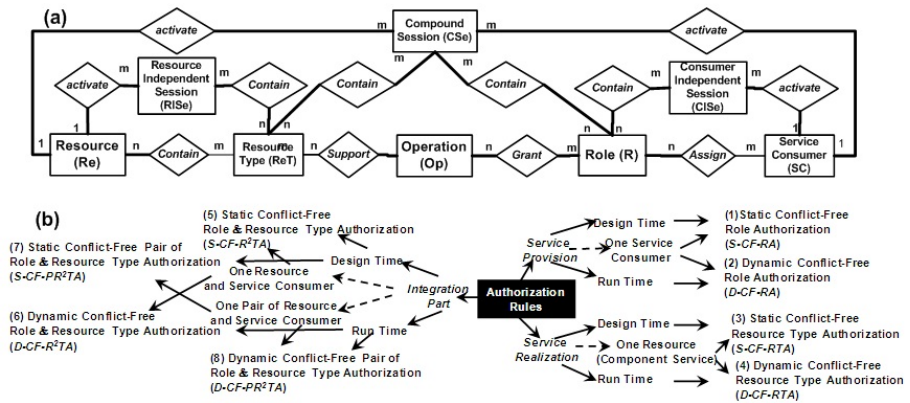


Fig. 1. (a) SOAC Conceptual Model and (b) Conflict-Free Authorization Rules

SOAC Engine has the following features: (1) *SOAC Engine* is developed based on our proposed conceptual model [1], *SOAC* (See (a) in Fig. 1), which is an extension of role based access control for the authorization of composite web services. In *SOAC Engine*, the authorization of a composite web service to a service consumer will consider "who can do what *under what kind of support*". (2) Two concepts, Role and Resource Type, are introduced in *SOAC Engine* in order to reduce the administrative overhead in the authorization management. They are mapped to component services and service consumers respectively based on their characteristics. (3) Authorization rules are embedded in *SOAC Engine* to prevent eight types of conflict of interest in terms of composite web service authorization at both design time and runtime (See (b) in Fig. 1).

2 System Architecture

The *SOAC Engine* (See Fig. 2) is developed with JAVA and Oracle, and resided in local web server which transfers the authorization messages between service consumer and the engine through **Application Interface**. **Authorization Management Interface** of *SOAC Engine* is used by administrator to (1) set the elements of SOAC and their relationships, (2) identify the conflicted relationships among elements, and (3) manage the authorization policies for both composite web service and resource. The *SOAC Engine* includes four component packages: (1) **Authorization Administrative Management** manages the elements of SOAC and their relationships, e.g., querying the assigned roles for specific service consumer. (2) **Credential Management** manages credentials for service consumers and resources. (3) **Authorization Decision Making** performs the policy compliance checking to make authorization decisions. The conflict of interest is checked in this packages. (4) The **Authorization Enforcement** controls the enforcement of authorization decisions related with the composite web service and its component services.

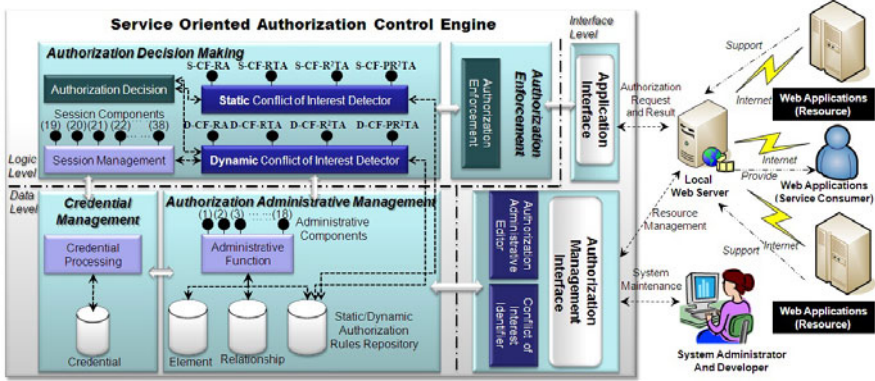


Fig. 2. System Architecture of SOAC Engine

Reference

[1] Sun, H., Zhao, W., Yang, J.: Managing Conflict of Interest in Service Composition. In: Proceedings of CoopIS 2010, pp. 273–290. Springer, Heidelberg (2010)

A Change Analysis Tool for Service-Based Business Processes

Yi Wang, Jian Yang, and Weiliang Zhao

Department of Computing, Faculty of Science,
Macquarie University, Sydney, NSW 2109, Australia

Abstract. We develop an enabling tool referred to as Service Change Analyzer for analyzing change impact in service-based business processes. The presented tool in this demo is an implementation of our proposed approach for change management which includes a service-oriented business process model, a taxonomy of changes and change impact patterns associated with services and business processes, and algorithms for calculating change impact scopes.

