Karl Aberer Zhiyong Peng
Elke A. Rundensteiner Yanchun Zhang
Xuhui Li (Eds.)

LNCS 4255

Web Information Systems – WISE 2006

7th International Conference on
Web Information Systems Engineering
Wuhan, China, October 2006, Proceedings

Springer

Lecture Notes in Computer Science 4255

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

Karl Aberer Zhiyong Peng
Elke A. Rundensteiner Yanchun Zhang
Xuhui Li (Eds.)

Web Information Systems – WISE 2006

7th International Conference on
Web Information Systems Engineering
Wuhan, China, October 23-26, 2006
Proceedings

Springer

Volume Editors

Karl Aberer
EPFL
Switzerland
E-mail: karl.aberer@epfl.ch

Zhiyong Peng
Wuhan University
China
E-mail: peng@whu.edu.cn

Elke A. Rundensteiner
Worcester Polytechnic Institute
USA
E-mail: rundenst@cs.wpi.edu

Yanchun Zhang
Victoria University
Australia
E-mail: yzhang@csm.vu.edu.au

Xuhui Li
Wuhan University
China
E-mail: lixuhui@whu.edu.cn

Library of Congress Control Number: 2006934596

CR Subject Classification (1998): H.4, H.2, H.3, H.5, K.4.4, C.2.4, I.2

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

ISSN 0302-9743
ISBN-10 3-540-48105-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-48105-8 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11912873 06/3142 5 4 3 2 1 0

Preface

WISE 2006 was held in Wuhan, China, during October 23–26, hosted by Wuhan University. As more and more diverse information becomes available on the Internet and in the Web, novel theoretical and technical approaches to building and improving Web information systems become of vital importance to the further development of information technology. Following the successful conferences from 2000 to 2005, WISE 2006 provided a premium forum for researchers, professionals, and industrial practitioners from around the world to share their rapidly evolving knowledge and to report on new advances in Web information systems.

WISE 2006 received 183 submissions from 20 countries worldwide. From these submissions, the Program Committee selected 37 regular papers and 17 short papers, corresponding to an acceptance ratio of 20% and 9%, respectively. This volume also includes invited keynote papers, given by three leading experts at WISE 2006: M. Tamer Özsu (University of Waterloo), Katsumi Tanaka (Kyoto University) and Lizhu Zhou (Tsinghua University).

The submission and review process worked as follows. Each submission was assigned to three or four Program Committee members for review. During the discussion phase PC members and PC chairs carefully analyzed in particular papers that received divergent evaluations. Based on the review scores and the results of the discussions, the Program Chairs made the final decision.

Three workshops were held in conjunction with WISE 2006. The workshop papers are published in a separate proceedings by Springer in its *Lecture Notes in Computer Science* series (LNCS 4256).

We are grateful to the Program Committee members who helped tremendously in reviewing the large number of submissions, to Yanchun Zhang, Xuhui Li and Qing Li for their great support in the conference organization, to Yurong Chen for setting up the Web site and processing the submissions and registrations. Finally, we would like to thank Wuhan University for their support in organizing the conference.

October 2006

Karl Aberer
Zhiyong Peng
Elke Rundensteiner

Organization

Organization Committee

General Co-chairs

Yanxiang He, Wuhan University, China
M. Tamer Özsu, University of Waterloo, Canada

Program Committee Co-chairs

Karl Aberer, EPFL, Switzerland
Zhiyong Peng, Wuhan University, China
Elke Rundensteiner, WPI, USA

Workshop Co-chairs

Ling Feng, University of Twente, Netherlands
Guoren Wang, Northeastern University, China

Publicity Co-chairs

Qing Li, City University of Hong Kong, China
Xiaoyong Du, Reming University, China

Publication Co-chairs

Yanchun Zhang, Victoria University, Australia
Xuhui Li, Wuhan University, China

WISE Society Representive

Xiaohua Jia, City University of Hong Kong, China

Local Organization Committee

Shi Ying, Wuhan University, China
Youyu Gong, Wuhan University, China
Chuanhe Huang, Wuhan University, China

Program Committee

Bernd Amann, France
Periklis Andritsos, Italy
Budak Arpinar, USA
Bharat K. Bhargava, USA

Alex Borgida, USA
Wojciech Cellary, Poland
Shermann S. M. Chan, Japan
Edward Chang, USA
Enhong Chen, China
Li Chen, USA
Songting Chen, USA
Vassilis Christophides, Greece
Kajal Claypool, USA
Alfredo Cuzzocrea, Italy
Ernesto Damiani, Italy
Alex Delis, USA
Oscar Díaz, Spain
Gill Dobbie, New Zealand
David Embley, USA
Piero Fraternali, Italy
Avi Gal, Israel
Dimitrios Georgakopoulos, USA
Dina Goldin, USA
Vivekanand Gopalkrishnan, Singapore
Peter Haase, Germany
Mohand-Said Hacid, France
Wook-Shin Han, Korea
Yanbo Han, China
Christian Huemer, Austria
Mizuho Iwaihara, Japan
H.V. Jagadish, USA
Qun Jin, Japan
Kamal Karlapalem, India
Hiroyuki Kitagawa, Japan
Nick Koudas, Canada
Harumi Kuno, USA
Herman Lam, USA
Wang-Chien Lee, USA
Shijun Li, China
XueMin Lin, Australia
Tok Wang Ling, Singapore
Chengfei Liu , Australia
Sanjay Madria, USA
Murali Mani, USA
Massimo Mecella, Italy
Carlo Meghini, Italy
Xiaofeng Meng, China
Mukesh K. Mohania, India
Wolfgang Nejdl, Germany
Anne Hee Hiong Ngu, USA
Zaiqing Nie, China
Moira C. Norrie, Switzerland
Tekin Ozsoyoglu, USA
Jignesh M. Patel, USA
Alexandra Poulovassilis, UK
Cartic Ramakrishnan, USA
Michael Rys, USA
Monica Scannapieco, Italy
Klaus-Dieter Schewe, New Zealand
Heiko Schuldt, Switzerland
Mark Segal, USA
Tony Shan, USA
Timothy K. Shih, Taiwan
Steffen Staab, Germany
Gerd Stumme, Germany
Hong Su, USA
Jianwen Su, USA
Aixin Sun, Singapore
Keishi Tajima, Japan
Hideyuki Takada, Japan
Olga de Troyer, Belgium
Michalis Vazirgiannis, Greece
Jari Veijalainen, Finland
Yannis Velegrakis, USA
Andreas Wombacher, Netherlands
Yuhong Xiong, China
Jian Yang, Australia
Masatoshi Yoshikawa, Japan
Ge Yu, China
Jeffrey Yu, China
Philip Yu, USA
Donghui Zhang, USA
Aoying Zhou, China
Xiaofang Zhou, Australia

Table of Contents

Keynote Papers

Session 1: Web Search

Session 2: Web Retrieval

Session 3: Web Workflows

Session 4: Web Services

Session 5: Web Mining

Session 6: Performant Web Systems

Session 7: Web Information Systems

Session 8: Web Document Analysis

Session 9: Quality, Security and Trust

Session 10: Semantic Web and Integration

Session 11: XML Query Processing

Session 12: Multimedia and User Interface

Internet-Scale Data Distribution: Some Research Problems

M. Tamer Özsu

David R. Cheriton School of Computer Science, University of Waterloo
tozsu@uwaterloo.ca

Abstract. The increasing growths of the Internet, the Web and mobile environments have had a push-pull effect. On the one hand, these developments have increased the variety and scale of distributed applications. The data management requirements of these applications are considerably different from those applications for which most of the existing distributed data management technology was developed. On the other hand, they represent significant changes in the underlying technologies on which these systems are built. There has been considerable attention to this issue in the last decade and there are important progress on a number of fronts. However, there are remaining issues that require attention. In this talk, I will review some of the developments, some areas in which there has been progress, and highlight some of the issues that still require attention.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, p. 1, 2006.

Towards Next-Generation Search Engines and Browsers – Search Beyond Media Types and Places

Katsumi Tanaka

Graduate School of Informatics, Kyoto University
tanaka@dl.kuis.kyoto-u.ac.jp

Abstract. In this keynote talk, the author will describe concepts and technologies for next-generation search engines and browsers for searching and browsing contents beyond media types and places. Currently, digital content are represented by different media such as text, images, video etc. Also, digital content are created, stored and used on a variety of places (devices) such as independent digital archives, World Wide Web, TV HDD/DVD recorders, personal PCs, digital appliances and mobile devices. The viewing styles of these content are different. That is, WWW pages are accessed and viewed in an active manner such as a conventional Web browser (reading, scrolling and clicking interface). On the other hand, TV content are accessed and viewed in a passive manner. As for searching these "ambient multimedia contents", currently, many commercial search engines cover only WWW content and personal PC contents, called "desktop search".

First, the author describes research issues necessary for searching “ambient multimedia contents”. The main research issues are (1) cross-media search, (2) ranking methods for contents without hyperlinks, and (3) integration of search results. As for cross-media search, the author describes query-free search, complementary-information retrieval, and cross-media meta-search.

Second, the author describes ways of browsing “ambient multimedia content”. The main topics of the second part are new browsers by media conversion of digital content, concurrent and comparative browsers for multiple contents. For example, the proposed browsers have an ability to automatically convert Web content into TV content, and vice versa.

The last part of the talk is concerned with mining metadata owned by search engines and its usage for computing the "trustness" of the searched results.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, p. 2, 2006.

Building a Domain Independent Platform for Collecting Domain Specific Data from the Web

Lizhu Zhou

Department of Computer Science and Technology, Tsinghua University
dcszlz@tsinghua.edu.cn

Abstract. World Wide Web has become a major information resource for both individuals and institutions. The freedom of presenting data on the Web by HTML makes the information of same domain, such as sales of book, scientific publications etc., be present on many Web sites with diverse format. Thus to collect the data for a particular domain from the Web is not a trivial task, and how to solve the problem is becoming a trendy research area. This talk first gives an overview of this new area by categorizing the information on the Web, and indicating the difficulties in collecting domain specific data from the Web. As a solution, the talk then continues to present a stepwise methodology for collecting domain specific data from the Web, and introduce its supporting system SESQ which is a domain independent tool for building topic specific search engines for applications. The talk shows full features of SESQ by two application examples. In conclusion, the talk briefs further research directions in this new Web data processing area.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, p. 3, 2006.

A Web Search Method Based on the Temporal Relation of Query Keywords

Tomoyo Kage and Kazutoshi Sumiya

University of Hyogo
1-1-12 Shinzaike-honcho, Himeji, 670-0092, Hyogo, Japan
nd05w005@stshse.u-hyogo.ac.jp
sumiya@shse.u-hyogo.ac.jp

Abstract. As use of the Web has become more popular, searching for particular content has been refined by allowing users to enter multiple keywords as queries. However, simple combinations of multiple query keywords may not generate satisfactory search results. We therefore propose a search method which automatically combines query keywords to generate queries by extracting the relations among query keywords. This method consists of two Web search processes: one to determine the temporal relations between query keywords, and one to generate queries based on the obtained temporal relations. We discuss these two processes along with experimental results and implementation issues regarding a prototype system.

Keywords: Information Retrieval, Temporal Relation, Web Archive, Query Generation.

1 Introduction

Current Web search engines based on the relations between keywords and Web pages only consider the presence or absence of keywords. For that reason, if users input some keywords as query keywords to a Web search engine, it outputs only Web pages that include all the query keywords. For example, suppose a user inputs {*cherry blossoms*, *autumn leaves*, *trip*} to a Web search engine. Current search engines will output Web pages that include all the query keywords; i.e., pages containing information about sightseeing. However, in some cases the user is likely to want information from travel agency pages which provide tour information or from personal sites where trip topics are discussed.

In this study, the automatic generation of meaningful queries using OR bindings makes it possible to get useful Web pages. We determine the relation between query keywords on Web pages as a temporal relation to generate the queries. This temporal relation is extracted from a time series of Web pages. The time series of Web pages we use are collected from Web archives[1] .

[1] http: //www.archive.org/

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 4–15, 2006.

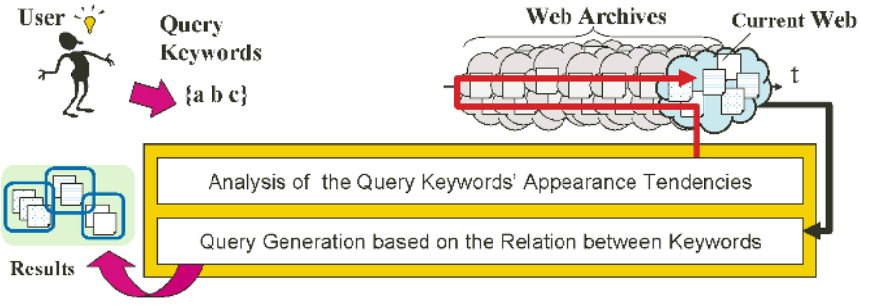


Fig. 1. Concept of Our Approach

In this paper, we propose using the temporal relations between query keywords to generate query keyword combinations (Fig. 1). We analyze the appearance tendencies of query keywords, generate queries by extracting the temporal relation, and then get Web pages which cannot be obtained by current engines.

Where the query keywords are {*cherry blossoms*, *autumn leaves*, *trip*}, the characteristic of {*cherry blossoms*, *autumn leaves*} alternately appearing in a time series of Web pages is extracted, and queries returning {R(*cherry blossoms* $\wedge$ *trip*) $\vee$ R(*autumn leaves* $\wedge$ *trip*)} are generated. That is, the system will find that there is a fairly small chance of these keywords appearing at the same time. Users can get travel agency pages including tour information through the query. Furthermore, this method is much more efficient than a method that generates all combinations joined by AND binding or OR binding. This is because only meaningful queries are generated by using the relation between query keywords.

Related work on query generation includes the multimedia retrieval of Web pages by query relaxation [8] and interval glue operations for video data retrieval [9]; these methods differ from ours, however, because their retrieval target is different. Regarding studies using time series pages, approaches specialized for the purpose of summarization [1][2][3][7] have been proposed. Furthermore, [5][6] aim at search engine ranking. They deal with temporal Web pages with the same target as our method, but the goals and methods differ from ours. Research on the temporal relation of keywords includes [10] and [4]. Temporal characteristics are extracted from the query logs in the search engine; however, we extract these from Web logs, and so their approach differs from ours. A search engine which applies clustering to search results is also available [11]. This search engine classifies subtopics for query keywords by analyzing Web page summaries, while we use the temporal features of query keywords.

The remainder of this paper is organized as follows. Section 2 describes the process to determine the temporal relation that we propose, and Section 3 describes the query generation. Section 4 looks at an experiment to test the proposed method and the design of the prototype system. In Section 5, we summarize this work.

2 Temporal Relation of Query Keywords

2.1 Definition

We define the temporal relations of query keywords that users input into a Web search engine. There may be several kinds of temporal relation between the keywords. We make connections between the relations. Three relations — co-occurring, ordered, and repeated — plus the independent relation, are all basic temporal relations. The strict co-occurring relation is a special case of co-occurring in that it is a strictly simultaneous co-occurrence, while the cyclic relation is a special case of repeated in that the time intervals are exactly the same. The temporal relations are defined as follows.

Co-occurring. The co-occurring relation is where keywords a and b both appear at the same time on time series pages. General co-occurrence means that multiple keywords appear on a Web page. However, we do not pinpoint a page because the scale is time. Consequently, if keywords a and b appear at more or less the same time, we say that they have a co-occurring relation. For example, a keyword pair {*SARS*, *corona*} has a co-occurring relation. In this case, a tight relation exists between the keywords, and a strict co-occurring relation is classified as a strong relation of this type. This is described in more detail below.

Ordered. The ordered relation is where keywords a and b are order dependent with respect to each other; i.e., one of them appears before the other in time series pages. That is, with two keywords, either "a appears before b" or "b appears before a". A concrete example of this relation is that of the keyword pair {*SBC*, *AT&T*}. In this way, a "causal" semantic relation is included and the relation expresses a topic change. We classify ordered as repeated, though, when there is a strong relation, as described in more detail below.

Repeated. The repeated relation is where keywords a and b appear in a circular pattern in chronological order; i.e., a, and b alternately appear on time series pages. For instance, the keywords pair {*disaster*, *volunteer*} or {*remodeling*, *reopening*} has this relation. A repeated relation includes the ordered relation because a keyword pair of the repeated relation has the ordered relation and appears two or more times. A cyclic relation is classified as a strong repeated relation. This is described in more detail below.

Independent. The independent relation is where keywords a and b appear independently on time series pages. That is, they appear in random order with respect to each other; for example, such a keyword pair might be {*pumpkin pie*, *recipes*}. We define the independent relation as a residual which includes relations other than the three relation types described above.

Strict Co-occurring. The strict co-occurring relation is where keywords a and b both appear on the same page from among the time series pages, and it is strongly related to the co-occurring relation described above. For example, {*Halloween*, *costume*} has such a relation.

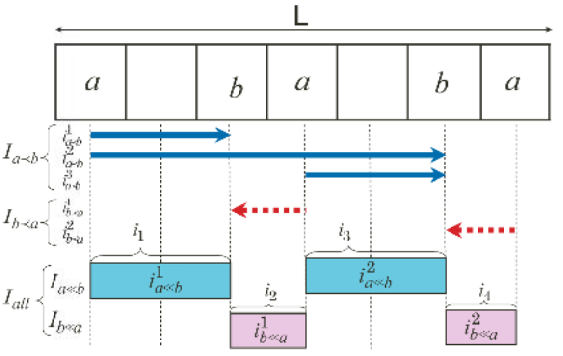


Fig. 2. Extraction of Intervals

Cyclic. The cyclic relation is where keywords a and b appear temporally and periodically, and it is strongly related to the repeated relation described above. This relation differs from the repeated one in that the keywords appear regularly. In particular, the time length of a to b is constant, as it is for b to a. That is, the time lengths of intervals a to b and b to a are approximate. For example, in the case of {*cherry blossom*, *autumn leaves*}, the time interval of spring to autumn is the same every year, as it is for autumn to spring. The same would hold for events (e.g.,{*Halloween*, *Christmas*}) which usually appear periodically.

2.2 Determination

In this study, we determine the temporal relation between the keyword pairs on each URL. We generate keyword pairs and operating intervals around a keyword pair. We then identify each pair as having either a co-occurring, ordered, or repeated relation. The determination is done in the order of co-occurring, ordered, and cyclic. The method of determination for a keyword pair a, b is described below.

First, we extract intervals $I_{a \prec b}$ and $I_{b \prec a}$ to determine the temporal relation between query keywords. $I_{a \prec b}$ is a set of $i_{a \prec b}$, and $i_{a \prec b}$ is an interval from "a page that includes a" to "a page that includes b". Correspondingly, $I_{b \prec a}$ is a set of $i_{b \prec a}$, and $i_{b \prec a}$ is an interval from "a page that includes b" to "a page that includes a". We then extract intervals $I_{a \ll b}$ and $I_{b \ll a}$. $I_{a \ll b}$ is a set of $i_{a \ll b}$, and $i_{a \ll b}$ is an interval satisfying a condition that a appears earlier than b. As before, $I_{b \ll a}$ is a set of $i_{b \ll a}$, and $i_{b \ll a}$ satisfies a condition that b appears earlier than a. The extraction process of $I_{a \ll b}$ is as follows (Fig. 2).

- **STEP 1** Extract $I_{a \prec b}$
- **STEP 2** Remove $i_{a \prec b}$ which can be regarded as having $i_{b \prec a}$

STEP 2 is necessary because $I_{a \ll b}$ is not extracted from the parts of intervals where $i_{b \prec a}$ are of $i_{a \prec b}$.

In addition, we determine the temporal relations through extracted intervals. The definition of Fig. 2 and the algorithm for determining the relations are explained in the following. We define the intervals of time-series pages as follows. L is the time length of time-series pages. $I_{a \prec b}$, $I_{b \prec a}$, $i_{a \prec b}$, $i_{b \prec a}$, $I_{a \ll b}$, $I_{b \ll a}$, $i_{a \ll b}$, and $i_{b \ll a}$ are as having been defined. The set of $I_{a \ll b}$ and $I_{b \ll a}$ is I_{all}, and $I_{all} = \{i_1, i_2, ..., i_n\}$.

```
//Determination of Co-occurring Relation
if ((cnt(less_time(I_{a≺b}))+cnt(less_time(I_{b≺a})) / (cnt(I_{a≺b})+cnt(I_{b≺a})) ≥ α)
   then //Determination of Strict Co-occurring Relation
      if (cnt(time_0(i_{a≺b})) / (cnt(I_{a≺b})+cnt(I_{b≺a})) ≥ α)
         then
            strict co-occurring
         else
            co-occurring
      fi
   else goto Determination of Ordered Relation
fi
```

Fig. 3. Determination of Co-occurring Relation

Co-occurring Relation. If the total of co-occurring keywords among all keywords that appear is large, we determine that the relation is co-occurring. We set a threshold value which can be regarded as the same time, and if the total within which both keywords appear is more than the threshold value, the relation is co-occurring. The algorithm is shown below (Fig. 3).

We define the function as follows. $time(i)$ returns the time length of i. $cnt(I)$ returns the total of intervals included in I. $less\_time(I)$ returns intervals (included in I) whose time length is less than a threshold value. $time\_0(i)$ returns the total of i whose time length is zero. α is a threshold value.

We use $i_{a<b}$ and $i_{b<a}$, which can be regarded as keyword-appearing intervals, to determine co-occurring. The appearing intervals of keyword pair a and b are calculated by $time(i_{a<b})$, and the total of co-occurring keywords is a summation of $i_{a<b}$ and $i_{b<a}$ where the value is less than the threshold. We extract the ratio of the total of co-occurring to the total of appearing, and a relation is regarded as co-occurring if the value is larger than threshold α. When a relation is found to be co-occurring, we determine whether it is strict co-occurring[2]. If the ratio of total of $time(i_{a<b}) = 0$ to the total of appearing is larger than threshold α, the relation is strict co-occurring.

Ordered Relation. If the relation is not determined as co-occurring, ordered relation determination is then done. If the order of a keyword pair appearing on time series pages is unique, the relation is determined as ordered. The method to determine an ordered relation is shown in Fig. 4 and proceeds as follows.

[2] We consider this to mean that both keywords appear on the same pages.

```
//Determination of Ordered Relation
if
   ( (Σ_{i=1}^{n} (time(i_i))) / L  ≥ β )
     then
       ordered a < b
     else
       if ( (Σ_{i=1}^{n} (time(i^i_{b≪a}))) / (Σ_{i=1}^{l} (time(i^i_{a≪b}))) ≥ γ )
         then
           ordered b < a
         else
           goto Determination of Repeated Relation
       fi
   fi
else
   independent
fi
```

Fig. 4. Determination of Ordered Relation

First, we calculate the ratio I_{all} to L (time length of all time-series pages) to determine whether a temporal relation is based on the keyword pair over the whole interval on the time series pages. The keyword pair is considered to depend on the order if the intervals keeping the order occupy the whole interval. If the value is less than β(threshold), the relation is independent, and the process is finished. Otherwise, the process is continued.

We next calculate the ratio of the summation of $I_{a \ll b}$ to L and the ratio of the summation of $I_{b \ll a}$. If the bias of two values is larger than γ(threshold), the relation is regarded as ordered.

Repeated Relation. When the relation is not determined as ordered, determination of whether it is a repeated relation occurs. If a keyword pair appears alternately, the keywords pair is repeated. The method of determining a repeated relation is shown in Fig. 5 and described below.

$reverse(i_j, i_{j+1})$ returns "true" or "false". If $((i_j == i_{a \ll b}) \wedge (i_{j+1} == i_{b \ll a}))$ or $((i_j == i_{b \ll a}) \wedge (i_{j+1} == i_{a \ll b}))$, $reverse(i_j, i_{j+1})$ returns "true" or "false". Otherwise, it returns "false". $variance(x)$ returns the variance value of x. β, θ, and δ are threshold values.

We determine whether the intervals keeping order $i_{a \ll b}$ and $i_{b \ll a}$ appear alternately. If the words of a keyword pair appear alternately, the intervals are added together. If the ratio of the summation to the time length of I_{all} exceeds θ, the relation is repeated; if not, the relation is independent.

When a relation is found to be repeated, we then determine whether it is cyclic. In the cyclic case, the period is constant so we calculate the variance of i_i (regarded as the time cycle). If the variance values of $i_{a \ll b}$ and $i_{a \ll b}$ are both less than δ, the relation is cyclic[3].

[3] The variance is a measure of the spread of the data.

```
//Determination of Repeated Relation
foreach (i_j)
   if (reverse(i_j, i_{j+1}) == true)
      s = Add(time(i_j));
      j++;
   fi
end
if ( s / Σ_{i=1}^{n} (time(i_i)) ≥ θ
   then //Determination of Cyclic Relation
      if ((variance(i_{a≪b})) < δ) ∧ (variance(i_{b≪a})) < δ)
         then
            cyclic
         else
            repeated
      fi
   else independent
fi
```

Fig. 5. Determination of Repeated Relation

3 Query Generation Based on Temporal Relation

3.1 Basic Query Generation

The Web retrieval method we propose generates queries based on the temporal relations between query keywords. We extract semantic relations (i.e., topic dependence and modification relation) from the temporal relations. And then we relate semantic relations to logical expression and generate queries. The relation is shown in Table 1. In this section, we describe the query generation by AND binding and OR binding. The basic queries for a keyword pair a, b based on the temporal relations are shown in Table 2.

The query $Q_{a|b}$ means the generated queries are $(a \vee b)$ and $(a \wedge b)$ because the keywords are considered to have topic dependence or a modification relation. $Q_{a\|b}$ means at least one word of the keyword pair must be found; because the keywords always co-occur, the search is not always required for both keywords. The keywords have strong topic dependence, but they do not have a modification relation.

$Q_{a<b}$ means the keyword pair is joined by AND and the second keyword b is joined by OR. The former is based on a view that the ordered relation includes semantic relations like a causal relationship. Alternatively, through the latter a user can use this query to retrieve Web pages which currently include only b but also included a in the past.

$Q_{a\odot b}$ means the generated queries are $(a \wedge b)$, (a), and (b). The first one is because the keywords are considered to have topic dependence along with the ordered relation. Another query is needed because the components of keyword pairs do not usually appear together on the same page. $Q_{a\circledcirc b}$ means the generated queries are (a) and (b) as in the case of two queries for a repeated relation.

Table 1. Relation of Logical Expression and Semantic Relation

Logical Expression	Semantic Relation
$R(a \wedge b)$	modification, commonness
$R(a \vee b)$	topic dependence
$R(a) \cup R(b)$	topic independence

Table 2. Queries Generated from Query Keywords

Temporal Relation	Result for Generated Query
$Q_{a\|b}$ (co-occurring)	$R(a \vee b) \cup R(a \wedge b)$
$Q_{a\|\|b}$ (strictly co-occurring)	$R(a \vee b)$
$Q_{a<b}$ (ordered)	$R(a \wedge b) \cup R(b)$
$Q_{a\odot b}$ (repeated)	$R(a \wedge b) \cup (R(a) \cup R(b))$
$Q_{a\circledcirc b}$ (cyclic)	$R(a) \cup R(b)$
$Q_{\{a,b\}}$ (independent)	$R(a \wedge b)$

However, it often happens that the keywords have an independent topic, and $(a \wedge b)$ is not generated.

$Q_{\{a,b\}}$ means the keyword pair is joined by AND. The keywords do not make any sense without co-occurrence because they share no connection. That is, the keywords have a modifier relation with each other.

3.2 Query Combination

We generate queries by combining the basic queries. When the total number of query keywords is more than three, queries are generated taking into consideration the relationships between the temporal relations.

Example 1. In the case of keywords a, b, and c, for the case discussed in the preceding section, $\{a, b\}$ is co-occurring, and $\{b, c\}$ and $\{c, a\}$ are independent. From the relationship between the temporal relations, we extract that c is independent and a pair of a and b is co-occurring. The result of a query combination of $R(Q_{a|b})$, $R(Q_{\{b,c\}})$, and $R(Q_{\{c,a\}})$ is $R(c \wedge (a \vee b)) \cup R(c \wedge a \wedge b)$. For example, if a is *SARS*, b is *corona*, and c is *virus*, the result for the generated query is as follows:

$R(virus \wedge (SARS \vee corona)) \cup R(virus \wedge SARS \wedge corona)$

Example 2. If $\{a, b\}$ is repeated, $\{b, c\}$ and $\{c, a\}$ are independent. The result of a query combination of $R(Q_{a\odot b})$, $R(Q_{\{b,c\}})$, and $R(Q_{\{c,a\}})$ is $R(a \wedge b \wedge c) \cup (R(c \wedge a) \cup R(c \wedge b))$. For example, if a is *cherry blossoms*, b is *autumn leaves*, and c is *castle*, the result for the generated queries is as follows:

$R(cherry\ blossoms \wedge autumn\ leaves \wedge castle) \cup$
$R(castle \wedge cherry\ blossoms\ ) \cup R(castle \wedge autumn\ leaves)$

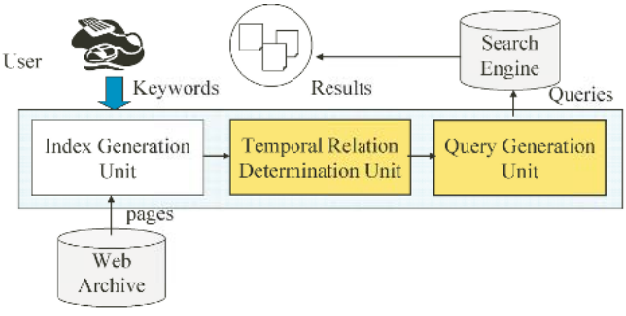


Fig. 6. System Architecture

4 Prototype System and Evaluation

4.1 Prototype System

Fig. 6 shows an overview of our prototype system. The system is organized into three units as follows:

1. Index Generation Unit
   This unit collects a time series of Web pages and generates an index. The pages are collected by following links from the Internet Archive (http://web.archive.org/web/*/a targeted URL). The collection area is two links deep within the same site to take into consideration changing URLs.
   The index generation unit indexes the words and records the times at which they were collected. The time dates are extracted from numerical values in the Internet Archive (http://web.archive.org/web/14 figures/targeted URL), and then are added to the indexes.
2. Temporal Relation Determination Unit
   This unit extracts the time intervals by referring to the generated index and determines the temporal relation for each keyword pair. The targeted sites are the top N URLs that include all query keywords.
3. Query Generation Unit
   This unit generates queries based on the temporal relation and performs retrieval using them. The targeted relations are the top M relations of the relations determined for N URLs.

4.2 Evaluation

We evaluated the extraction of the temporal relations and Web searching based on our method (Table 2). In this experiment, we manually picked out Web sites because at present we cannot generate a complete index of all Web pages stored in Web archives. We applied temporal relations that accounted for more than 30 percent of the determined relations to the query generation. In this regard, we assumed that $R(a \vee b)$ was equal to $R(a) \cup R(b)$ because the former

Table 3. Precision and Recall [Experiment 1]

	Query Result	Precision	Recall	F score
original	R(SARS ∧ corona ∧ virus)	0.95	0.35	0.51
proposed	R(SARS ∧ corona ∧ virus) ∪ R(virus ∧ SARS) ∪ R(virus ∧ corona)	0.90	0.84	0.87
other	R(virus ∧ SARS) ∪ R(virus ∧ corona)	0.87	0.60	0.71
other	R(virus ∧ corona) ∪ R(SARS ∧ corona)	0.92	0.63	0.75
other	R(SARS ∧ corona) ∪ R(virus ∧ SARS)	0.90	0.63	0.74
other	R(SARS ∧ corona)	0.95	0.35	0.51
other	R(virus ∧ corona)	0.90	0.33	0.48
other	R(virus ∧ SARS)	0.85	0.31	0.54

was heavily dependent on the search engine. We evaluated the search results by comparing the precision, recall, and F score (2pr/(p+r)) of results when using generated queries to the results when using other combinations of query keywords ("original" is the current query). In addition, the total number of answer pages used to compute recall was the summation of answers obtained from queries generated for all combinations of keywords. We used Google as the search engine and examined the top 20 URLs.

Experiment 1. We first used {*SARS, corona, virus*} as the query keywords and the URLs of websites concerning infectious diseases that included the keywords as the target Web pages. The keyword pair {*SARS, corona*} had a co-occurring relation [4]. In contrast, the keywords {*SARS, virus*} and {*corona, virus*} were independent because these words are widely found on medical and public health sites.

As shown in Table 3, the F score of the proposed method was higher than that of any other query. We also found that two queries of the proposed method output specific Web pages. When one query, (*virus* ∧ *SARS*), of this method was used, the answer set included news announcing general contents. These were pages on which gSARS virush was written rather than gcorona virush. When another query, (*virus* ∧ *corona*), of this method was used, the answer set included information about the corona virus of pets (dogs and cats). At the same time, ten out of twenty Web pages included both (*SARS* ∧ *corona* ∧ *virus*) and (*SARS* ∧ *corona*). Hence, proposed query — i.e., (*SARS* ∧ *corona* ∧ *virus*) — to return correct Web pages and (*virus* ∧ *SARS*) and (*virus* ∧ *corona*) to return specific Web pages were appropriate queries.

Experiment 2. Next, we used {*cherry blossoms*C*autumn leaves*C*castle*} as query keywords and the URLs of pages including these keywords as the target Web pages. The keyword pair {*cherry blossoms*, *autumn leaves*} had a repeated relation. We had expected that the relation would be a cyclic relation, but it depended on Web page conditions in the Web archive. In contrast, the keyword pairs {*cherry blossoms*, *castle*} and {*autumn leaves*, *castle*} were independent.

[4] Note that this experiment was run when SARS first appeared and that SARS was later found to be caused by a corona virus.

Table 4. Precision and Recall [Experiment 2]

	Query Result	Precision	Recall	F score
original	R(cherry blossoms ∧ autumn leaves ∧ castle)	0.75	0.31	0.44
proposed	R(cherry blossoms ∧ autumn leaves ∧ castle) ∪ R(castle ∧ cherry blossoms) ∪ R(castle ∧ autumn leaves)	0.78	0.90	0.83
other	R(castle ∧ cherry blossoms) ∪ R(castle ∧ autumn leaves)	0.80	0.67	0.73
other	R(castle ∧ autumn leaves) ∪ R(cherry blossoms ∧ autumn leaves)	0.57	0.56	0.56
other	R(cherry blossoms ∧ autumn leaves) ∪ R(castle ∧ cherry blossoms)	0.69	0.43	0.53
other	R(castle ∧ cherry blossoms)	0.90	0.38	0.53
other	R(castle ∧ autumn leaves)	0.70	0.29	0.41
other	R(cherry blossoms ∧ autumn leaves)	0.50	0.21	0.29

As shown in Table 4, the F score of the proposed method was higher than that of any other query. We also found that two queries (*castle* ∧ *cherry blossoms*) and (*castle* ∧ *autumn leaves*) of the proposed method output specific Web pages.

When part of proposed query (*castle* ∧ *cherry blossoms*) of this method was used, the answer set included information about castles with a view of cherry blossoms. When another part of proposed query (*castle* ∧ *autumn leaves*) of this method was used, the answer set included information about castles with a view of autumn leaves. The query (*cherry blossoms* ∧ *autumn leaves*), which used unrelated query keywords, returned specific Web pages that were not appropriate for the query; e.g., pages including information on the ancient Japanese story.

Hence, the proposed query (*cherry blossoms* ∧ *autumn leaves* ∧ *castle*) returned correct Web pages and (*castle* ∧ *cherry blossoms*) and (*castle* ∧ *autumn leaves*) returned specific Web pages that were appropriate for the query.

5 Conclusion and Future Work

The Web search method we propose improves the effective recall rate by using query generation based on the temporal relations between keywords. We have found that search results from this method differ depending on the combination of query keywords. Our future work will be aimed at finding ways to generate a greater variety of queries, improve the algorithm, support more complex temporal relations, and solve the identification problem that arises when time series pages are the same.

Acknowledgments. This research was supported in part by a Grant-in-Aid for Scientific Research (B)(2) 16300028 from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

1. Jatowt, A., and Ishizuka, M.: Summarization of Dynamic Content in Web Collections. In Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy (2004) 245–254

2. Jatowt, A., and Ishizuka, M.: Temporal Web Page Summarization. In Proceedings of the 5th International Conference on Web Information Systems Engineering, Brisbane, Australia (2004) 303–312
3. Jatowt, A., and Ishizuka, M.: Web page summarization using dynamic content. In Proceedings of the 13th International World Wide Web Conference (poster) (2004) 344–345
4. Chien, S., and Immorlica, N.: Semantic Similarity Between Search Engine Queries Using Temporal Correlation. In Proceedings of the 14th International Conference on World Wide Web (WWW2005) (2005) 2–11
5. Jatowt, A., Kawai, Y., and Tanaka, K.: Temporal Ranking of Search Engine Results. In Proceedings of the Fifth International Conference on Web Information Systems Engineering (WISE2005) (2005) 43–52
6. Jatowt, A., Kawai, Y., and Tanaka, K.: Using Web Archive for Improving Search Engine Results. In Proceedings of the Eighth Asia Pacific Web Conference (APWeb2006) (2006) 893–898
7. Jatowt, A., Khoo, KB, and Ishizuka, M.: Change Summarization in Web Collections. In Proceedings of the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Ottawa, Canada (2004) 653?662
8. Kuwabara, A., and Tanaka, K. RelaxImage: A Cross-Media Meta-Search Engine for Searching Images from Web Based on Query Relaxation. In Proceedings of the 21st International Conference on Data Engineering (ICDE2005) (2005)
9. Pradhan, S., Tajima, K., and Tanaka, K.: A Query Model to Synthesize Answer Intervals from Indexed Video Units. IEEE Transactions on Knowledge and Data Engineering, Vol.13, No.5 (2001) 824–838
10. Vlachos, M., Meek, C., Vagena, Z., and Gunopulos, D.: Identifying Similarities, Periodicities and Bursts for Online Search Queries. In Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD2004) (2004) 131–142
11. Clusty.com: http://clusty.com/

Meta-search Based Web Resource Discovery for Object-Level Vertical Search*

Ling Lin, Gang Li, and Lizhu Zhou

Tsinghua University, Beijing 100084, PRC
linling03@mails.tsinghua.edu.cn
dcszlz@tsinghua.edu.cn

Abstract. Object-level vertical search engine has been the research focus recently where the resource collecting problem is still an open area. It is difficult to adapt the traditional link-based web crawler for this task because of the sparse linkage and data-centered webpage of the relevant resources. In this paper, we propose a meta-search based method enhanced with auxiliary crawling to address the problem caused by sparse linkage of the relevant resources. And to retrieve the data-centered webpages efficiently, domain schema is defined to describe the target resource, and representative data instances are selected for meta-search query composing. Moreover, evaluation criteria for the domain resource survey are also proposed as the guideline for query composing and auxiliary crawling, which enable the resource discovery to be automatically performed by computers. Experiment results on real-world data show that our method is effective and efficient.

1 Introduction

Searching for desired information is a primary activity on the Web. Most existing search engines(SEs) index and query web resources at webpage level. In recent years object-level vertical SEs are paid more attention, and there are several successful systems such as Yahoo!/MSN Shopping, MSRA Libra, SESQ[9]. Generally speaking, the object-level vertical SEs discover the target web resources, extract structural data objects from the acquired webpages and then provide more precise query upon these data objects[1,2]. Among the above procedures, data extraction and web database schema matching have been hot spots of researches. However, to the best of our knowledge, rather few papers have focused on the frontier task of resource discovery. In this paper, we try to propose a meta-search based search strategy for the resource collecting task of object-level vertical search and discuss the major issues involved with our method.

Before introducing our method, there are several possible solutions to be considered. Utilizing existing focused web crawlers[5,6,7] is a first candidate because they use a best-first search strategy to find the relevant webpages. However, there

* This work is supported in part by National Natural Science Foundation of China 60520130299.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 16–27, 2006.

are two reasons that they are inapplicable. Firstly, target resources of object-level vertical search often reside in the websites which have competition relationship and do not link to each other, such as E-Commerce portals. The link-based search strategy will result in low efficiency. Secondly, most current analysis algorithms for webpage content are based on VSM using term-vectors to represent the feature of texts. Such algorithms are not suitable for those data-centered webpages because usually there are no continuous or long-enough texts within them for accurate analysis. Therefore, other methods should be designed to deal with the above two challenges of object-level vertical search. A second straightforward way to perform the resource collecting task is to exploit existing web directories. However, the websites organized by web directory can be limited in the range of topics and the quantity of websites. The situation will be worse if the target domain/topic does not fit into the taxonomy or category of the web directories. Another reasonable method is to use webpage classifiers to filter out the target webpages for further data extraction[2]. However, such method requires a prior acquisition of a large amount of webpages. Although the requirement can be achieved by using width-first crawling strategy adopted in page-level general SEs, it is still a consuming task and therefore impractical for many application scenarios.

In this paper we believe that the target resource of object-level vertical search is the website whose webpages contain extractable data objects. With this idea, we propose a novel resource discovery method which only requires human effort at the primary step of the whole process to define the target domain schema and collect initial data seeds. The main innovation of our work can be presented in 3 aspects. (1)We use existing page-level general SEs to find the relevant websites by submitting queries composed of representative data instances of the predefined domain schema. (2)The major challenges facing our method are the data instance selection and query composing because they determines the quality and quantity of retrieved resources. To address this problem, a survey of domain resource property is performed to get the numerical evaluations according to our proposed metrics. (3)Moreover, an auxiliary crawling for more resources will be utilized if the results of domain survey indicate its necessity. Experiments on real-world data show promising results of our method.

2 Related Work

Although there are few researches addressing the same problem discussed in this paper, some works are related with ours by common theoretical background or similar ideas in the following three aspects.

Many researches about object level vertical SEs focus on data extraction from webpages. One important idea concluded from these works[3,4] is that the target domain of object-level vertical search can be abstracted and described as schema, which lays out the basis for our method. Our work focus on web resource discovery and is positively complementary to the existing works.

Our work shares certain similar idea with focused website crawling[5] in that

target resources are to be discovered at website level, but the purely link-based search strategy is inefficient for our task. Researches in [8,13] report that relevant resources may share a co-citation relationship or scatter in different web communities. Therefore, in [8] meta-search is used for the starting URLs supplementation in the focused crawling. Experiments in [7] also show that back-crawling to hub resources brings bursts of rapid retrieval of target webpages. In this paper, we integrate the above ideas into a more comprehensive framework to address the problem of sparse linkage among target resources.

There are many notable works on meta-search in aspects such as database selection[10] and result merging[11], etc. Our method is different in that it is based on page-level general SEs but not specific SEs[10]. And result merging is not a concernful issue here because we aim to retrieve a larger number of resources. However, some ranking models[11] can be utilized to judge the quality of retrieved resources for our domain survey. On the other hand, new problem of meta-search arises for our method in query composing as how to use the keyword based SEs for the discovery of relevant data-centered webpages.

3 The Task of Meta-search Based Resource Discovery

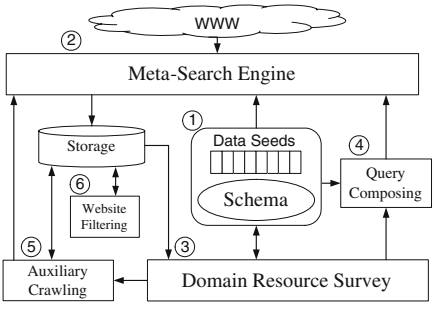


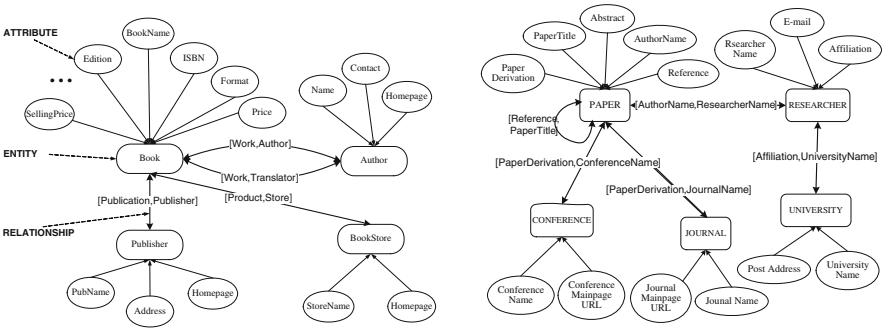
Fig. 1. System Architecture and Work Process

Based on above consideration, the task of meta-search based resource discovery are designed as the following steps. (1)Domain schema is defined and small amount of data seeds are collected manually. (2)The data seeds are used as the queries submitted to meta-search engine(MSE), (3)then the returned results are stored for domain resource survey. According to results of domain survey, (4)better data seeds are selected for later query composing, and (5)auxiliary crawling is performed to find more websites if it is necessary. (6)Website filtering use domain-relevant knowledge to filter out irrelevant websites returned from MSE. Fig. 1 shows the system architecture and work process. In the rest of this paper, we will give detailed discussion of the above subtasks from (1) to (5). The techniques of website filtering in (6) are domain-relevant and relatively independent of the above meta-search based framework, therefore the details are omitted for lack of space.

3.1 Target Domain Description

Domain representation is defined manually in the primary step of domain resource collecting task. Although a same data object may have different formats and displays at different websites, they can be integrated and abstracted into a common schema. The domain schema we use is defined as ordered tuples.

Definition 1. ***Schema*** *describes the entities and the relationship between the entities, where the entity is the abstraction concept of target data objects and the relationship is defined upon two attributes of entities.*
Entity ::=
<EntityName, Attribute1[cardinality], Attribute2[cardinality]... >
Relationship ::=
<RelationshipName, role1: Entity1.Attribute i, role2: Entity2.Attribute j>



(a) "Book Shopping" Domain (b) "Academic Paper" Domain

Fig. 2. Schemas of Sample Domains

Two examples are illustrated for the "book shopping" domain in Fig. 2(a) and "academic paper" domain in Fig. 2(b) respectively. Therefore, the target resources are the websites which contain the extractable data objects of the same or compatible schemas.

3.2 Domain Resource Survey

According to domain schema, data instances are collected for domain survey. The survey is performed by querying MSE with a small amount of data seeds and analyzing the returned results. In this section we discuss the issues why data seeds should be used and consequently how they may affect the resource retrieval for different target domains.

Querying MSE with Data Seeds. Analogous to the "starting URL seeds" in link-based crawling, we use ***data seeds*** to denote the data instances of domain schema submitted to MSE. Our MSE is composed of keyword based page-level general SEs such as Google, Yahoo! etc. Therefore, data seeds, instead of other keywords such as a generalized description of user interests, are used for the queries because they can better represent the content of the target resource, thus achieve higher efficiency and accuracy.

Two Examples. (1)Take book shopping for example. Search the query word *"book shopping"* and the first 10 results returned by SE include only 6 book shopping

websites. On the contrary, use [*"Harry Potter and the Order of the Phoenix" "J. K. Rowling" Hardcover 1232*] as the query, which represents a book instance with its attribute values of *BookName, Author, Format* and *Pages*, all the first 10 results are target websites which sell this book. (2)In the academic paper domain, use an instance of paper with the values of its title and first author, e.g. [*"Efficient Mining of Constrained Correlated Sets" "Laks V. S. Lakshmanan"*], the first 10 returned results include websites like *portal of ACM, Citeseer, infomatik, sigmod, IEEE society*, etc. On the contrary, use *"academic paper of computer science"* as query and none of the above websites are included in the first 10 results.[1]

The reason why using data seeds as query can achieve better results can be explained from the following two aspects.

1. From SE's aspect, the index is built directly upon the content of webpages.
2. From data seeds' aspect, the words within the context of entity attributes' values represent explicit semantic of search target. For example, a single word *"apple"* can have multiple meanings while in the context *"apple 1.8GHz, 512M 80G"*, it explicitly stands for a computer brand.

Domain Resource Evaluation. Before discussing the evaluation metrics, some notations are to be introduced. Suppose MSE are composed of h SEs. For each query, only top k returned results of each SE are taken for the sake of accuracy and efficiency. Only the domains name of the returned URLs are recorded although they usually point to webpages. The set of distinct relevant websites returned by the s^{th} SE *w.r.t.* the d^{th} data seed is denoted as R_{sd}. The result whether a website is hit by a query is expressed by a 0-1 matrix, denoted as HIT matrix. Suppose n data seeds are submitted to the MSE and totally m websites are retrieved ($m = |\bigcup_{s=1}^{h}\bigcup_{d=1}^{n} R_{sd}|$), then the HIT matrix is m rows and n columns. The value of element hit_{ij} is defined as

$$hit_{ij} = \begin{cases} 1 & d_j \in ED_{w_i} \\ 0 & otherwise \end{cases}$$

in which d_j is the j^{th} data seed used for corresponding query, w_i is the i^{th} retrieved website, and ED_{w_i} is the **Effective Data (ED)** of w_i. Here, ED_{w_i} denotes the part of data on w_i indexed by SEs and returned in top results. In most cases, a SE could hardly cover **All Data (AD)** one website has. We assume that (1)The ratio of ED/AD increases as the importance of website increases to the same SE; (2)Data objects inside the ED of their website are regarded more popular than those outside. Therefore, if w_i is retrieved by query composed by d_j, we have $d_j \in ED_{w_i} \subseteq AD_{w_i}$. Note that we do not concern about the precise value of ED/AD but do exploit the phenomenon that important websites and data objects are more highly indexed by SEs. Based on these notations, three metrics are propose to evaluate domain resources.

[1] Searches in the above two examples were performed on Google on April 19, 2006.

– **Data Popularity**

One important criteria for choosing data seeds is their capability of retrieving as many websites as possible. Good data seeds are usually the popular data instances covered by more websites. Hence the data popularity is defined according to the HIT matrix as the following way.

Definition 2. ***Data-seed Popularity of the*** j^{th} ***data seed*** (DSP_j) *is defined as*

$$DSP_j = \frac{\sum_{i=1}^{m} hit_{ij}}{k \times h} \tag{1}$$

Definition 3. ***Data Popularity*** (DP) *of target domain is defined as*

$$DP = \frac{\sum_{j=1}^{n} DSP_j}{n} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} hit_{ij}}{n \times k \times h} \tag{2}$$

DSP_j is defined as the percentage of websites hit by the j^{th} data seed $(\sum_{i=1}^{m} hit_{ij})$ out of its maximum capability of retrieving websites$(k \times h)$, and DP indicates the average DSP performance of target domain. The value domain of DSP_j and DP is $[0, 1]$. Here the maximum retrieval capability of a data seed is calculated as an ideal situation when there is no overlapping between the top k results of the h SEs, which is also the upper bound of m when $n = 1$. Note that we use $k \times h$ instead of m, because DP values may be biased by the group performance of an improperly-chosen set of data seeds with a low m but can reflect their real performance faithfully with a standard $k \times h$. DP and DSP can be influenced by and consequently reflect the following aspects of target domain.

1. The ability of data sharing among resources. For domains like *academic paper, book/electronics shopping* where the same data object can be covered by multiple websites, DP values are higher. In these domains DSP will be higher for particular data instances such as best-sellers, important papers frequently cited, etc. They can be good candidates for data seeds. On the contrary, domains like *animal/pet sale* and *job finding* have lower DP because data objects tend to be unique, thus good data seeds are difficult to find.
2. The update frequency of data objects. The update frequency of data objects can vary significantly for different domains. Using out-of-date data seeds will result in low DSP and reduce the efficiency of resource discovery, especially for the domains in which data are not archived and frequently updated.

– **Data-Centrality of Domain Resource**

Since one of the ultimate purposes of object-level vertical resource discovery is to collect as many extractable data objects as possible, the distribution of target data over different websites is another important issue to be concerned about. The data distribution can be reflected by the amount of ED for retrieved websites. Note that we only consider the data which can be indexed by the

SEs and ignore the situation of Hidden Web because they are harder to be retrieved or estimated. One reasonable and straightforward way to estimate the data centrality is to evaluate how much data seeds are covered by the retrieved websites.

Definition 4. ***Data-seed Coverage of the*** i^{th} ***website*** *(DC_i) can be defined as*

$$DC_i = \frac{\sum_{j=1}^{n} hit_{ij}}{n} \tag{3}$$

Definition 5. ***Data Centrality*** *(DC) is defined as the highest DC_i of all retrieved websites*

$$DC = MAX_{i=1}^{m}\{DC_i\} \tag{4}$$

The value domain of DC_i and DC is also [0,1]. High DC indicates that in the domain there exist resources where a majority of the target data are centralized, it is the ideal situation and resource collecting should be oriented towards these resources. Otherwise, the domain is a data-distributed one and a larger number of resources are required to retrieve enough data.

One note worth mention is that according to our work, in the domains with higher DC for most data objects, the DSP values of different data seeds seems more stable. This may indicate that for data-centralized domain, the result of meta-search based resource discovery is less sensitive to the selected data seeds.

– **Resource Authority Distribution**

The value of DC_i evaluates only one aspect of the resource quality and it is determined by the ED indexed by SEs. Since different SEs usually adopt different strategies to index and rank the websites, to be more fair, the evaluation by MSE should also be considered. Democratic selection model is often applied for meta-search rank fusion, where each SE is regarded as a voter and each resource as a candidate[11]. Consequently, a resource with more votes can be viewed as an authoritative one. We borrow the idea here to define the authority resource in the following way.

Definition 6. *Given a set of data seeds, a resource is regarded as* ***Authority*** *if and only if it is covered by at least l (**authority degree**) SEs, where $2 \leq l \leq h$.*

In reality, SEs index different parts of the web and the extent of their overlapping and disjointedness may vary a lot in different domains. The distribution of authority can be estimated by the percentage of authority resources.

Definition 7. *The* ***Authority Distribution (AD)*** *of target domain is*

$$AD = \frac{\sum_{c=1}^{C_h^l} \frac{|\bigcap_{s\in S_c} \bigcup_{d=1}^{n} R_{sd}|}{|\bigcup_{s\in S_c} \bigcup_{d=1}^{n} R_{sd}|}}{C_h^l} \tag{5}$$

where S_c denotes the c^{th} set of distinct combination of l SEs out of all h SEs.

For a domain with higher AD, there are larger amount of authority resources and these websites should be retrieved in higher priority. On the other hand, a domain with lower AD posts harder work to resource collection because more websites should be found to perform reliable and fair collecting of data.

Note that an *authority* website does not guarantee a high DC_i and vice versa. The reason is that DC and AD are defined from different aspects and affected by websites' strategy of cooperation with SEs. An authority may be covered by multiple SEs while the data objects within the website are not intensively indexed.

Deciding Data Seeds Selection and Auxiliary Crawling. After the domain resource survey, data seeds selection can be decided according to the values of DP and DC. For domains with significant varying values of DP and DC over different data seeds, the efficiency of meta-search based resource discovery is more sensitive to the selected data seeds. Therefore, data seeds with higher DSP should be chosen.

However, as mentioned before that in domains where data objects tend to be unique, good data seeds are hard to find. Therefore, other query composing method can be used as supplementation.(1)Using attribute names of target entity instead of the attribute values, such as *"price"* for *"$5.9"*.(2)Using <EntityName> according to the domain schema instead of the value of entity's name for the domains whose data objects tend to be unique, such as *"horse"* instead of *"Big Tim"* which is the name of a horse. These methods are based on the observation that in the webpages containing target data objects, attribute names and entity name are often displayed around the attribute values.

Besides, the values of DC and AD may decide whether an auxiliary crawling for more resources is necessary. Domains with lower DC and AD values are data-distributed and authority-distributed. The directly retrieved websites from MSE may be limited by the selected data seeds, which means more relevant resources may be missed. Therefore, an auxiliary crawling can be performed to pursue a higher resource recall.

3.3 Auxiliary Backward-Forward Website Crawling

Websites retrieved from MSE are influenced by the the ratio of ED/AD especially in the domains with immense instance space but low AD and DC values. Relevant resources may be missed by directly querying MSE. Fortunately, many relevant resources are organized in various hub pages by manual or automatic effects, which can be exploited with auxiliary crawling to address the above problem.

Bipartite Graph Model and Auxiliary Crawling. Given a query, the relating *authority* and *hub* web resources tend to form a **bipartite** subgraph of the WWW, which can be described as $G(V, E)$ and $V = V_a \bigcup V_h$ where the

vertices V_a denotes authorities and V_h denotes hubs. In this paper, our bipartite model satisfies the following constraints. (1)V_a is a website, and V_h is a webpage. (2)Each edge E denotes a inter-site link while intra-site links are ignored. The process of auxiliary backward-forward website crawling includes the following steps. (1)Directly retrieved resources from querying MSE is used as the **root set** R. (2)Backlinks of R are acquired using SEs for hub identification, namely **backward crawling** ($V_a \rightarrow V_h$). (3)Hubs are identified and **Sibling-links** are extracted and added to the **base set** B as candidate authorities for further website filtering, namely **forward crawling** ($V_h \rightarrow V_a$). Sibling-link denotes the link in a hub page which points to the co-cited authorities.

Hub Page Identification and Sibling-Link Extraction. The major problem of auxiliary crawling is hub identification and sibling-link extraction. To retrieve more authorities, we allow hubs to be identified with more flexible style, which can be categorized into the following types.

1. Strict-formatted regular hub page. All sibling website links share the same semantic and identical structure.
2. Regular style hub page with ignorable variance, including (1)variances caused by HTML script writing errors, such as an unwanted
; (2)variances of added element, such as small labels attached around special links, e.g. "Hot"; (3)variances of layout style on the link element itself, such as an emphasized anchor, etc.
3. Free style hub pages which are not designed to be hubs originally but also provide lists of relevant links, such as a favorite list from a personal homepage, a paragraph of text embedded with the discussed websites' links, etc.

The algorithm starts from the location of root link element in the possible hub page, and then search the HTML tree in a bottom-up way to discover possible sibling-links. In order to pursue a higher recall of various hubs, the algorithm allows the repeated sibling-link pattern to be discovered with ignorable variances of type 2, and use bounded recursive pattern comparison to deal with more complex sibling-link display style of type 3. More detailed algorithm description is omitted here for lack of space.

4 Experimental Results

The proposed algorithms are implemented in Java 1.5 platform and the experiments are performed on an AMD Sempron(TM) 1.81GHz processor with 1G RAM running Windows XP Professional Edition. All the data are reported according to the experiments performed from April 3 to April 17, 2006.

The proposed MSE is composed of 4 SEs including AltaVista, Google, MSN and Yahoo. The target domains in test are online book shopping and academic paper of computer science. The domain schemas are defined as Fig. 2 in Section 3.1. For the data seeds of book shopping domain, 100 best sellers over 17

bookstores from Yahoo! Directory are collected as "Popular Book", and another 100 random book instances are collected for compare as "Random Book". For paper domain, the data seeds are collected with the help of SESQ system[9]. 100 papers published during year 1971˜2004 are selected randomly as "Random Paper", and another 100 papers published during year 2003˜2004 with relatively high references are selected as "New Paper".

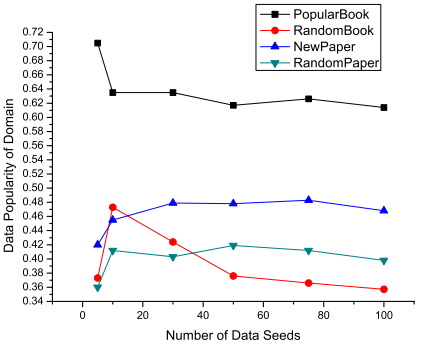


Fig. 3. *DP* of Book and Paper Domain

Firstly, the domain evaluation and query composing are tested using the initial data instances according to domain schema as discussed in Section 3.2. For each query, top 10 results are considered for every SE. Fig. 3 shows the *DP* values of book and paper domain with the 4 sets of data seeds mentioned above. It can be seen that "Popular Book" has a much higher score on the ability of retrieving websites than the "Random Book", which is consistent with our expectation. For the paper domain, it's interesting that the *DP* value of "Random Paper" does not decrease much from the value of "New Paper", which can be explained in the following experiments. Generally speaking, *DP* decreases as the number of data seeds increases. Considering our definition of *DP* in Section 3, this phenomenon indicates that the efficiency of retrieving new resource websites by adding more data seeds is decreasing as the size of data seeds increases. Another interesting thing is that within the interval of 5-10 data seeds, the *DP* values of "Random Book", "Random Paper" and "New Paper" increase slightly, which may reveal that for data seeds of lower quality, increasing the number of data seeds can make a compensation for their low *DP* within a limited extension.

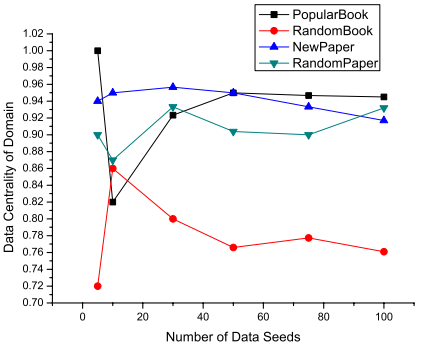


Fig. 4. *DC* of Book and Paper Domain

Fig. 4 shows the DC value of book domain and paper domain. The property of data distribution is quite different. Data seeds in paper domain are highly covered by retrieved websites, and again the difference between two sets of data seeds is small. On the contrary, for book domain the data centrality varies a lot between "Popular Book" and "Random Book". This explains partially why the *DP* value of paper domain is much more stable than book domain as the quantity and quality of data seeds change.

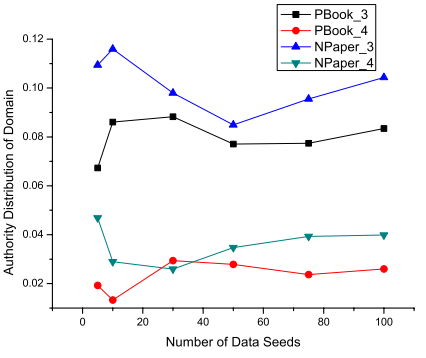


Fig. 5. *AD* of Book and Paper Domain

The *AD* evaluation is performed on the domains with the above queries and the authority degree $l = \{3, 4\}$. Lines in Fig. 5 are labelled as "<domain>_ l". The number of authority websites decreases as l increases. The results indicate an authority-distributed property of book domain, which reflects the fact that E-Commerce websites present competition relationship and different SEs index and rank the websites in different ways. On the contrary, in academic domain there exist common authority websites accepted by different SEs, which makes them reliable sources of target data objects.

From the domain survey we find that paper domain is more centralized in the distribution of authority and data, while in book domain the quality and quantity of websites returned directly from MSE is more sensitive to different data seeds. Therefore, in later resource collecting "Popular Book" is chosen and auxiliary crawling is performed on book domain. The experiments are shown in Table 1. The *RootSet* is constructed by the websites with authority degree $l \geq 3$. The *Backlinks* are acquired using SEs with the keyword "link:<url>". *Hubs* are identified with a sibling-link pattern repeated no less than 5 times, sibling-links in *BaseSet* are checked by website filtering to find *RelevantSites*. For comparison, websites from top 100 results for each SE is collected as MSE_Sites. The relevant sites not contained in MSE_Sites are regarded as *NewSites*.

Table 1. Experiment Results of Auxiliary Crawling on Book Domain

DataSeeds	RootSet	Backlinks	Hubs	BaseSet	RelevantSites	NewSites	MSE_Sites
5	7	144	48	1774	143	122	193
10	13	275	113	6291	316	266	391
30	30	594	272	18376	789	674	722

Auxiliary crawling with data-seeds>30 are not examined for it costs too much time and network burden. It can be seen that the number of *RelevantSites* is noteworthy compared with the MSE_Sites with a majority of *NewSites*, which proves the effectiveness of auxiliary crawling. Auxiliary crawling yields various relevant resources, such as publishers' websites, even the hidden websites where publicly accessible "browse" is not offered. However, current precision of relevant websites out of *BaseSet* is lower because of the hub identification algorithm in which recall takes precedence over precision. One solution is to utilize more sophisticated features for hub identification, using not only HTML structure but also domain-relevant knowledge to process the texts in the hub pages.

5 Conclusion

In this paper we present a meta-search based web resource discovery method for object-level vertical search. Our main contribution includes the following points. Firstly, we use domain schema and data seeds for the target websites retrieval, which provides a good description for the data objects contained in these websites. Secondly, we propose a numerical evaluation method for the domain resource survey, which enables the resource discovery work to be done automatically. Thirdly, a backward-forward crawling is used to improve the recall effectively even when the number of data seeds is small. Experimental results on real-world data show that our method is reasonable and effective.

The future work of our research is to develop more sophisticated and robust algorithms for hub identification and website filtering. Another interesting work is to try automatic data extraction from the retrieved webpages with the help of domain schema and the querying data instances.

References

1. Z. Nie, Y. Zhang, J.R. Wen, W.Y. Ma: Object-Level Ranking: Bringing Order to Web Objects. In 14^{th} International World Wide Web Conference, Chiba, 2005.
2. J.R. Wen: Object-Level Vertical Search. In $2^n d$ NICT China-Japan Joint Research Forum on Digital Content and Web Computing, Beijing, 2006.
3. X. Meng, H. Lu, H. Wang, M. Gu: SG-WRAP: A Schema-Guided Wrapper Generator. In 18^{th} International Conference on Data Engineering, San Jose, 2002.
4. Zhao Li, Wee Keong Ng, Aixin Sun: Web data extraction based on structural similarity. Knowledge and Information Systems, 8(4): 438-461 (2005).
5. M. Ester, H.P. Kriegel, M. Schubert: Accurate and Efficient Crawling for Relevant Websites. In 30^{th} International Conference on Very Large Data Bases, Toronto, 2004.
6. S. Chakrabarti, M. Berg, B. Dom: Focused Crawling: a new Approach to Topic-Specific Web Resource Discovery. In 8^{th} International World Wide Web Conference. Toronto, 1999.
7. M. Diligenti, F. Coetzee, S. Lawrence, C.L. Giles, M. Gori: Focused Crawling Using Context Graphs. In 26^{th} International Conference on Very Large Data Bases. Cairo, 2000.
8. J. Qin, Y. Zhou, M. Chau: Building Domain-Specific Web Collections for Scientific Digital Libraries: a Meta-Search Enhanced Focused Crawling Method. In 4^{th} ACM/IEEE Joint Conference on Digital Libraries, Tucson, 2004.
9. Q. Guo, L. Zhou, H. Guo, J. Zhang: SESQ: A Novel System for Building Domain Specific Web Search Engines. In 8^{th} Asia Pacific Web Conference, Harbin, 2006.
10. W. Meng, Z. Wu, C. Yu, H. Li: A Highly Scalable and Effective Method for Metasearch. ACM Transactions on Information Systems. 19(3):310-335 (2001).
11. J.A. Aslam, M. Montague: Models for Metasearch. In 24^{th} SIGIR, New Orleans, 2001
12. J.M. Kleinberg: Authoritative Sources in a Hyperlinked Environment. Journal of the ACM. 46(5):604-632 (1999).
13. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins: Trawling the Web for Emerging Cyber-Communities. In 8^{th} International World Wide Web Conference, Toronto, 1999.

PreCN: Preprocessing Candidate Networks for Efficient Keyword Search over Databases

Jun Zhang[1,2,3], Zhaohui Peng[1,2], Shan Wang[1,2], and Huijing Nie[1,2]

[1] School of Information, Renmin University of China, Beijing 100872, P. R. China
{zhangjun11,pengch,swang,hjnie}@ruc.edu.cn
[2] Key Laboratory of Data Engineering and Knowledge Engineering(Renmin University of China), MOE, Beijing 100872, P.R. China
[3] Computer Science and Technology College, Dalian Maritime University, Dalian 116026, P.R. China

Abstract. Keyword Search Over Relational Databases(KSORD) has attracted much research interest since casual users or Web users can use the techniques to easily access databases through free-form keyword queries, just like searching the Web. However, it is a critical issue that how to improve the performance of KSORD systems. In this paper, we focus on the performance improvement of schema-graph-based online KSORD systems and propose a novel Preprocessing Candidate Network(PreCN) approach to support efficient keyword search over relational databases. Based on a given database schema, PreCN reduces CN generation time by preprocessing the maximum Tuple Sets Graph(G_{ts}) to generate CNs in advance and to store them in the database. When a user query comes, its CNs will be quickly retrieved from the database instead of being temporarily generated through a breadth-first traversal of its G_{ts}. Extensive experiments show that the approach PreCN is efficient and effective.

1 Introduction

Without knowing the database schema or writing SQL queries, casual users or Web users can use Keyword Search Over Relational Databases(KSORD) techniques to access databases in a fashion similar to using search engines to search the Web[1]. In fact, the amount of information stored in Deep Web is 400 or 500 times larger than that in the visible Web[2]. If database systems support keyword search, publishing or searching a database is expected to be simpler and easier in the web, and the deep web problem can be alleviated[1].

KSORD systems may be classified into two types according to their query processing mechanism. One is offline systems,which retrieve results for a keyword query from an intermediate representation generated by "crawling" the database in advance, such as EKSO[3]. The other is online systems, which convert a keyword query into many SQL queries and retrieve the database itself, such as SEEKER[4], DISCOVER[6], IR-Style [7] and BANKS[9,10]. The offline KSORD Systems execute queries relatively efficiently, but they can't query the

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 28–39, 2006.

up-to-date data in time, and also need a long preprocessing time to generate the intermediate representation and large physical space to store it. However, the online KSORD systems can retrieve the latest data from databases, but their execution may be inefficient because those converted SQL queries usually contain many join operators. Furthermore, online KSORD systems can be classified into two categories[1] according to their data model, data-graph-based systems like BANKS[9,10] and schema-graph-based systems like SEEKER[4], DISCOVER[6], IR-Style[7] and DBXplorer[8].

We focus on how to improve the efficiency of schema-graph-based online KSORD(SO-KSORD) systems(Fig. 1), which can be measured by user query response time(T_{res}) mainly composed of three parts: $T_{res} = T_{ts} + T_{cn} + T_{sql}$, where T_{ts} denotes the time to create Tuple Sets(TSs), T_{cn} denotes the time to generate Candidate Networks(CNs, see Definition 6)[6,7], and T_{sql} denotes the time to execute converted SQL queries from CNs.

In our experiments, T_{sql} occupies a major proportion of T_{res} varying from 68% to 91% with various queries, T_{cn} occupies a proportion varying from 8% to 31%, and T_{ts} occupies a proportion of about 1%. T_{ts} is determined by the Information Retrieval(IR) engine of RDBMS, and usually very small. T_{sql} is the most important factor determining the execution efficiency of SO-KSORD systems, and we have studied how to reduce T_{sql} in another paper[12]. It is clear that T_{cn} is also an important factor affecting the execution efficiency of SO-KSORD systems. In this paper, we aim to reduce T_{cn} to improve the performance of SO-KSORD systems.

Currently, for any user keyword query, the existing SO-KSORD systems generate CNs temporarily through a breadth-first traversal of Tuple Sets Graph(G_{ts}, see Definition 5) and execute all the generated CNs. However, it can take much time to generate CNs for a user keyword query when the database schema becomes complicated or there are many keywords in a user query.

By our observations, in SEEKER[4] and IR-Style[7], if the same G_{ts} is generated for two queries with the same query Keywords Number(KeywNum) and Maximum allowed CN size(MaxCNsize), the same set of CNs will also be generated for them. Furthermore, for a given database schema and its G_s, there are certain G_{ts} patterns for all keyword queries in the database. So, it is possible to preprocess G_{ts} patterns to generate CNs in advance for a user keyword query.

Therefore, we propose a novel Preprocessing Candidate Network(PreCN) approach to support efficient keyword search over relational databases. The goal of PreCN is to reduce the CN generation time T_{cn} for a user keyword query. PreCN preprocesses the maximum G_{ts} based on a given G_s through a breadth-first traversal of the G_{ts}, generating all CNs under the limitation of Maximum allowed Keywords Number(MaxKeywNum) and MaxCNsize in advance and storing those CNs in the database. When a user query comes, its CNs will be quickly retrieved from the database instead of being temporarily generated through a breadth-first traversal of its G_{ts}. Our extensive experiments show that our approach is efficient and effective.

The rest of this paper is organized as follows. Section 2 reviews existing work and analyzes their limitations and drawbacks. Section 3 introduces several basic definitions. Section 4 describes our novel approach in detail. Our extensive experiments are presented in Section 5, and we conclude with summary and future work in section 6.

2 Related Work

DBXplorer[8] exploits the RDBMS schema to find out "join trees" which have potential answers to a user query. DISCOVER[6] also uses schema graph information to enumerate candidate networks. The two systems execute all CNs and return all results to users. IR-Style[7] extends DISCOVER to perform relevance-ranked and top-k keyword queries, and it reduces T_{sql} in T_{res}, compared with DISCOVER. SEEKER[4] also produces top-k results. SEEKER extends keyword query language to search metadata and non-text columns(e.g. numeric columns and date columns) besides text columns in relational databases, and adopts a new scoring formula to compute the similarity between a user query and the query data.

A drawback of these existing SO-KSORD systems is that they generate CNs temporarily through a breadth-first traversal of G_{ts} for any user query. Thus, they can take much time to generate CNs and perform inefficiently when the database schema becomes complicated or there are many keywords in a query.

As for AND semantic keyword queries in DISCOVER[6], a new preprocessing technology[5] is proposed to generate tuple sets joining tree patterns in advance through a breadth-first traversal of G_s and to store them in the database. When a user keyword query comes, proper tuple sets joining tree patterns are retrieved from the database and evaluated according to the specific tuple sets created temporarily for the query, thus CNs are generated for the query.

Our preprocessing technology is different from the above approach. As for OR semantic keyword queries in IR-Style[7] and SEEKER[4], we find out G_{ts} patterns as user keyword query patterns for a given G_s, and preprocess the maximum G_{ts} pattern to generate all CNs under the limitation of MaxCNsize and MaxKeywNum through a breadth-first traversal of the maximum G_{ts} and to store them in the database. When a user query comes, its proper CNs are directly retrieved from the database. So, our approach is simpler and more efficient than that in [5].

Offline systems(such as EKSO[3]) usually preprocess the data to generate an intermediate representation for the database. However, what our PreCN approach preprocesses is the database schema information, not the data themselves. As we know, the database schema is usually stable, but the data is changeful. So, the PreCN approach not only needs less physical storage space to store the preprocessed results, but also can still make the SO-KSORD systems search the up-to-date data in the database.

3 Basic Concepts

We give several basic definitions that we will use in the following sections[6,7,8].

Definition 1. Keyword Query(Q_k). *A Q_k is a set of query keywords, denoted as $Q_k(k_1, k_2, ..., k_m)$, $k_j (1 \leq j \leq m)$ is a query keyword. Suppose Q_k is OR semantics among the query keywords.*

Definition 2. Database Schema (S_{db}). *S_{db} is denoted as $S_{db}(R, FK)$, R is the set of relations in the database schema, and FK is the set of primary key to foreign key relationships among relations in R. Furthermore, suppose $R = R_t \cup R_c$, where R_t is the set of relations which have text attributes with full-text indexes, and R_c contains the other relations in S_{db}. Take DBLP(Fig. 2) as an example, $R_t = \{Papers, Authors, Confs, YearConfs\}$, $R_c = \{Cites, Write\}$.*

Definition 3. Schema Graph (G_s). *Denoted as $G_s(V, E)$, G_s is a directed graph that captures the primary key to foreign key relationships in S_{db}. It has nodes in V for each relation in R and edges in E for each primary key to foreign key relationship in FK. Furthermore, suppose $V = V_t \cup V_c$, where V_t is the set of nodes from the relations of R_t, and V_c is the set of nodes from the relations of R_c. V_t and V_c are called relation nodes.*

Definition 4. Tuple Set (TS). *By utilizing the full-text search functionality of RDBMS, all TSs are created from corresponding relations in R_t of S_{db} for a user query Q_k. Only those non-empty result sets are left, called as non-free Tuple Sets(nfTSs). Each nfTS inherits the corresponding relation's primary key and foreign key relationships.*

Definition 5. Tuple Set Graph (G_{ts}). *Denoted as $G_{ts}(V, E)$, G_{ts} is created for a user query Q_k, which is also a directed graph with all nodes and edges in $G_s(V, E)$. In addition, it also has nodes for each nfTS and edges for primary key to foreign key relationships like $nfTS_i \rightarrow nfTS_j$, $nfTS_i \rightarrow r_j$, or $r_j \rightarrow nfTS_i$ ($r_j \in R$). Furthermore, suppose $V = V_t \cup V_c \cup V_{ts}$, where V_t and V_c are defined in definition 3. V_{ts} is the set of nodes from nfTSs, and the nodes in V_{ts} are called tuple set nodes.*

Definition 6. Candidate Network(CN). *A CN is a joining network of tuple sets which is generated through a breadth-first traversal of G_{ts}, and satisfies the following four conditions(see DISCOVER[6], IR-Style[7]). CNs also can be considered as join expressions to be used to create Joining Network of Tuples(JNTs) which are potential answers to a keyword query Q_k. Those nodes with zero out-degree are leaf nodes from nfTSs, which have occurrences of the query keywords and non-zero score for the query. On the contrary, the others are inner nodes. The size of a CN is the number of nodes in the CN. The four conditions which a CN should satisfy are as follows:*

(i) The number of nfTSs in a CN does not exceed the number of query keywords m in Q_k. This constraint guarantees that a minimum number of CNs are generated while not missing any result that contains all the keywords.

(ii) All leaf nodes should come from nfTSs. This constraint ensures CN's "minimality".

(iii) A CN doesn't contain a construct of the form $r_i \rightarrow r_j \leftarrow r_i$ *(*$r_i \in R$*,* $r_j \in R$*). This constraint ensures every JNT from a CN wouldn't contain the same tuple more than once.*

(iv) The size of a CN is not greater than the value of a given MaxCNsize.

4 PreCN Approach

4.1 System Architecture

In the PreCN approach, we divide a SO-KSORD system into two parts(see Fig. 1): search engine(Fig. 1(a)) and CN preprocessor(Fig. 1(b)).

For a given G_s, CN preprocessor creates all possible tuple sets and constructs the $max(PG_{ts})$. Then, it traverses the $max(PG_{ts})$ in a breadth-first manner to generate all CNs under the limitation of system MaxCNsize and MaxKeywNum(e.g. SysMaxCNsize = 6 and SysMaxKeywNum = 6) in advance, and stores those CNs in the database. Once CN preprocessor finishes preprocessing work, user queries will profit from this approach to reduce T_{cn}.

When a user query is submitted to the search engine of the SO-KSORD system, Tuple Set creator(TS Creator) creates tuple sets for each relation with text attributes and full-text indexes, and only those non-empty tuple sets are left. If the query's KeywNum and MaxCNsize are not greater than SysMaxKeywNum and SysMaxCNsize respectively (that means the query has been preprocessed), its CNs will be quickly retrieved from the database, otherwise, the query' CNs will be temporarily generated through a breadth-first traversal of its G_{ts}. Finally, CN executor can run a top-k algorithm to get top-k results for this query.

In the following subsections, we discuss our approach in detail.