Keywords: Service-oriented computing, Web service, Change management.

1 Introduction

Change management is critical and challenging in the context of service-oriented computing paradigm when multiple services are supported by a single business process [1]. The complex dependencies between services and business processes have not been carefully investigated in the existing work. In particular, it is still lacking of an effective tool to support change impact analysis for coupled services and business processes. In this demo, we present a tool: **S**ervice **C**hange **A**nalyzer (**SCA**) for enabling impact analysis when a change happens. The SCA is built up based on our previous work for change analysis, in which a service-oriented business process model is developed, types of changes associated with services and business processes are identified, and a set of change impact patterns for capturing change effect are specified [2]. The SCA accepts service changes as its input and it can give the detailed analyzed results for the change impact scope and provide suggestions for possible change impact patterns. This tool provides developers a standard practice to change the complicated change management tasks into a series of simple standard procedures. With the help of the tool SCA, the time and cost of change management tasks can be dramatically reduced.

2 System Architecture

The SCA is a JAVA based tool which focuses on a type of dependencies that multiple services are supported by a single business process. The SCA can accept various types of service changes as its input based on our identified change

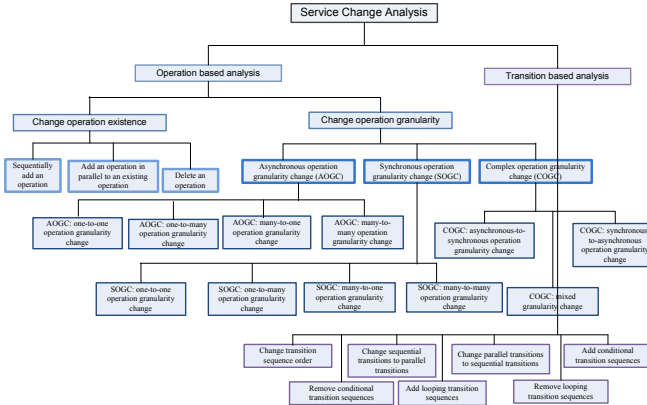


Fig. 1. Hierarchy diagram

taxonomy. For each type of service change, the SCA provides the corresponding interface for users to specify the details of the change. For each input service change, the SCA can calculate the impact scopes and provide suggestions for the potentially used change impact patterns.

The SCA has two major modules as: **operation based analysis** and **transition based analysis** (Figure 1). The **operation based analysis** is realized by: *change operation existence* and *change operation granularity*. The change operation existence module has three sub modules as: sequentially add an operation, add an operation in parallel to an existing operation, and delete an operation. The change operation granularity module consists of three sub modules as AOGC, SOGC, and COGC, which deal with the impact analysis for operation granularity changes. The **transition based analysis** has seven sub modules that handle impact analysis relating to transition changes.

3 Demonstration Overview

This demo will provide a set of running scenarios. The developed service change analyzer accepts 21 types of service changes [2] as its input. For an input change, the analyzer can calculate and show the impact scopes including the affected activities and data connections in the associated business process and the potentially affected services. Possible impact patterns to handle the change are also suggested.

References

1. Papazoglou, M.P.: The challenges of service evolution. In: Bellahsene, Z., Léonard, M. (eds.) CAISE 2008. LNCS, vol. 5074, pp. 1–15. Springer, Heidelberg (2008)
2. Wang, Y., Yang, J., Zhao, W.: Change impact analysis for service based business processes. In: SOCA 2010, pp. 1–8 (2010)

A Novel Approach for Interacting with Linked Open Data

Armin Haller¹ and Tudor Groza²

¹ CSIRO ICT Centre

`armin.haller@csiro.au`

² School of ITEE, The University of Queensland

`tudor.groza@uq.edu.au`

Abstract. The continuous growth of the Linked Data Web brings us closer to the original vision of the Web, as an interconnected network of machine-readable resources. However, the ability to easily manipulate Linked Open Data (LOD) is still an open issue. In this demonstration we will show through a lifecycle model how to enable a domain-agnostic read/write interaction with LOD. Our solution uses an ontology to build a binding front-end for a given RDF model, in addition to RDFa to maintain the semantics of the resulting form/widget components. On the processing side, a RESTful Web service is provided to seamlessly manage semantic widgets and their associated data, and hence enable the read/write data interaction mechanism.