4.2 CN Preprocessing in CN Preprocessor

By our observations, for a given S_{db} and its G_s, we have found that there are certain G_{ts} patterns for all keyword queries in the database. So, it is possible to preprocess G_{ts} patterns to generate CNs in advance for a user keyword query. At first, we give two definitions as follows:

Definition 7. Pattern of Tuple Set Graph (PG_{ts}). *For a given* $S_{db}(R_t \cup R_c, FK)$ *and its* $G_s(V, E)$*, at most t nfTSs can be created for any* Q_k*(here* $t = |R_t|$*), and at most* $2^t - 1$ *unrepeatable combinations of nfTSs can be created for all kinds of user queries. Each* G_{ts} *which is created from each combination of nfTSs is called a Pattern of Tuple Set Graph (*PG_{ts}*). Each* PG_{ts} *can also be considered as a kind of query pattern, that means two queries are the same query pattern if the same* G_{ts} *is created for them.*

Definition 8. Maximum Pattern of Tuple Set Graph($max(PG_{ts})$). *For a given* $S_{db}(R_t \cup R_c, FK)$ *and its* $G_s(V, E)$*, the* PG_{ts} *which has the most tuple sets is called the Maximum Pattern of Tuple Set Graph (*$max(PG_{ts})$*).*

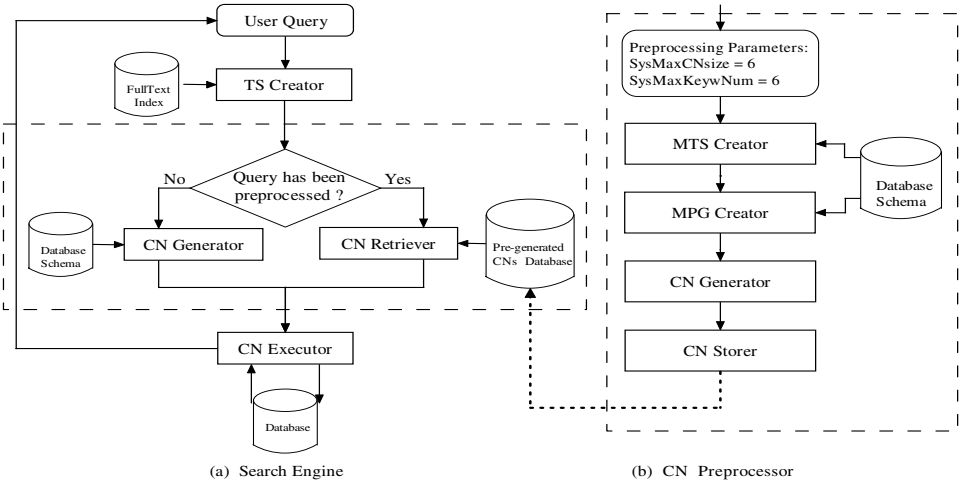


Fig. 1. Improved architecture for SO-KSORD system

Example 1. *Take DBLP as an example.* S_{db} *=*$(R_t \cup R_c, FK)$*, where* R_t *=* $\{$*Papers, Authors, Confs, YearConfs*$\}$*,* R_c *=* $\{$*Cites, Write*$\}$*.* G_s *is shown in Fig. 2. All combinations of nfTSs related to* PG_{ts} *are shown in table 1. The* $max(PG_{ts})$ *is created from the 15-th pattern which has the most TSs (see Fig. 3).*

A straight-forward method of generating CNs in advance is to preprocess all PG_{ts} for a given S_{db} and its G_s. However, the problem with this method is that it will take too much time for preprocessing all PG_{ts}. A better method is to preprocess only the $max(PG_{ts})$. The following theorem and lemmas show that the latter method is reasonable.

Let $nfTS(Q_k)$ denote the set of nfTSs created for Q_k, $nfTS(CN_i)$ denote the set of nfTSs used in CN_i, $|X|$ denote the number of elements in this set X, $sizeof(CN_i)$ denote the size of CN_i, $G_{ts}(Q_k)$ denote the G_{ts} created for Q_k, and $CN(Q_k, KeywNum, MaxCNsize)$ denote the set of CNs generated for Q_k under the limitation of KeywNum and MaxCNsize.

Theorem 1. *For a given* $S_{db}(R_t \cup R_c, FK)$ *and* G_s*,* $CN(Q_{k1}, KeywNum_1, MaxCNsize_1) \supseteq CN(Q_{k2}, KeywNum_2, MaxCNsize_2)$*, if*
(i) $nfTS(Q_{k1}) \supseteq nfTS(Q_{k2})$*,*
(ii) $KeywNum_1 \geq KeywNum_2$*, and*
(iii) $MaxCNsize_1 \geq MaxCNsize_2$*.*

Proof: Here, suppose both $nfTS_i$ and $nfTS_j$ are the same only if they are created from the same relation in R_t and have the same tuple set name, even though they have different tuples. According to Definition 5 and the condition (i), $G_{ts}(Q_{k1}) \supseteq G_{ts}(Q_{k2})$ holds. Suppose $CN_i \in CN(Q_{k2}, KeywNum_2, MaxCNsize_2)$. Because CN_i is generated through a breadth-first traversal of $G_{ts}(Q_{k2})$, CN_i can also be generated through a breadth-first traversal of

Table 1. All Combinations of nfTSs Related to PG_{ts} for DBLP Schema

PG_{ts}#	Combinations of nfTSs	nfTS Number
1	$\{TS\_Authors\}$	1
2	$\{TS\_Confs\}$	1
3	$\{TS\_YearConfs\}$	1
4	$\{TS\_Papers\}$	1
5	$\{TS\_Authors, TS\_Confs\}$	2
6	$\{TS\_Authors, TS\_YearConfs\}$	2
7	$\{TS\_Confs, TS\_YearConfs\}$	2
8	$\{TS\_Authors, TS\_Papers\}$	2
9	$\{TS\_Confs, TS\_Papers\}$	2
10	$\{TS\_YearConfs, TS\_Papers\}$	2
11	$\{TS\_Author, TS\_Confs, TS\_YearConfs\}$	3
12	$\{TS\_Author, TS\_Confs, TS\_Papers\}$	3
13	$\{TS\_Author, TS\_YearConfs, TS\_Papers\}$	3
14	$\{TS\_Confs, TS\_YearConfs, TS\_Papers\}$	3
15	$\{TS\_Author, TS\_Confs, TS\_YearConfs, TS\_Papers\}$	4

$G_{ts}(Q_{k1})$ (see CN generation algorithm in [6,7]). According to the CN definition(see Definition 6), $|nfTS(CN_i)| \leq KeywNum_2$ and $|sizeof(CN_i)| \leq MaxCNsize_2$ hold. According to the conditions (ii) and (iii), $|nfTS(CN_i)| \leq KeywNum_1$ and $|sizeof(CN_i)| \leq MaxCNsize_1$ hold. As a result, $CN_i \in CN(Q_{k1}, KeywNum_1, MaxCNsize_1)$ holds according to Definition 6. Then $CN(Q_{k1}, KeywNum_1, MaxCNsize_1) \supseteq CN(Q_{k2}, KeywNum_2, MaxCNsize_2)$.

The two following lemmas can be educed from Theorem 1. Their proofs are omitted due to lack of space.

Lemma 1. *For a given S_{db} and G_s, under the same limitation of SysMaxCNsize(System Maximum allowed CN size) and SysMaxKeywNum(System Maximum allowed Keywords Number), the set of CNs generated from $max(PG_{ts})$ (denoted as $CN(max(PG_{ts}), SysMaxKeywNum, SysMaxCNsize)$) contains all CNs generated from the other patterns of tuple set graphs.*

Lemma 2. *For a given S_{db} and G_s, $CN(Q_k, KeywNum, MaxCNsize)$ can be retrieved from $CN(max(PG_{ts}), SysMaxKeywNum, SysMaxCNsize)$, if*
(i) $KeywNum \leq SysMaxKeywNum$, and
(ii) $MaxCNsize \leq SysMaxCNsize$.

Lemma 1 shows that only $max(PG_{ts})$ needs to be preprocessed instead of all G_{ts} patterns. If all G_{ts} patterns are preprocessed, not only more time is needed to do preprocessing work but also more physical storage space is needed to store CNs and their related information. As a result, more time is also needed to retrieve CNs for a user query from the stored database.

Lemma 2 shows that our PreCN approach is effective, because the set of CNs for a user query can be retrieved directly from the stored database under the

limitation of SysMaxCNsize and SysMaxKeywNum, instead of being temporarily generated from the query's G_{ts}.

Preprocessing Capacity. According to Definition 6, the CNs generated for Q_k are determined by three factors: G_{ts} created for Q_k, KeywNum and MaxCNsize. We define the system preprocessing capacity as the number of user queries whose CNs can be retrieved directly from the database. Also, we define two system parameters: SysMaxCNsize and SysMaxKeywNum. Since the $max(PG_{ts})$ is determined by a given database schema S_{db} and its G_s, SysMaxCNsize and SysMaxKeywNum should be great enough in order to benefit as many as possible user queries. Generally, SysMaxCNsize should increase as SysMaxKeywNum increases, and also, the former should not be less than the latter. However, with the growth of the two system parameters, both the number of generated CNs and the time to preprocess the $max(PG_{ts})$ will dramatically increase. Specially, when SysMaxCNsize and SysMaxKeywNum reach a certain value, CN preprocessor could take too long time to finish preprocessing work due to the limitation of the performance of the computer used in the experiments.

It is known that Web searchers generally use queries of about 2 terms[11]. In our experiments, we found that the top 100 results for two-keywords queries distributed in those CNs whose sizes varied from 1 to 5. For various data sets, SysMaxCNsize and SysMaxCNsize can be set to different values. In our experimental DBLP data set, it is practical that the two parameters are set to 6.

CN Preprocessing Algorithm(CNPA). Figure 1(b) describes the process of CN preprocessor. SysMaxCNsize and SysMaxKeywNum are the two input parameters of CN preprocessor. For a given $S_{db}(R_t \cup R_c, FK)$ and G_s, firstly, MTS creator creates all possible empty tuple sets. There are at most t such tuple sets (t = $|R_t|$). Secondly, MPG creator constructs $max(PG_{ts})$ according to the result from MTS creator. Thirdly, CN generator generates all CNs through a breadth-first traversal of $max(PG_{ts})$ under the limitation of SysMaxCNsize and SysMaxKeywNum. Finally, CN storer stores generated CNs into the database. We describe CN Preprocessing Algorithm(CNPA) formally as Algorithm 1.

Algorithm 1. CN Preprocessing Algorithm

Input: $S_{db}(R_t \cup R_c, FK)$, G_s, SysMaxCNsize, SysMaxKeywNum
Output: $CN(max(PG_{ts}))$: the set of CNs for $max(PG_{ts})$
Begin

```
1: for each r_i in R_t do
2:    Create an empty tuple set named TS_r_i;
3: end for
4: Construct max(PG_ts);
5: //Here the CN generating algorithm is the same with [6,7]
6: Generate CN(max(PG_ts)) through a breadth-first traversal of max(PG_ts);
7: Store CN(max(PG_ts)) into the database;
8: Return CN(max(PG_ts));
```

End.

The performance of CNPA is determined by three factors: the complexity of G_s, SysMaxCNsize and SysMaxKeywNum. The more complex the given G_s is, the more time CNPA will take to do preprocessing work. With the growth of SysMaxCNsize or SysMaxKeywNum, the time taken by CNPA will increase dramatically (see Fig. 4). The number of CNs generated by CNPA is also determined by the above three factors(see Fig. 5).

4.3 CN Generating in Search Engine

According to Lemma 2, if a user query's KeywNum and MaxCNsize are not greater than SysMaxKeywNum and SysMaxCNsize respectively, its CNs will be quickly retrieved from the database, otherwise, the query' CNs will be temporarily generated through a breadth-first traversal of its G_{ts}. The new CN generating algorithm(CNGA) in the search engine of the SO-KSORD system is shown in Algorithm 2. This algorithm performs efficiently even though hundreds or thousands of CNs are stored in the database, because CN retriever can be converted into an efficient SQL query to retrieve CNs from the database. Our experiments show the efficiency of CNGA. In fact, if a query needs to generate CNs by using the original CN generator instead of CN retriever in CNGA, it will take too much time to generate its CNs.

Algorithm 2. New CN Generation Algorithm

Input: $Q_k(K_1, K_2, ..., K_m)$, m is the number of query keywords,
$nfTS(Q_k)$: the set of nfTSs created for Q_k, MaxCNsize
Output: $CN(Q_k)$: the set of CNs for Q_k
Begin

```
1: if not (m ≤ SysKeywNum and MaxCNsize ≤ SysMaxCNsize) then
2:    //original CN Generator
3:    Generate CN(Q_k) directly from G_ts(Q_k);
4:    Return CN(Q_k);
5: end if
6: //CN Retriever
7: for each CN_i stored in the database do
8:    if Distinct(nfTS(CN_i)) ⊆ nfTS(Q_k) then
9:       if sizeof(CN_i) ≤ MaxCNsize then
10:         if |nfTS(CN_i)| ≤ m) then
11:            Add CN_i to CN(Q_k);
12:         end if
13:      end if
14:   end if
15: end for
16: Return CN(Q_k);
```

End.

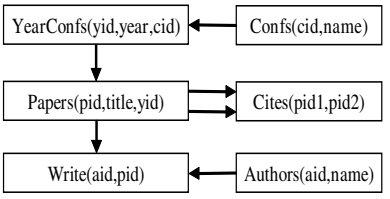


Fig. 2. The DBLP G_s

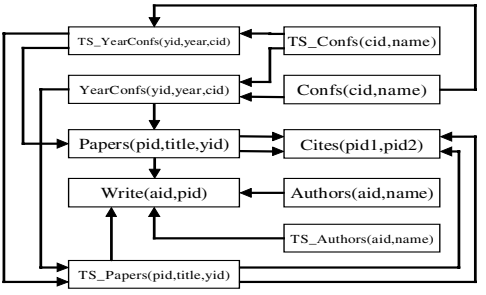


Fig. 3. The DBLP $max(PG_{ts})$

5 Experimental Evaluation

5.1 Experimental Environment

We ran our experiments using the Oracle 9i RDBMS on the platform of Windows XP SP2 and IBM NoteBook computer with Intel Pentium 1.86GHZ CPU and 1.0GB of RAM memory. Based on SEEKER[4] system, we implemented our PreCN approach in Java and connected to the RDBMS through JDBC. The IR engine is the Oracle9i Text extension.

We used the DBLP data set for our experiments, which we decomposed into relations according to the schema[7] shown in Fig. 2. YearConfs is an instance of a conference in a particular year with about 6,000 records, Confs lists about 1,000 conferences, Papers lists about 590,000 papers, Cites is a relation with about 8,000 records that describes each paper pid2 cited by a paper pid1, and Authors lists about 200,000 authors, and Write lists about 590,000 records that describes each paper written by the authors. All the data are extracted from the DBLP xml file downloaded from ftp://ftp.informatik.uni-trier.de/pub/users/Ley/bib/.

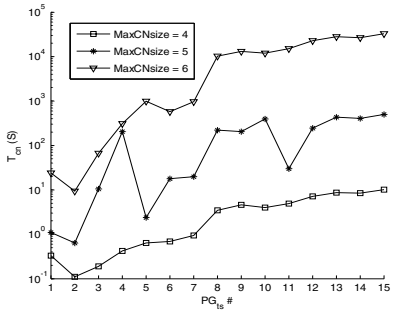


Fig. 4. Preprocessing Efficiency. Let MaxKeywNum = MaxCNsize, Vary PG_{ts} #.

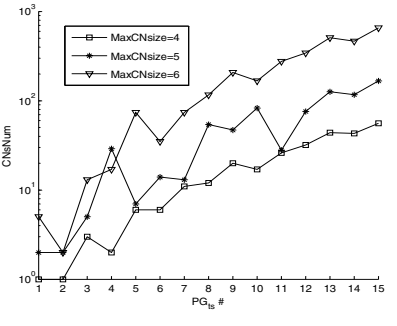


Fig. 5. Generated CN Number. Let MaxKeywNum = MaxCNsize, Vary PG_{ts} #.

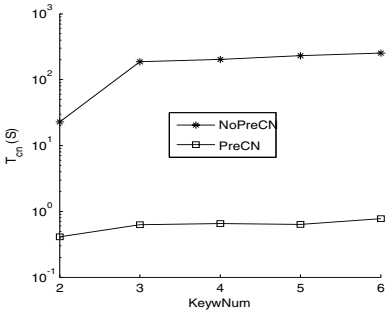


Fig. 6. Effectiveness. Fix MaxCNsize = 5, Vary KeywNum.

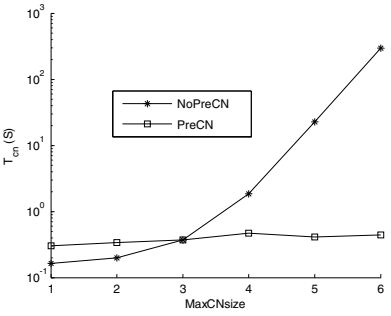


Fig. 7. Effectiveness. Fix KeywNum = 2, Vary MaxCNsize.

5.2 Results and Discussion

Preprocessing Efficiency. Figure 4 shows the efficiency of various PG_{ts} preprocessing. Figure 5 shows the number of generated CNs from various PG_{ts}. In these experiments, MaxKeywNum was always set to the same value as MaxCNsize. From the two figures, we can get to know that the more complex the PG_{ts} is, the more time it takes to do CN preprocessing work and the more CNs it produces. Also, with the growing value of MaxCNsize, the time to preprocess PG_{ts} increases dramatically. We can see, when both MaxCNsize and MaxKeywNum were set to 6, the 15-th PG_{ts} ($max(PG_{ts})$) took about 9 hours to finish preprocessing work and produced 657 CNs. If all the G_{ts} patterns are preprocessed, they will take about 38 hours.

Effectiveness of Preprocessing. Each point value is on average over 30 queries in the two following figures. Figure 6 shows the effectiveness of various KeywNum values. As KeywNum increases, PreCN performs stably and efficiently, and orders of magnitude faster than original CN generator which generates the set of CNs for a user query through a breadth-first traversal of its G_{ts}. In our experiments, T_{cn} with PreCN approach is less than 1 second on average under the limitation of MaxCNsize set to 5. On the contrary, that will be about 200 seconds without PreCN approach when KeywNum is greater than 2. Even though KeywNum is 2, the time to generate CNs without PreCN approach reaches 23 seconds.

Figure 7 shows the effectiveness of various MaxCNsize. As MaxCNsize increases, T_{cn} without PreCN approach will increase dramatically. The reason is that the number of generated CNs will increase dramatically with the growth of MaxCNsize. On the contrary, T_{cn} with PreCN approach will keep stable and less than 1 second, because CN retriever performs stably and is slightly affected by the number of retrieved results. When MaxCNsize is small than 3, T_{cn} without PreCN approach is less than that with PreCN. However, MaxCNsize usually should be set to 5 or a greater value, because top 100 results for a two-keywords query usually distribute in the CNs whose sizes vary from 1 to 5. So, the PreCN approach is necessary and effective to reduce T_{cn} for a user query.

6 Conclusion and Future Work

We have presented a novel approach PreCN to support efficient top-k keyword queries in SO-KSORD systems. Under the limitation of SysMaxCNsize and SysKeywNum, PreCN preprocesses the $max(PG_{ts})$ to produce CNs in advance so that the set of CNs for user queries will be quickly retrieved from the database. This method depends on the database schema which is usually stable, therefore, it can also keep the advantage of SO-KSORD systems which can retrieve up-to-date data in time. Our experiments show that this approach is efficient and effective. In the future work, we seek to increase SysMaxCNsize and SysKeywNum to enlarge the preprocessing capacity of PreCN, and test our PreCN approach in other data set.

Acknowledgement

This work is supported by the National Natural Science Foundation of China under Grant No.60473069 and 60496325, and the project of China Grid(No.CNGI-04-15-7A).

References

1. S. Wang, K. Zhang. Searching Databases with Keywords. Journal of Computer Science and Technology, 20(1). 2005:55-62
2. M.K.Bergman. The deep web: Surfacing hidden value. White Paper, Bright Planet, 2000.
3. S. Qi, W. Jennifer. Indexing Relational Database Content Offline for Efficient Keyword-Based Search. Proceeding of IDEAS, 2005:297-306.
4. J. Wen, S. Wang. SEEKER: Keyword-based Information Retrieval Over Relational Data-bases. Journal of Software. 2005
5. K. Zhang. Research on New Preprocess-ing Technology for Keyword Search in Databases. PhD thesis of Renmin University of China. 2005.
6. V. Hristidis, Y. Papakonstantinou: DISCOVER: Keyword Search in Relational Databases. VLDB, 2002:670-681.
7. V. Hristidis, L. Gravano, Y. Papakonstantinou. Efficient IR-Style Keyword Search over Relational Databases. VLDB, 2003:850-861.
8. S. Agrawal, S. Chaudhuri, and G. Das. DBXplorer:A System for keyword Search over Relational Databases.ICDE, 2002:5-16.
9. G. Bhalotia, A. Hulgeri, C. Nakhe et al. Keyword Searching and Browsing in Databases using BANKS.ICDE, 2002:431-440.
10. V. Kacholia, S. Pandit, S. Chakrabarti et al. Bidirectional Expansion For Keyword Search on Graph Databases. VLDB 2005:505-516.
11. B. Jansen, A. Spink, T. Saracevic. Real life, real users, and real needs: A study and analysis of user queries on the web. Information Processing and Management. 36(2), 2000:207-227.
12. J. Zhang, Z. Peng, S. Wang, H. Nie. CLASCN: Candidate Network Selection Supporting Efficient Top-k Keyword Queries over Databases. Technical Report, School of Information,Renmin University of China, 2006.

Searching Coordinate Terms with Their Context from the Web

Hiroaki Ohshima, Satoshi Oyama, and Katsumi Tanaka

Department of Social Informatics, Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo, Kyoto 606-8501, Japan
Tel.: +81-75-753-5385, Fax: +81-75-753-4957
{ohshima, oyama, tanaka}@dl.kuis.kyoto-u.ac.jp

Abstract. We propose a method for searching coordinate terms using a traditional Web search engine. "Coordinate terms" are terms which have the same hypernym. There are several research methods that acquire coordinate terms, but they need parsed corpora or a lot of computation time. Our system does not need any preprocessing and can rapidly acquire coordinate terms for any query term. It uses a conventional Web search engine to do two searches where queries are generated by connecting the user's query term with a conjunction "*OR*". It also obtains background context shared by the query term and each returned coordinate term.

1 Introduction

Any term can be described by other terms. For any term, we can think of many other terms related to it. The relationships between terms are very important, so many dictionaries describe the relationships, and a lot of research has been done on automatically acquiring terms related to other terms. One of the relationships between terms is coordination, and our goal is to search coordinate terms from a query term. Our definition of "coordinate terms" is the following: when term X is a coordinate term of term Y, there exists a term Z that is the hypernym of both X and Y. For example, *tomato*, *potato*, and *carrot* are coordinate terms because they all have the same hypernym *vegetable*.

We propose a method for searching coordinate terms using a traditional Web search engine. When a user submits a term as a query, two Web search queries are made by connecting the user's query term and the conjunction "*OR*" before and after the term.

A lot of other research has been done on acquiring coordinate terms automatically. Most of the methods need large text or Web page corpora, and some of them need a lot of computation for mutual information or cooccurrence of terms. For example, Church [1] proposed a method of finding semantically related terms by using mutual information. Lin [2] proposed a method of making clusters of similar words. Similarities between words are calculated and similar words clusters are generated. Modification relation is used to calculate the similarities, so a large corpus with modification relation is necessary. Shinzato [3]

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 40–47, 2006.

proposed a method of acquiring coordinate terms using HTML structure that needs many HTML documents. Zoubin [4] proposed a system called Bayesian Sets that acquires coordinate terms. It finds clusters of terms based on Bayesian inference. The algorithm is very simple and fast but requires a data set such as the Grolier encyclopedia or EachMovie. There is also some research on acquiring hypernym/hyponym relationships[5][6][7] that propose systems that can also be used to acquire coordinate terms. These systems also need large corpora or a lot of computation.

Our method analyzes only text in the titles and the snippets from Web search results. That is, our method does not need any prepared corpora, any preprocessing, or any other external information. It has two advantages. One is speed. Our method can acquire coordinate terms very quickly. If the Internet connection is fast enough, a search will finish in about three or four seconds. Most of the time is spent querying the Web search engine. Analyzing text takes little time because the text requiring analysis is very small. The other advantage of our method is its large range of potential adaptation. As long as there are a certain amount of documents about the user's query term in the Web, our system can find coordinate terms in any field. Since the Web keeps developing, our method, which can find coordinate terms based on "current" Web content, has a large advantage and great potential.

A difference between our method and others is the purpose. Other methods mainly aim to construct a large dictionary by using large corpora. We, on the other hand, are attempting "quick mining" of the Web. Web search engines store a great deal of information in their databases. We think that a few simple contrivances enable us to obtain a lot of meaningful knowledge from these databases. Our method analyzes only the data in a Web search engine, obtaining the data on demand. This is a very small effort, but if it works well, users can use such information very casually. For example, query formulation or query modification for a Web search often needs to obtain unknown terms from somewhere. Our method is very simple and very fast, so it will be used in such a situation.

Google Sets [1] is an interesting tool for acquiring coordinate terms. When we submit a small subset of the items, Google Sets returns the whole set. The details of the Google Sets algorithm are not disclosed, but it seems that a large-scale clustering algorithm is applied to collected Web pages where many clusters of terms have already been generated. This can be described as automatic coordinate term dictionary construction. In any case, we can get coordinate terms from Google Sets, so we compared some Google Sets search results with our method's search results.

We also obtain context shared by a query term and each returned term. When a user's query is *Bordeaux*, our system can acquire coordinate terms such as *Burgundy*, *Paris*, *cabernet*, and *blue*. *Bordeaux* has several meanings, such as a city in France, a region famous for wine, wine made in Bordeaux, and a color like red wine. Based on the difference in the meanings, there are different contexts shared by *Bordeaux* and its coordinate terms. For example, the context shared

[1] http://labs.google.com/sets

by *Bordeaux* and *Burgundy* is that both of them are famous wine regions. The context will be also characterized by other terms related to wine. The context shared by *bordeaux* and *blue* is both of them are colors. If such context is shown with the results of our coordinate term search, it helps users understand why they are regarded as coordinate terms, so we try to find the context shared by a query term and each returned coordinate term.

2 Searching Coordinate Terms

Our goal is to retrieve coordinate terms for query terms. Our approach is based on two assumptions. One is that the conjunction "*OR*" can connect two coordinate terms. The other is that when two terms "A" and "B" are coordinate, there must exist both "A *or* B" pattern and "B *or* A" pattern. Neither assumption is anything special. In practice, the idea behind the second assumption is also very useful in finding an appropriate word. When a proper coordinate term is a complex word, it is necessary to cut the complex word out of a sentence. We will discuss this later. Our method consists of the following four steps.

Step 1. A user submits a query.
Step 2. Two Web search queries are made.
Step 3. The Web search results are obtained and analyzed.
Step 4. Candidates for coordinate terms are scored.

User's query
First, a user submits a query term, which can be a single word, a complex word, or an abbreviation. It can be a noun, a verb, an adjective, or an adverb. Some examples of query terms are *piano*, *Swan Lake*, *beautiful*, *get married*.

Making two queries using a Web search engine
Next, two Web search queries are made by connecting a user's query term and the conjunction "*OR*" before and after the term. When the query term is *Swan Lake*, two queries for a Web search engine are **"Swan Lake or"** and **"or Swan Lake"**. For example, Google[2] allows phrase searches with quotation marks enclosing the query. Thus the quotation marks are contained in the queries.

Obtaining and analyzing the Web search results
Conventional Web search engines return listed items as search results. Each search result item usually consists of a page title, a snippet of content, and a URL. For example, Google allows a maximum of 100 result items from one search. If we gets 100 result items for each of the Web queries there are only two Web searches. Therefore, we analyze a maximum of 200 titles and snippets. The snippet shows a small amount of content from the searched Web page and usually consists of sentences that contain the user's query words. Here are two examples of snippets: one for the query *"Swan Lake or"* and the other for the query *"or Swan Lake"*.

[2] http://www.google.com/

Table 1. Terms appearing for query *Swan Lake*

candidates for coordinate terms	C_A	C_B
Sleeping	6	0
Sleeping Beauty was	1	0
Sleeping Beauty was such	1	0
Sleeping Beauty was such great	1	0
Sleeping Beauty	5	5
Beauty	0	5
or Sleeping Beauty	0	1
Juliet or Sleeping Beauty	0	1
and Juliet or Sleeping Beauty	0	1

1. That partly explains why Tchaikovsky's *Swan Lake or* **Sleeping Beauty** was such great music outshining any other ballet music. ...
2. ... just right for Coppelia or Cinderella or the first act of Giselle but not the girl of your dreams in Romeo and Juliet or **Sleeping Beauty** *or Swan Lake.*

We count the number of appearances of terms that are connected to the user's query term by the conjunction "*OR*". With *Swan Lake*, in the snippet for *"Swan Lake or"*, the words "*Sleeping Beauty was such great*" immediately follow the words "*Swan Lake or*". In this case, we count out each of the following: *Sleeping*, *Sleeping Beauty*, *Sleeping Beauty was*, *Sleeping Beauty was such*, *Sleeping Beauty was such great*. In the snippet for *"or Swan Lake"*, we count terms that appear immediately before "*or Swan Lake*". They all are candidates for coordinate terms of *Swan Lake*.

Table 1 shows the number of appearances of candidate terms, where C_A denotes the number of appearances after *OR*, and C_B denotes the number of appearances before *OR*. *Sleeping Beauty* is the only proper coordinate term of *Swan Lake* in this table. It also clearly shows that there are many appearances of both C_A and C_B for *Sleeping Beauty*, but no appearances of C_A or C_B for any other term. Thus, even if a coordinate term is a complex word, our method can find the proper cut points. This is because of the assumption that there must be a coordinate term on either side of "*OR*".

Scoring candidates for coordinate terms
After counting, each candidate coordinate term is scored. Scores are calculated by the geometrical average of C_A and C_B.

3 Experimental Result

3.1 Comparing Results with Google Sets

We show the results of some coordinate term searches. Each table shows a query term, result terms from Google Sets, and result terms from our method. Terms listed as results by our method are candidate terms with scores over 0 (that appear more than once). Terms listed as results of Google Sets are part of the whole result.

Table 2. Search results from Google Sets and our method

Query: Swan Lake		
Google Sets	Our method	
Nutcracker	Nutcracker	13.3
Sleeping Beauty	Sleeping Beauty	5.0
Petrushka	Giselle	4.5
Raymonda	Romeo and Juliet	2.0
Les Sylphides	Hamlet	2.0
West Glacier	Coppelia	1.4
Martin City	Nutcracker Suite	1.4
Marion	Tchaikovsky symphony	1.0
The Nutcracker		
Romeo Juliet		

Query: George Bush		
Google Sets	Our method	
Bill Clinton	John Kerry	15.6
Ronald Reagan	Al Gore	6.9
Al Gore	Saddam Hussein	2.4
Richard Nixon	Cheney	2.0
Jimmy Carter	Bill Clinton	2.0
Ralph Nader	Dick Cheney	1.4
Laurel Springs School	Tony Blair	1.4
Organized Religion	Dick	1.0
Abraham Lincoln	Bob Dole	1.0
George Washington	Dole	1.0

Query: China		
Google Sets	Our method	
India	India	19.1
Japan	Russia	6.3
Indonesia	Japan	5.7
Australia	Hong Kong	3.5
Fiji	Taiwan	3.2
Malaysia	Europe	1.7
Taiwan	central Asia	1.4
Korea	dinnerware	1.4
Canada		
Singapore		

Query: piano		
Google Sets	Our method	
piano	harpsichord	16.7
violin	organ	14.1
guitar	keyboard	9.2
cello	voice	8.5
organ	harp	7.5
flute	guitar	6.5
harpsichord	orchestra	5.3
Saxophone	music	3.0
Voice	ensemble	2.8
Bass	accordion	1.4

Query: wise		
Google Sets	Our method	
LM Chapters	foolish	19.1
RE SEED	prudent	6.3
Dept	waste	5.7
Waves Beams	stupid	3.5
Scott	learned	3.2
Lee	installshield	1.7
Stephens	moral	1.4
Big Stone Gap	fool	1.4
Augustus		
McCarthy		

Query: beautiful		
Google Sets	Our method	
comfortable	ugly	6.5
inspirational	handsome	5.3
affordable	interesting	1.4
custom made	nature	1.0
solid mahogany		
elegant		
fun		
smart		
charming		
dazzling silver		

Query: audacious		
Google Sets	Our method	
bodacious	fierce	5.2
brazen faced	bold	2.4
brassy	daring	2.0
bold faced	ambitious	1.7
venturous	aggressive	1.4
barefaced	outrageous	1.4
insolent	xmms	1.0
brazen	reckless	1.0
daring	brazen	1.0
venturesome	extravagant	1.0

Query: walk		
Google Sets	Our method	
trot	run	38.5
lope	bike	18.8
untitled	ride	7.4
canter	jog	3.5
run	talk	2.4
bike	cycle	2.2
rollerblade	offer	2.0
swim	bus	1.7
gallop	mail	1.7
	call	1.0

Table 2 shows the search results for several query terms. For *Swan Lake*, *China*, and *piano*, the two methods returned similar results. Results for *George Bush* are a little different. Google Sets found successive US presidents. Our method found people actually related to *George Bush*. For *audacious* and *walk*, results for the two methods are nearly identical. Results for *wise* from Google Sets are strange. Such a search might find a cluster consisting of people who are *wise*. Our method found both similar terms like *prudent* and terms with opposite meanings like *foolish*. Results for *beautiful* from our method also consist of similar terms and terms with opposite meanings. Google Sets, on the other hand, found only adjectives similar to *beautiful*.

In summary, our method is at least very precise. Google Sets finds terms with similar meanings, but our method finds terms with both similar and opposite meanings. Our method returns fewer results than Google Sets.

3.2 Adding Background Term to the Web Queries

If a user's query term is a word with multiple meanings, our method's results include coordinate terms for each meaning of the query term. For example, the

Table 3. Experimental results for jaguar with background terms

Query: jaguar							
Background: (N/A)		Background: car		Background: animal		Background: Mac	
panther	17.5	BMW	5.5	leopard	16.7	Panther	36.1
leopard	4.5	Mercedes	4.6	puma	7.4	Mac OS X	9.8
Land Rover	4.2	Land Rover	3.9	cougar	4.9	OS X	4.0
tiger	3.2	Ferrari	3.5	eagle	2.4	Mac OS	2.8
car	2.0	Ford	3.2	mountain lion	2.4	Mac OS X Panther	1.7
puma	1.4	Lexus	2.8	signs	2.0	Puma	1.4
eagle	1.4	Porsche	2.4	bird	1.4	OS	1.4

leftmost column in table 3 shows results for *jaguar*. *Jaguar* has several meanings; an automobile manufacturer, an animal, and a version of Mac OS X. *Land Rover* and *Mercedes* are coordinate terms of *Jaguar* as an automobile manufacturer. *Panther*, *leopard*, and *tiger* are coordinate terms of *jaguar* as an animal. *Panther*, *tiger*, and *puma* are coordinate terms of *jaguar* as a version of Mac OS X.

Google Sets can accept 1 to 5 terms, so if a user wants coordinate terms of *Jaguar* as an automobile manufacturer, *Jaguar* and *Land Rover* are suitable queries. Although our method can accept only one term, we are able to acquire coordinate terms for each meaning of a query term by adding "background terms" to the queries for the Web search. When a user's query is *jaguar*, we usually submit *"jaguar or"* and *"or jaguar"* to the search. But if a user indicates that *jaguar* means the animal, we change the queries for the Web search into **"jaguar or"** $\wedge$ **animal** and **"or jaguar"** $\wedge$ **animal**. Consequently, the searched coordinate terms relate to the background terms. The rest of table 3 shows the results for which background terms are set. In all of the three, the returned result terms are appropriate coordinate terms for each meaning.

4 Context Shared by Coordinate Terms

When considering coordinate terms in a search result, a user may wonder why a listed term is regarded as a coordinate term. Showing the context shared by the query term and the coordinate term will help the user understand what kind of relationship they have to each other. Our method extracts such context and shows it with the results. The context is represented as terms that characterize the context. When a coordinate term is found, titles and snippets are obtained from a Web search engine. These titles and snippets are sentences containing the query term and the coordinate term. Other terms that appear frequently in the snippets will also characterize the context. For every term in the titles and the snippets, the weighting is calculated. It is based on TF-IDF. Term Frequency (TF) of a term is the number of appearances in the titles and snippets. Inverse Document Frequency (IDF) is calculated from all the snippets.

The left panels in table 4 show coordinate term search results for *Bordeaux* and *Oracle*. *Bordeaux* is a term which has several meanings, so the contexts are different based on the meaning. The right panels in table 4 shows context terms for several coordinate terms. The context shared by *Bordeaux* and *Burgundy*

Table 4. Experimental results and context for Bordeaux and Oracle

Query: Bordeaux	
Burgundy	10.5
Paris	3.5
Cabernet	3.5
Lyon	2.4
fixed copper	2.2
Bergerac	1.7
blue	1.4
Montpellier	1.4
Merlot	1.4
Metz	1.4
light red wine	1.0

Query: Bordeaux											
Burgundy		Paris		Cabernet		fixed copper		blue			
Burgundy	85	Paris	42	Cabernet	40	fix	40	Grosfillex	46		
wine	33	Dordogne	28	Sauvignon	21	copper	32	blue	34		
live	28	train	24	Noir	16	apply	24	bistro	31		
region	22	gateway	15	Pinot	16	branch	20	chair	31		
Champagne	21	fish	15	Syrah	16	twig	20	furniture	31		
roast	20	enchant	15	wine	16	mixture	19	Miami	31		
Languedoc	20	river	13	bartend	15	infect	18	side	31		
home	18	TEFL	13	Merlot	13	apple	17	patio	15		
feature	16	European	13	substitute	12	spray	16	DealTime	15		
Latour	15	Sarlat	13	Zinfandel	12	leaf	15	color	13		

Query: Oracle	
SQL Server	20.0
SQL	13.0
MySQL	10.2
DB	7.4
SAP	5.5
Sybase	5.3
Access	3.5
PostgreSQL	1.7
MS SQL	1.4
System	1.4
Microsoft SQL	1.0

Query: Oracle									
SQL Server		MySQL		SAP		Sybase			
server	181	MySQL	136	SAP	104	Sybase	138		
SQL	166	JSPMaker	85	MBA	69	database	31		
skill	56	sp	69	app	60	vBulletin	31		
cheaper	55	JSP	68	zone	46	compliant	31		
tricky	55	quick	68	consult	46	monitor	31		
iSeries	53	general	61	strategy	46	servlet	27		
perception	47	weblog	48	choice	33	export	27		
techtip	46	workgroup	46	business	33	core	27		
data	41	download	40	forum	32	ensure	27		
key	40	install	33	edge	31	connect	26		

is characterized by the terms *wine*, *live*, *region*, and so on. Because both *Bordeaux* and *Burgundy* are famous wine regions, these terms properly represent the context. Other results also show basically appropriate context. The context for *Paris* may be about travel, that for *Cabernet* consists of many kinds of grapes. *Fixed copper* is one of the oldest fungicides, and the terms characterizing its context indicate it. The context for *blue* is not so good, but at least in this context bordeaux and blue may be used for some furniture.

For *Oracle*, the represented context differs a little between each coordinate term. Price would be a shared contextual topic between *Oracle* and *SQL Server*. Both *Oracle* and *MySQL* may be contextual terms for JSP. *Oracle* and *SAP* would share some contextual business issues because both of them are ERP package venders. The topics common to *Oracle* and *Sybase* would be very technical ones.

Thus, returned contextual terms are useful to a user in understanding coordinate terms.

5 Conclusion

We proposed a method of searching coordinate terms. After receiving a query term from a user, our system submits two Web search queries by connecting the query term and the conjunction "*OR*". The title and snippets of the Web search results from a conventional Web search engine are analyzed, and proper coordinate terms for the query term are found. Our method does not need any preprocessing or any corpora because it obtains all the data through two Web searches. It searches very quickly and can accept a term in any field. Some

experimental results are shown. By comparing our method's results with those of Google Sets, we prove our system works well. Our method is very precise. It can search coordinate terms for a word with multiple meanings by adding "background terms" to the queries for the Web search. Our method can also obtain the background context shared by the query term and each returned coordinate term.

Acknowledgments

This work was supported in part by the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT) project "Software Technologies for Search and Integration across Heterogeneous-Media Archives," by MEXT Grants-in-Aid for Scientific Research (Nos. 18049041 and 16700097), and by a 21st Century COE Program at Kyoto University called "Informatics Research Center for Development of Knowledge Society Infrastructure."