1 Introduction

The emergence of the Linked Data Web [2] over the last years has led to an explosion of datasets being openly published on the Web. Interacting with the large diversity of existing Linked Open Data (LOD) in a read/write fashion is, however, still an open research topic. Some work has already been done in the area (e.g., [3][1][6]), in general by creating (HTML) forms as a medium between RDF data and humans. While this bridging concept represents probably the best solution, the underlying technical aspects of current approaches share a common drawback: they either require manual mapping between domain concepts and form components, or are specifically tailored for a particular domain (i.e., forms generated for a specific schema / ontology describing a dataset). This demonstration presents a novel solution aimed to bring us a step closer to the original vision, by providing a lifecycle and associated mechanism to enable read/write interaction with Linked Data. This interaction takes place mainly in a “local” dataset context (i.e., in a context of a dataset exposed by an organization), but taking advantage also of the “global” Linked Data Web context.

2 A Lifecycle Model for Data Manipulation in RDFa

We propose a lifecycle, which is applicable on arbitrary RDF graphs (hence being domain-agnostic) and comprises three phases, i.e., widget generation, deployment and usage, all using as back-end the ActiveRaUL Web service [4].

I. Widget Generation. The first step of the lifecycle creates a widget model in RDF, using the RaUL ontology [5]. The RaUL ontology (defining the widget model) consists of two parts: (i) **Form controls** describing the structure of the widget, and (ii) a **Data model** defining the structure of the exchanged data. The *data model* expresses RDF statements, which are referenced from the *form controls* via a data binding mechanism. The *data model* gives meaning to the data used in the *form controls* by uniquely referencing standard ontologies on the Semantic Web.

II. Widget Deployment. The ActiveRaUL service offers an endpoint to deploy a widget with a POST request to its `/public/forms` resource and gives the user two options for the payload: (i) a handcrafted **RaUL-based widget model** that can be deployed within a public namespace `/public/forms` of the ActiveRaUL service backend, with the goal of re-using the generic form models in other Semantic Web applications and eventually becoming a standard ontological widget model (e.g. a user registration form shared between many sites), and (ii) an **Arbitrary RDF graph**, which sent to the same endpoint will trigger the ActiveRaUL service to perform a best-effort generation of a widget according to some mapping rules.

The ActiveRaUL service supports different data representations, such as RDF/XML, RDF/JSON or RDF/N3. If the POST request is successful, the service will return the URL of the newly created widget in the HTTP Location header of the response, for example, `/public/forms/addproduct`.

III. Widget Usage. A widget can be retrieved by sending an HTTP GET request to ActiveRaUL using the URL returned after the successful deployment. It depends on the HTTP Accept header to determine what kind of representation is sent back to the client. Whenever the user fills in the form or changes a previously submitted one, the ActiveRaUL client-side JavaScript API processes the form and sends the appropriate HTTP message to the ActiveRaUL service.

Data Reuse. An important feature in this lifecycle is the reuse of existing data. RaUL-based widgets can explicitly reference data from the Linked Data Web. Binding form controls to ontological relations (e.g., *foaf:givenName*), enables us to use the same mechanism to reference external data. In practice, all form controls that define an *owl:sameAs* relation are treated as references to an external vocabulary. This relation also asserts that the individual which is assigned a URI by the ActiveRaUL Web service is in fact the same individual having a different URI somewhere on the Linked Data Web. The client-side Javascript RDFa API will try to retrieve data about the resource from the URI provided in the *textbox* and fill-in the remaining form controls at runtime with this data.