References

1. Church, K.W., Hanks, P.: Word association norms, mutual information, and lexicography. Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (1998) 76–83
2. Lin, D.: Automatic retrieval and clustering of similar words. Proceedings of the 36th annual meeting on Association for Computational Linguistics (1998) 768–774
3. Shinzato, K., Torisawa, K.: A simple www-based method for semantic word class acquisition. Proceedings of the Recent Advances in Natural Language Processing (RANLP05) (2005) 493–500
4. Ghahramani, Z., Heller, K.: Bayesian sets. Proceedings of the Nineteenth Annual Conference on Neural Information Processing Systems (NIPS2005) (2005)
5. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. Proceedings of the Fourteenth International Conference on Computational Linguistics (1992) 539–545
6. Shinzato, K., Torisawa, K.: Acquiring hyponymy relations from web documents. Proceedings of Human Language Technology Conference/North American chapter of the Association for Computational Linguistics annual meeting (HLT-NAACL04) (2004) 73–80
7. Sanderson, M., Croft, B.: Deriving concept hierarchies from text. Proceedings of the 22nd ACM SIGIR Conference (SIGIR'99) (1999) 206–213

A Semantic Matching of Information Segments for Tolerating Error Chinese Words*

Maoyuan Zhang[1, 3], Chunyan Zou[2], Zhengding Lu[1], and Zhigang Wang[1]

[1] Department of Computer Science and Technology,
HuaZhong University of Science and Technology,
430074 Wuhan, P.R. China
zmydragon@163.com
[2] School of Foreign Languages, HuaZhong Normal University,
430079 Wuhan, P.R. China
[3] Schoole of Management, HuaZhong University of Science and Technology,
430074 Wuhan, P.R. China

Abstract. There exist new words and error words in Chinese information of web pages. In this paper, we introduce our definition of semantic similarity between sememes and their theorems. On the base of proving the theorems, the influence of the parameter is analyzed. Moreover, this paper presents a novel definition of the word similarity based on the sememe similarity, which can be used to match the new Chinese words with the existing Chinese words and match the error Chinese words with correct Chinese words. And also, based on the novel word similarity, a matching method of information segments is presented to recognize the category of Chinese web information segments, in which new words and error words occur. In addition, the experiment of the matching methods is presented. Therefore, the novel matching method is an efficient method both in theory and from experimental results.

1 Introduction

A huge amount of web information exists on the Internet, and the information discovery methods of web pages have been researched to gain the valuable information exactly. Semantic matching is a puzzling problem, not only for information discovery but also for natural language processing. For example, an ontology search engine [1] used vector-matching algorithm, a XML document clustering [2] used the matching method to gain the similarity between documents, and a visual speech recognition system [3] used the matching method to distinguish a word from other words.

Since information segments are composed of words, semantic matching of information segments is based on the methods of word semantic matching. Methods of word semantic matching can be categorized into two groups: the word matching methods based on occurrence frequency and those based on conceptual relationship.

* This work was partially supported by National Natural Science Foundation of China under Grant 60403027.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 48 – 59, 2006.

The former matching methods use the word occurrence frequency in documents or the common characters between words to calculate the similarity between words. Some of the former semantic matching methods [4][5][6] did not take the structural relationship of the lexical taxonomy into consideration.

The latter matching methods use the word conceptual relationship in the semantic network to calculate the similarity between words. As for the semantic network, assume that a lexical taxonomy is structured in a tree like hierarchy with a node for a concept, and Rada et al. [7] has proven that the minimum number of edges separating concepts c1 and c2 is a metric for measuring the conceptual distance between c1 and c2. Their work forms the basis of the relationship-based matching methods. Some semantic matching methods [8][9][10][11], considered the conceptual distance in the hierarchy of the lexical taxonomy, but did not take into account the conceptual depth in the hierarchy of the lexical taxonomy. Furthermore, a word similarity method based on different ontologies [12] considered both the conceptual distance and the conceptual depth in the taxonomy hierarchy, but had to modify and expand its semantic networks when a new word appeared.

There exist some problems for Chinese information in web pages. On the one hand, the information of web pages is likely to change, and new Chinese words may appear with the development of the human knowledge. On the other hand, the information of web pages may include error Chinese words led by contrived factors and natural factors. For instance, someone propagandizing illegal information inserted irrelative Chinese characters into the words to prevent information extraction tools from gaining the meanings of the Chinese words. Hence, facing those problems, the semantic matching methods should be of the adaptive ability to match the new Chinese words with the existing Chinese words, and be of the semantic tolerance ability to gain the correct semantic meanings from the error Chinese words.

The rest of this paper is organized as follows. In Section 2, we introduce our definition of semantic similarity between sememes. In Section 3, we introduce and prove some theorems on the semantic similarity between sememes, and then analyze the influence of the parameter on the base of the theorems. In Section 4, we present a novel definition of the word similarity based on the sememe similarity, which can be used to match the new Chinese words with the existing Chinese words and match the error Chinese words with correct Chinese words. And then, a matching method of information segments is presented to recognize the category of web information segments. In Section 5, the experiment of the matching methods is presented.

2 Semantic Similarity Between Sememes

2.1 Sememe Network

HowNet is an online bilingual common sense hierarchical ontology describing semantic relations between concepts (represented by Chinese and English words) and also describing semantic relations between the attributes of concepts [13]. As the most basic language unit, one or more Chinese sememes form a Chinese word with some certain meanings, one or more Chinese words form a phrase, and one or more words

and phrases form a sentence. Each Chinese sememe or word has its own meaning, from which the meaning of a phrase or a sentence that contains them originates. So there is a tight semantic relationship between Chinese sememes and words.

From medicinal information in web pages, we extracted the Chinese words about medicine, such as a Chinese word "药名", which means "name of medicine" in English. Furthermore, we extracted the sememes from the Chinese words about medicine, such as sememe "药" and "名", which mean "medicine" and "name" respectively in English. Based on the sememes and the Chinese words about medicine, we constructed a HowNet—medicine according to the constructional principle of the HowNet. HowNet—medicine is a special kind of HowNet for the medicinal information, and describes semantic relations between concepts (*represented by Chinese sememes and words*) and also semantic relations between the attributes of concepts. The difference between HowNet—medicine and HowNet lies in that the previous one is based on Chinese sememes and words.

2.2 Similarity Function of Sememes

If we assign "exactly the same" with a value of 1 and "no similarity" as 0, then the interval of similarity is [0,1]. The values of argument of similarity function may cover a large range up to infinity, so it is intuitive that the similarity function is a nonlinear function. Hence, The nonlinearity of the similarity function is taken into account in the derivation of the formula for semantic similarity between sememes.

Sememe Similarity Based on Path Length. Given two sememes $seme_1$ and $seme_2$, we need to find the semantic similarity of them. We can do this by analysis of the knowledge base, as follows: Sememes are associated with concepts in the hierarchical HowNet—medicine. Hence, we can find the first concept in the hierarchical semantic network that subsumes the concepts representing the compared sememes. One direct method for similarity calculation is to find the minimum length of path connecting the two concepts representing the compared sememes.

Definition 1: Suppose $Seme_1$ and $Seme_2$ are two sememes, and the minimum length of path connecting the two concepts representing them is L. Then the sememe similarity based on path length is defined as

$$Sim1(Seme1, Seme2) = f_1(L) = e^{-\alpha L} \ . \tag{1}$$

where constant $\alpha>0$ and $L \in [0,+\infty)$.

Expanded Sememe Similarity. The depth of the sememe is derived by counting the levels from the concept representing the sememe to the top of the lexical hierarchy. Sememes at upper layers of hierarchical semantic nets have more general concepts and less semantic similarity between sememes than sememes at lower layers. This must be taken into account in calculating the sememe similarity. We therefore need to scale down $Sim1(seme_1, seme_2)$ for sememes at upper layers(lower depth) and to scale up $Sim1(seme_1, seme_2)$ for sememes at lower layers(higher depth). Moreover, the similarity is constrained to [0,1].

Definition 2: Function $f_2(h_1, h_2)$ is defined as

$$f_2(h1,h2)=\frac{e^{\beta(h1+h2)/2}-e^{-\beta(h1+h2)/2}}{e^{\beta(h1+h2)/2}+e^{-\beta(h1+h2)/2}}\text{, where constant } \beta>0 \text{ and } h1,h2\in[0,+\infty).$$

Definition 3: Suppose $Seme_1$ and $Seme_2$ are two sememes, their depths are h_1 and h_2 respectively, and the minimum length of path connecting the two concepts representing them is L. Then the expanded sememe similarity is defined as

$$\begin{aligned}\text{Sim2}(\text{seme1},\text{seme2}) &= f(f_1(L), f_2(h1,h2))\\ &= f_1(L)\times f_2(h1,h2).\end{aligned} \tag{2}$$

3 Theorems on the Sememe Similarity

In order to analyze the parameterβ 's influence on the expanded sememe similarity, we will introduce some theorems. Some functions are defined as follows before introducing some theorems.

Definition 4: Function u(h) is defined as $u(h)=\frac{e^{\beta h}-e^{-\beta h}}{e^{\beta h}+e^{-\beta h}}$, where the constant β>0 and the argument $h\in[0,+\infty)$.

Definition 5: Function w(h) is defined as $w(h)=4h/(e^{\beta h}+e^{-\beta h})^2$, where the constant β>0 and the argument $h\in[0,+\infty)$.

Definition 6: Function v(h) is defined as

$$v(\beta,h)=u(h+d)-u(h)=\frac{e^{\beta(h+d)}-e^{-\beta(h+d)}}{e^{\beta(h+d)}+e^{-\beta(h+d)}}-\frac{e^{\beta h}-e^{-\beta h}}{e^{\beta h}+e^{-\beta h}},$$

where argument $\beta, h\in[0,+\infty)$, and constant d≥0.

3.1 Theorems and Their Proofs

Based on those function definitions above, we introduce and prove some theorems as follows.

Theorem 1: The range of the function u(h), defined by the Definition 4, is [0,1]. Moreover, if $u(h_o)\geq a$, where a is a constant and h_o is a value of the argument h, then $\beta\geq\frac{1}{2h_o}\ln\frac{1+a}{1-a}$.

Proof: Differentiating u(h) with respect to variable h as follows,

$$\begin{aligned}du/dh &= (1/(e^{\beta h}+e^{-\beta h})^2)[\beta(e^{\beta h}+e^{-\beta h})^2-\beta(e^{\beta h}-e^{-\beta h})^2]\\ &= 4\beta/(e^{\beta h}+e^{-\beta h})^2 \quad .\end{aligned} \tag{3}$$

Since β>0, it can be obtained that $du/dh > 0$. That is, the function u(h) is a monotonically increasing function, which is shown in the figure 1.

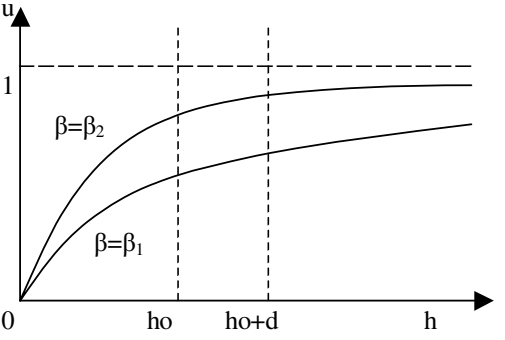


Fig. 1. The curve of function u(h)

According to the above conclusion, the function u(h) has the minimum value 0 when h=0, and has the maximum value 1 when $h \to +\infty$. Hence, the function's range is [0,1].

Differentiating the derivative of function u(h) with respect to variable h as follows,

$$\begin{aligned} d^2u/dh^2 &= (-8\beta/(e^{\beta h}+e^{-\beta h})^3)(\beta e^{\beta h}-\beta e^{-\beta h}) \\ &= -8\beta^2(e^{\beta h}-e^{-\beta h})/(e^{\beta h}+e^{-\beta h})^3 . \end{aligned} \tag{4}$$

Since βh>0, it can be obtained that $e^{\beta h}-e^{-\beta h} \le 0$.

So $d^2u/dh^2 \le 0$, that is to say, the derivative function of u(h) is a monotonically decreasing function.

If $u(h_o) \ge a$, then $(e^{\beta h_o}-e^{-\beta h_o})/(e^{\beta h_o}+e^{-\beta h_o}) \ge a$, that is, $(e^{2\beta h_o}-1)/(e^{2\beta h_o}+1) \ge a$.

Hence, it can be easily proved that $e^{2\beta h_o} \ge (1+a)/(1-a)$, that is to say $\beta \ge \frac{1}{2h_o}\ln\frac{1+a}{1-a}$.

This completes the proof.

Theorem 2: There exist natural number b and c, where b=c/(2β) and 1.54<c<1.55, such that the function w(h) defined by the Definition 5 is monotonically increasing when $h \in [0,b]$ and is monotonically decreasing when $h \in (b,+\infty)$.

Proof: Differentiating function w(h) with respect to variable h as follows,

$$\begin{aligned} dw/dh &= (4/(e^{\beta h}+e^{-\beta h})^4)[(e^{\beta h}+e^{-\beta h})^2 - \\ &\quad 2\beta h(e^{\beta h}+e^{-\beta h})(e^{\beta h}-e^{-\beta h})] \\ &= (4/(e^{\beta h}+e^{-\beta h})^3)[(e^{\beta h}+e^{-\beta h}) - 2\beta h(e^{\beta h}-e^{-\beta h})] \\ &= (4e^{\beta h}/(e^{\beta h}+e^{-\beta h})^3)[(1+e^{-2\beta h}) - 2\beta h(1-e^{-2\beta h})] \\ &= (4e^{\beta h}/(e^{\beta h}+e^{-\beta h})^3)[(1+2\beta h)e^{-2\beta h}+1-2\beta h]. \end{aligned} \tag{5}$$

Let us introduce a new function

$$g(x) = (1+x)e^{-x}+1-x, \text{ where } x \geq 0. \tag{6}$$

Differentiating g(x) with respect to variable x as follows,

$$dg/dx = e^{-x} - (1+x)e^{-x} - 1 = -1 - xe^{-x} < 0$$

So the function g(x) is a monotonically decreasing function.

According to the above conclusion, the function u(h) has the maximum value 2 when x=0, and has the minimum value, a negative infinity, when $x \to +\infty$. Moreover, the equation g(x)=0 has an unique solution.

Suppose the unique solution is c, then g(x)>0 when 0≤x≤c, and g(x)<0 when x>c. When x=1.54, g(x)= 0.0045>0. When x=1.55, g(x)= -0.0088<0. So it can be obtained that 1.54<c<1.55.

From equation (5) and (6), it is easily given that

$$dw/dh = (4e^{\beta h}/(e^{\beta h}+e^{-\beta h})^3) \times g(2\beta h).$$

Hence, when 0≤2βh<c, it is obtained that g(2βh)>0 and then $dw/dh > 0$. Moreover, when 2βh>c, it is obtained that g(2βh)<0 and then $dw/dh < 0$.

Suppose b=c/(2β). Therefore, we can obtain that the function w(h) is monotonically increasing when $h \in [0, b]$ and is monotonically decreasing when $h \in (b, +\infty)$.

This completes the proof.

Theorem 3: As to the function v(β, h), defined by the Definition 6, suppose h_o is a value of the variable h. Then, the function v(β, h) is monotonically decreasing with respect to variable β when $h=h_o$ and $2\beta h_o > c$, where the constant c>1.55.

Proof: Let us introduce the function w(h) from the theorem 2 as follows

$$w(h) = 4h/(e^{\beta h}+e^{-\beta h})^2.$$

Differentiating v(β, h) with respect to variable β as follows

$$\begin{aligned} \partial v/\partial \beta &= [4(h+d)/(e^{\beta(h+d)}+e^{-\beta(h+d)})^2] - [4h/(e^{\beta h}+e^{-\beta h})^2] \\ &= w(h+d) - w(h) \end{aligned}$$

According to the theorem 2, w(h+d)≤w(h), when h≥c/(2β).

So that

$$\partial v / \partial \beta = w(h+d) - w(h) \le 0 \text{, when } h \ge c/(2\beta).$$

Since $2\beta h_o > c$, it is easy to see that $h_o > c/(2\beta)$.

Hence, it is obtained that the function v(β, h) is monotonically decreasing with respect to variable β when $h=h_o$ and $2\beta h_o > c$

This completes the proof.

Corollary 1: Suppose $Seme_1$ and $Seme_2$ are two sememes. Then the range of the similarity function $Sim2(Seme_1, Seme_2) = f(f_1(L), f_2(h_1, h_2))$, which is defined by Definition 3, is [0,1], and the similarity function Sim2 is nonlinear.

Proof: Let $h=(h_1+h_2)/2$, then

$$f_2(h1, h2) = \frac{e^{\beta(h1+h2)/2} - e^{-\beta(h1+h2)/2}}{e^{\beta(h1+h2)/2} + e^{-\beta(h1+h2)/2}} = u(h) \cdot$$

According to the theorem 1, $0 \le u(h) \le 1$, and thus $0 \le f_2(h1, h2) \le 1$.

In addition, $0 \le f_1(l) = e^{-\alpha l} \le 1$.

So that

$$0 \le f(f_1(L), f_2(h1, h2)) = f_1(L) * f_2(h1, h2) \le 1.$$

That is to say, the range of the similarity function $Sim2(seme_1, seme_2)$ is [0,1].

As to the function $Sim2(seme_1, seme_2) = f(f_1(L), f_2(h_1, h_2))$, the domains of argument L, h_1 and h2 are $[0,+\infty)$. If the function is linear, then $Sim2(seme_1, seme_2) \to +\infty$ when $L \to +\infty$. This leads to a contradiction, so the similarity function Sim2 is nonlinear.

This completes the proof.

3.2 Analyses of Parameter β 's Influence

Suppose the maximal depth of the semantic network is h_{max}, then the value of the function u(h) for $h > h_{max}$, is not significant to the similarity calculating, but the value of the function u(h) for $h \in [0, h_{max}]$ has influence to the similarity calculating. Hence, the range of the function u(h) for $h \in [0, h_{max}]$ should be large enough to be close to the interval [0,1], in order to gain better effect of influence. That is to say, the value of $u(h_{max})$ should be large enough to be close to 1.

On one hand, the value of $u(h_{max})$ should be large enough. In order to make $u(h_{max}) > a$, according to the theorem 1, the parameter β should satisfy that $\beta \ge \frac{1}{2h_{max}} \ln \frac{1+a}{1-a}$. For instance, if a =0.95 and h_{max} =10, then β≥0.183.

On the other hand, it is not promised that the larger the value of $u(h_{max})$ is, the better the parameter β 's influence effect is. As shown in the figure 1, suppose $\beta_2 \ge \beta_1$, $2h_o\beta_1 > c$, c>1.55, and d=1. Then the increment of the function u(h) is equals to $v(\beta_1, h_o)$ (shown in the theorem 3) when $\beta=\beta_1$ and h increases from h_o to h_o+d,. And also, the increment of the function u(h) is equals to $v(\beta_2, h_o)$ (shown in the theorem 3) when $\beta=\beta_2$ and h increases from h_o to h_o+d. According to the theorem 3 and the assume $\beta_2 \ge \beta_1$, it can be obtained that $v(\beta_2, h_o) \le v(\beta_1, h_o)$. Therefore, the larger is the value of

β, the smaller is the difference ($u(h_o+d)-u(h_o)$). Hence, an excessively large value of β can decreases the depth's influence on the similarity to some extent.

According to the above conclusion, the parameter β should satisfy that $\beta \geq \frac{1}{2h_{\max}} \ln \frac{1+a}{1-a}$ and should not be set as an excessively large value, to gain better effect of the depth's influence on the similarity.

4 Semantic Matching Method Based on Sememes

4.1 Word Similarity Based on Sememes

Definition 7: Suppose w is a word, and assume $Seme_1$, $Seme_2$, …, $Seme_n$ are the sememes forming the word w. Then the word w can be denoted by the sememe vector, which is defined as

$$SemeV=(Seme_1, Seme_2, \ldots, Seme_n) . \quad (7)$$

Definition 8: Suppose Seme is a sememe, and SemeV is a sememe vector ($Seme_1$, $Seme_2$, …, $Seme_n$). Then the similarity function between a sememe and a sememe vector, is defined as

$$Sim3(Seme, SemeV) = \max_{j=1}^{n} Sim2(Seme, Seme_j) \cdot \quad (8)$$

where the function Sim2 is defined by Definition 3.

Definition 9: Suppose the sememe vector of word w_1 is $SemeV_1=(Seme_{11}, Seme_{12}, \ldots, Seme_{1m})$, and the sememe vector of word w_2 is $SemeV_2=(Seme_{21}, Seme_{22}, \ldots, Seme_{2n})$. Then the word similarity based on sememes is defined as

$$Sim4(w1, w2) = \frac{1}{|SemeV_1|} \sum_{i=1}^{m} Sim3(Seme_{1i}, SemeV_2) \cdot \quad (9)$$

where $|SemeV_1|$ denotes the dimension of the vector $SemeV_1$.

4.2 The Solution of Matching for New Words and Error Words

With the development the human knowledge, the new Chinese words are formed by the sememes just as the existing Chinese words are, so the new Chinese words can also be denoted by the sememe vectors as well as the existing Chinese words. On the other hand, the error Chinese words are formed by the sememes just as the correct Chinese words are, so the error Chinese words can also be denoted by the sememe vectors as well as the correct Chinese words.

HowNet describes semantic relations between concepts (represented by Chinese and English words). Facing the new Chinese words, HowNet cannot give the semantic relation between the new words and the existing words, and the semantic similarity based on words cannot be applied. Hence, in order to calculate the similarity between words, we have to modify the HowNet by adding the new words. Moreover, HowNet cannot give the semantic relations between the error Chinese words and the correct Chinese words.

HowNet—medicine, introduced in Section 2, describes semantic relations between concepts (represented by Chinese sememes and words). Facing new Chinese words, we can extract the sememes forming the words, gain the semantic relations between sememes from HowNet—medicine, and give the semantic similarity between the new Chinese words and the existing Chinese words by applying the word similarity function based on sememes (defined by Definition 9). Facing error Chinese words, in the same way, we can give the semantic similarity between the error Chinese words and the correct Chinese words.

4.3 The Matching Method of Chinese Information Segments in Web Pages

Before recognizing which item an information segment belongs to, we set up a set of information items. For instance, the item set of medicine information includes an item on medicine name, an item on medicine usage and so on. Moreover, we set up a feature word set for every information item, and the feature word set is composed of some feature words to denote the information item. As shown in figure 2, F_1 is a feature set for the information item $item_1$.

The sememe-based semantic matching of Chinese information segments includes four layers. The first layer is named as word-sememe conversion, extracts the sememes from the words in the Chinese information segment and then uses the sememe vectors to denote the words. The second layer calculates the semantic similarity between sememe vectors and the feature words in the feature sets by using Eq.9, and then sums the similarities for the same sememe vector and the same feature set. For instance, m_{11} is the sum of similarities for $SemeV_1$ and set F_1. The third layer compares the sums of the same feature set. For instance, after comparing m_{1j}, m_{2j}, …, m_{ij}, …,m_{pj}, y_j is set as m_{ij} if m_{ij} is the largest. The fourth layer combines all components and output the vector $y=(y_1, y_2, \ldots, y_q)$. And then, we can recognize which item the Chinese information segment belongs to, by judging which component of vector y is the largest.

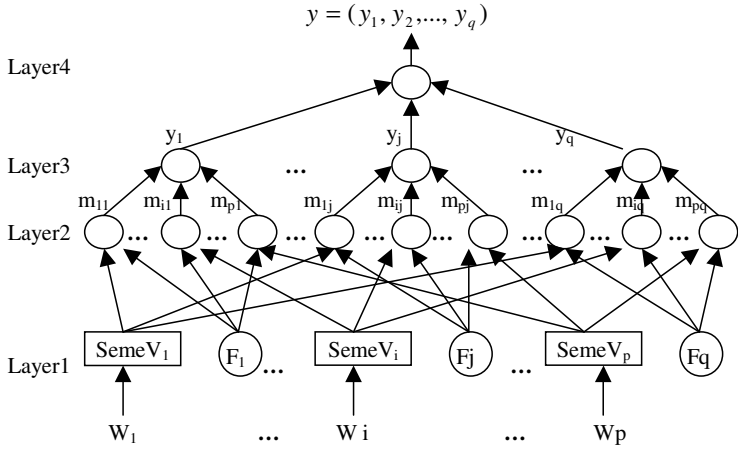


Fig. 2. The sememe-based semantic matching of Chinese information segments

4.4 Web-MIND System

We have successfully designed a monitoring system for Chinese medicine information in web pages, named as Web-MIND system. It can search, extract and analyze the illegal medicine information from the web pages, for the medicine administration to monitor the Chinese medicine information in web pages.

The Web-MIND system is composed of five components, which are search engine, segmentation of Chinese words, classification of web pages, information extraction of web pages, and matching of information segments. The search engine component applies the search engine "google" to search the Chinese web pages about medicine. The segmentation component of Chinese words uses a segmentation method of Chinese words based on language situation, which we have presented in the paper [14], to segment the Chinese sentences into Chinese words. The classification component uses a fuzzy classification method based on feature selection for web pages, which we have presented in the paper [15], to filter out all non-medicine web pages. The information extraction component extracts the information segments about medicine. The matching component of information segments uses the matching method of Chinese web information segments, which we present in this paper, to recognize which category the information segments belong to.

5 Experiments

The Web-MIND system got 930 Chinese web pages on medicine information from the Internet. Some new Chinese words and error Chinese words appearing in these pages did not exist in both HowNet and HowNet—medicine. After the information extraction component of the Web-MIND system extracting the information segments about medicine, the Web-MIND system adopted two methods to recognize which item the information segments belong to, and compared their accuracies. The two methods are a semantic matching method based on words and the semantic matching method based on sememes. Here, parameters were set as α=0.2 and β=0.6.

As shown in table 1, the two methods' accuracies were compared with respect to the category of information segments, which were medicine information about name, about efficacy, about caution, about usage and about manufacturer.

Table 1. The two methods' results

Information segments about	Quantity	Method A		Method B	
		Correct	Accuracy	Correct	Accuracy
Name	930	793	85.3%	830	89.2%
Efficacy	930	774	83.2%	823	88.5%
Caution	760	629	82.8%	668	87.9%
Usage	760	641	84.3%	676	89.1%
Manufacturer	640	544	85.0%	569	88.9%
Total	4020	3381	84.1%	3566	88.7%

On the one hand, the total accuracy of method A was 84.1% while the total accuracy of method B was 88.7%. Hence, the method B was more efficient than the method A with respect to the total accuracy.

On the other hand, the max deviation of the method A's accuracy was equal to 84.1%-82.8%=1.3%, while the max deviation of the method B's accuracy was equal to 88.7%-87.9%=0.8%. As to the information on efficacy and on caution, in which some new words came into being with much probability, the method B was more efficient than the method A as shown in the figure 3. Hence, the method B had more stable accuracies than the method A.

According to the above conclusion, the semantic matching method based on sememes is an efficient matching method to recognize which item the Chinese information segments belong to, especially for the information including some new Chinese words and error Chinese words.

6 Conclusion

This paper presents a novel semantic similarity between sememes, and analyzes the influence of the parameter by introducing and proving some theorems. On the base of semantic similarity between sememes, a semantic matching method of Chinese information segments is presented to recognize which item Chinese web information segments belong to, in which new words and error words occur. The semantic matching method is an efficient matching method, by applied successfully to the Web-MIND system.

References

1. Gao, M., Liu, C., Chen, F., An ontology search engine based on semantic analysis, 3rd International Conference on Information Technology and Applications, Sydney, Australia, (2005)256 – 259
2. Yang, J., Cheung, W.K., Chen, X., Integrating element and term semantics for similarity-based XML document clustering, The 2005 IEEE/WIC/ACM International Conference on Web Intelligence, University of Technology of Compiegne, France, IEEE Computer Society Press (2005) 222 – 228.
3. Da, L.G, Facon, J., Borges, D.L, Visual speech recognition: a solution from feature extraction to words classification, Proceeding of Symposium on Computer Graphics and Image Processing, XVI Brazilian (2003) 399 – 405.
4. Shen, H.T., Shu, Y., Yu, B, Efficient semantic-based content search in P2P network, IEEE Transactions on Knowledge and Data Engineering, 16 (7) (2004) 813 – 826.
5. Yi, S., Huang, B., Tatchan Weng, XML application schema matching using similarity measure and relaxation labeling, Information Sciences, 169(1-2) (2005) 27-46.
6. Nakashima, T., Classification of characteristic words of electronic newspaper based on the directed relation, 2001 IEEE Pacific Rim Conference on Communications, Computers and signal Processing, Victoria, B.C., Canada, IEEE Computer Society Press (2001) 591 – 594.
7. Rada, R., Mili, H., Bichnell, E., Blettner, M., Development and application of a metric on semantic nets, IEEE Transaction on Systems, Man, and Cybernetics, 9(1) (1989) 17-30.

8. Cross, V., Fuzzy semantic distance measures between ontological concepts, 2004. Processing NAFIPS '04. IEEE Annual Meeting of the Fuzzy Information, Alberta, Canada, IEEE Computer Society Press (2004).635 – 640.
9. Soo, V., Yang, S., Chen, S., Fu, Y., Ontology acquisition and semantic retrieval from semantic annotated Chinese poetry, Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries, Tuscon, AZ, USA, IEEE Computer Society Press (2004) 345 – 346.
10. Vladimir, A.O., Ontology based semantic similarity comparison of documents, 14th International Workshop on Database and Expert Systems Applications (DEXA'03), Prague, Czech Republic (2003) 735-738.
11. Cheng, L., Lu, Z., Wen, K., The exploration and application about amphibolous matching based on semantics, Journal Huazhong University of Science & Technology (Nature Science Edition), 31 (2) (2003) 23-25.
12. Rodriguez, M.A., Egenhofer, M.J., Determining semantic similarity among entity classes from different ontologies, IEEE Transactions on Knowledge and Data Engineering, 15 (2) (2003) 442 – 456.
13. Guan Y., Wang, X., Kong, X., Zhao, J., Quantifying semantic similarity of Chinese words from HowNet, Proceedings of the First International Conference on Machine Learning and Cybernetics, Beijing, China, IEEE Computer Society (2002) 234-239.
14. Zhang, M.Y., Lu, Z.D., Zou, C.Y., A Chinese word segmentation based on language situation in processing ambiguous words, Information Sciences, 162(3--4) (2004) 275-285.
15. Zhang, M.Y., Lu Z.D., A fuzzy classification based on feature selection for web pages, The 2004 IEEE/WIC/ACM International Conference on Web intelligence, Beijing, China, IEEE Computer Society Press (2004) 469-472.

Block-Based Similarity Search on the Web Using Manifold-Ranking

Xiaojun Wan, Jianwu Yang, and Jianguo Xiao

Institute of Computer Science and Technology,
Peking University, Beijing 100871, China
{wanxiaojun,yangjianwu,xiaojianguo}@icst.pku.edu.cn

Abstract. Similarity search on the web aims to find web pages similar to a query page and return a ranked list of similar web pages. The popular approach to web page similarity search is to calculate the pairwise similarity between web pages using the Cosine measure and then rank the web pages by their similarity values with the query page. In this paper, we proposed a novel similarity search approach based on manifold-ranking of page blocks to re-rank the initially retrieved web pages. First, web pages are segmented into semantic blocks with the VIPS algorithm. Second, the blocks get their ranking scores based on the manifold-ranking algorithm. Finally, web pages are re-ranked according to the overall retrieval scores obtained by fusing the ranking scores of the corresponding blocks. The proposed approach evaluates web page similarity at a finer granularity of page block instead of at the traditionally coarse granularity of the whole web page. Moreover, it can make full use of the intrinsic global manifold structure of the blocks to rank the blocks more appropriately. Experimental results on the ODP data demonstrate that the proposed approach can significantly outperform the popular Cosine measure. Semantic block is validated to be a better unit than the whole web page in the manifold-ranking process.

1 Introduction

The goal of similarity search on the web is to find web pages similar to a query page specified by the user and return a ranked list of similar web pages to the user. The typical kind of similarity search is K nearest neighbor search (i.e. K-NN search), which is to find K documents most similar to the query document.

Web page similarity search is widely used to improve traditional web search engines by allowing the user to use a web page as a query and thus releasing the burden of extracting key words from the web page to formulate the query for the user. Traditional web search engines take a query of several terms as input and return a set of relevant pages that match the query terms. However, the query terms are usually difficult to define and prone to be inappropriate. Note that it happens very often that when users already have an interesting web page, they just want to see more relevant web pages, and they can use the web page as a query and perform similarity search on the web to get similar pages. A few search engines have provided the functionalities of web page similarity search or recommendation. For example, Google[1] can perform an advanced search with "related" option to find web pages similar to a user-specified web page.

[1] http://www.google.com

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 60 – 71, 2006.

The retrieval performance of a similarity search engine relies heavily on the measure for evaluating web page similarity. Most popular measures for document similarity, such as the Cosine measure, the Jaccard measure and the Dice measure [1, 19], can be directly applied to evaluate web page similarity by using only textual content of web pages, among which the standard Cosine measure is considered best for document similarity search. Haveliwala et al. [11] adopted this intuitive idea and performed experiments to evaluate system performance based on a new evaluation strategy. Web page structure denoted by HTML tags has also been used to help evaluate web page similarity [8, 13]. Tombros and Ali [18] explored the factors of the textual content and the structural information when evaluating web page similarity. Another different source widely used to determine similarity is the link structure between web pages, such work including [9, 10, 12, 15, 21]. In this study, we do not make use of the link structure between web pages.

In most previous approaches, a web page is considered as a single unit for similarity calculation. However, a web page as a whole may not be appropriate to represent a single topic because a web page usually contains various contents such as navigation, decoration, interaction, contact information, which may be unrelated to the main topic of the web page. Furthermore, a web page often contains multiple topics that are not necessarily relevant to one another. Therefore, detecting the semantic content structure of a web page could potentially improve the performance of web similarity search. Moreover, most approaches find relevant web pages to a query page only by the pairwise comparison between the web page and the query page, thus ignoring the intrinsic global manifold structure of the whole set of web pages. In order to address the above two limitations, we calculate web page similarity at a finer granularity of page block instead of at the coarse granularity of the whole web page, each block representing a semantic unit with coherent text reflecting a single topic. And a manifold-ranking process is employed to make full use of the relationships between the blocks of web pages. In the manifold-ranking process, blocks can spread their ranking scores to their nearby neighbors via a weighted network.

In more details, the proposed similarity search approach based on manifold-ranking of blocks consists of two processes: initial ranking and re-ranking. In the initial ranking process, a small number of web pages are initially retrieved based on the popular Cosine measure. In the re-ranking process, the query page and the initially retrieved web pages are segmented into blocks by the VIPS algorithm [5, 17], and then the manifold-ranking algorithm [23, 24] is applied on the blocks and each block obtains its ranking score. Lastly, a web page gets its final retrieval score by fusing the ranking scores of the blocks in the page. The initially retrieved web pages are re-ranked and the re-ranked list is returned to users. Experimental results on the ODP data show the improved performance of the proposed approach over the Cosine measure. Page block is validated to be a more suitable unit than the whole web page in the manifold-ranking process.

The rest of this paper is organized as follows: The proposed approach is described in detail in Section 2. Section 3 gives the experiments and results. Lastly, we present our conclusion in Section 4.

2 The Proposed Approach

2.1 Overview

The aim of the proposed approach is two-fold: one is to evaluate the similarity between a query page and a web page at a finer granularity by segmenting the web pages into semantic blocks, which addresses the limitation of present similarity metrics based on the whole web page, usually characterized as a series of blocks with semantically coherent content; the other is to evaluate the similarity between a query page and a web page by exploring the relationships (i.e. the intrinsic manifold structure) between all the obtained blocks in the feature space, which addresses the limitation of present similarity metrics based only on pairwise comparison.

The proposed approach first segments the web pages (including the query page) into blocks using the VIPS algorithm, and then applies a manifold-ranking process on the blocks. All the blocks of a web page obtain their ranking scores and the retrieval score of the web page is acquired by fusing the ranking scores of its blocks.

Note that it may have high computational cost to apply the manifold-ranking process to all web pages in the collection, so the above manifold-ranking process is taken as a re-ranking process. First, we use popular similarity measures (e.g. Cosine, Jaccard, Dice, etc.) to efficiently obtain an initial ranking list of web pages, and then the initial k web pages in the returned list are re-ranked by applying the above manifold-ranking process.

Formally, given a query web page q and a web collection C, the proposed approach consists of the following four steps:

1. **Initial Ranking:** The initial ranking process uses a popular similarity measure (i.e. Cosine) to get a set of top ranked web pages $D_{\text{init}} \subseteq C$ in response to the query page q, $|D_{\text{init}}|=k$. Each web page $d_i \in D_{\text{init}}$ $(1 \le i \le k)$ is associated with an initial retrieval score $InitScore(d_i)$.

2. **Page Segmentation:** By using the VIPS algorithm, the query page q is segmented into a set of blocks $\chi_q = \{x_1, x_2, \ldots, x_p\}$, and all web pages in D_{init} are segmented respectively and the set of blocks corresponding to the web pages in D_{init} is $\chi_{D_{\text{init}}} = \{x_{p+1}, x_{p+2}, \ldots, x_n\}$.

3. **Manifold-Ranking:** The manifold-ranking process in applied on the total set of blocks: $\chi = \chi_q \cup \chi_{D_{\text{init}}}$, and each block x_j $(p+1 \le j \le n)$ in $\chi_{D_{\text{init}}}$ gets its ranking score f_j^*.

4. **Score Fusion:** The final score $FinalScore(d_i)$ of a web page $d_i \in D_{\text{init}}$ $(1 \le i \le k)$ is computed by fusing the ranking scores of all the blocks within d_i. The documents in D_{init} are re-ranked according to their final scores and the re-ranked list is returned.

The steps 2-4 are key steps and compose the re-ranking process. They will be illustrated in detail in next sections, respectively.

2.2 The Page Segmentation Process

Several kinds of methods have been proposed for web page segmentation, among which the most popular ones are DOM-based segmentation [7], location-based

segmentation [14] and Vision-based Page Segmentation (VIPS) [5, 17]. Compared with other segmentation algorithms, VIPS excels in both an appropriate partition granularity and coherent semantic aggregation. Recently, the VIPS algorithm has been successfully applied to various web information retrieval tasks with encouraging results [2, 3, 4, 6, 22]. In this study, we adopt the VIPS algorithm to segment web pages into semantic blocks.

The VIPS algorithm makes full use of page layout features such as font, color and size and takes advantage of visual cues to obtain the vision-based content structure of a web page. The algorithm can successfully bridge the gap between the DOM structure and the semantic structure. The page is partitioned based on visual separators and structured as a hierarchy closely related to how a user will browse the page. Content related parts could be grouped together even if they are in different branches of the DOM tree.

The VIPS algorithm first extracts all the suitable nodes from the HTML DOM tree, and then finds the separators between these nodes. Here, separators denote the horizontal or vertical lines in a web page that visually do not cross any node. Based on these separators, the semantic tree of the web page is constructed. A value called degree of coherence (DoC) is assigned for each node to indicate how coherent it is. Consequently, VIPS can efficiently keep related content together while separating semantically different blocks from each other. In particular, the algorithm applies an iterative process started from the subtree within BODY in the DOM structure to build the content structure as follows:

1) **Visual Block Extraction:** This step aims at finding all appropriate visual blocks contained in the current subtree. Normally, every node inside a current node can represent a visual block. However, some "huge" nodes such as TABLE and P may act only for organization purpose and are not appropriate to represent a single visual block. Therefore, in these cases the current node should be further divided and replaced by its children. This process is iterated until all appropriate nodes are found to represent the visual blocks in the web page.

2) **Visual Separator Detection:** When all blocks are extracted, they are put into a pool for separator detection. *Visual separators* are defined as horizontal or vertical lines in a web page that visually cross with no blocks in the pool. An appropriate weight is set to each separator according to elaborately designed patterns (i.e. Distance pattern, Tag pattern, Font pattern and Color pattern) and those with highest weight are selected as the actual separators.

3) **Content Structure Construction:** When the actual separators are detected, visual blocks on the same sides of all the separators are merged and represented as a node in the content structure. The DoC of each node is also defined through similar methods as described in Step 1. After that, each node is checked whether it meets the granularity requirement. For every node that fails, we go to Step 1 on Visual Block Extraction phase again to further construct the sub content structure within the node. If all the nodes meet the requirement, the iterative process is then stopped and the vision-based content structure for the whole page is obtained. The common requirement for DoC is that DoC > PDoC, if PDoC is pre-defined.

Each block in VIPS is represented as a node in a tree. The root is the whole page; inner nodes are the top level coarser blocks, and all leaf nodes consist of a flat segmentation of a web page. The granularity of segmentation in VIPS is controlled by a

predefined degree of coherence (PDoC), which plays a role as a threshold of the most appropriate granularity for different applications (usually set to 5). The segmentation only stops when the DoCs of all blocks are no smaller than the PDoCs. Figure 1 shows the result of using VIPS to segment a sample CNN web page [17].