3 Conclusion

In this demonstration we have introduced a lifecycle model to manipulate data in the Linked Data Web. We proposed RDFa annotated Web widgets which follow the principles of Linked Open Data. We developed a client-side Javascript RDFa

API based on jQuery that parses RaUL annotated HTML widgets and performs a data binding based on the user input or data referenced in the Linked Data Web. The RDFa API interacts with the ActiveRaUL service, a generic RESTful Web service enabling developers to deploy and manage their Web forms and the data model associated with these forms.

References

1. Berners-Lee, T., Cyganiak, R., Hausenblas, M., Presbrey, J., Sneviratne, O., Ureche, O.-E.: On Integration Issues of Site-specific APIs into the Web Of Data. Technical report, DERI (2009)
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data – The Story So Far. In: International Journal on Semantic Web and Information Systems, IJSWIS (2009)
3. de Ohra, B.: Automated mapping between RDF and forms (2005), http://www.dehora.net/journal/2005/08/automated_mapping_between_rdf_and_forms_part_i.XHTML
4. Haller, A., Rosenberg, F.: A Semantic Web Enabled form model and restful service implementation. In: Proceedings of SOCA 2010 (2010)
5. Haller, A., Umbrich, J., Hausenblas, M.: RaUL: RDFa User Interface Language – A data processing model for web applications. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 400–410. Springer, Heidelberg (2010)
6. Hausenblas, M.: RDFS Forms Vocabulary (2010), <http://rdfs.org/ns/rdfforms/XHTML>

Cloud Broker: Helping You Buy Better

Mohan Baruwal Chhetri, Quoc Bao Vo, Ryszard Kowalczyk, and Cam Lan Do

Swinburne University of Technology, Hawthorn VIC 3122, Australia
{mchhetri,bvo,rkowalczyk}@swin.edu.au, ronaldo@gmail.com

Abstract. In this paper we present the cloud broker which can help decision makers assess the feasibility of adopting cloud computing in their organizations. It matches their requirements in terms of infrastructure, costs, geographic location and other requisite criteria, to the capabilities of the cloud service providers. Additionally, it allows them to test benchmark applications on selected cloud providers to get a better estimate of their cost, configuration and performance.

Keywords: Cloud Broker, comparison, cost, configuration, performance, benchmark evaluation.

1 Introduction

Cloud computing offers computing as a utility and transforms the way in which new application services are provisioned by promising infinite scalability, ease of development and low infrastructure setup cost. Because of its growing popularity, many companies have entered the market and offer their services under different provisioning models and pricing schemes. The diversity of the cloud service offerings leads to the practical question of *which cloud offering to choose and why?* A cloud broker assists prospective consumers in choosing the cloud provider that best suits their needs by matching their specific requirements to the different cloud service offerings. Additionally, it also allows the consumers to use the *try before you buy* model to test benchmark applications on the short-listed providers to get a better estimate of the cost, configuration and performance of the different providers.

2 System Overview

The cloud broker offers cloud brokerage as a service for Infrastructure as a service (IaaS) customers. It offers the following two key services which will be demonstrated in the demo:

Comparison of Cloud Providers. The cloud broker maintains an active IaaS provider catalogue which contains details about the various providers including their capabilities, business models, pricing models and liabilities. It builds a cost/configuration matrix using a common set of metrics, which it uses to answer specific user queries by matching the specific requirements of the IaaS consumers

to the capabilities of the providers. Such a tool is particularly beneficial for enterprises who want to get a quick overview of the available cloud offerings.

Evaluation of Benchmark Applications. Although most IaaS providers publicly advertise the different service configurations they support and the corresponding prices, their precise nature and the resultant application runtime behaviour is still largely unknown. Given the blackbox nature of these cloud platforms it becomes important for service consumers to be able to test applications under different load conditions on the different cloud providers which may be located in different geographical regions to ensure that the QoS that has been promised to the end consumers is actually achievable.

A detailed description of the cloud broker design and implementation is available at <http://www.it.swin.edu.au/centres/ciamas/tiki-index.php?page=CB>. In the current implementation we support two benchmark applications - the popular enterprise benchmark application TPC-W¹ and SciMark² which is the benchmark application for numeric and scientific computing. Some screenshots of the demonstrator are available at the above link. More will be made available soon.