VIPS is also very efficient. Since we trace down the DOM structure for visual block extraction and do not analyze every basic DOM node, the algorithm is totally top-down. Furthermore, the PDoC can be pre-defined, which brings significant flexibility to segmentation and greatly improve the performance.

Fig. 1. VIPS segmentation of a sample web page

2.3 The Manifold-Ranking Process

The manifold-ranking method [23, 24] is a universal ranking algorithm and it is initially used to rank data points along their underlying manifold structure. The prior

assumption of manifold-ranking is: (1) nearby points are likely to have the same ranking scores; (2) points on the same structure (typically referred to as a cluster or a manifold) are likely to have the same ranking scores. An intuitive description of manifold-ranking is as follows: A weighted network is formed on the data, and a positive rank score is assigned to each known relevant point and zero to the remaining points which are to be ranked. All points then spread their ranking score to their nearby neighbors via the weighted network. The spread process is repeated until a globally stable state is achieved, and all points obtain their final ranking scores.

In our context, the data points are denoted by the blocks in the query page q and the web pages in D_{init}. The manifold-ranking process in our context can be formalized as follows:

Given a set of data points $\chi = \chi_q \cup \chi_{D_{\text{init}}} = \{x_1, x_2, ...x_p, x_{p+1}, ..., x_n\} \subset R^m$, the first p points represent the blocks in the query page q and the rest n-p points represent the blocks in the web pages in D_{init}. Let $f: \chi \to R$ denote a ranking function which assigns to each point x_j ($1 \le j \le n$) a ranking value f_j. We can view f as a vector $f=[f_1, \ldots, f_n]^T$. We also define a vector $y=[y_1, \ldots, y_n]^T$, in which $y_j=1$ ($1 \le j \le p$) for the blocks in q and $y_j=InitScore(d_i)$ ($p+1 \le j \le n$) for the blocks in any web page d_i in D_{init}, where $x_j \in d_i$, $d_i \in D_{\text{init}}$, which means that the initial retrieval score of a web page is used as the initial ranking scores of the blocks in the web page. The manifold-ranking algorithm goes as follows:

1. *Compute the pairwise similarity among points (blocks) using the standard Cosine measure.*
2. *Connect any two points with an edge. We define the affinity matrix W by* $W_{ij}=sim_{cosine}(x_i,x_j)$ *if there is an edge linking* x_i *and* x_j*. Note that we let* $W_{ii}=0$ *to avoid loops in the graph built in next step.*
3. *Symmetrically normalize W by* $S=D^{-1/2}WD^{-1/2}$ *in which D is the diagonal matrix with (i,i)-element equal to the sum of the i-th row of W.*
4. *Iterate* $f(t+1)=\alpha Sf(t)+(1-\alpha)y$ *until convergence, where* α *is a parameter in* (0,1).
5. *Let* f_j^* *denote the limit of the sequence* $\{f_j(t)\}$*. Each block* x_j ($p+1 \le j \le n$) *gets its ranking score* f_j^*.

In the above iterative algorithm, the normalization in the third step is necessary to prove the algorithm's convergence. The fourth step is the key step of the algorithm, where all points spread their ranking score to their neighbors via the weighted network. The parameter of manifold-ranking weight α specifies the relative contributions to the ranking scores from neighbors and the initial ranking scores. Note that *self-reinforcement* is avoided since the diagonal elements of the affinity matrix are set to zero.

The theorem in [24] guarantees that the sequence $\{f(t)\}$ converges to

$$f^* = \beta(I - \alpha S)^{-1} y \tag{1}$$

where β=1-α. Although f^* can be expressed in a closed form, for large scale problems, the iteration algorithm is preferable due to computational efficiency. Usually the

convergence of the iteration algorithm is achieved when the difference between the scores computed at two successive iterations for any point falls below a given threshold (0.0001 in this study).

Using Taylor expansion, we have

$$\begin{aligned} f^* &= \beta(I - \alpha S)^{-1} y \\ &= \beta(I + \alpha S + \alpha^2 S^2 + K) y \\ &= \beta(y + \alpha S y + \alpha S(\alpha S y) + K) \end{aligned} \tag{2}$$

From the above equation, if we omit the constant coefficient β, f^* can be regarded as the sum of a series of infinite terms. The first term is simply the vector *y*, and the second term is to spread the ranking scores of the blocks to their nearby blocks, and the third term is to further spread the ranking scores, etc. Thus the effect of the blocks in the documents is gradually incorporated into the ranking score.

2.4 The Score Fusion Process

The final retrieval score of a web page $d_i \in D_{\text{init}}$ is computed by fusing the ranking scores of its blocks as follows:

$$FinalScore(d_i) = \frac{\sum_{x_j \in d_i} \lambda_j f_j^*}{|d_i|} \tag{3}$$

wher $\lambda_j = \sigma \frac{\text{size of block } x_j \text{ in page } d_i}{\text{distance from the center of } x_j \text{ to the center of screen}}$ empirically measures the importance of the block x_j in the web page d_i [4], where σ is a normalization factor to make the sum of λ_j for blocks in d_i to be 1, i.e. $\sum_{x_j \in d_i} \lambda_j = 1$. The bigger λ_j is, the more important the block is. Note that λ_j can be viewed as a probability that the user is focused on the block x_j when viewing the page d_i. $|d_i|$ represents the number of blocks in the page d_i. This normalization avoids favoring pages with more blocks.

Finally, the web pages in D_{init} are re-ranked according to their final scores and the re-ranked list is returned[2].

3 Experiments

3.1 Experimental Setup

In the experiments, the proposed approach ("MR+Block") is compared with two baseline approaches: "Cosine" and "MR+Page". The "Cosine" baseline does not apply the

[2] Only the top *k* web pages (i.e. the pages in D_{init}) in the original ranked list are re-ranked and the rest web pages in the original ranked list still hold their initial ranks.

manifold-ranking process and directly ranks the web pages by their Cosine similarity with the query page. The "MR+Page" baseline uses the whole web page instead of page block in the manifold-ranking process, and thus it does not apply the steps of page segmentation and score fusion as in the proposed approach.

The Cosine measure is the most popular measure for document similarity based on the vector space model (VSM). After removing all the tags, each web page d is represented by a vector with each dimension referring to a unique term and the weight $w_{d,t}$ associated with the term t is calculated by the $tf_{d,t} * idf_t$ formula, where $tf_{d,t}$ is the number of occurrences of term t in document d and $idf_t=1+log(N/n_t)$ is the inverse document frequency, where N is the total number of documents in the collection and n_t is the number of documents containing term t. The Cosine similarity between two web pages can be defined as the normalized inner product of the two corresponding vectors. Note that other similarity measures can also be explored, but this study focuses only on the widely-used Cosine measure.

To perform the experiments, a ground truth data set is required. As in [11], we built the ground truth data set from the directory hierarchy of Open Directory Project (ODP)[3]. The ODP maintains hierarchical directories with a large number of web pages, and all the web pages within a directory belong to the same category. Document similarity is implicitly encoded in these hierarchical directories, e.g. for a web page in the directory of "\Computers\Programming\Databases", another web page in the same directory is more similar to this page than any web page in the directory of "\Computers\Programming\Graphics". We downloaded 9610 web pages from subdirectories of five top directories: "\Business", "\Computers", "\Recreation", "\Science" and "\Sports". Each web page belonged to a three-level directory, e.g. "\Science\Agriculture\Animals". There are on average about 25 web pages in each directory. We used the VIPS tool[4] for page segmentation. The average number of blocks in a web page is 6.59. We extracted plain texts from the whole web page and each block by removing all tags. Stop words were removed and Porter's stemmer [16] was used for word stemming. 170 web pages were randomly collected as queries in test set. Two kinds of relevance are defined as follows:

1-level relevance: Given a query page q in the directory of "\A\B\C", only the web pages in the same directory of "\A\B\C" are considered as relevant (similar) pages, and all other pages are considered as irrelevant (dissimilar) pages.

2-level relevance: Given a query page q in the directory of "\A\B\C", the web pages in all the subdirectories of "\A\B" (e.g. "\A\B\D") are considered as relevant (similar) pages, and all other pages are considered as irrelevant (dissimilar) pages.

Based on the above relevance levels, two relevance lists for a query page were built respectively and the retrieval performance can be obtained on either relevance list.

The total web pages were considered as the collection for search, a ranked list of 500 web pages was returned in response to each query page based on a specified

[3] http://dmoz.org
[4] http://www.ews.uiuc.edu/~dengcai2/VIPS/VIPS.html

retrieval approach. For the proposed manifold-ranking process, the number of initially retrieved web pages was typically set to 50, i.e. $|D_{\text{init}}|=k=50$.

As in TREC[5] experiments, we used the average precisions at top N results, i.e. *P@5* and *P@10*, as evaluation metrics. The precision at top N results for a query is calculated as follows:

$$P@N = \frac{|C \cap R|}{|R|} \tag{4}$$

where R is the set of top N retrieved documents, and C is the set of similar documents defined above for a given query document. The precision is calculated for each query and then the values are averaged across all queries.

Note that based on two relevance lists, we can obtain two *P@5* values and two *P@10* values, respectively. Usually the precision values based on the 2-level relevance list are higher than that based on the 1-level relevance list.

3.2 Experimental Results

The precision values of the proposed approach ("MR+Block") and two baseline approaches (i.e. "Cosine" and "MR+Page") are compared in Table 1, when the manifold-ranking weight α is set to 0.2 (The weight is tuned on the training set). Seen from Table 1, the proposed approach significantly outperforms the two baseline approaches over three out of four metrics. We can also see that the "MR+Page" baseline achieves higher precision values than the "Cosine" baseline based on 2-level relevance, while it achieves lower precision values than the "Cosine" baseline based on 1-level relevance, which shows that the manifold-ranking process on the granularity of the whole web page can not significantly improve the retrieval performance.

The *P@10* values of two MR-based approaches (i.e. "MR+Block" & "MR+Page") with different manifold-ranking weight α are shown and compared in Figures 2 and 3. Seen from the figures, with appropriate values of the manifold-ranking weight (α <0.5), the proposed approach (i.e. "MR+Block") can always outperform the approach of "MR+Page". Similar trends can be observed over the *P@5* metric and we omit the comparison figures due to page limit. The observations demonstrate that page block is a more appropriate unit than the whole web page in the manifold-ranking process, which can be explained by that a web page usually contains various contents and multiple topics and is not appropriate to be considered as a single unit, while each block represents a single topic with coherent text and thus is a finer granularity for the manifold-ranking process.

Figure 4 explores the influence of the number of initially retrieved documents (i.e. k) on the performances of our proposed approach (i.e. "MR+Block") at two relevance levels. Seen from the figure, when k is larger than 75, the performances almost do not alter any more. This shows that a small number of initially retrieved documents work well in the re-ranking process and it will not significantly improve the retrieval performance by increasing the number of initially retrieved documents.

[5] http://trec.nist.gov

Table 1. Precision values of the proposed approach and baseline approaches

		Baseline1 (Cosine)	Baseline2: (MR+Page)	Our Approach: (MR+Block)
1-level Relevance	P@5	0.515	0.503	0.526
	P@10	0.386	0.385	0.406*#
2-level Relevance	P@5	0.689	0.698	0.721*#
	P@10	0.599	0.618*	0.641*#

(* indicates that the performance change over baseline1-"Cosine" is statistically significant. # indicates that the performance change over baseline2-"MR+Page" is statistically significant.)

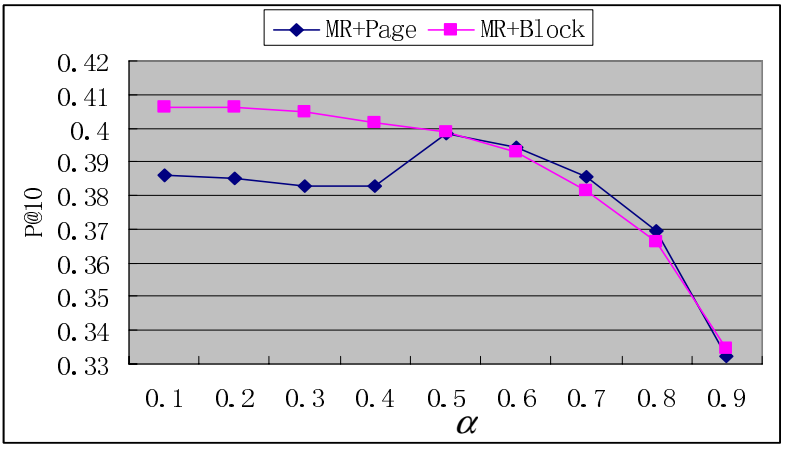


Fig. 2. P@10 comparison of MR-based approaches with different manifold-ranking weights based on 1-level relevance

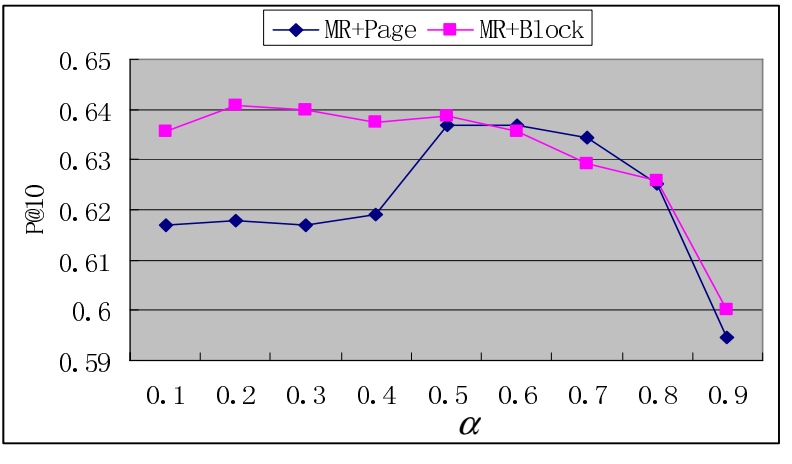


Fig. 3. P@10 comparison of MR-based approaches with different manifold-ranking weights based on 2-level relevance

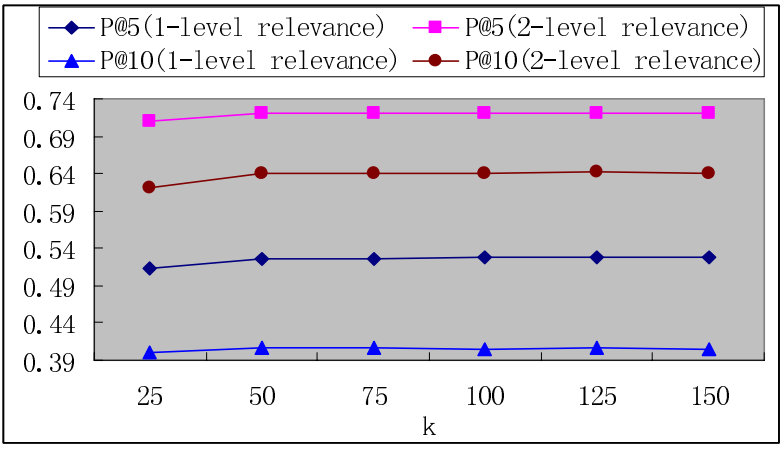


Fig. 4. Performance comparison of the proposed approach with different number of initially retrieved documents

4 Conclusion

In this paper, we propose a novel approach for web page similarity search. The proposed approach re-ranks a small number of initially retrieved web pages based on manifold-ranking of page blocks. The manifold-ranking process can make full use of the relationships among page blocks to improve the retrieval performance. The experimental results demonstrate the favorable performance of the proposed approach.

References

1. Baeza-Yates, R., and Ribeiro-Neto, B. (1999) Modern Information Retrival. ACM Press and Addison Wesley
2. Cai, D., He, X., Li, Z., Ma, W.-Y. and Wen, J.-R. (2004) Hierarchical clustering of WWW image search results using visual, textual and link analysis. In Proceedings of the 12th ACM International Conference on Multimedia
3. Cai, D., He, X., Ma, W.-Y., Wen, J.-R. and Zhang, H.-J. (2004) Organizing WWW images based on the analysis of page layout and web link structure. In Proceedings of the 2004 IEEE International Conference on Multimedia and EXPO (ICME'2004)
4. Cai, D., He, X., Wen, J.-R. and Ma, W.-Y. (2004) Block-level link analysis. In Proceedings of the 27th Annual International ACM SIGIR Conference (SIGIR'2004)
5. Cai, D., Yu, S., Wen, J.-R., and Ma, W.-Y. (2003) VIPS: a vision based page segmentation algorithm. Microsoft Technical Report, MSR-TR-2003-79
6. Cai, D., Yu, S., Wen, J.-R. and Ma, W.-Y. (2004) Block-based web Search. In Proceedings of the 27th Annual International ACM SIGIR Conference (SIGIR'2004)
7. Chen, J., Zhou, B., Shi, J., Zhang, H.-J. and Qiu, F. (2001) Function-based object model towards website adaptation. In Proceedings of the 10th World Wide Web conference (WWW10)

8. Cruz, I.F., Borisov, S., Marks, M. A. and Webb, T.R. (1998) Measuring structural similarity among web documents: preliminary results. In Proceedings of the 7th International Conference on Electronic Publishing, 513–524
9. Dean, J., Henzinger, M. R. Finding related pages in the World Wide Web. In Proceedings of the Eighth International Conference on World Wide Web, 1467-1479.
10. Fogaras, D., Rácz, B. (2004) Scaling link-based similarity search. Technical Report
11. Haveliwala, T. H., Gionis, A., Klein, D., Indyk, P. (2002) Evaluating strategies for similarity search on the Web. In Proceedings of WWW2002, 432-442.
12. Jeh, G. and Widom, J. (2002) SimRank: A measure of structural-context similarity. In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
13. Joshi, S., Agrawal, N., Krishnapuram, R. and Negi, S. (2003) A bag of paths model for measuring structural similarity in web documents. In Proceedings of the 9th ACM SIGKDD Conference, 577–582
14. Kovacevic, M., Diligenti, M., Gori, M. and Milutinovic, V. (2002) Recognition of common areas in a web page using visual information: a possible application in a page classification. In Proceedings of 2002 IEEE International Conference on Data Mining (ICDM'02), Maebashi City, Japan
15. Lin, Z., Lyu, M. R., King, I. (2006) PageSim: a novel link-based measure of web page similarity. In Proceeding of the 15th International World Wide Web Conference
16. Porter, M. F. (1980) An algorithm for suffix stripping. Program, 14(3): 130-137
17. Song, R., Liu, H., Wen, J.-R. and Ma, W.-Y. (2004) Learning block importance models for web pages. In Proceeding of the Thirteenth World Wide Web conference (WWW 2004), 203-211
18. Tombros, A. and Ali, Z. (2005) Factors affecting web page similarity. In Proceedings of ECIR2005.
19. van Rijsbergen, C. J. (1979) Information Retrieval. Butterworths, London
20. Wan, X. (2004) Link-based search of similar pages on the web. Master Thesis. Dalhouse University
21. Xue, G.-R., Zeng, H.-J., Chen, Z. and Yu, Y. (2004) MRSSA: an iterative algorithm for similarity spreading over interrelated objects. In Proceedings of CIKM2004
22. Yu, S., Cai, D., Wen, J.-R. and Ma, W.-Y. (2003) Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In Proceedings of the Twelfth International World Wide Web Conference (WWW2003)
23. Zhou, D., Bousquet, O., Lal, T. N., Weston, J. and SchÖlkopf, B. (2003) Learning with local and global consistency. In Proceedings of NIPS-2003
24. Zhou, D., Weston, J., Gretton, A., Bousquet, O. and SchÖlkopf, B. (2003). Ranking on data manifolds. In Proceedings of NIPS-2003

Design and Implementation of Preference-Based Search

Paolo Viappiani and Boi Faltings

Artificial Intelligence Laboratory (LIA)
Ecole Polytechnique Fdrale de Lausanne (EPFL)
1015 Lausanne, Switzerland
paolo.viappiani,boi.faltings

Abstract. Preference-based search is the problem of finding an item that matches best with a user's preferences. User studies show that example-based tools for preference-based search can achieve significantly higher accuracy when they are complemented with suggestions chosen to inform users about the available choices. We present `FlatFinder`, an implementation of an example-based tool and discuss how such a tool as well as suggestions can be efficiently implemented even for large product databases.

1 Introduction

People frequently use the world-wide web to search through a large collection of items. The most common search facility available on the web is based on a form that is directly mapped to a database query and returns a ranked list of the most suitable options. The user has the option to return to the initial page and change his preferences and then carry out a new search. This is the case for example when searching for flights on the most popular travel web sites[1][2]. Such tools are only as good as the query the user formulates. A study [1] has shown that among the users of such sites only 18% are satisfied with their final choice.

In most cases, users do not know exactly what they are looking for: they might consider different trade-offs or they might even have conflicting desires about the features the item should have. In fact psychological studies have shown that people construct their preferences [7] while learning about the available products. Therefore preference-based search should also help users in formulating accurate preferences.

We believe that the key issues for implementing successful preference-based search systems are:

1. *preference modeling:* the formalism chosen to model preferences
2. *preference elicitation:* how to acquire or learn preferences from the user
3. *usability:* ease of use of the interface
4. *scalability:* do the algorithms used by the tool scale up for large databases?

[1] `http://www.travelocity.com/`
[2] `http://www.expedia.com`

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 72–83, 2006.

The first point, *preference modeling*, requires the designer to choose from possible preference representations. The user expresses the preferences using an interface by qualitative statements; these are then translated into the internal preference model.

Preference elicitation is crucial. Decision theory [4] provides a method that guarantees perfect decision accuracy by first eliciting a model of the user's preferences through a series of questions, and then determining the optimal choice based on this model. However, even for simple items such as cameras, eliciting such a model would require hundreds of questions, and few users would be ready to undergo such a lengthy process. Therefore, we need to provide users a concise way to express preferences that would be *usable*. At the same time we would benefit from approaches able to mitigate the inaccuracies inevitably created when translating a user's qualitative statement into a quantitative model of preferences suitable for ranking items.

First, we describe the *example-critiquing* approach to preference based search. We then discuss how to model preferences and to compute suggestions. Finally we will discuss how to implement scalable tools and present a prototype for student accommodation search called *FlatFinder*.

1.1 Example-Critiquing

People are usually not able to state preferences up-front and behavioral decision theory studies [7] have shown that people *construct* their preferences as they see the available options. The elicitation through an interaction based on examples is therefore faster and often more precise than a comprehensive preference

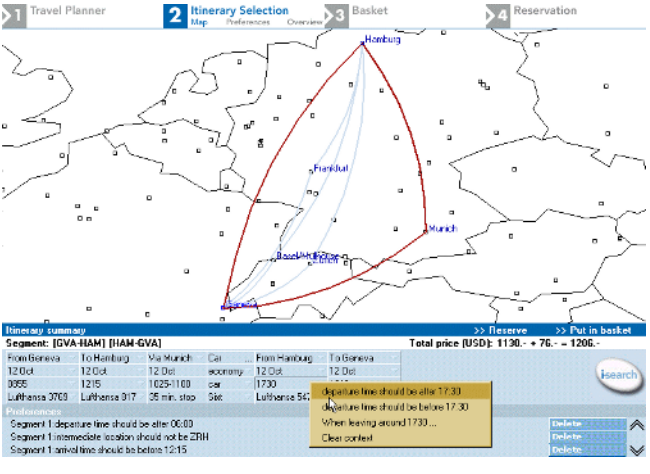


Fig. 1. Isy-travel is an example-critiquing tool for planning business-trips. Here the user is planning a one-day trip from Geneva to Hamburg. The preference model is shown at the bottom, and the user can state additional preferences by clicking on features of the shown example.

elicitation procedure; the latter may construct its model based on an incoherent set of answers!

Figure 1 shows Isy-travel, a commercial tool for business travelers [9]. Here, the user is shown examples of options that fit the current preference model well. The idea is that an example either is the most preferred one, or there is some aspect in which it can be improved. Thus, on any of the examples, any attribute can be selected as a basis for critiquing. For instance, if the arrival time is too late, then this can be critiqued. The critique then becomes an additional preference in the model.

This form of example critiquing has been proposed by various researchers, including the ATA system of Linden et al. [6], SmartClient [9], and more recently dynamic critiquing [13].

The advantage of such a system in the elicitation of preferences is that examples help users reason about their own preferences, revise them if are inconsistent, have an idea of which preferences can be satisfied and make trade-off decisions.

Example-critiquing has been shown to provide a means for effective preference elicitation. Pu and Li [8] have shown that example-critiquing with its tradeoff support enables consumers to more accurately find what they want (with up to 57% increase in accuracy) and be more confident in their choices, while requiring a level of cognitive effort that is comparable to simple interfaces such as a ranked list [11].

2 Incremental Preference Model Acquisition

In example-critiquing, a preference is stated as reaction to displayed options. A critiques can either be *negative reactions* to the options shown, when none of them satisfy the preference, or *positive reactions*, when an option satisfies the preference.

For instance, if the tool shows the user examples that all arrive at London Stansted airport, and she requests to land in Heathrow, that critique would be a *negative reaction*. If the system indeed showed one flight landing in Heathrow, by stating that preference she would be *positively* reacting to the shown examples.

If certain preferences are missing from the current model of the user, the system provides examples that do not satisfy those unknown preferences. If the user is aware of all of her preferences, she can realize the necessity to state them to the system by posting what we have called a *negative reaction* critique. However our intuition is that this is not always the case, because the user might not not know all the available options. Moreover, stating a preference costs some user effort (in our prototype, 2 selections and 2 clicks) and rationally she would do that only if she perceives this as beneficial.

To use a metaphor, the process of example-critiquing is hill-climbing: the user states preferences as long as he perceives it as bringing to a better solution. However, the process might end in a local optimum; a situation in which the user can no longer see potential improvement. For example, a user looking for a notebook computer might start looking for a low price, and thus find that all

models weigh about 3 kg. Since all of the presented models have about the same weight, he or she might never bother to look for lighter models. This influence of current examples prevents the user from refocussing the search in another direction; this is known as the *anchoring effect* [14].

For these reasons we display two sets of examples:

- **candidate examples** that are optimal for the preference model, and
- **suggested examples** that are chosen to stimulate the expression of preferences.

We conducted user studies to evaluate the decision accuracy of example critiquing with and without suggestions. We found [12] that with the aid of suggestions, users state more preferences and, more importantly, achieve a much higher decision accuracy (up to 70%). Subsequently, we looked at the logs of the user study to check frequency of the different type of critiquing described before. In most of the cases (55%) a preference is stated as positive critiques.

2.1 Suggestions: Diversity and Lookahead Principle

The importance of the diversity of the example shown was recognized by Linden, S. Hanks and N. Lesh ([6]) who explicitly generated examples that showed the extreme values of certain attributes, called *extreme examples*. However, an extreme example might often be an unreasonable choice: it could be a cheap flight that leaves in the early morning, a student accommodation where the student has to work for the family, an apartment extremely far from the city. Moreover, in problems with many attributes, there will too many extreme or diverse examples to choose from, while we have to limit the display of examples to few of them.

We assume that user is minimizing her own effort and will add preferences to the model only when she can expect them to have an impact on the solutions. This is the case when:

- she can see several options that differ in a possible preference, and
- these options are relevant, i.e. they could be reasonable choices, and
- these options are not already optimal, so a new preference is required to make them optimal.

In all other cases, stating an additional preference is likely to be irrelevant. When all options would lead to the same evaluation, or when the preference only has an effect on options that would not be eligible anyway, stating it would only be wasted effort. This leads us to the following *look-ahead* principle as a basis for suggestion strategies:

> Suggestions should not be optimal under the current preference model, but should provide a high likelihood of optimality when an additional preference is added.

We stress that this is a heuristic principle based on assumptions about human behavior that we cannot formally prove. However the high decision accuracy

achieved in user studies [12] is an important motivation. Furthermore, examining the incremental critiques more in detail in the user studies ([15]), we found that in most cases there was a displayed example that became optimal because of the addition of a preference. This provides another clue of the validity of the principle.

In the following sections we present our model of treating preferences, the suggestion strategies, the implementation and the scalability issues.

3 Theoretical Model

3.1 Modeling Items and Preferences

We assume that items are modeled by a fixed set of m attributes that each take values in associated domains. Domains can be *enumerated*, consisting of a set of discrete elements, or *numeric*. In this paper, we consider preferences on individual attributes and independent of one another (i.e. we do not consider conditional preferences). A preference r is an order relation of the values of an attribute a.

For a practical preference-based search tool, it is convenient to express preferences in a concise way. We consider total orders (each pair is comparable) and express them by a numerical cost function c, $d_k \rightarrow \Re^+$, that maps a domain value d_k of an attribute a_k to a real number. A preference always applies to the same attribute a_k; we use the notation $c_i(o)$ to express the cost that the function assigns to the value of option o for that attribute.

Whenever o_1 is preferred to o_2 according to preference i, the first will have lower cost (for preference i) than the second: $c_i(o_1) < c_i(o_2)$.

An overall ranking of options can be obtained by combining the penalty functions for all stated preferences. Some researchers [3] have proposed the use of machine learning algorithms for finding the best aggregate function for a particular user. In our systems, we combine them using a weighted sum, which corresponds well to standard multi-attribute utility theory [4]. Thus, if $\mathcal{R}_c = \{c_1, .., c_s\}$ is the set of the cost functions of all preferences that the user has stated, we compute the cost $C(o) = \sum_{c_i \in \mathcal{R}_c} w_i \cdot c_i(o)$. Option o_1 is preferred over option o_2 whenever it has a lower cost, i.e. $C(o_1) < C(o_2)$.

The user states preferences in a qualitative way (for example "the price should be less than 500 dollar"). We map these qualitative statements into parameterized functions that are standardized to fit average users. These are chosen with respect to the application domain.

Similar models for modeling preferences in databases have been proposed in [5]. Preference modeling for example-critiquing is discussed in more detail in [10].

3.2 Model-Based Suggestion Strategy

In [12] we proposed different strategies that use the concept of pareto-optimality to implement the look-ahead principle stated in the introduction: suggestions

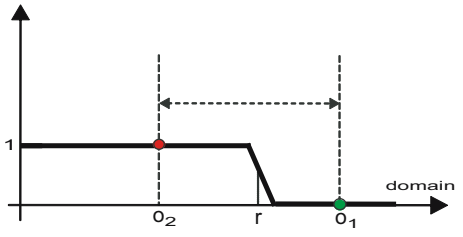


Fig. 2. For an ordered attribute, a new preference will prefer o_1 over o_2 if the reference value r falls between the values of the attribute, and the preference is of the right polarity

should not be optimal yet, but have a high likelihood of becoming optimal when an additional preference is added. We call them *model-based* suggestion strategies because they specifically choose examples to stimulate the expression of additional preferences based on the current preference *model*.

We model preferences by standardized functions that correctly reflect the preference order of individual attribute values but may be numerically inaccurate. When generating suggestions, we would like to use a model that is not sensitive to this numerical error. Pareto-optimality is the strongest concept that would be applicable regardless of the numerical details of the penalty functions.

An option o is **dominated by** another option $\bar{o}$ (equiv. $\bar{o}$ dominates o) if

- $\bar{o}$ is not worse than o according to all preferences in the preference model: $\forall c_i \in R : c_i(\bar{o}) \leq c_i(o)$
- $\bar{o}$ is strictly better than o for at least one preference: $\exists c_j \in R : c_j(\bar{o}) < c_j(o)$

An option o is **pareto-optimal** if it is not dominated by any other option. The dominance relation is a partial order of the options that we will denote with the $\succ$ operator; Pareto-optimal options can also be seen as the set of maximal options with respect to the dominance relation.

In our applications, users initially state only a subset R of their true preference model $\overline{R}$. When a preference is added, dominated options with respect to R can become Pareto-optimal. The look-ahead principle can be formulated as follows: an ideal suggestion is an option that is Pareto-optimal with respect to the full preference model $\overline{R}$, but is dominated in R, the partial preference model.

The model based suggestions try to guess the chance that a (dominated) option has to become Pareto optimal. To become pareto-optimal when a new preference is added to the model, an option has to be strictly better than any dominating option with respect to this new preference.

We use a heuristic estimation of the probability that a hidden preference on attribute a_i makes o better than o^+ according to that preference, hence escaping the dominance relation. Such a heuristic considers the difference between the attribute values: the higher this difference, the more likely that new preference will make the option preferred. The reasoning is illustrated in Figure 2. The chances that a new preference will treat o_1 and o_2 differently depends on the difference

between their values. Assuming that the shape of such a penalty function is a step function with sharp increase from 0 to 1, if the reference point falls at any point with equal probability, the chance of breaking the dominance is directly proportional to this difference.

4 Algorithms and Scalability

In this section we analyze the algorithms to compute the candidates and suggestion examples and their complexity; we consider the problem of scalability and show some faster approximations.

4.1 Generation of Candidates

As said earlier, candidates are the best examples corresponding to the current set of preferences R. The generation of candidates can be addressed by a a *top-k query* to the database. The set of options retrieved $\{o_1, .., o_k\}$ is such that $C(o_1) \leq C(o_2) \leq .. \leq C(o_k)$ and for any other option $\bar{o}$ in the database $C(\bar{o}) \geq C(o_k)$.

While the trivial approach would compute the score of each option in the database, the *Fagin* algorithm [2] can do this with middleware complexity $O(N^{(m-1)/m}k^{1/m})$ where m is the number of attributes and k the number of candidates we want to generate. This algorithm can be applied to a different aggregate function as long as it satisfies some properties (monotonicity, strictness).

4.2 Generation of Suggestions

The model based strategy requires the analysis of the dominance binary relation, by making a series of pairwise checks between the options of the catalog. Each one evaluates two options, say o_1 and o_2, to determine whether o_1 dominates o_2, o_2 dominates o_1, they are equally preferred for all the preferences, or they are not comparable (i.e. there is no dominance in either direction). This is done by considering iteratively each of the preferences and comparing o_1 and o_2 for at most m comparisons (as soon as two preferences give opposite order of o_1 and o_2, they are not comparable), where m is the number of preferences, so the complexity is $O(m)$.

Since the dominance relation is a partial order, we can exploit asymmetry and transitivity to save some of the pairwise checks, however in the worst case we have to make $n(n-1)/2$ checks. So the complexity of the complete dominance analysis is $O(n^2m)$.

The algorithm for model-based suggestions is presented in the Algorithm 1. *Update* is responsible for updating the value for the estimation of the probability $p(o, a_i)$ of becoming Pareto optimal given that the missing preference is on a_i; its precise definition depends on the particular assumptions on the possible preferences.

In the *probabilistic* strategy, the update multiplies the current value by $w_{a_i} * \delta(o, o_d)$, where $\delta(o, o_d)$ is the heuristic estimation presented in the previous section (based on the normalized distance between the attribute values) and w_{a_i} is

a weight representing the probability that there is a preference on that attribute. Intuitively the more dominating options there are, the more $p(o, a_i)$ decreases. For more details, refer to Viappiani et. al. [16].

In another model-based strategy, the *attribute* strategy, we assume that the preferences that the users can state are only of the kind `LessThan` or `GreaterThan` (the user cannot express preferences for a value in the middle), therefore we check whether the current option has a value that is either smaller or bigger than any value of the dominating options: only in this case can a preference break all the dominance relations simultaneously. In this strategy, `update` will take the minimum of the absolute values of the $\bar{\delta}$, and returns 0 if they are of different sign.

Algorithm 1. Model-based suggestions(int n)
δ heuristics based on the normalized differences

for all $option \in OPTIONS$ **do**
  $p(option) = 0$
  for all $a_i \in A_u = \{$attributes with no preferences$\}$ **do**
    $p(option, a_i) = 1$ //contribution for attribute a_i
    for all $o_d \in O_D = \{o_d \in O : o_d \succ option\}$ **do**
      //we iterate over the set of options that dominates o
      $\bar{\delta} \leftarrow \delta_{a_i}(o, o_d)$
      $p(option, a_i) = \text{update}(p(option, a_i), \bar{\delta})$
  $p(option) = 1 - \prod_{a_i}(1 - p(option, a_i))$
$suggList \leftarrow$ order options according p
return first n options in $suggList$;

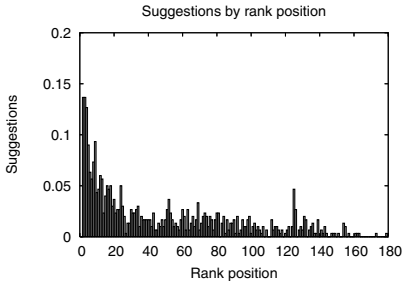
Our model based strategies have complexity $O(nmd)$, where m is the number of attributes and d is the number of dominating options. It is difficult to calculate an average value for d in function of n, because it depends on the data and correlation between attribute values. While generally the set of dominating options is much smaller than the set of options, in the worst case they can be a linear fraction of n. Sorting the options according to the resulting probability (to select the best n) costs $nlogn$ in term of complexity.

The overall complexity is $O(n^2)$: while this complexity was not a problem for our prototype, we expect it to be more problematic as the item collections grow. Approximations will be necessary for large databases.

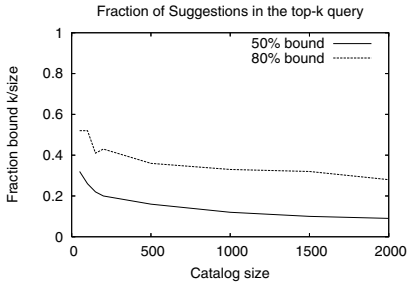
We propose the following approximations:

- to select suggestions from the top k options
- to replace pareto-dominance with utility dominance
- to assume dominating options as a fixed number of options at the top

Approximation 1: Select Suggestions from the Top. From experimental tests, it emerged that suggestions are not evenly distributed according to their current costs, but they are often at the top. This is not surprising: since suggestions are options that should be reasonable, we will expect them to be not too far in the ranking from the current candidates.



(a) The position of suggestions in the overall ranking; repeated queries are performed on a database of student accommodations with 187 options, 3 model-based suggestions are retrieved.



(b) For different size of catalogs, the bound k required to guarantee that respectively 50% and 80% of the suggestions are in the top k positions.

Fig. 3. Suggestions are concentrated in the top position in the overall ranking; this is plausible because suggestions must consider the preferences in the current model. On average, 50% of the suggestions are in the top 16% of the overall ranking, 80% in the top 47%.

In the actual use of the preference based search tool on the apartment database used by `FlatFinder` (our current prototype for example-critiquing with suggestions), in more than the 80% of the cases suggestions are among the top half of the options (ordered according to the optimal query) and in more than the 60% in the top quarter of the options.

We considered how to mitigate the impact of very large data sets by only considering suggestions in the subset of the catalog that contains the *top-k* options for the preference query. Since an option can be dominated only by options that have lower cost $C()$, the preliminary phase of dominance analysis is also simplified. We can run the algorithms presented before on a database containing only the options returned by the *top-k* query with a given bound k_b. Using this approximation, the computation is going to do $O(k_b^2)$ checks.

We ran several generations of suggestions with our database of student accommodation and random databases of different size. We look for the *bound* k_b for which a certain fraction of suggestions lie in the top k_b positions in the ranking. Choosing the bound k_b in this way, we can guarantee a limited decrease in the quality of suggestions that does not depend on the size of the catalog.

Figure 3b shows that the bound grows more slowly as the catalog size increases. Using interpolation we found that the bound for ensuring 50% of suggestions approximately increases as the function $2.1n^{lg(1.5)}$. Substituting this function into k_b^2, we found that the complexity is $O(n^{1.2})$, that is significantly lower than n^2. Empirically we found that these bounds for random data are higher than with real data, with correlation between the attributes values. Figure 3a shows the rank position of the suggestions retrieved on a real database.

This method results in a significant improvement in complexity. However, such complexity might be still costly for some applications.

Approximation 2: Utility Dominance. Instead of pareto dominance, we can use other forms of dominance. In particular, we might use the total ordering established by the combination function defined in the preference modeling formalism, such as a weighted sum $C()$. We call this *utility-domination*, and the utility-optimal option is the most preferred one. The utility dominance approximation consists of checking the probability of breaking utility dominance. The advantage is that the dominating set is easily computed: once we have the ranking for the current preferences, the utility-dominating set will be composed by all the options prior in the ranking.

However, we still have to make, for each option, a comparison to its utility-dominators. These are 1 for the first option in the rank, 2 for the second, and n for the last. So, even if the calculation of dominators is faster, this strategy is still quadratic $O(n^2)$. We have a total complexity of $O(n^2)$, of the same order as the complete method, even if faster in practice.

Approximation 3: Top-Domination. This approximation strategy looks at the probability of breaking the dominance with the options at the top of the ranking. This method requires testing each option against a fixed number k_d of best options to check if their dominance could be broken. We don't have to look for the dominating set: these are always the options at the top in the ranking. For each option we need to make a constant number of comparisons to these top options, on m attributes. Therefore this method is much faster with a complexity $O(nm)$.

The difference with approximation 1 is that there we tested only a subset of the options against their actual dominating set, while here we test all the options against a constant set of dominators.

4.3 Evaluation of the Approximations

We evaluated the approximated techniques on the ability to find Pareto-optimal options considering the unknown preferences. We ran simulations with the

Table 1. The hit-rate according to our look-ahead principle for the different approximation strategies and the complete method of generation of suggestions. For comparison, we show the case in which we simply display more candidates.

method	note	hit rate
model-based suggestions		77%
approximation 1	$k_b = 50\%$	74%
	$k_b = 25\%$	72%
	$k_b = 12.5\%$	64%
approximation 2	utilitarian	66%
approximation 3	k_d=3	23%
	k_d=6	31%
	k_d=12	49%
random suggestions		12%
no suggestions	more candidates	31%

apartment database and with randomly generated data. We generated random models consisting of 2 to 7 preferences and we calculated the number of times the method successfully fulfilled our lookahead strategy (the *hit rate*) when one preference is not stated yet (i.e. the frequency of finding, among the suggestions selected, an option that became Pareto-optimal by the addition of one among the missing preferences).

The first approximation gives almost the same hit rate as the complete search, so it could often be a reasonable choice. In cases in which a very low complexity is required, the third approximation with $k_d = 12$ can be considered: it is much faster and still achieves a significantly greater hit rate that just showing more options from the top. Utility-dominance does not provide a good balance between computation benefit and losses and would rarely be the method of choice.

5 Conclusions

Preference-based search is a ubiquitous problem on the web. We have presented tools based on examples that can achieve higher decision accuracy than the traditional form filling approach. User studies show that such tools can greatly help the user, especially when they integrate the display of suggestions that make the users aware of possible choices and stimulate the preferences expression. The principle we follow is that good suggestions are items that become optimal when other possible preferences are considered. Our suggestion strategies are based on the concept of Pareto optimality.

However few web sites currently support preference based search. The main issue is that such personalized search tools are hard to implement for large databases and user populations. To overcome this problem, we presented three approximations that simplify the computation, making the generation of suggestions scale.

The first approximation considers possible suggestions only in the top options returned by a *top-k query*. By reducing the considered options to one fourth, it can maintain a hit-rate (72%) close to the optimum with a significant reduction of the number of pairwise checks required in the computation. The second considers utility dominance instead of the standard dominance relation, simplifying the computation of dominating options. However, it has a lower hit-rate (66%).

Finally, the third considers as dominating options the ones with lowest cost. It can be a reasonable choice for large databases as it achieves a fast computation of suggestions with a complexity of $O(nm)$ and provides a hit rate of 49%.

References

1. M. S. D. W. Equity. Transportation e-commerce and the task of fulfilment, 2000.
2. R. Fagin. Fuzzy queries in multimedia database systems. In *PODS '98*, pages 1–10, USA, 1998. ACM Press.
3. S.-w. H. Hwanjo Yu and K. C.-C. Chang. Rankfp: A framework for supporting rank formulation and processing. In *ICDE 2005*, pages 514–515, 2005.

4. R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* John Wiley and Sons, New York, 1976.
5. W. Kiesling. Foundations of preferences in database systems. In *VLDB 2002*, pages 311–322, 2002.
6. G. Linden, S. Hanks, and N. Lesh. Interactive assessment of user preference models: The automated travel assistant. In *User Modeling '97*, 1997.
7. J. Payne, J. Bettman, and E. Johnson. *The Adaptive Decision Maker*. Cambridge University Press, 1993.
8. P. Pu and L. Chen. Integrating tradeoff support in product search tools for e-commerce sites. In J. Riedl, M. J. Kearns, and M. K. Reiter, editors, *ACM Conference on Electronic Commerce*, pages 269–278. ACM, 2005.
9. P. Pu and B. Faltings. Enriching buyers' experiences: the smartclient approach. In *SIGCHI conference on Human factors in computing systems*, pages 289–296. ACM Press New York, NY, USA, 2000.
10. P. Pu and B. Faltings. Decision tradeoff using example-critiquing and constraint programming. *Constraints: An International Journal*, 9(4), 2004.
11. P. Pu and P. Kumar. Evaluating example-based search tools. In *ACM Conference on Electronic Commerce (EC'04)*, 2004.
12. P. Pu, P. Viappiani, and B. Faltings. Increasing user decision accuracy using suggestions. In *CHI 2006*, April 2006.
13. J. Reilly, K. McCarthy, L. McGinty, and B. Smyth. Dynamic critiquing. In P. Funk and P. A. González-Calero, editors, *ECCBR*, volume 3155 of *Lecture Notes in Computer Science*, pages 763–777. Springer, 2004.
14. A. Tversky. Judgement under uncertainity: Heuristics and biases, 1974.
15. P. Viappiani, B. Faltings, and P. Pu. The lookahead principle for preference elicitation: Experimental results. In *Seventh International Conference on Flexible Query Answering Systems (FQAS)*, 2006.
16. P. Viappiani, B. Faltings, V. Schickel-Zuber, and P. Pu. Stimulating preference expression using suggestions. In *Mixed-Initiative Problem-Solving Assistants*, volume FSS07-05 of *AAAI Fall Symposium Serie*, pages 128–133. AAAI, 2005.

Topic-Based Website Feature Analysis for Enterprise Search from the Web*

Baoli Dong[1,2], Huimei Liu[3], Zhaoyong Hou[3], and Xizhe Liu[2]

[1] Department of Mechanical Engeering, Taiyuan University of Science and Technology, Taiyuan, China
[2] Institute of Manufacturing Engineering, Zhejiang University, Hangzhou, China
[3] School of Science, Taiyuan University of Technology, Taiyuan, China
tydbl@hotmail.com, hmliu@tyut.edu.cn

Abstract. Efficient and accurate enterprise search is a challenging and important problem for specified resources available on the web. Domain-specific enterprise websites are similar in the topic structures and textual contents. Considering the semantic information of website content terms, a novel website feature vector modelling method representing website topic were proposed on the basis of vector space model. The feature vector elements integrated textual semantic information about topic content and structure information through different semantic terms and weighting schema respectively. The contrast recognition performances demonstrate that this feature analysis approach to website topic gives full potentials for specific enterprise web search.

1 Introduction

Enterprise search from the web has gained a prominent status in enterprise information systems and e-commerce ones due to improving the accuracy and service quality of specific resource discovery. Specific website search is typically regarded as the binary classification or filtering problem [1]. That the actual complex web environment may lead to a problem: the accuracy and efficient of topic feature analysis. Therefore, the topic feature extraction, modelling and classification algorithm are important for website classification.

Vector space model (VSM) is a kind of term-based feature vector description. The similarity degree between feature vectors is determined by vector distance. In this way, the simple and direct approach to website classification is to take the whole site content as a super web page, and represented with a VSM-based single feature vector [2], however this method performs poorly [3]. Moreover, every topic included in the site can be predefined as a dimension of feature space. To each topic, the element value of topic frequency vector (TFV) represents the number of pages within the site having that particular topic [4]. Thus, TFV represents website topic feature. This needs to analyze the topics of all pages s in a site and is very time consuming.

* The work was supported partially by the Natural Science Foundation of China (No. 60374057) and Key Program of the Ministry of Education of China (No.211CERS-8).

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 84–89, 2006.

The websites are also viewed as the reticular directed graph or hierarchy tree models composed of nodes and links. The node is the web page or the smaller information unit. The link is the website internal hyperlink, which has certain semantic structure information defined through texts, pictures, etc. For example, viewing the website as a labeled tree where the labels are drawn from a set of page classes [4], utilizing the website internal physics and logic link structure to merge different page topics and determine website topic [5], exploiting a multiscale tree model to describe internal link structure for website classification [6]. However, the complicated statistics and computations to website link structure limit the practical application of these methods above mentioned.

The website content and structure characteristics reflect its topic. To specific sites, our solution is employing term-based hybrid feature vector to represent websie topic. The relevant topic feature model, algorithm and experiments are introduced in the rest of the paper briefly.

2 Key Technologies of Website Topic Analysis

2.1 Website Topic Feature Vector(WTFV)

Website topic information is mainly hidden in the main body structure and content, and the kind of website structure contains the potential characteristic correlated with its topic. The contents of same topic sites are similar to some degree, and the internal hyperlink structure also exist similarities obviously. Anchor texts of navigation toolbars or menus in homepages are indicative for the structure information. Different topic sites have respective structure feature terms represented by anchor texts. As the top-level page and content index page, the homepage contains abundant structure characteristic information. Anchor texts in navigation toolbars or menus of homepages are indicative for the website topic feature. An example homepage of manufacturing enterprise website is illustrated with Figure 1.

From the point of view of text, the content and structure features are separate. the structure portion is denoted through hyperlink texts, and the content portion

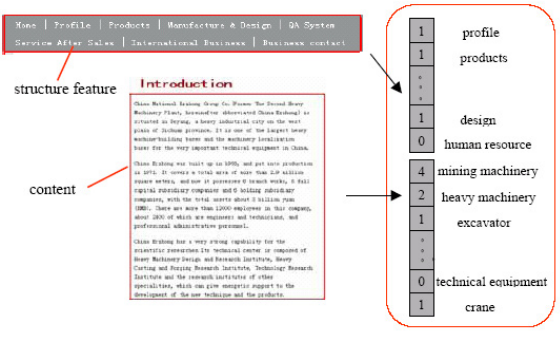


Fig. 1. Example of text information representation for website topic feature

is denoted by text keywords of the main body content. The website topic feature vector can be denoted as following

$$\boldsymbol{V} = \{w_1, \ldots, w_i, w_1^{'}, \ldots, w_j^{'}\} = \{w_1, w_2, \ldots, w_n, \} \quad \mathrm{n} = i + j \tag{1}$$

Here, vector elements of WTFV consist of a union of two types: structure elements w_i and content ones $w_j^{'}$. They are weight values of structure feature term t_i and content feature term $t_j^{'}$ respectively, and i, j is the dimension of structure and content feature set. t_i covers link structure information indicated by anchor texts like "*profile* ","*product*", "*after service*", "*news*", etc. w_i is determined by Boolean function to represent structure characteristics, namely let w_i present 1 if occurs in sites and 0 otherwise. $t_j^{'}$ is about textual content information represented by content keywords, which can be achieved through the domain topic thesaurus and mutual information through statistic samples. Taken the mechanical manufacturing enterprise site in Figure 1 for instance, the relevant major keywords could include "*mining machinery*", "*heavy machinery*", "*excavator*", "*crane*", etc, and which are assigned by TF-IDF weighting schema considering term frequency and the term significance throughout all samples.

2.2 Classification Algorithm Selection

The training sample set of website classification is relatively smaller. Support vector machine(SVM) can solve the supervised learning problems that are lack of learning samples [7]. Meanwhile, The idea of SVM is constructing the maximal margin hyperplane as optimal hyperplane of sample classification in the high-dimensional space like document and website. The hyperplane separates as far as possible two kinds of samples, simultaneously uses the maximization of minimum example distances as classification criteria to obtain two optimal subsets. So we choose SVM algorithm for website classification firstly.

The input feature vector can be defined as $\{\boldsymbol{V}_1, \ldots, \boldsymbol{V}_m\}$, SVM normalizes the classification planar equation to the best classification decision function

$$f(x) = sgn(\sum_{support} \alpha_i^* y_i K(w_i, w) + b^*) \tag{2}$$

Here α_i^* is the globally optimal solution of Lagrange multiplier. The SVM-based classification plane is only obtained through directly calculating Kernel function $K(w, w_i)$. To our experiments, RBF is selected as the kernel function

$$K(w, w_i) = \exp(-c\frac{|w - w_i|}{\sigma^2}) \tag{3}$$

Furthermore, we also adopt the centriod-based calssification algorithm [8]. Namley, given a set of topic samples as template set and the corresponding WTFV vector set, the arithmetic mean of this vector set is taken as the center vector $\boldsymbol{V}_{\mathrm{c}}$ that represents topic template set.

$$\boldsymbol{V}_{\mathrm{c}} = \frac{1}{m}\sum_{k=1}^{m} \boldsymbol{V}_k \tag{4}$$

To a new website with its feature vector $\boldsymbol{V}$, the similarity degree between $\boldsymbol{V}$ and center vector $\boldsymbol{V}_{\mathrm{c}}$ is commonly measured using the cosine function, given by

$$Sim(\boldsymbol{V}_i, \boldsymbol{V}_j) = \frac{\boldsymbol{V}_i \cdot \boldsymbol{V}_j}{|\boldsymbol{V}_i| \times |\boldsymbol{V}_j|} = \frac{\sum_{k=1}^{n} w_{ik} * w_{jk}}{\sqrt{(\sum_{k=1}^{n} w_{ik}^2)(\sum_{k=1}^{n} w_{jk}^2)}} \tag{5}$$

Whether the website topic is coincident with the predefined topic can be determined by similarity threshold.

3 Experiments

3.1 Analytical Performance Experiment of Structure Feature

We study on the analytical capacity of structure feature added in WTFV. The experimental data set includes two kinds: manufacturing enterprise site and manufacturing-topic general portal. Both are basically uniform in the topic content. Therefore, the vector composition takes structure features as the majority of feature elements. We divide the structure feature terms into two parts: one is related to manufacturing enterprise sites(referenced by Figure 1); the other is related to general portals like"*product list* ","*exhibition*", "*open tender*", "*trade news*", etc. The feature dimension of every part is the same, and the remainders are content features. Feature vectors of test samples are set up through web search, parsing and feature extraction. The cosine measures between feature vectors are calculated through formula 5. The similarity matrix between samples is shown in Figure 2. Meanwhile, to the same test data set, we implement the similar experiment only considering content feature terms. The dimension of content features is equal with the former and the weighting way is TF-IDF. The similarity matrix between samples in this case is as Figure 3 shown.

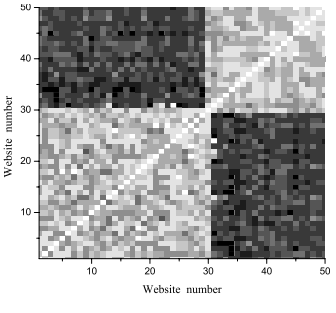


Fig. 2. WTFV-based similarity matrix

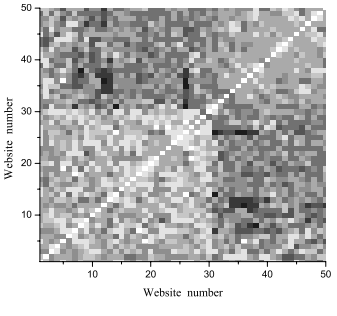


Fig. 3. Content-based similarity matrix

In the grey-scale map, different grey levels represent cosine distances between feature vectors. Although content-based key terms of two kinds of sites are very

similar, the genre of website is discerned conveniently through similarity measures after taking WTFV model. While the websites are represented with content features alone, the contrast of similarity value between them is not obvious. Therefore, structure feature is very effective in this situation where the topics of websites are difficult to analyze through text contents. This is especially significant for subject subdivision of specific website classification.

3.2 Contrast Experiments of Website Recognition

The data set contains manufacturing-topic positive samples, and the other negative samples consist of government sites, media portals and non-manufacturing sites selected randomly from subject lists in general portals like Sohu. All websites have independent domain names. Moreover, to guarantee relevant degree between object pages and website topic, the search depth of URL is defined as 1 to 2 levels of link hierarchy structure in the whole site and the total size of pages downloaded is limited.

In the contrast experiments, three approaches to website classification are taken considering different feature models and classification algorithms. The VSM-Centroid approach means that the feature model only considers the key terms of main body text in websites and determine the topic measure through cosine similarity threshold functions. The WTFV-Centroid approach is different from the former adopting WTFV as website topic feature model and the centroid classification alogorithm. And WTFV-SVM means the website topic analysis employs WTFV and SVM algorithm. The topic recognition results about *recall*, *precision*, F performance indexes are shown as Figure 4.

We find that the topic recognition performance of WTFV-SVM is superior to the others. It only needs to download a small amount of web pages and the feature space is relatively small, but still achieves higher classification accuracy. The structure characteristics considered in vector space model make for improving the website topic analysis and web search performance for specific sites. The trade-off between the size of feature space and the ability of feature description is a big problem for feature vector modelling. WTFV solves this problem well through adding semantic structure feature elements.

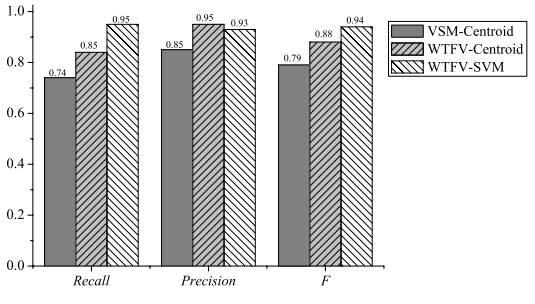


Fig. 4. Classification results of domain sites by different models and algorithms

4 Conclusion

In this paper, we proposed a website feature modelling method for enterprise search considering website topic structure and content semantic text information. Such inherent structure information is simplified as vector elements by link texts. Feature elements have different semantic information and strengthen the website topic description ability. Such vector expression is direct and succinct. It simplifies the feature description and avoids the unfavorable factors in the practical enterprise website search and classification. For instance, the complicated link structure analysis and computation. The experiment results show that this model and relevant techniques contribute to improve the accuracy of website topic recognition. Relevant technologies have applied in enterprise web resource discovery service [9].

References

1. Chakrabarti,S.,Dom,B.,van den Berg,M.: Focused Crawling: a New Approach to Topic-specific Web Resource Discovery. Computer Networks. **31** (1999) 1623-1640
2. Ester,M.,Kriegel,H.-P.,Schubert,M.: Website Mining: A New Way to Spot Competitors, Customers and Suppliers in the World Wide Web. In: Proc. 8th ACM SIGKDD 02,Edmonton (2002) 249-258
3. Kriegel,H.-P.,Schubert,M.: Classification of Websites as Sets of Feature Vectors. In: Proc. International Conference on Databases and Applications (DBA'2004),Innsbruck (2004) 127-132
4. Ester,M.,Kriegel,H.-P.,Schubert,M.:Accurate and Efficient Crawling for Relevant Websites. In: Proc. 30th International Conference on Very Large Databases (VLDB'04), Toronto (2004) 396-407
5. Chen,X.Q.,Yu,Z.H.,Bai,S.,et al.: Automatic Information Extraction and Classification of Web Sites. In: Proc. JSCL-99,Beijing (1999) 87-92
6. Tian,Y.H.,Huang,T.J.,Gao,W.: A Web Site Representation and Mining Algorithm Using a Multiscale Tree Model. Journal of Software. **15** (2004) 1393-1404
7. Joachims,T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: Proc. ECML-98, Chemnitz (1998) 137-142
8. Han,E.-H.,Karypis,G.: Centroid-based Document Classification: Analysis and Experimental Results. In: Proc. PKDD'00,London (2000) 424-431
9. Dong,B.L.,Liu,H.M.: Implementation Web Resource Service to Product Design.In: Proc. International Conference on Programming Language for Machine Tools,Shanghai (2006) 972-977

Fault-Tolerant Orchestration of Transactional Web Services

An Liu[1,2,3], Liusheng Huang[1,2], Qing Li[2,3], and Mingjun Xiao[1,2]

[1] Department of Computer Science and Technology
University of Science & Technology of China, Hefei, China
[2] Joint Research Lab of Excellence
CityU-USTC Advanced Research Institute, Suzhou, China
[3] Department of Computer Science
City University of Hong Kong, Hong Kong, China
liuan@ustc.edu, lshuang@ustc.edu.cn, itqli@cityu.edu.hk,
xiaomj@ustc.edu.cn

Abstract. As composite services are often long-running, loosely coupled, and cross application and administrative boundaries, they are susceptible to a wide variety of failures. This paper presents a solution for fault-tolerant web services orchestration by using relaxed atomic execution and exception handling. To achieve atomic execution, a scalable commit protocol is proposed, which allows heterogeneous transactional web services to participate in a composition. A recovery algorithm is given to ensure a reliable service orchestration in the presence of failures.

1 Introduction

Web services are a rapidly emerging technology for supporting interoperable machine-to-machine interactions over a network. As a single web service cannot always fulfill complex functional requirements, it is necessary and becomes increasing popular for organizations to provide new value-added services (composite services) by composing some pre-existing web services (component services). This process is also known as orchestration [11].

As composite services are often long-running, loosely coupled, and cross application and administrative boundaries, they are susceptible to a wide variety of failures. Therefore, a main problem that remains is how to ensure a reliable orchestration and execution even in the presence of failures.

In this paper we propose a solution for reliable web services orchestration by using relaxed atomic execution and exception handling. We exploit the transactional properties of web services and design a *scalable commit protocol* to achieve atomic execution. We give a recovery algorithm, which ensures a reliable service orchestration in the presence of failures.

The rest of the paper is organized as follows. Section 2 introduces a motivating example that will be used throughout the reminder of the paper. Section 3 presents the

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 90 – 101, 2006.

scalable commit protocol and the relaxed atomic execution. The exception handling mechanism and recovery algorithm are presented in Section 4. Section 5 discusses related work and Section 6 concludes the paper.

2 Motivating Scenario

As a motivating example, consider a touring arrangement scenario, where tourists want to visit an attraction in a remote city X. Firstly, a flight booking (FB) service reserves some flight tickets to city Y as there is no direct flight to X. Therefore, a car booking (CB) service needs to reserve a car from Y to X. After the public transportation has been done, a hotel booking (HB) service is invoked to book a hotel room. At last, an appropriate transportation (e.g. taxi or bicycle) needs to be arranged by a transportation booking (TB) service, according to the distance between the hotel and the attraction or simply user preference. These services constitute a composite service. The orchestration of the composite service is illustrated in Fig. 1. Note that the ellipses labeled "And" mean that HB can start when both FB and CB terminate.

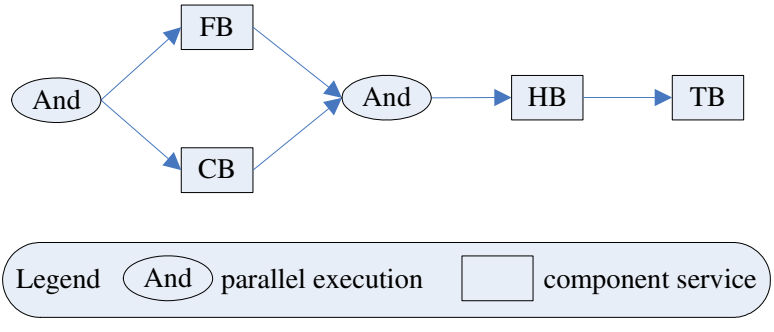


Fig. 1. A Composite Service for travel arrangement

As web services are built on the Internet, which is not a reliable media, and due to many other reasons, there may be many errors during the service execution. For example, service CB may be unavailable when being invoked. How to ensure a reliable composition is a challenging problem. In this paper, we propose a fault-tolerant mechanism to achieve this goal.

3 Relaxed Atomic Execution

In this section, we present the first mechanism to ensure the fault-tolerant orchestration, namely, relaxed atomic execution. In Section 3.1, we develop a taxonomy of transactional web services. In Section 3.2, we present a scalable commit protocol to achieve atomic execution. In Section 3.3, we discuss how to use acceptable termination states to achieve relaxed atomic execution.

3.1 Transactional Web Services

As stated in [1], a transactional web service is a web service that emphasizes transactional properties for its characterization and correct usage. The transactional properties are generally based on the notions of compensatable, retriable, and pivot transactions [9], or atomic, quasi-atomic, and non-atomic tasks [8]. Depending on their exposed operation interfaces during different lifecycles, we extend these basic classifications here to be able to distinguish among several types of transactional web services.

We divide the life cycle of a web service into two simple phases: active and completed. Furthermore, we are concerned about whether the service supports 2PC in the active phrase and whether its effects can be semantically undone through compensation when the service has already terminated. To declare these characteristics, services should provide corresponding operation interfaces including, e.g., prepare-method, commit-method, abort-method for 2PC, and compensate-method for compensation.

Based on the above-mentioned interfaces, we distinguish between *atomic*, *semantic-atomic*, *weak-atomic*, and *pivot* services. Atomic services support both 2PC and compensation, and thus are associated with the "all or nothing" semantics through invoking abort-method or compensate-method. Semantic-atomic services only provide compensate-method that can semantically undo their effects to achieve partial atomicity, while weak-atomic services only support 2PC and their effects cannot be eliminated once they terminate. A service is a pivot service if it is not atomic, not semantic-atomic nor weak-atomic. Thus, pivot services cannot rollback no matter which phase of lifecycle they are in.

Note that, these interfaces may have additional restrictions in the real web services applications. For example, service providers may predefine the time duration of blocking of resources associated with the prepare-method or only allow certain requesters to invoke compensate-method within a certain period of time. We assume here all these interfaces can be invoked by anybody in anytime since we focus on a general method to ensure atomicity by exploiting the transactional properties of web services.

3.2 Atomic Execution

One mechanism for fault-tolerant orchestration is the atomic execution, that is, the composite services execute with the "all or nothing" semantics. (A relaxed atomic execution will be discussed in next sub-section.) As stated in [15], "the most beneficial way of applying fault tolerance is by associating its measures with system structuring units as this decreases system complexity and makes it easier for developers to apply fault tolerance." However, the structure of the orchestration may be very complicated. Therefore, we first list some elementary structures in the orchestration (ref. Fig.2), based on which an arbitrary complex structure can be constructed and then discuss the detailed process of atomic execution case by case.

The atomic execution of the linear structure illustrated in Fig.2.(a) is straightforward. If service A fails, the composite service terminates unsuccessfully; otherwise, B starts. If B fails, A must be compensated for since it has been committed and the composite service terminates unsuccessfully; otherwise, the composite service terminates successfully. Hence, to ensure the atomic execution, A must be atomic or semantic-atomic, and B can be of any type.

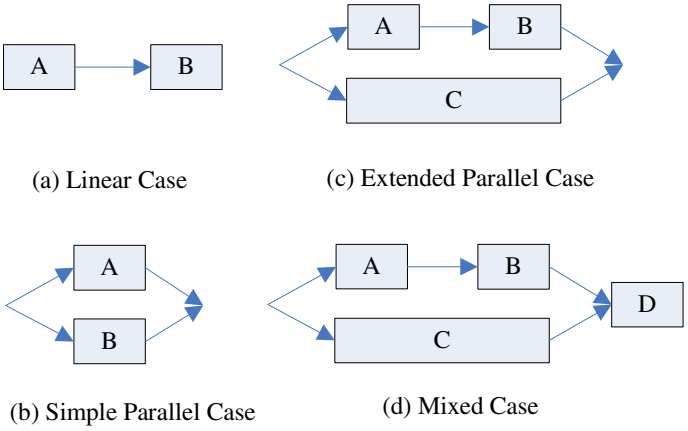


Fig. 2. Elementary structures of an orchestration

Consider now the simple parallel structure illustrated in Fig.2.(b). As we discussed earlier, not every service provides 2PC interfaces and thus the 2PC cannot be applied here directly. Therefore, we propose a ***scalable commit protocol (SCP)*** that can be applicable to services with any transactional properties. To describe the protocol, we first introduce two roles: coordinator and participant. The former represents the composite service with respect to the simple parallel structure and the latter represents the component services. If a component service is atomic, semantic-atomic, or weak-atomic, the participant is referred to as a-participant, sa-participant, and wa-participant, respectively. Similarly, if the component service is pivot, the participant is referred to as p-participant. Note that there is at most one pivot service in the simple parallel case (thus only one p-participant)[1]; otherwise the atomicity will be violated.

The SCP has four phases, which are enumerated below and illustrated in Fig.3:

Phase 1: The coordinator sends a <prepare> message to each a-participant and wa-participant. When an a-participant/wa-participant receives the <prepare> message, it checks if it can commit the transaction. If so, it sends a <vote-commit> message to the coordinator and blocks corresponding resources until it receives next message from the coordinator; otherwise, it sends a <vote-abort> message to the coordinator.

Phase 2: When the coordinator receives <vote-commit> message from all the a-participants/wa-participants, it sends a <commit> message to each a-participants/sa-participant. However, if it receives at least one <vote-abort> message from any of the a-participants/wa-participants, it asks all the a-participants/wa-participants to abort local transaction by sending a <abort> message and then aborts the global transaction.

When an a-participant/sa-participant receives a <commit> message from the coordinator, it commits the local transaction. If the local transaction has committed suc cessfully, it sends a <ack-commit> message to the coordinator; otherwise, it sends a <ack-abort> message to the coordinator.

[1] As we will see later, there may be multiple pivots in the whole orchestration.

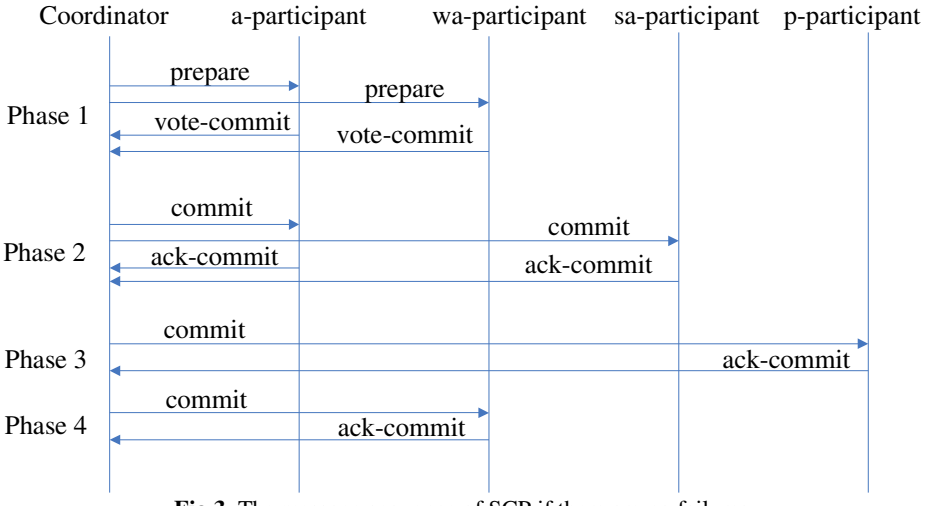


Fig.3. The message sequence of SCP if there are no failures

Phase 3: When the coordinator receives <ack-commit> message from all the a-participants/sa-participants, it sends a <commit> message to the p-participant. However, if it receives at least one <ack-abort> message from any of the a-participants/sa-participants, it asks all the wa-participants (a-participants/sa-participants) to abort local transaction by sending a <abort> (<compensate>) message and then aborts the global transaction.

When the p-participant receives a <commit> message from the coordinator, it commits the local transaction. If the local transaction has committed successfully, it sends a <ack-commit> message to the coordinator; otherwise, it sends a <ack-abort> message to the coordinator.

Phase 4: When the coordinator receives a <ack-commit> message from the p-participant, it sends a <commit> message to each wa-participant. However, if it receives a <ack-abort> message from the p-participant, it asks each of the wa-participants (a-participants/sa-participants) to abort local transaction by sending a <abort> (<compensate>) message and then aborts the global transaction.

When a wa-participant receives a <commit> message from the coordinator, it commits the local transaction and sends a <ack-commit> message to the coordinator. □

When a participant receives a <abort> (<compensate>) message from the coordinator, it aborts (compensates for) the local transaction. Finally, these component services and also the composite service terminate in a consistent state.

Note that, the coordinator's action of sending a message means that it invokes a corresponding method provided by participants. In particular, <prepare>, <commit>, <abort>, and <compensate> messages are mapped to prepare-method, commit-method, abort-method, and compensate-method respectively. Also note that we assume here there is no message loss in the SCP.

If there are no failures, SCP requires 4n+2m+2 messages as compared to 4(n+m+1) messages needed by the standard 2PC [16], where n is the number of a-participants/wa-participants and m is the number of sa-participants.

Based on the above-described SCP, all the component services terminate either successfully or unsuccessfully at the same time. While ensuring the atomic execution, SCP allows heterogeneous transactional services participate in the parallel case.

Consider now the extended parallel structure illustrated in Fig.2.(c). According to the discussion about linear structure, service A must be a-participant or sa-participant. The conservative atomic execution would let B and C enter into the first phase of the SCP when A has committed successfully. However, if B is sa-participant or p-participant, C can start the SCP earlier. This is obvious and can be easily deduced from the process of the SCP, the discussions of which are omitted here.

The last elementary structure illustrated in Fig.2(d) is an extension of the extended parallel structure. As D may fail, services A, B, and C need to be compensated for. Thus, A, B, and C must be a-participant or sa-participant, and D can be any type.

Summarizing all the above discussions, for an arbitrary orchestration structure, its atomic execution can be ensured as long as every component service has appropriate transactional properties.

3.3 Relaxed Atomic Execution

In the last sub-section, we have discussed the atomic execution of service orchestration. However, the "all or nothing" semantics may be too strict in the context of web services composition. In particular, different users may have different requirements and therefore accept different termination states of component services. This is especially important in B2C applications. Thus, we adopt and adapt *acceptable termination states* (ATS) [14] to accommodate the relaxed atomicity.

Assume that a service has only two termination states: successful and failed. Also assume that a composite service S is composed of n services: S_1, S_2, …, S_n. The termination state of S is a n-tuple (x_1, x_2, …, x_n), where x_i = ST (i=1, 2, …, n) if S_i has completed successfully and x_i = FT if S_i has not completed successfully. An acceptable termination state of a composite service is a termination state (x_1, x_2, …, x_n) where the composite service is considered to terminate successfully as long as the termination state of its component service S_i is x_i. The acceptable termination states (ATS) of a composite service is the set of all possible acceptable termination states. For example, the ATS of the composite service illustrated in Fig.1 may be {(ST,ST,ST,ST),(ST,ST,ST,FT)}, where S_1, S_2, S_3, and S_4 are FB, CB, HB, and TB, respectively.

We now present the process of relaxed atomic execution based on the elementary structures described in Section 3.2. Referring back to Fig. 2, for the linear structure, if the ATS shows that service B could terminate in FT state (i.e., B fails), then there is no need to compensate for A when B fails; therefore, A can be any type of transactional services. For the simple parallel structure, the SCP needs a few adaptations. Instead of waiting for <vote-commit> or <ack-commit> message from all participants, the coordinator only needs these messages from those participants whose acceptable termination state is successful. Relaxed atomic execution for the extended parallel case and mixed case can be directly resolved based on the linear case and simple parallel case, and the details are omitted here for briefness.

As the ATS can vary dynamically according to the different requirements of service providers or end users, the relaxed atomic execution makes the composite service more flexible. In addition, it reduces the possibility to compensate for successfully terminate services and to abort the whole application. Moreover, services that do not have 2PC interface or compensate-method could participate in the orchestration.

4 Fault-Tolerant Orchestration

This section describes how to build fault-tolerant orchestration by using relaxed atomic execution and exception handling. In Section 4.1, we discuss the exception handling mechanism in the composite service. Sections 4.2 and 4.3 describe the recovery algorithm and an illustrating example respectively.

4.1 Exception Handling

The relaxed atomic execution is in fact a backward error recovery mechanism. To make the orchestration more fault-tolerant, we introduce the forward error recovery mechanism by using exception handling. Just as the try-throw-catch constructs for exception handling in the programming language such as Java, we need to specify the scope in the service orchestration as a logical unit where exceptions will be caught and handled. Therefore, we adopt and adapt the term *sphere* to specify such a scope.

The concept of "spheres of control" was suggested in [6], which defines logical boundaries around an arbitrary set of operations that can be unilaterally committed or aborted. In short, a sphere in service orchestration can be viewed as a scope in a composite service and involves multiple component services that are composed by the elementary structures as discussed in Section 3.2. Note that spheres can be nested, but cannot be overlapped. The nested sphere could be described in a tree structure if we consider the outermost sphere as the root and its direct inner spheres as its children.[2]

Take the same touring scenario as an example, we may specify four spheres: sp_1, sp_2, sp_3, and sp_4, as illustrated in Fig.4(a). The corresponding sphere tree is shown in Fig.4(b).

In short, a sphere can handle the exceptions thrown by component services or other spheres. In general, not only the exception type but also the corresponding handling resolutions should be specified. The former is generally application-specific and thus we focus on the latter in this paper.

Summarizing the exception handling resolutions proposed in [3], [8] and [17], we distinguish between four types of resolutions in the context of web services orchestration, namely: *skip*, *retry*, *alternate* and *replace*.

- ***Skip*:** It is used to specify that certain services need not to be executed due to some reasons, such as the cost or time constraints. We assume that this resolution is always given by the root-coordinator as it knows the ATS of the composite service and other global information. Note that only these services whose ATS is (FT) is allowed to be skipped.

[2] We assume that every sphere is controlled by a coordinator, and the coordinator of the outermost sphere is root-coordinator while other coordinators are sub-coordinators.

- ***Retry*:** It is used to specify that certain services need to be re-invoked. Generally, it should also specify the upper bound of retry times or the deadline for retrying.
- ***Alternate*:** It allows a different service that has a similar function as the original's is invoked. For example, consider sp_2, if the HB is temporarily unavailable, we may invoke another HB provided by another organization.
- ***Replace*:** It means a different approach is taken. For example, consider sp_1, if there is no flight ticket to city X is available, some train tickets may be booked through another service TTB instead.

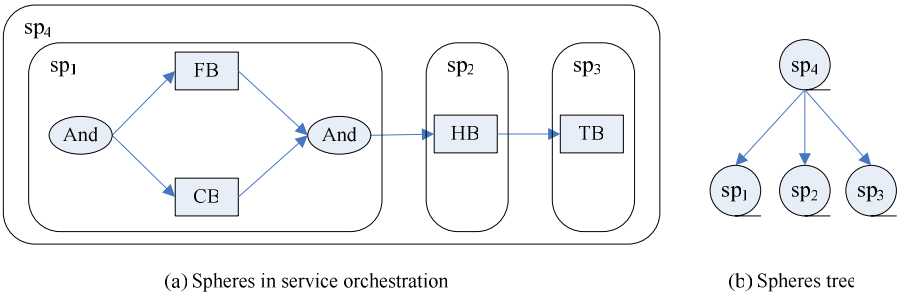


Fig. 4. Spheres in service orchestration

When a sphere catches an exception, it checks if there is an exception handler for this exception. If so, it uses the corresponding resolutions and then the execution can go on; otherwise, it propagates the exception to its parent sphere. The process continues until either an exception handler is found or the exception is propagated to the root sphere. The root sphere aborts the whole process if it cannot handle this exception.

When a sphere needs to abort, it compensates for successful-terminated component services and cancels active component services (we call this *default abort* method). However, the sphere may provide a self-defined abort method, and it always tries this method first and then the default one. For example, when sp_1 needs to abort, it may not compensate for FB (flight booking) and CB (car booking) services; instead, it may give the flight tickets and the arranged car to another tourist.

Such exception handling resolutions affect the atomic execution as they may change services' transactional properties. The *skip* method only ignores some services and has the same effect on atomic execution as the ATS. The *retry* method does not introduce new services and thus has no effect. The *alternate* and *replace* methods introduce new services and thus have to ensure the transactional properties of the new service can satisfy the criteria discussed in the last section. The self-defined abort method can relax the requirements of transactional properties as it does not require services to have a compensate-method.

4.2 Recovery Algorithm

We are now in position to give the whole recovery algorithm. The algorithm here assumes that there is no error during exception handling.