3 Demo Scenario

The demonstration shows how the cloud broker can assist decision-makers in small and medium enterprises to assess the feasibility of the adoption of cloud computing in their organization. In the first part of the demo, users specify their specific infrastructure requirements and query the cloud broker for matching cloud offerings. The cloud broker returns the list of potential cloud providers that closely match the user query. In the second part of the demo, the user specifies which of the returned cloud providers to test for performance. The user selects the benchmark application that closely matches its own application and specifies the workload to test for. Different users can specify the load conditions they want to simulate and the geographical location in which they want to deploy the application. The cloud broker deploys the benchmark application, runs the test cases and returns the results to the user.

4 Conclusion

The cloud broker helps prospective cloud consumers to compare and contrast the different cloud providers according to their specific requirements. It also helps them test benchmark applications on different cloud providers to get a better estimate of the cost and the performance. These two simple services can help the decision-makers make more informed decisions on whether to migrate the applications to the cloud or not.

Acknowledgement. This work was partially supported by a grant from the Smart Services CRC within the Service Aggregation project.

¹ <http://www.tpc.org/tpcw>

² <http://math.nist.gov/scimark2/>

FACTUS: Faceted Twitter User Search Using Twitter Lists

Takahiro Komamizu¹, Yuto Yamaguchi¹, Toshiyuki Amagasa^{1,2},
and Hiroyuki Kitagawa^{1,2}

¹ Graduate School of System and Information Engineering, University of Tsukuba

² Center of Computational Science, University of Tsukuba

{taka-coma,yuto_ymgc}@kde.cs.tsukuba.ac.jp,

{amagasa,kitagawa}@cs.tsukuba.ac.jp

Abstract. In this demonstration, we present FACTUS, a tag-based Twitter user search system using Twitter lists. FACTUS provides a faceted search over Twitter users. To this end, the system first extracts tags, such as “music” and “sports,” from Twitter lists, and assigns the tags to Twitter users. Then, facets are computed based on the tags, thereby enabling us to browse Twitter users using facets.

Keywords: Twitter, Twitter User Search, Twitter Lists, Faceted Search.

1 Introduction

Twitter [1] is one of the most famous social networking services, and more than 200 million users are using the service to post and/or subscribe short messages. There are many accounts, including famous people, researchers, bots, and so on. For this reason, it is important for one to find out interesting users accounts. In fact, user search is officially supported by the system [2], but its functionality is not sufficient to find interesting users for us.

For this problem, our research group has proposed a tagging scheme for Twitter accounts using Twitter lists [3]. In this scheme, tags, such as “sports” and “music,” are extracted from Twitter lists, and are assigned to accounts based on the correlation between accounts and lists, and that between lists and tags.

Based on the scheme, we have developed FACTUS, a faceted search system over Twitter users. In the system tags are used to compute facets, and users can explore Twitter users by specifying interesting tags in facets. The system also supports keyword search. So, a user can arbitrary combine the two methods to browse Twitter users. To the best of our knowledge, FACTUS is the first work that applies faceted search to Twitter accounts. We show how the faceted search facilitates finding Twitter users.

2 The System

The architecture of FACTUS is shown in Fig. 1. We assume that data of Twitter accounts and Twitter lists are obtained from Twitter service and are stored in *Twitter Database*.

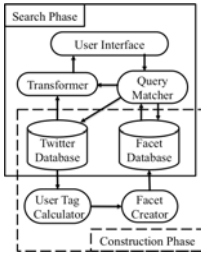


Fig. 1. Architecture



Fig. 2. Screen shot

The process in FACTUS is roughly divided into two phases. In the construction phase, *User Tag Calculator* obtains Twitter lists from the Twitter Database, extract tags, and assigns the tags to the accounts based on point-wise mutual information between accounts and tags [3]. Tags are also delivered to the *Facet Creator*. It computes the relationship between the tags and categories (http://twitter.com/#!/who_to_follow/interests) based on the point-wise mutual information. As a result, we can use the categories as facets and tags with high point-wise mutual information to the categories are values of facets.

In the search phase, a user iteratively selects an interesting tag in a facet. The *User Interface* (Fig. 2) receives the information, and passes it to the *Query Matcher*. The Query Matcher retrieves such accounts from the *Twitter Database* and facets from *Facet Database* that match to the given tag. Finally, the *Transformer* transforms received facets and Twitter accounts into the next screen for the next browsing.