- **Recovery Algorithm**
  - **Sub-coordinators**:

```
Var: msg: message between coordinators; s: a component service
Begin
 wait(msg);
 if msg is exception {
  if the coordinator cannot handle the exception, send msg to its parent;
  else {//the coordinator has learned that what action should be taken
   case (retry s): retry s;
   case (alternate s): select another service to execute;
   case (replace s): abort current sphere and then invoke the specified service;
  }
 }
End
```

  - **Root-Coordinator**

```
Var: msg: message between coordinators; s: a component service
Begin
 wait(msg);
 if msg is exception {
  if the coordinator cannot handle the exception, abort the root-sphere;
  else {
   case (skip s): tell corresponding spheres to ignore s and then continue normal execution;
   case (retry s): retry s;
   case (alternate s): select another service to execute;
   case (replace s): abort root-sphere and then invoke the specified service;
  }
 }
End
```

- **Correctness of the Algorithm:** When an exception is thrown, it is either handled by the current sphere according to predefined resolutions or propagated to its parent sphere. After it has been handled successfully, the composite service continues its normal execution. If the exception eventually cannot be handled, the root-sphere aborts the whole process. Based on the relaxed atomic execution, the composite service will terminate in a correct state.

4.3 Touring Example Revisited

To illustrate our approach, we consider our touring arrangement example again, for which the spheres in the composite service are as illustrated in Fig.4. The exception-resolution pairs for every sphere are given in Table 1. Assume that services FB, CB, HB, and TB are atomic, semantic-atomic, weak-atomic, and atomic, respectively; also assume that the ATS of the composite service is {(ST,ST,ST,ST),(ST,ST,ST,FT)}. As shown in Table 1, the *replace* method introduces a new semantic-atomic service (TTB) to book train tickets for the two exceptional cases of FB and CB.

More specifically, we discuss below the exception handling for each of the cases:

- FB fails (i.e., the available tickets are not enough). FB_NoFlightTicket exception is thrown by FB and caught by sp_1. The latter learns from an exception-resolution table (ERT) that it should first abort and then invoke TTB. When TTB terminates successfully, the composite service goes on to execute the HB service.

- HB fails (i.e., not enough room). HB-NoRoom exception is thrown by HB and caught by sp_2. The latter learns from the ERT that it should invoke another service HB_1 that also offers hotel rooms. When HB_1 terminates successfully, the composite service goes on to execute the TB service.
- TB fails (i.e., the service TB is unavailable now – maybe due to a server crash). TB_Unavailable exception is thrown by TB and caught by sp_3. The latter retries it for n times but it is still unavailable. Then sp_3 propagates this exception to its parent, sp_4, which learns from the ERT that such a failure can be ignored. At last, the composite service terminates successfully. Alternatively, in the case that TB must terminate successfully and thus the *skip*(TB) method does not exist, sp_4 shall abort the whole composite service as this exception cannot be handled. According to the transactional properties of these component services, it is obvious that the composite service can terminate in a correct state after abortion.

Table 1. Exception-resolution pair of every sphere

Sphere	Exception	Resolution
sp_1	FB_NoFlightTicket	*replace*(TTB)
sp_1	CB_NoCar	*replace*(TTB)
sp_2	HB_NoRoom	*alternate*(HB)
sp_3	TB_Unavailable	*retry*(TB,n)
sp_4	TB_Unavailable	*skip*(TB)

5 Related Work

Much work has been done recently to address the requirements of transactional support for web services, such as the Business Transaction Protocol (BTP) [5], Web Services Transactions Specifications [4], and Web Services Transaction Management (WS-TXM) [2]. These proposals focused on how to relax some ACID properties and defined a standardized way for services to interact with their coordinators. However, the heterogeneous transactional properties of web services were not considered.

Mikalsen et al. described a framework called WSTx which takes different transaction attitudes into account [10]. In their approach, client application declares their transactional requirements and service provides declare their individual transactional capabilities and semantics. However, they only ensure failure of atomicity, which is not enough for fault-tolerant web services composition. This shortcoming also exists in the work of [1], where the authors used ATS to express required failure atomicity. The transaction model proposed by Fauvet et al. in [7] adopted the original THP to avoid resource blocking and extended the strict atomicity by defining a parallel composition operator with minimality and maximality constraints. Obviously, the two constraints have less semantics than that of ATS used by our model. In [15], a forward recovery based transaction model was proposed. However, the heterogeneous transactional properties of web services were also ignored, which may lead to unrecoverable composite services.

6 Conclusion and Future Work

In this paper, we have proposed a solution for fault-tolerant web services orchestration. This solution supports both backward error recovery via relaxed atomic execution and forward error recovery via exception handling, and is therefore more flexible and powerful than existing approaches proposed in the literature. To achieve atomic execution, a *scalable commit protocol* (SCP) has been proposed, which allows heterogeneous transactional web services to participate in a composition. The acceptable termination state is used to relax the atomicity and makes the orchestration more flexible. A recovery algorithm has also been given, which ensures a reliable service orchestration in the presence of failures.

To make our solution more applicable in practice, there are at least two issues which need to be further studied:

- **Transactional characteristic advertisement.** Some earlier attempts ([10], [12]) have aimed at extending the WSDL to express transactional properties of web services. However, these proposed specifications are not yet widely accepted. We argue that this is an important issue which needs to be refined in the future.
- **Efficiency.** Our preliminary SCP has the same weakness as that of the 2PC, namely, resources may be blocked for an unknown duration. However, this weakness can be eliminated by using such methods as the Tentative Hold Protocol (THP) [13] and the reservation-based coordination protocol [18]. To design an efficient scalable commit protocol is one of the main directions of our future work.

Acknowledgements

The research described here is supported by the National Basic Research Fund of China ("973" Program) under Grant No.2003CB317006, and has been benefited from various discussions among the group members of the Joint Research Lab between CityU and USTC in their advanced research institute in Suzhou (China), particularly Mr. Lin Baoping and Liu Hai.

References

1. Bhiri, S., Perrin, O., and Godart, C. 2005. Ensuring Required Failure Atomicity of Composite Web Services. In Proceedings of 14th International Conference on World Wide Web (WWW'05). Chiba, Japan.
2. Bunting, D., et al. 2003. Web Services Transaction Management (WS-TXM) Ver1.0. Available at (2006-4-20): http://developers.sun.com/techtopics/webservices/wscaf/wstxm.pdf
3. Chiu, D.K.W., Li, Q., and Karlapalem, K. 1999. A Meta Modeling Approach for Workflow Management System Supporting Exception Handling". Information Systems, Vol. 24, No. 2, pp.159-184.
4. Curbera, F., et al. 2003. The Next Step in Web Services. Communications of the ACM, Vol. 46, No. 10, pp.29-34.
5. Dalal, S., et al. 2003. Coordinating Business Transactions on the Web. IEEE Internet Computing, Vol. 7, No. 1, pp.30-39.

6. Davies, C.T. 1978. Data Processing Spheres of Control. IBM Systems Journal, Vol. 17, No. 2, pp.179-198.
7. Fauvet, M., et al. 2005. Handling Transactional Properties in Web Service Composition. In Proceedings of 6th International Conference on Web Information Systems Engineering (WISE'05). New York, NY, USA.
8. Hagen, C. and Alonso, G. 2000. Exception Handling in Workflow Management Systems. IEEE Transactions on Software Engineering, Vol. 26, No. 10, pp. 943-958.
9. Mehrotra, S., et al. 1992. A Transaction Model for Multidatabase Systems. In Proceedings of 12th International Conference on Distributed Computing Systems (ICDCS'92). Yokohama, Japan.
10. Mikalsen, T., Tai, T., and Rouvellou, I. 2002. Transactional Attitudes: Reliable Composition of Autonomous Web Services. In Proceedings of the workshop on Dependable Middleware-based Systems (WDMS'02) at the Dependable Systems and Network Con-ference (DSN'02). Washington D.C., USA.
11. Peltz, C. 2003. Web Services Orchestration and Choreography. IEEE Computer, Vol. 36, No. 10, pp.46-52.
12. Pires, P.F., Benevides, M.R.F., and Mattoso, M. 2002. Building Reliable Web Services Compositions. Web, Web-Services, and Database Systems, NODe 2002 Web and Database-Related Workshops. Erfurt, Germany.
13. Roberts, J., et al. 2001. Tentative Hold Protocol Part 2: Technical Specification. Available at (2006-4-20): http://www.w3.org/TR/tenthold-2/
14. Rusinkiewicz, M. and Sheth, A. 1995. Specification and Execution of Transactional Workflows. In Modern Database Systems: The Object Model, Interoperability, and Be-yond. pp. 592-620.
15. Tartanoglu, F., et al. 2003. Coordinated Forward Error Recovery for Composite Web Services. In Proceedings of 22nd International Symposium on Reliable Distributed Sys-tems (SRDS'03). Florence, Italy.
16. Weikum, G. and Vossen, G. 2002. Transactional Information Systems: Theory, Algorithms, and the Practice of Concurrency Control and Recovery. Morgan-Kaufmann Publishers, San Francisco, CA.
17. Zeng, L., et al. 2005. Policy-Driven Exception-Management for Composite Web Services. In Proceedings of 7th IEEE International Conference on E-Commerce Technology (CEC'05). Munich, Germany.
18. Zhao, W., Moser, L.E., and Melliar-Smith, P.M. 2005. A Reservation-Based Coordina-tion Protocol for Web Services. In Proceedings of 3rd IEEE International Conference on Web Services (ICWS'05). Orlando, FL, USA.

Supporting Effective Operation of E-Governmental Services Through Workflow and Knowledge Management

Dong Yang, Lixin Tong, Yan Ye, and Hongwei Wu

Department of Industrial Engineering, Shanghai Jiao Tong University,
200030 Shanghai, China
{dongyang, culizn, yeyan, hom}@sjtu.edu.cn

Abstract. The improvement in efficiency of governmental administrative processes is a key to successful application of e-government. Workflow technology offers such a mean to realize automation of business processes. However, it is inappropriate to employ workflow technology alone for automating governmental processes that are typically of knowledge-intensive characteristics. In this paper, we study an approach of automating or semi-automating e-governmental processes by combining workflow technology and knowledge management, especially semantic web technology such as OWL, SWRL. The proposed approach can be classified into several parts: (1) The government processes are modeled using UML activity diagram with extended UML profiles. (2) The Problem-Solving-Method (PSM) tasks are represented with OWL-S. (3) The application ontology for applying for social security cards is developed in OWL. 4) Based on the proposed approach, the architecture of knowledge-driven e-governmental processes management system is presented and a prototype system is implemented using workflow engines, Jena and JESS.

Keywords: Knowledge-intensive business processes, e-government.

1 Introduction

Electronic government aims to enhance the efficiency of governmental administrative processes, improve qualities of government services and reduce operational costs through application of ICT (Information and Communication Technology) technology. With the prosperity of ICT, especially the Internet, recent years almost all government agencies around the world have set up their own websites or portals to promote communication between citizens and them, trying to serve best for their customers. It is estimated that until July, 2005 the number of websites built by government agencies at all levels in China amounts to 16000, increasing at the rate of 14.6 percent a year [1]. These websites or portals in China mainly present static information such as information publishing and guidelines for specific services to citizens. Additionally, interaction channels such as message boards and online feedbacks are offered to collect from citizens and enterprises feedbacks and advices on operation of services. Furthermore, some information integration means such as

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 102 – 113, 2006.

central databases are adopted to eliminate the information islands existing among applications within government agencies. For example, Shanghai, the largest commercial city in China, has established a central database called CIS (Shanghai Citizen Information System) to realize the consistency of citizen data among government agencies.

However, several weaknesses exist that hinder the development of e-government in China. Firstly, most government agencies fail to provide a mechanism for citizens or enterprises to track the progress of the cases or services for which they apply, such as granting full old-age pension. Second, the internal processes within government agencies are not streamlined. An obvious example is that much time was wasted passing documents or information manually between the tasks of the fragmented processes. Thus, up to now the automation of e-government processes has not been indeed achieved. The current strategy to implement e-government in Shanghai by the construction of a central database is not enough to exploit the potentials of electronic government.

To better serve citizens and automate government processes, a perspective of citizen-oriented, processes-center integration is needed. Workflow technology has been proven to be effective in integrating or automating business processes and production processes (such as ERP) [2,3]. Nevertheless, automation of governmental processes by employing workflow technology alone is inappropriate. The reason for this is that e-government processes differ essentially from business processes in that they are knowledge-intensive processes [4,5]. The knowledge-intensive processes are characterized by frequent dependencies on knowledge such as regulation and rules to fulfill tasks, and they can be identified by high complexity of processes and high intensity of knowledge [5]. The typical knowledge-intensive processes are order configuration, new product development, market sales, etc. In the context of e-government, government administrative processes often involve a kind of knowledge-intensive tasks that are identified in the CommonKADS [6] as PSM (Problem-Solving-Method) tasks. The example PSM tasks include assessment, configuration, diagnosis, etc [6]. A variety of PSM tasks exist in the context of electronic government processes, such as assessment of full old-aged pension, the classification of legal cases. However, the means for automating these tasks and thus improving processing efficiencies of these tasks still lack. Although Workflow technology offers a way of automating or semi-automating business processes, only normal, well-structured tasks are supported. To effectively automate or manage overall e-government processes involving both normal and knowledge-intensive tasks, the combination of workflow technology with knowledge management is indispensable to achieve this goal. In this paper, we discuss the undertaken project, KD-GPMS (Knowledge-Driven e-Government Processes Management System), aiming to develop a novel approach to the automation of e-government processes by leveraging workflow technology to manage the overall processes and semantic web technologies to realize automatic reasoning of PSM tasks.

This paper is organized as follows. Related work is summarized in next section. The motivation of this paper is described in section 3. In section 4, we present the approach of modeling e-government processes in UML activity diagram with extended UML profiles to support the modeling of the knowledge-intensive tasks. The construction of e-government application ontology using OWL and the

representation of PSM tasks in OWL-S are dealt with in section 5. Section 6 presents the architecture of KD-GPMS system, and then describes an implemented prototype. A conclusion is provided in section 7.

2 Related Work

There have been a number of studies on knowledge management applied to e-government domain [7-10]. The SmartGov project [7] developed a Knowledge based platform to assist public sector employees to generate online transaction related to e-forms. It utilized ontology to construct the domain map associated with both knowledge units (such as guidelines, lessons) and e-form services [8]. The eGov project realized a one-stop government platform where government services specified by GovML (Governmental Markup language) are registered and deployed [9]. The OntoGov project developed a holistic framework for reconfigure and evolution of e-government services by using semantically rich representation (ontology-based) of domain knowledge, and service knowledge [10].

Recently research on the application of process modeling and workflow management to e-government domain has attracted a lot of interest [11-13]. In the IMPULSE project [11], an inter-workflow model is proposed for coordination of government services across boundaries of public agencies. [12] models service flow management using XML by identifying service points that captures specific tasks provide by public authorities. [13] shows that the Process Definition Language (PDL) can be applied to analyze inconsistencies and errors in government administrative processes such as the application of the license renewal.

Nevertheless, there are few works concerning the combination of workflow management and knowledge management. The research most closely related to our work is [5], which also take full potential of both workflow and knowledge management to automate or semi-automate knowledge-intensive government processes. But, our work differs from it in that we consider automatic reasoning of PSM tasks within government services, integrated into uniform workflow management framework. Additionally, semantic web technologies such as OWL, SWRL are utilized in our study to construct knowledge bases in e-government domain.

3 Motivating Examples

To illustrate the modeling of e-government processes, an example process about applying SSC services (Social Security Card) is firstly given in this section. A card, called SSC (Social Security Card), is granted to the citizens in Shanghai as a uniform credential in obtaining governmental services offered by various government organizations and agencies, such as housing benefit, health insurance, labor employment, thus avoiding confusion caused by existing proprietary cards required to fulfill these individual services. The workflow process for applying a SSC runs as follows:

1) A citizen fill in an application form by providing requisite information such as identity card, housing account, health insurance account.
2) When his/her information is unavailable in the CIS, the process terminates.

3) Otherwise, social workers check the consistency between the information on the form and those in CIS. If inconsistency exits, additional paper-based evidences are required.
4) After the consistency checks, social workers will assess whether he/she is qualified for this application according to the rules or regulations such as the restriction on legal ages, residence, etc. The process automatically terminates in case the application cannot be approved.
5) In the case of the approval of his/her case, the citizen travels to the nearest CBO on the agreed upon date to take picture and fingerprint, and to make payment.
6) The collected information is transferred to the SSC center for making cards.
7) Through a web-based Internet browser, the citizen can know exactly whether his/her card has been manufactured. After it has been made, he/she may choose to either travel to the CBO for the card or wait for it to be mailed according to the address that he/she offered in filling the application form.

We use the CommonKADS knowledge engineering method to identify the knowledge-intensive tasks within this e-government process. For example, through analysis of this process, the task "assess application" is identified as a knowledge-intensive task, i.e. PSM task, owing to its close dependency on government regulations about SSC.

4 Modeling E-Government Processes

The UML activity diagram is employed to model knowledge-intensive e-government processes in our project. To distinguish normal tasks from knowledge-intensive tasks, UML profiles [14] extension mechanism is considered to extend the UML activity diagram. The main advantage of UML profile is to offer a way to tailor the UML meta-model for different domains, thereby enabling domain experts or modelers to model specific domains with their familiar terms or vocabularies. Moreover, the existing UML tools can be utilized by reusing UML notations without developing purpose-specific modeling tools. The UML profiles can be defined by the stereotypes, constraints and tagged value elements. A stereotype is a meta-class defined by the UML meta-model, which is used to create domain-specific class and may has own specific graphical symbol. Constraints specified in OCL (Object Constraint language), or other languages (such as natural languages, pseudo-code) impose restrictions on the stereotypes.

Figure 1 shows the workflow model for applying for a SSC using the UML activity diagram. Several stereotypes are defined, among which are *<<wfTask>>*, *<<PSMTask>>*, etc. The stereotype *<<wfTask>>* indicates the regular tasks managed and monitored by WfMS, whereas the stereotype *<<PSMTask>>* considers the knowledge-intensive tasks governed by PSM engine for automatic reasoning. The activity "*Assess application*", for example, belongs to the knowledge-intensive tasks. The workflow models depicting the SSC service are exported as XMI model files [15], and then they serve as an input to the WfMS where the tasks within the SSC service are enacted and navigated, and possibly the PSM engine is triggered for automatic reasoning.

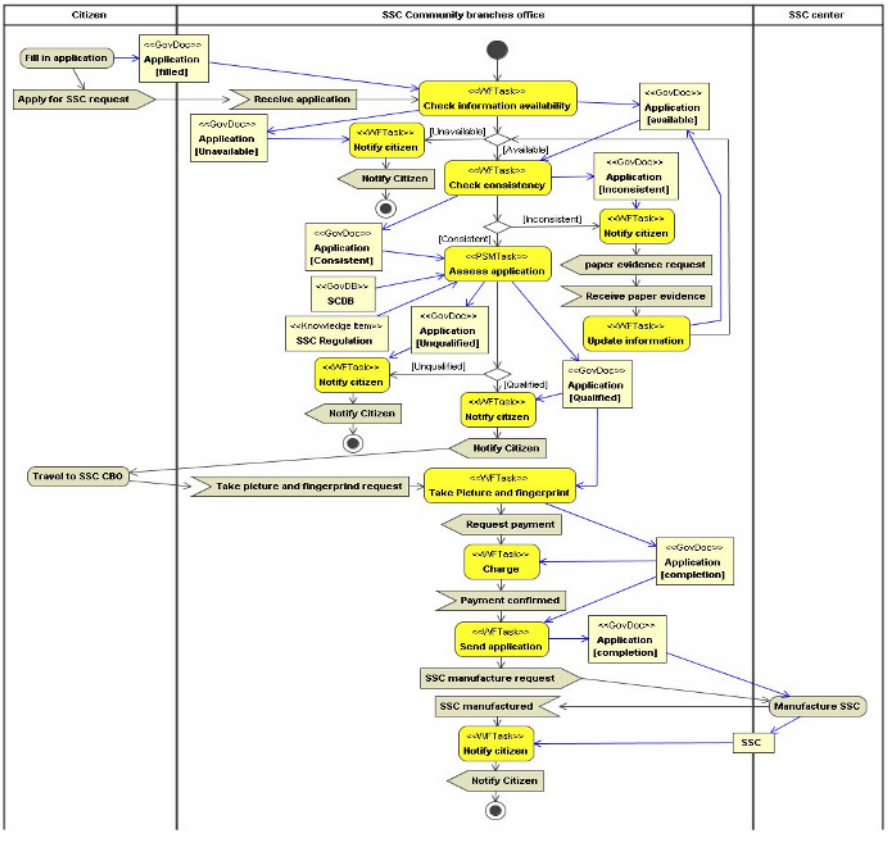


Fig. 1. Workflow models for applying a SSC

5 E-Government Knowledge Modeling

Ontology is defined as a formal and explicit representation of a conceptualization [16]. It allows one to represent, in a formal way, concepts of a domain of concern and their relations. Compared to the content-based knowledge management system (KMS), the ontology-based KMS enables hierarchical construction of domain concepts, thereby facilitating reuse and reasoning of domain knowledge [17]. In the KD-GPMS, the ontology-based method is adopted to construct the KMS for e-government.

For knowledge-intensive tasks identified in section 3, knowledge items required by these tasks can be derived according to the CommonKADS method. For example, the knowledge items on which the task '*Assess application*" depends are the two regulations: Shanghai SSC Policy Method, and Shanghai SSC Policy Supplement. Further, e-government ontology can be identified, refined on the basis of these derived knowledge items.

5.1 E-Government Ontology

To support reuse of domain concepts and terms across various e-government applications, the e-government ontologies within our project are hierarchically constructed, which are classified into three parts: general ontology, domain ontology and application ontology. The general ontology contains generic concepts shared by all domains, such as location, time, event, state. The domain ontology includes the terms and vocabularies specific to e-government domain, for example, *GovDocument*, *GovService*, *citizen*, *credential*. In addition to including those ontologies from domain ontology, the application ontology also defines methods-specific or task-specific concepts required to solve an application problem. The terms *SSC Application Form*, *SSC decisions* are such ontologies. To realize the reasoning on PSM tasks, the roles defined in PSMs are required to be mapped to these method-specific or task-specific ontologies. To avoid semantics confusion, a formal definition of ontology is required.

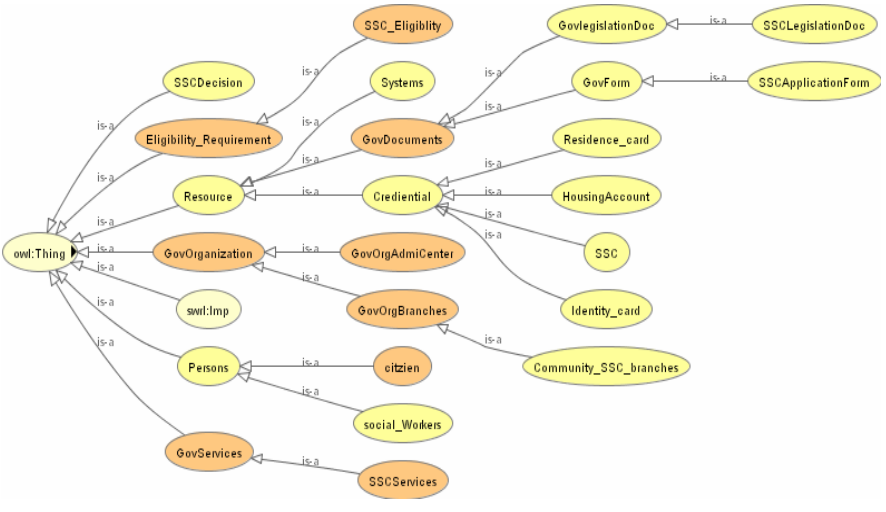


Fig. 2. A taxonomy of domain ontologies

The OWL (Web Ontology language) [18] is employed to formally model ontology for this application owing to its semantic of description logics and capability of reasoning on consistencies. Figure 2 shows a part of taxonomy of the e-government application ontologies in our project.

5.2 PSM Task and PSM Method

A PSM task specifies the goal of solving a problem whereas a PSM method describes how to realize this goal by decomposing this task into a set of subtasks (or inferences) and control constructs among them. To reuse reasoning parts of knowledge-base systems, PSM methods for typical PSM tasks have been defined independently of particular applications by knowledge engineering communities, thereby acting as

building blocks to construct complex applications. As a result, concrete applications for knowledge-intensive tasks can directly adopt or customize them. For example, for the PSM task "*assessment*", there exists a PSM method, named *assessment-with-abstract* [6], to solve this problem. This method consists of five sub-tasks (inference actions) and control flows among them, as shown in figure 3. These inference actions are *abstract*, *specify*, *select*, *evaluate* and *match*. The *abstract* inference is used to simplify data within cases in terms of some rules. The *specify* inference determines those rules related to specific data (such as age, residence, incoming, etc.) within cases. The *select* inference action is employed to choose one of these rules to evaluate cases. The *match* inference is utilized to decide whether the cases are approved or not in terms of the results of evaluations. We utilize the OWL-S (Ontology Web Language for Services) [19] to represent control constructs of PSM methods since the OWL-S offering rich semantic description of services enables flexible automated service discovery and composition. For the task "*Assess application*", for example, the control flows of its corresponding PSM method, i.e. *Assessment-with-abstract*, are described in figure 4 with the OWL-S.

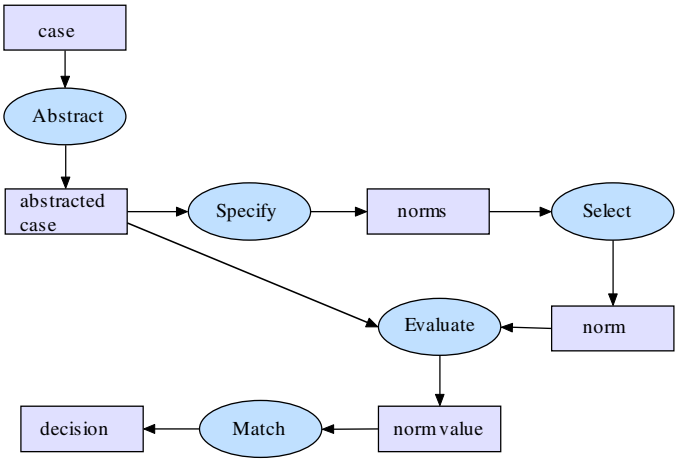


Fig. 3. Control construct for the PSM method "*assessment-with-abstract*"

5.3 E-Government Rules

As a standard rule language for semantic web, the SWRL [20, 21] allows users to write Horn-like rules expressed in terms of OWL concepts. There are two main advantages for the use of the SWRL to express e-government rules in our project: (1) The facts that SWRL rules make use of OWL concepts to represent the predicates within the rules favors better integration with e-government domain ontology. For example, users can edit both ontologies and rules in an integrated environment such as the Protégé tool. (2) Our system has freedom to choose any specific inference engine (such as Jena, Jess) without modifying SWRL rules for the SWRL is a

```
<process:CompositeProcess rdf:ID="Assessment_Process">
  <rdfs:label>The process for assessment PSM methods</rdfs:label>
  <process:composedOf>
    <process:Sequence>
      <process:components rdf:parseType="Collection">
        <process:AtomicProcess rdf:about="#Abstract" />
        <process:AtomicProcess rdf:about="#Secify" />
        <process:CompositeProcess rdf:about="#EvaluateProcess" />
        <process:AtomicProcess rdf:about="#MakeDecision"/>
      </process:components>
    </process:Sequence>
  </process:composedOf>
  </process:CompositeProcess>
  <process:CompositeProcess rdf:ID="EvaluateProcess">
  <process:composedOf>
    <process:Repeat-Until>
      <untilCondition rdf:about="#NumberOfNormEqualZero"/>
      <untilProcess rdf:about="#SelectEvaluateMatch"/>
    </process:Repeat-Until>
  </process:composedOf>
  </process:CompositeProcess>
  <process:CompositeProcess rdf:ID="SelectEvaluateMatch">
  <process:composedOf>
  <process:Sequence>
    <process:components  rdf:parseType="Collection">
      <process:AtomicProcess rdf:about="#Select" />
      <process:AtomicProcess rdf:about="#Evaluate" />
      <process:AtomicProcess rdf:about="#Match" />
    </process:components>
    </process:Sequence>
  </process:composedOf>
  </process:CompositeProcess>
```

Fig. 4. Control construct for the method "*Assessment*" in the OWL-S

description language independently of any implementation languages internal to inference engines. To illustrate the representation of e-government rules in the SWRL, let us consider an essential regulation for applying for a SSC: "those who are 18 years old and live in Shanghai can apply for social security cards". This can be expressed as a SWRL rule:

$$citizen(?x) \wedge hasage(?x, ?y) \wedge swrlb:greaterThanOrEqual\ (?y, 18) \wedge \\ \wedge\ hasResidence(?x, \mathrm{SH}) \wedge\ SSCServices(?z) \rightarrow\ appliedfor(?z, ?x)$$

6 System Architectures and Implementation

According to the approach above, we develop a prototype system for the KD-GPMS project, taking advantage of both workflow technology and knowledge management. Figure 5 shows the architecture of this system, which consists of two parts:

(1) The front-end:

The front-end is an Internet-based web interface for citizens to fill in information required to complete an e-government service. It comprises HTML pages and sub programs for verifying the correctness of these forms.

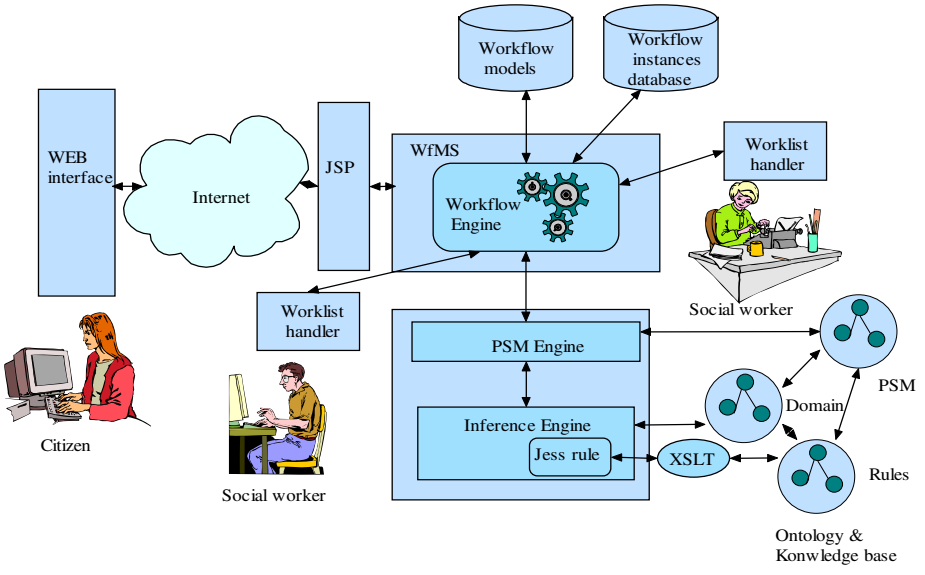


Fig. 5. The architecture of the prototype system

(2) The back-end:

The back-end of this system is composed of the JSP application, the workflow management system (WfMS) and the PSM engine, whose functions are explained as follows:

- JSP application

  The JSP (Java Server Pages) application module processes the requests such as the forms submitted by citizen, triggers the workflow engine for enactment of e-government services, and then forwards the results to the client front end.

- Workflow engine and WfMS

  The workflow engine starts the execution of cases once it receives the triggering message from the JSP application. The engine schedules those tasks within workflow processes to be executed according to workflow models that have been defined in UML activity diagram and saved in XMI file format, assigns tasks to proper persons, receives the processing result of tasks, and monitors the progress of the whole cases. However, in the case of a task as a

knowledge-intensive one, the workflow engine will trigger the PSM engine to solve a problem and then receive feedback from it.

Apart from the workflow engine, the WfMS also contains workflow monitoring and performance tools for managers to check statistical results about the performance of services.

- PSM engine

The PSM engine takes the responsibility for solving a PSM task. This engine module is implemented by OWL-S API, which provides necessary functionalities to manipulate OWL-S files. It reads the files in OWL-S, coordinates execution of the subtasks within PSM methods, and starts the Jess [22], a rule engine for Java platform that implements Rete algorithm to process rules. The Jess engine actually performs the primitive reasoning with e-government rules expressed in the SWRL, where the SWRL rules are transformed into Jess rules by using a XSLT processor, and the domain knowledge in the Protégé are mapped to Jess facts. In this prototype implementation, we utilize the JessTab [23], a plugin for integration of Jess inference engine into the Protégé environment, enabling the reasoning on knowledge bases in the Protégé-OWL. Additionally, the ontologies and knowledge in the Protégé are manipulated through Jena [24], a Java framework offering a programmatic environment for RDF, OWL.

7 Conclusion

In this paper, we survey the undertaken project KD-GPMS that integrates workflow and knowledge modeling techniques to automate the whole e-government processes involving both regular task and knowledge-intensive tasks, aiming to improve processing efficiencies of government services. This work adopts the UML activity diagram with extended UML profiles to model government processes and semantic web technologies to model e-government domain knowledge for problem solving. A prototype system has been implemented using workflow engines and Jess, Jena, Protégé tools. In the future, we plan to develop PSM methods as semantic web services such that they can be reused by a variety of government services typically involving a wide range of knowledge-intensive tasks such as assessment, classification and planning.

Acknowledgements. This study is funded by National Science Foundation of China (NSFC) under No. 70471023.

References

1. CINIC.: 16th Statistical Survey Report on the Internet Development in China, China Internet Network Information Center. http://www.cnnic.net.cn/en/index/index.htm, January, 2006
2. Geograkopoulos D., Hornick M, Sheth A.: An Overview of Workflow Management : From Process Modeling to Workflow Automation Infrastructure. Distributed and Parallel Database, 3(2), 1995

3. W.M.P. van der Aalst, K. van Hee: Workflow Management: Models, Methods, and Systems. MIT Press, 2002
4. Klischewski, R., Lenk, K.: Understanding and Modelling Flexibility in Administrative Processes. In: Lenk, K., Traunmüller R. (Eds.): Electronic Government (EGOV 2002). Lecture Notes in Computer Science, Vol. 2456. Springer-Verlag, Berlin Heidelberg New York (2002) 129-136.
5. Papavassiliou, G., Ntioudis, S., Abecker, A., Mentzas, G.: Supporting Knowledge-Intensive Work in Public Administration Processes. Knowledge and Process Management, 10(3), 2003
6. Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R. Shadbold, N., van der Velde, W., Wielinda, B.: Knowledge Engineering and Management, The CommonKADS Methodology. The MIT Press, Cambrigde (2000)
7. E. Tambouris, G. Boukis, C. Vassilakis, G. Lepouras, S. Rouvas, R. Canadas, S. Heredia, J. C.Lopez Usero: SMARTGOV: A Governmental Knowledge-based Platform for Public Sector Online Services, Proceedings of the KMGov2002 Workshop, May 23-24, 2002
8. John F., Nick A., Ann M., Andy M., Tomás P.L., Pablo F.P., Rafael C., Jesús S.: Knowledge Management Applied to E-government Services: The Use of an Ontology. Wimmer, M. (Ed.): Knowledge Management in Electronic Government (KMGov 2003), LNAI 2645, pp. 116–126, 2003.
9. Tambouris, E.: An Integrated platform for Realising Online One-Stop Government: The eGov Project. Proceedings of the DEXA International Workshop On the Way to electronic Government. IEEE Computer Socity Press, 2001, pp. 359-363
10. Efthimios T., Stelios G., Gregory K., Dimitris A., Andreas A., Ljiljana S., Gregoris M.: Ontology-Enabled E-gov Service Configuration: An Overview of the OntoGov Project. Wimmer, M. (Ed.): Knowledge Management in Electronic Government (KMGov 2004), LNCS 3035. Springer-Verlag (2004) 122-127
11. Aljosa P., Sara D., Jose A. E.: IMPULSE: Interworkflow Model for e-Government. In: Lenk, K., Traunmüller R. (Eds.): Electronic Government (EGOV 2002). Lecture Notes in Computer Science, Vol. 2456. Springer-Verlag, Berlin Heidelberg New York (2002) 472-479
12. Klischewski, R., Wetzel, I.: Serviceflow Management: Caring for the Citizen's Concern in Designing E-Government Transaction Processes. Proceedings of Hawaii International Conference on System Sciences (HICSS-35), IEEE, 2002
13. Osterweil, L., Schweik, C.M., Sondheimer, N. and Thomas, C.: Analyzing Processes for E-Government Development: The Emergence of Process Modeling Languages. Journal of E-Government,1(4), 2005, 63-87
14. Object Management Group: UML - Unified Modeling Language v2.0. http://www.omg.org/
15. XMI - XML Metadata Interchange Specification v1.2, http://www.omg.org/cgi-bin/doc?formal/2002-01-01. last visited May 2004
16. Gruber, T. R.: Ontolingua: A Mechanism to Support Portable Ontologies. Technical Report KSL 91-66, Stanford University, Knowledge Systems Laboratory, 1991
17. Dieter F.: Ontology-Based Knowledge Management. IEEE Computer, 2002(11), pp. 56-59
18. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P., Stein, L.A.: OWL Web Ontology Language Reference. W3C Recommendation (2004).
19. The OWL Services Coalition, OWL-S: Semantic Mark-up for Web Services v.1.0. Available at http://www.daml.org/services/owl-s/1.0/owl-s.pdf
20. SWRL Specification. http://www.w3.org/Submission/SWRL/

21. O'Connor, M., Knublauch, H., Tu, S.W., Grossof, B., Dean, M., Grosso, W., Musen, M.: Supporting Rule System Interoperability on the Semantic Web with SWRL. Gil, Y., Motta E. (Eds.): The Semantic Web - ISWC 2005. Lecture Notes in Computer Science, Vol. 3729. Springer-Verlag (2005) 974-986.
22. Jess Rule Engine, http://herzberg.ca.sandia.gov/
23. Jess Protégé Tab: http://www.ida.liu.se/~her/JessTab/JessTab.ppt
24. Jena, http://jena.sourceforge.net/

DOPA: A Data-Driven and Ontology-Based Method for Ad Hoc Process Awareness in Web Information Systems*

Meimei Li, Hongyan Li**, Lv-an Tang, and Baojun Qiu

National Laboratory on Machine Perception,
School of Electronics Engineering and Computer Science,
Peking University, 100871, P.R. China
{Limm,Lihy,Tangla,Qiubj}@cis.pku.edu.cn

Abstract. The knowledge in Web Information Systems (WISs) makes that business process can not be described by a fixed model. Due to not integrating with domain knowledge seamlessly, traditional workflow management technology can not manage the ad hoc process very well. In order to solve that problem, we propose a data-driven and ontology-based method to support process awareness. Using an ontology-based model we integrate domain knowledge and WIS structure into process. A set of reasoning rules are designed to regulate the task transfer. Learning is done from the cases to help users make decision. A smart tool used for ontology building and WIS generation is developed and we prove the feasibility of this method in a real Web-based application.

1 Introduction

The concept of Web Information System (WIS) was proposed in 1998[1]. As a new generation of information system, WIS obtains information resources and exhibits business data via Web technologies. In the early time, most WISs were developed by data-oriented approach. Early in 1990s, business process reengineering illustrates the increased emphasis on process. Thus system engineers are resorting to a process-oriented approach to develop a WIS.

However, in some knowledge-intensive domains such as hospital, law and stock, the business process is ad hoc. That means the process model can not be defined in advance. Take a Hospital Information System as an example: doctors diagnose according to their own experience and patients' symptoms. The different illness and clinical experience make process different every time. It is impossible to design a model to deal with all the possibilities.

One of the reasons leading to that problem is ignoring the influence of data to business process. In knowledge-intensive WISs, each process is dynamically driven by the interaction of function and data. So, case handling trends to be

* Supported by Innovation Found For Technology Based Firms under grant number 06C26225110506.

** Corresponding author.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 114–125, 2006.

driven by data flow instead of totally by control flow. However, it is very crucial to describe those data and the relations between data and process. In fact, the data itself, the relations among them, their influence to process, etc, can be considered as knowledge. Further, the cases of process are a kind of knowledge too. A typical object describing knowledge is ontology. Many researchers have proposed some ontology-based process models [3][4]. However, the research is still in its infancy although most methods make progress in different aspects.