3 Demonstration Scenario

We demonstrate FACTUS using real Twitter data crawled from twitter.com. Audiences can enjoy browsing real Twitter users based on the tags extracted from Twitter lists. The system also supports keyword based account search.

Acknowledgments. This research has been supported in part by the Grant-in-Aid for Scientific Research from MEXT (#23700102, #21240005).

References

1. Twitter, <http://twitter.com>
2. Twitter Search, <http://search.twitter.com/>
3. Yamaguchi, Y., Amagasa, T., Kitagawa, H.: Tag-based User Topic Discovery using Twitter Lists. In: Memon, N., Alhajj, R., Ting, I.H. (eds.) ASONAM. IEEE Computer Society, Los Alamitos (to appear, 2011)

Author Index

- Adams, Brett 227
Amagasa, Toshiyuki 343
- Bai, Quan 249, 306
Bal, Daniella 129
Bal, Malissa 129
Bao Vo, Quoc 341
Baruwal Chhetri, Mohan 341
Bhiri, Sami 199
- Casteleyn, Sven 185
Christophides, Vassilis 29
Colman, Alan 143
Curran, James R. 213
- Davis, Joseph 330
de Knijff, Jeroen 241
Derguech, Wassim 199
De Troyer, Olga 185
Ding, Jun 265
Dong, Wei 332
du Mouza, Cedric 29
Dustdar, Schahram 290
Dyreson, Curtis E. 172
- Fafalios, Pavlos 101
Fokoue, Achille 57
Frasincar, Flavius 129, 241
Fung, Gabriel 282
- Gong, Xueqing 116
Grigera, Julián 257
Groza, Tudor 338
Gu, Yanhui 158
- Hachey, Ben 213
Haller, Armin 338
Han, Jun 143
Haselmann, Till 43
Hmedeh, Zeinab 29
Hoang, Dat Dac 332
Hogenboom, Alexander 129
Hogenboom, Frederik 129, 241
Hu, Fanghuai 265
Huang, Liusheng 322
Hummer, Waldemar 290
- Kapuruge, Malinda 143
Kitagawa, Hiroyuki 343
Kitsuregawa, Masaru 158
Komamizu, Takahiro 343
Kotani, Daisuke 298
Kowalczyk, Ryszard 341
- Lan Do, Cam 341
Leitner, Philipp 290
Li, Qing 322
Lin, Winston H. 330
Lin, Yuming 116
Liu, An 322
Liu, Chengfei 273
- Meijer, Kevin 241
Mekala, Kalyan G. 172
Miller, James 1
Mu, Yi 306
- Nakamura, Satoshi 298
Navarro, Antonio 257
Nguyen, Thin 227
Noor, Talal H. 314
- Paik, Hye-Young 332
Paret, Elien 185
Peng, Peng 72
Phung, Dinh 227
- Qian, Weining 116
- Radford, Will 213
Rivero, José Matías 257
Robles Luna, Esteban 257
Roshanbin, Narges 1
Rossi, Gustavo 257
Ruan, Tong 265
- Satzger, Benjamin 290
Schill, Alexander 87
Scholl, Michel 29
Schuster, Daniel 87
Shao, Zhiqing 265
Sheng, Quan Z. 314
Shi, Guizhi 334

- Srivatsa, Mudhakar 57
Su, Xing 306
Sun, Haiyang 334

Tanaka, Katsumi 298
Thom, James A. 87
Travers, Nicolas 29
Tzitzikas, Yannis 101

Urbansky, David 87

van Bunningen, Arthur 129
Van Woensel, William 185
Venkatesh, Svetha 227
Vossen, Gottfried 43
Vouzoukidou, Nelly 29

Wang, Yi 336

Xu, Guandong 158

Yamaguchi, Yuto 343
Yang, Jian 334, 336
Yang, Zhenglu 158
Ye, Dayong 249
Yongchareon, Sira 273
Young, Robert 57

Zhang, Jingwei 116
Zhang, Minjie 249, 306
Zhang, Xiuzhen 15
Zhang, Yanchun 158
Zhao, Dongyan 72
Zhao, Weiliang 334, 336
Zhao, Xiaohui 273
Zhou, Aoying 116
Zhou, Xiaofang 282
Zhou, Yun 15
Zhu, Jia 282
Zou, Lei 72