Another important matter is describing process in Web environment. WIS has its own components, which may be Web pages, navigators, messages, etc. Thus, the tasks of a process, transfer between tasks, and the constraints, etc, need depend on those special components to carry out. The process finished by different users is also a concentration but not only the process done by one user.

In order to support the process awareness in WISs in a better way, especially the ad hoc processes in knowledge-intensive domain, we adopt a data-driven and ontology-based method, DOPA in short, and the chief features are:

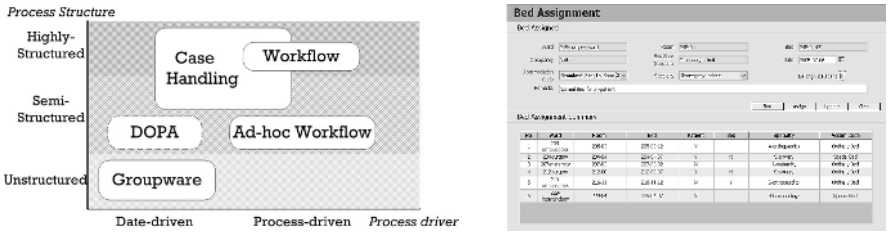
- Knowledge-supported: A concept model described by ontology combines the process closely with related knowledge and WIS structures;
- Data-driven: Depending on data transmission and estimation, support the dynamic processes;
- Rule-restricted: A set of rules are used to make sure the process is correct and effective;
- Tool-visualized: The graphical tool supports the "What you see is what you get" in model building.

The remainder of paper is organized as follows: section 2 presents some methods of process awareness; section 3 describes WIS Ontology including DOMAIN Ontology, STRUCTURE Ontology and PROCESS Ontology; section 4 explains how the case is handled through reasoning and learning from WIS Ontology; section 5 introduces experiment and application; at last, section 6 summarizes the paper and discusses future work.

2 Related Work

Process-Aware Information Systems (PAISs) support business processes in organizations based on explicit knowledge of both the organization and the processes [14]. Four typical process-aware methods are: Workflow, Groupware, Ad-hoc Workflow and Case Handling. Figure 1(a) compares them from two dimensionalities: process structure and process driver.

In terms of "process driver", Workflow and Ad-hoc Workflow drive the process according to the control structure defined in models while Groupware takes data as the basis of process. But due to its weak abilities on control, Groupware can not adapt to the complexity in WIS very well. From the dimensionality of "process structure", the highly-structured model in Workflow must be designed in advice. The main idea of Ad-hoc Workflow to support the semi-structured process is manually changing the process by end users, such as Caramba [8]. This method increases the workloads. Compared with others, Case Handling supports



(a) (b)

Fig. 1. (a) Process-aware Methods. (b) The Web page of Bed Assignment.

both semi and highly structured processes driven by data and process [9]. But it may be a trouble for Case Handling if the processes are ad hoc structured.

Although the process-aware methods have evolved rapidly, unfortunately, they seldom consider the domain knowledge. In order to integrate the knowledge, many people are engaged in building the ontology-based process models. The model created by Gianluigi Greco [4] provides a semantic vision of the application context and of the processes themselves. But, it belongs to the "Workflow" method mentioned in Fig.1(a), so it is hard to handle ad hoc processes.

At the same time, much work has been done to support BPM in Web application [5][7][10][11]. OOHDM and UWE base on UML to describe the control flow among operations done by one user in a time. WebRatio is a tool to develop data-intensive Web application using ontology [3][5]. It adopts Web Modeling Language (WebML)[6] integrating process modeling into the design of Web application in hypermedia framework. WSDM is equipped with a Task Modeling step and defines various relations among tasks corresponding to each possibility.

However, those examples are oriented toward simple personal applications, whose primary function is to exhibit data. Compared with the web sites, WIS focuses on more complex operations of data, abundant domain semantic and business logic. Besides, although some of them e.g. WebRatio extends the data model it is still limited to metadata. But in WIS data are so important for process that they can not be ignored.

3 WIS Ontology

WIS Ontology includes *DOMAIN Ontology*, *STRUCTURE Ontology* and *PROCESS Ontology*. The DOMAIN Ontology describes the domain data and relations between them, while the STRUCTURE Ontology describes the structure of WIS components and PROCESS Ontology describes such a task network that complies with the business logic and routine.

3.1 DOMAIN and STRUCTURE Ontology

The details of DOMAIN and STRUCTURE Ontology in WIS development can be found in [12]. Here is only a general picture:

- **DOMAIN Ontology**: It is used to describe the objects in a special domain and relations between them. For example, in hospital, each person corresponds to an organization like pharmacy, surgery or orthopaedics. The drugs and equipments are distributed to some organizations or persons. This kind of knowledge is presented by DOMAIN Ontology. Because it is domain-specific, it should refer to the domain standards, e.g. the Reference Information Model (RIM) in HL7 [15].
- **STRUCTURE Ontology**: In WIS, Web pages are the interfaces to communicate with people. Through the pictures, text fields, tables and buttons, etc, they tell people what they have known and what they can do. The layout of such elements is the structure of a Web page. The navigations between the Web pages lead users to implement functions step by step. Those pages and navigations consist of the structure of WIS.

3.2 PROCESS Ontology

In order to illustrate the scenario, an example in hospital is given first. Fig.1(b) is the page of Bed Assignment in the internet Total Hospital Information System (iTHIS) [2]. The page indicates a task in the hospital process and implicates much knowledge. The textfield "Bed" in it is a basic item in STRUCTURE Ontology which has the attributes such as type, size, position and constraint. The item corresponds to a data object "Bed" in DOMAIN Ontology.

In WIS, tasks like "Bed Assignment" of processes are implemented by a set of Web pages connected by navigations. But whether the task is triggered and when is implemented rely on the state of data objects. As a result, PROCESS Ontology should base on DOMAIN Ontology and STRUCTURE Ontology. The definitions are as following:

Definition 1. *The business process **BP** is a 5-tuples,* $\mathbf{BP} =< \mathbf{TS}, \mathbf{DS}, \mathbf{RS}, \mathbf{ST}, \mathbf{FT} >$*, where TS, DS and RS are respectively a set of tasks, data and relations between tasks. Relations represent the fore-and-aft order of two tasks. They are not defined in advance but during the running time dynamically. ST means the task at the beginning of BP, e.g. Patient Admission and FT means the final tasks, e.g. Leave hospital. They comply with the constraints* $Cont(ST) = 1; Cont(FT) \geq 1$ *Cont(s) is the function to get the element number of s. In the whole process, all data is created by tasks, so the DS is the union of GD in each task (see definition 2), which is* $DS = \bigcup_{\forall t \in TS}(t.GD)$.

Definition 2. *The task **T** is a 6-tuples,* $\mathbf{T} =< \mathbf{TN}, \mathbf{TC}, \mathbf{GD}, \mathbf{MD}, \mathbf{P\_S}, \mathbf{Ro} >$*, where TN is the name of the task, TC means the indispensable conditions for implementing the task. GD is the data generated after running, e.g. the information of bed assignment while MD is the data needed for running, e.g. the information of admission and charge. P_S is the Web Page Set to implement task function, and Ro indicates the legal role, e.g. nurse can assign the bed. For special tasks, there are* $Cont(ST, MD) = 0; \forall t \in TS \wedge t \neq ST, Cont(t.MD) \geq 1$*; And the GD of a task is considered as the union of PGD in each page of task's P_S, so* $GD = \bigcup_{\forall p \in P\_S.PS}(p.PGD)$.

Definition 3. *The Web Page Set* ***P_S*** *is a 3-tuples,* $\mathbf{P\_S} =< \mathbf{PS}, \mathbf{LS}, \mathbf{SP} >$, *where PS is the set of all Web pages in P_S, and LS is the set of all navigations between pages in PS. SP means the page at the beginning. From SP, all the other pages in PS must be arrived through navigations in LS, so* $\forall p \in P.PS \wedge p \neq SP, \exists(p_i, l_i), Link(SP.o_{k0}, p_1.o_{k1} \in LS \wedge Link(p_j \cdot o_{kj}, p_{j+1} \cdot o_{k(j+1)} \in LS \wedge Link(p_n \cdot o_{kn}, p \cdot o_k) \in LS (j \in (2, n-1))$. *Link (a,b) is a navigation from a to b. What is more, in the same period, all the pages can be accessed only by the same user.*

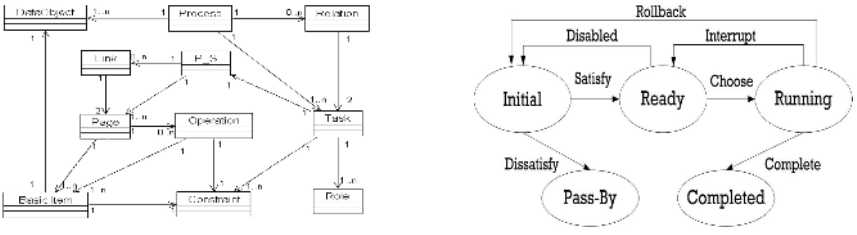
Definition 4. *The Web page* ***P*** *is a 4-tuples,* $\mathbf{P} =< \mathbf{BIS}, \mathbf{OS}, \mathbf{PGD}, \mathbf{PAD} >$, *where BIS and OS are respectively set of basic items, like droplist of specialty, and operation items like button of assign. PGD means the data generated after running the page and PAD is the accessorial data to give some help, like information in the table of Bed Assignment Summary. The operations in a page have different types. The data objects related to SELECT or INITIATE are used to provide information to users so that they belong to PAD. While INSERT and UPDATE either create a new object instance or give a new value to some data object, so the data objects related to them belong to PGD. Some data objects can belong to both, such as the PGD of INSERT can be the PAD of UPDATE. So there are* $PAD = \bigcup_{\forall bi \in BIS}(bi.D) - PGD$; $PGD = \bigcup_{\forall o.OTY = `Insert' or `Update'}(o.OBIS.D)$; $\forall p$, $\exists o_1, o_2(o_1 \in p \wedge o_2 \in p \wedge o_1.OBIS = o_2.OBIS \wedge o_1.OBIS \subset p.PAD \wedge o_2.OBIS \subset p.PGD)$.

Definition 5. *The operation item* ***O*** *is a 5-tuples,* $\mathbf{O} =< \mathbf{ON}, \mathbf{OBI}, \mathbf{OC}, \mathbf{OTY}, \mathbf{OBIS} >$, *where ON is the name of the operation,OBI means the basic item on which the action is triggered, OC indicates the execution conditions, e.g. for operation of assignment OC is that the values of bed, specialty, room and ward can not be NULL, OTY is the type of O, which are mainly the operations to Database, and OBIS is the execution parameters, e.g. droplist of specialty is one of parameters of operation "Assign". For all sets in the definition, there are* $\forall s \in \{TS, DS, PS, BIS, OBIS\} Cont(s) \geq 1$.

Definition 6. *The navigation* ***L*** *is a 4-tuples,* $\mathbf{L} =< \mathbf{P_{Sou}}, \mathbf{P_{Tar}}, \mathbf{LC}, \mathbf{LO} >$, *where* P_{Sou} *is the current Web page showed to user, and* P_{Tar} *is the Web page showed after the navigation, LC is the condition deciding the direction of navigation, such as whether the operation successes or not, and LO is the operation item on which L is triggered. E.g. the operation "update" is triggered, and if the action successes, it will return to the same page with new values generated by updating.*

Definition 7. *The Relation* ***R*** *is a 4-tuples,* $\mathbf{R} =< \mathbf{R_{Sou}}, \mathbf{R_{Tar}}, \mathbf{RD}, \mathbf{RO} >$, *where* R_{Sou} *is the former task which is implemented first,* R_{Tar} *is the latter task which is triggered after* R_{Sou} *is finished, RO is the trigger operation and RD is the data transferred from* R_{Sou} *to* R_{Tar}. *After a task is finished, the MD and the PAD in the SP of the* R_{Tar} *should be transferred for further estimation and implementation. As a result, there is* $RD = t.MD \bigcup t.P\_S.SP.PAD$.

In order to illustrate the elements and their relations of the definition above clearly, Fig.2(a) uses a UML model to describe the PROCESS Ontology.



(a) (b)

Fig. 2. (a) UML Model of PROCESS Ontology. (b) Running States of a Task.

4 Reasoning and Learning from WIS Ontology

As we have said that the case is not driven by a well-defined model, then how to establish the relations among tasks during the running time? A set of rules are established for that problem. Certain rules are established by domain experts. While others are reasoned according to the data conditions or recommended through learning.

4.1 Reasoning

When a task is completed, case must transfer to others. During this process, four steps would be taken one by one:

1. Find out tasks which may run after task t as candidate task set (*CandidataTasks(t)*);
2. For each *ct* in CandidateTasks(t), test whether the generated data objects meet TC of *ct* ;
3. Get rid of ungratified ones from CandidateTasks(t), and post task set *PostTasks(t)* is left;
4. Transfer data to every *pt* in PostTasks(t) and send a request.

All the four steps will cause the transformation of running state. After step 1, *ct* is initiated. Estimated by step 2, *ct* is passed if result fails. Otherwise, it turns to be ready. The ready *ct* becomes to *pt*. According to the strategy of distribution, when request is chosen by a worker and resources are all prepared, *pt* begins to run. The state transformation of a task in the life cycle is shown in Fig.2(b).

Rules are used to express the semantic of each state transformation, and are described as logic sentences, e.g. $a \Rightarrow_A b$, where a is the condition of the rule, b is the result and A means the action needed to be done. In step 1, it needs to find what is *ct*. If there is certain relation between *t* and *ct*, the rule is like Rule-1, and $\mapsto$ means it is the compelling rule. Otherwise, frequent choices of other users can give you some advice, and you can choose a better one or not from those advices, so $\rightarrow$ means it is an optional rule, like Rule-2.

- **Rule-1**
  Complete (t) $\mapsto_A$ Ready (ct), A is: **(1)** Put *ct* into PostTasks(t);**(2)** Transfer RD from t to ct ; **(3)** Send a request and wait in ct.TaskList; **(4)** Put $< t, ct >$ in Task-Path.
- **Rule-2**
  $Complete(t) \rightarrow_A Ready(ct)$, A is the same as Rule-1.
  If there are not rules like Rule-1 or Rule-2, we will choose *ct* according to state of data objects. Instead of all tasks, we choose a subset as *ct* whose MD intersects with TGD of t .The feasibility can be explained in two aspects:(1) in the view of t, TGD implies changes of data and the new state will affect the future process; (2) in the view of *ct*, if MD only contains those data unchanged for a long time, the task may be finished or has less possibility to run again. So, the more potential tasks are the ones whose MD includes new information. Rule-3 to Rule-5 indicate this situation.
- **Rule-3**
  $ct.MD \bigcap t.TGD \neq \Phi \Longrightarrow Initiate(ct)$.
- **Rule-4**
  $Initiate(ct) \wedge ct.TC$ is contented $\Longrightarrow_A Ready(ct)$, A is the same as Rule-1.
- **Rule-5**
  $Initiate(ct) \wedge ct.TC$ is not contented $\Longrightarrow Pass(ct)$
  RD will be transferred from t to ct if ct is ready. There are various manners to carry the point, and the typical one is using database. Task t stores its TGD in database from which *ct* fetches what it wants. Each task maintains a $TaskList$ to save the requests of cases. After *ct* is confirmed, a 2-tuples $< pre\_task, post\_task >$ is added to the task sequence Task-Path that case has passed through. The transformation from Ready to Running is described by Rule-6.
- **Rule-6**
  $Ready(ct) \wedge$ All resource are prepared $\wedge$ request of case is chosen$\Longrightarrow$
  $Running(ct)$, A is **(1)** Tag request in ct.TaskList to inform others that request is replied;**(2)** Jump to ct.SP to run the task.
  Tasks do not run as well as we expect. Abnormities usually happen, such as human interruption. Therefore, we save those temporary data for restarting like Rule-7. Another problem is caused by "DELETE" operation. Elimination of the existing data in the former task may lead rollback or collapse of latter task. Rule-8 and rule-9 are used to solve this problem. But if the latter one has finished, no influence will be exerted.
- **Rule-7**
  $Running(ct) \wedge$ user triggers to interrupt the task $\Longrightarrow_A Interrupt(ct)$, A is **(1)** Store the data which have been generated; **(2)** Release the resources which have been occupied.
- **Rule-8**
  $Ready(ct) \wedge RO.OTY ==$ "DELETE" $\Longrightarrow_A Initiate(ct)$, A is Remove request from ct.TaskList.

– **Rule-9**
$Running(ct) \wedge RO.OTY ==$"DELETE" $\Longrightarrow_A Ready(ct)$, A is **(1)** Delete all the data which have been generated by ct; **(2)** Remove request from ct.TaskList.

4.2 Learning

In Task-Path, a sequence of 2-tuples records the trace of a case, such as $\{< t0, t1 >, < t1, t2 >, < t2, t5 >, < t5, t10 >, < t10, t1 >, < t1, t3 >, < t1, t4 >, < t3, t5 >, < t4, t6 >, < t4, t7 >, < t5, t8 >, < t8, t9 >, < t6, t8 >, < t8, t9 >\}$. In ordinary workflow log, this example will be $\{t0, t1, t2, t5, t10, t1, t3, t4, t5, t8, t9, t6, t8, t9\}$. It only describes the order of tasks but not the "Relation" between tasks. Note that, it is easier to discover priority, parallel and circle from Task-Path than workflow log. Through synthesis of different Task-Paths, we aim to get a conceptual model from which the system can learn new rules to help users make decisions. Unlike strict model such as Petri-net, our model dose not emphasize whether each junction is AND/OR Join or AND/OR Split, but adopts a set of weights to illuminate the parallel degree. If the weight is larger than the value given by user, the path segment will be added into the rule set for reasoning. Fig.3 is a simple example. First, Task-Path needs preprocessing. It aims to:(1)Simplify circle: Replace the segment of circle with a node, thus the model is converted to a directed acyclic graph (DAG);(2)Get rid of dead nodes: The second element of 2-tuples may not be implemented in reality (see Fig.2(b)). In DAG, dead nodes are the leaf nodes which are not final tasks; (3) Mark the running order to help analyze AND/OR logic. Fig.4(a) is the model of Task-Path mentioned at the beginning of this section and Fig.4(b) is the result of preprocessing.

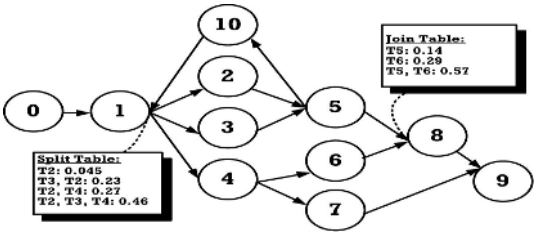


Fig. 3. The model is a directed graph. The node is task, and the edge is Relation between tasks. In each node, there are a split table and a join table, recording the weight of AND/OR logic. E.g. "$T2, T4 : 0.27$" means the weight of the situation that T2 and T4 implement together is 0.27.

Next, depended on the models in Fig.4 (b), we generate the model in Fig.3. The method of breadth-first search is used to each model, and the split table of parent node is edited due to the fact that all brother nodes are AND split. For join logic, if the order of some join edge is larger than that of some split edge, e.g. 10 of $< t6, t8 >$ is larger than 9 of $< t8, t9 >$, the logic of t8 is OR join, otherwise it is AND join. The algorithms of pretreatment and model generation are in Fig.5.

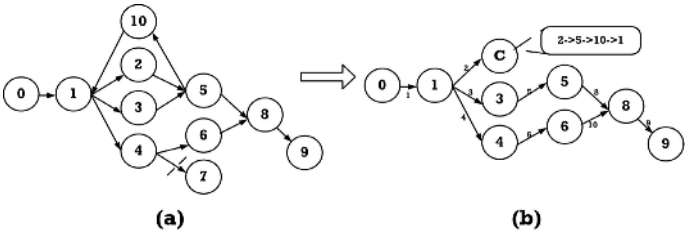


Fig. 4. (a) Model of Task-Path. (b) Result of Preprocessing.

```
PreProcess (Task-Path): Graph as Fig.4(b)
// node of graph is Node=<Taskname, PostList>
// element of PostList is <Node, order>
1.  Path' = Simple_Circle(Task-Path);
2.  Root = Node (t0, null vector);
3.  Graph = Initial (Root);
4.  for each 2-tuples ri=<pre,post> in Path' , do
5.      Pre = Find (pre);
6.      Post = Find (post);
7.      if Post = null then
8.          Post = Node(post, null vector);
9.          Pre.PostList.add(Post,i);
10.     else if Post in Pre.PostList
11.         Pre.PostList.set(Post, min(i,order of Post));
12.     else
13.         Pre.PostList.add(Post,i);
14.     endif
15. endfor
16. for each leaf node ln in Graph' do
17.     if ln is not the final task then
18.         while ln is not leaf node do
19.             Parent(ln).PostList.remove(ln);
20.             Delete(ln);
21.             ln = Parent(ln);
22.         endwhile
23.     endif
24. endfor
25. return Graph;
```

```
GenerateModel(Graph, start task, final tasks):
Process_Graph as Fig.3
// node of Process_Graph is Process_Node =
//<Taskname, Split table, Join table>
1.  Process_Root = Process_Node(t0,null vector, null
    vector);
2.  Process_Graph = Initial(Process_Root);
3.  for each Graph of a case do
4.      n = Root;
5.      while n is not the last node in Graph do
6.          Update_Split_Table (n);
7.          for each child node cn of n do
8.              if cn is a circle then
9.                  GenerateModel(cn,n,n);
10.             else
11.                 Update_Join_Table (cn);
12.             endif
13.         endfor
14.         n = cn;
15.     endwhile
16. endfor
17. return Process_Graph;
```

Fig. 5. Algorithms of Preprocessing and Model Generation

5 Experiments and Application

In order to verify the feasibility of DOPA, we combine experiment and application together. Our main experimental environment is the project WISE which aims to construct WIS automatically from users' view. The main parts of WISE are:

WISE Builder: A visual graphic tool helps the engineers build WIS Ontology. It supports "What You See Is What You Get" when defining model [12];

Data Mapper: A data mapping module used to generate the SQL codes for data access and manipulation. All the actions of SELECT, INSERT, UPDATE and DELETE are automatically carried out with the help of this part [13];

Code Generator: A powerful component aims to generate most of codes in target system.Its function includes: Convert the models to Web pages and generate function modules. What is more, it assists Process Builder to generate process engine, etc;

Process Builder: A useful part defines PROCESS Ontology and rules, and creates the process engine with reasoning and learning ability. The engine embedded in target system will support the process-awareness in WIS. Fig.6(a) shows the architecture of WISE.

Our experiments focus on a simple process in hospital with seven single-page tasks: User Login, Main Page, Patient Admission, Bed Arrangement, Diagnose, Blood Test, and Leave Hospital. To compare, three experiments have been done independently:

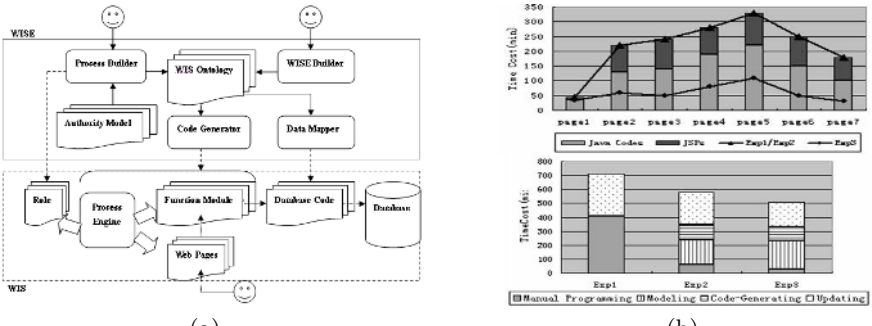
Experiment 1: Two professional programmers with 3 years experience on JSP and WIS develop on IBM Websphere Studio Application Developer 5.0, the requirement documents and design details are provided;

Experiment 2: A medicine student who is familiar with the application domain and a professional programmer depends on a workflow management tool;

Experiment 3: All the conditions are the same as experiment 2 except that the tool is WISE.

The three experiments share the same database and hardware environment. The time cost can divide into two parts: the time of generating function pages and generating process engine. The top figure in Fig.6(b) records the time cost for generating the seven Web pages and we suppose that the WFM tool in Experiment 2 don't support the auto-generation of function modules. The time cost for generation of process engine is shown in the bottom figure in Fig.6(b) and consist of manual programming time, modeling time and code-generating time. It also shows the time needed for ten times changes in process.

Note that, Experiment 3 generates JSPs and Java Codes automatically, so the actual workloads are reduced much. The time costs for process generation in



(a) (b)

Fig. 6. (a) Architecture of WISE. (b) Time Cost in Experiments (Unit: min).

three experiments are similar, but the time for updating is different. Although the time for each change in Experiment1 or Experiment 2 is not long, ten times of the time are needed for ten times of changes. However, in Experiment 3 some changes can be reasoned through rules so maybe only 6˜7 times of changes are needed in ten times. Thus the whole time is less than the other two.

6 Summary and Future Work

DOPA is a novel method for ad hoc process awareness in knowledge-intensive WIS. This paper delves deeply into modeling process ontology and handling cases through reasoning and learning. The chief contributions include: propose an ontology-based model for business process which integrates domain knowledge and WIS structure; present a data-driven method to implement process awareness through reasoning from PROCESS Ontology based on a set of rules and learning from cases to help users make decision; and develop a smart tool used for ontology building and WIS generation.

The research on supporting the ad hoc process awareness in WIS is just at beginning and future efforts are oriented to the following aspects:

1. Improve the abnormity handling: dead circle is an instance of abnormity. Further, wrong operations of users or the disconnection of internet will also disturb the process. So rules need be refined;
2. Enhance the reasoning and learning: more knowledge will be imported to control the process so reasoning will be more powerful such as supporting ontology query. Moreover, the results of learning can be used to assist analysis and optimization;
3. Establish the verification: because the process is ad hoc, verification is needed to pledge the correctness of the resulting process, especially for the hospital business processes which require the steps to be verified in order to ensure the suitable patient care.

References

[1] Tomas Isakowitz, Michael Bieber, Fabio Vitali: Web Information Systems. Communications of the ACM, Vol.41(7), A CM Press (1998) 78-80.
[2] Hongyan Li, Ming Xue, Ying Ying: A Web-based and Integrated Hospital Information System. In Proceedings of IDEAS04-DH, China, (2004) 157-162.
[3] Marco Brambilla: Extending Hypertext Conceptual Models with Process-Oriented Primitives. Lecture Notes in Computer Science Vol.2813, (2003) 246-262.
[4] Gianluigi Greco, Antonella Guzzo, Luigi Pontieri, and Domenico Sacc: An Ontology-Driven Process Modelling Framework. In proceedings of International Conference on Database and Expert Systems Applications-DEXA, Zaragoza, Aug. 30-Sep. 3 (2004).
[5] Piero Fraternali: Tools and Approaches for Developing Data-Intensive Web Application. ACM Computing Surveys, Vol.31, No.3 (1999).

[6] Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): a modeling language for designing Web sites. WWW9/Computer Networks 33(1-6) (2000) 137-157.
[7] Hans Albrecht Schmid, Gustavo Rossi: Modeling and Designing Processes in E-Commerce Applications. IEEE International Computing Vol.8 (1): JAN-FEB (2004) 19-27.
[8] Dustdar S: Caramba-A Process-aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Team. Distributed and Parallel Databases Vol.15(1), (2004) 45-66.
[9] Wil M.P.van der Aalst, Mathias Weske, Dolf Grnbauer: Case Handling: A New Paradigm for Business Process Support. Data & Knowledge Engineering Vol.53(2), (2005) 129-162.
[10] Olga De Troyer, Sven Casteleyn: Modeling Complex Processes for Web Applications using WSDM. In Proceedings of the 3rd International Workshop on Web-Oriented Software Technologies, (2003) 27-50.
[11] Nora Koch, Andreas Kraus: The Expressive Power of UML-based Web Engineering. In Proceedings of the 2nd International Workshop on Web-Oriented Software Technologies, IWWOST, (2002) 105-119.
[12] Tang Lv-an, Li Hongyan, Pan Zhiyong, Tan Shaohua: PODWIS: A Personalized Tool for Ontology Development in Domain Specific Web Information System. LNCS Vol.3399 , (2005) 680-694.
[13] Hongyan Li, Ming Xue, etc: Understanding User Operations on Web Page in WISE. In proceedings of 6th International Conference in Web-Age Information Management, LNCS Vol.3739 (2005) 846-851.
[14] Wil M.P. van der Aalst, Michael Rosemann, Marlon Dumas: Deadline-based Escalation in Process-Aware Information Systems. BPM Center Report, (2005), http://www.BPMcenter.org.
[15] HL7 Homepage: http://www.hl7.org.

A Transaction-Aware Coordination Protocol for Web Services Composition

Wei Xu, Wenqing Cheng, and Wei Liu

Department of Electronics and Information Engineering,
Huazhong University of Science and Technology, 430074, Wuhan, P.R. China
{Xuwei, Chengwq, Liuwei}@hust.edu.cn

Abstract. With the rapid development of WWW, Web Service is becoming a new application model for decentralized computing based on Internet. However, the tradeoff problem between consistency and resource utilization is the primary obstructer for building a transactional environment for Web Services Compositions. Since a resource may not be acceptable to lock exclusively by an unknown Internet user, we propose a Transaction-Aware Tentative Hold Protocol (*taTHP*) to perceive the transaction context information and play a more active role in transaction coordination. With the capability of forecasting the will-succeed transactions in a fairly small scope of candidates, taTHP is able to achieve more resource utilization with smaller complaints about the transaction coordination. Finally, a comprehensive comparison is carried out to demonstrate the improvement of the proposed protocol. And it can be concluded from the result that taTHP is provided with better efficiency and satisfactory quality of service.

1 Introduction

Web Services are nowadays emerging as a major technology to construct the next generation distributed computing environment. However, strict ACIDity (Atomicity, Consistency, Isolation, Durability)[1] in traditional transaction management is not appropriate for such a world of autonomous Web Service transactions, where the resources may not be acceptable to lock exclusively for such long period.

To solve this problem, a number of approaches have been made to propose some new specific protocols such as TIP[2], BTP[3], WS-T[4] and THP[5][6]. Among them, THP is more promising for its unique policy in isolation relaxation of transaction. Based on the idea of "tentative reservation", THP grants maximum autonomy of isolation to all the participants. It allows multiple clients to place tentative holds on the same resource item and verify availability before completing the transaction. Whenever one client completes the purchase of this item, the others may receive notifications that their holds are no longer available. Exchanging the negotiation information prior to the actual transaction itself, THP decreases the effect of outdated data, reduces the potential for cancellations, and improves the odds of successfully completing transactions.

However, granting exorbitant reservations of the same item will result in less satisfactory quality of service. It blocks customers to look for other optional services

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 126 – 131, 2006.

by giving the appearance of successful reservation. In order to avoid this situation, several constraints, which restrict the number of holds and the length of duration, are introduced to THP to achieve better effect of transaction coordination. However, there is still a problem unmentioned in those efforts, i. e. which kind of requests should be granted or rejected.

Hence, an improved transaction protocol based on THP, namely taTHP, is proposed in this paper. Being aware of the *Success Probability* (SP) of each transaction, the service provider is able to select "proper" transactions in a small scope of candidates to increase the resource utilization without complaints.

The rest of this paper is organized as follows: The next section presents the problem of THP. Subsequently, Section 3 discusses the computation model and coordination algorithm of taTHP in detail. And in Section 4, we demonstrate the efficacy of the proposed protocol through the simulation. Finally, Section 5 concludes the paper.

2 Problem of THP

As mentioned above, the *Overhold Size* H_{Size} and the *Hold Duration* $H_{Duration}$ govern the behavior of THP. The first parameter is the length of reservation queue of each resource item. Big H_{Size} grants a high degree of isolation relaxation, which expands the scope of transaction candidates to achieve better resource utilization. It is certain for THP to get better $N_{success}$ (number of success transactions) than strict transaction models.

However, with the increase of H_{Size}, more transactions are given the appearances of successful reservation instead of the potential competition. As a result, many transactions are blocked to receive timely rejection notifications from the unacquirable services. It causes the increase of *Average Lifetime of Failure Transactions*, namely T_{ALTF}, which has great negative effect on the customer's satisfaction of transactional service.

Another parameter $H_{Duration}$ is introduced to THP to keep reservation queue healthy. Small $H_{Duration}$ is helpful to get short T_{ALTF}. But $N_{success}$ will decrease significantly due to the strict time limit. It is more suitable for $H_{Duration}$ to prevent services from malicious attacks rather than improve the efficiency of coordination.

Now, the question is how to achieve better $N_{success}$ with lower T_{ALTF}. Obviously, the FRFR (*First Request, First Reservation*) policy of THP is not suitable for this target. A good coordination protocol should have the capability of picking up will-succeed transactions in a fairly small scope of candidates.

3 Transaction-Aware THP

As discussed above, THP is unable to tell which kind of requests should be granted or rejected due to the lack of the awareness of transaction context. To overcome this shortcoming, a Transaction-Aware protocol and its reservation policy are presented in this section. It enables THP forecast the success probability of each incoming transaction.

In THP, the SP of a Web Service transaction is determined by two factors: the commit-time of the transaction and the *Resource Acquiring Probability* (RAP) of each service. To the first factor, early commitment of a transaction makes it avoid resource competition and get preemption of services as soon as possible. Unfortunately, this factor is unpredictable and depends on the decisions of individual customers. Relatively, the second factor RAP, which is the integration of RAPs of all the services in a transaction, is more calculable than the forecast of the commit-time. The smaller RAP is, the less opportunity exists for the transaction to succeed in the reservation competition.

Therefore, being aware of the SP of a transaction, taTHP enables a service provider to choose suitable transaction requesters, provide them with its resources and make a wiser decision in an active bi-directional interaction pattern in Web Service transaction coordination.

3.1 Success Probability Model

In order to give the computation model of success probability of a transaction, several denotations are introduced firstly in this section. A Web Service transaction is normally denoted by the symbol T, which consists of a set of Web Service resources r_i, $i \in \{1,\ldots, m\}$. For ith resource r_i, the number of the resource items is n_i, which is the sum of the used number u_i (t) and the available number a_i(t) at time t. We also denote the number of reservation holds at time t by h_i(t) and the biggest number of transaction candidates is no more than $a_i(t)*H_{Size}$. In addition, while there isn't any available resource item, the RAP P_{ri}(t) of the resource r_i is set to zero.

Obviously, the success probability P_T(t) has a close relationship with the P_{ri}(t) of each service involved in transaction T. P_{ri}(t) can be achieved as follows:

$$P_{ri}(t) = \begin{cases} a_i(t) / h_i(t) & (h_i(t) \neq 0) \\ 1 & (h_i(t)=0,\ a_i(t) \neq 0 \text{ or } a_i(t) > h_i(t)) \\ 0 & (h_i(t)=0,\ a_i(t)=0) \end{cases} \quad (1)$$

Each service provider has the casting vote to the success of transaction by recalculating RAP on demand and the success probability of transaction can be defined as an integration of RAPs of all the services in a transaction. It can be defined as the following equation:

$$P(t) = \prod_{i=1}^{m} Pri(t) \quad (2)$$

Normally, a transaction with high success probability P_T(t) is more competitive than low ones. "Bad" transactions always consist of some exiguous services, which are the bottleneck of all the transactions. It is not advisable for any service provider to participant a transaction including this kind of services. With the proposed Transaction-Aware policy, service provider is able to choose transactions and make a more wise decision than passive manner.

However, there is still a problem in dealing with "long" transactions. From (2), it can be concluded that a long transaction with many service reservations has much lower success probability than a short one. The new policy is likely to form a

discrimination against long transactions, which is not the original intention of the proposed protocol. Hence, an improved computation model is defined as following:

$$P'_T(t) = min\{P_{ri}(t)\},\ i \in \{1,\ldots, m\} \tag{3}$$

The improved success probability forecast function $P'_T(t)$ is the *Minimal Resource Acquiring Probability* (MRAP) in the transaction. That means, the service providers pay more attention to the worst resource acquiring probability rather than the length of the transaction. Accordingly, we seldom use equation (2) to compare the success probabilities between transactions unless they have the same MRAPs.

3.2 Algorithm in Coordination

The coordination algorithm of taTHP is different from THP on the aspect of transaction-aware policy in the phase of reservation. In THP, when the number of reservation holds reaches the overhold size, further requests will be abandoned. But in taTHP, the further requests of transaction T_i will be compared with the transaction $T^* \in T_{Queue}$ in the reservation queue and be decided which one to doff. The following simplified pseudo-code is the coordination algorithm executed in each service provider.

```
Coordination Algorithm
begin
  for(each reservation q_i of transaction T_i)do
    for(each available resource r_k)do
    begin
      if(h_k(t)< H_Size)then
        grant the hold to transaction T_i;
      if(a_k(t)=0)then
        roll back T_i and withdraw all q_n in T_i;
      if(h_k(t)= H_Size)then
      begin
        get the T_Queue of resource r_k;
        update P'_T*(t), T*∈ T_Queue;
        for(each transaction T*∈ T_Queue)do
           if( P'_T(t) > min(P'_T*(t)))then
           begin
              grant reservation hold to T_i;
              roll back T* and withdraw all q_n in T*;
           end
           else if(P'_T(t) = min(P'_T*(t))) AND
                    P_T(t) > P_T*(t))then
           begin
              grant reservation hold to T_i;
              roll back T* and withdraw all q_n in T*;
           end
           else
              roll back T_i and withdraw all q_n in T_i;
       end;
    end;
end.
```

4 Simulation and Evaluation

Simulation analysis results are given in this section to demonstrate the performance of the proposed taTHP. We consider the typical scenario of a business trip where a customer requires an airline flight, a rented car and a hotel service independently. And most of these services can be found and booked though Internet. The customer who takes a business trip such as attending an international conference requests the tentative reservations on these three items. But, he will not confirm this order until being satisfied with the entire package. In most of cases, customer will spend a long time on comparing his choices with other services in respect of price, service quality, delivery time, etc. And the duration of his decision-making is unpredictable. In the simulation, the system initializes 200 transaction instances of 20 different transactions in 40 time epochs, which follow a Poisson process with the average occurrence rate of transactions equals to 5. To facilitate the simulation, we assume that each service provider shares the same overhold size and hold duration parameters, which change in a uniform manner. And the length of transactions is randomly chosen from 1 to 5. The duration of each transaction instance is also constructed randomly with the uniform distribution of parameters 1 to 40.

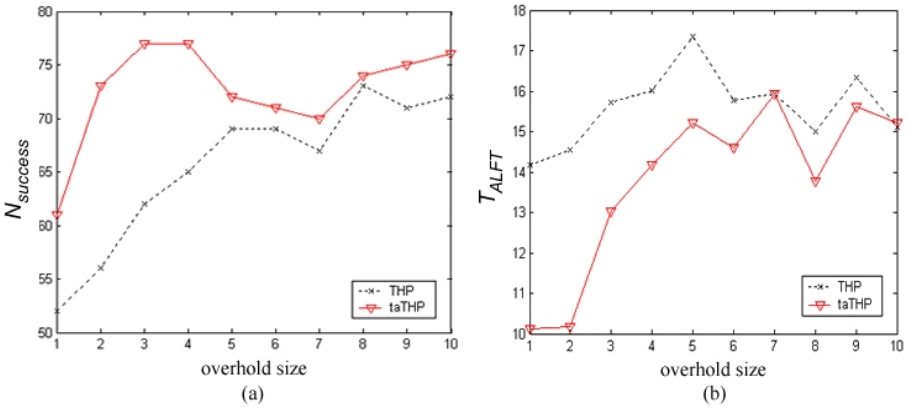


Fig. 1. Comparison between THP and taTHP

In Figure 1, the comparison is carried out between THP and taTHP. Obviously, the transaction-aware protocol is more powerful than THP on the aspects of $N_{success}$ and T_{ALTF}. From the plot (see Figure 1a), it can be observed that the curve of $N_{success}$ of taTHP is cliffier than that of THP in the beginning of the increase of H_{Size}. Hence, the new policy of choosing will-succeed transactions in a small scope of candidates achieves more significant improvement than the fixed FRFR policy when the H_{Size} is limited to a low level. With the continuous increase of H_{Size}, the difference disappears for the wide range of transaction candidates.

Furthermore, Figure 1b shows the comparison of average lifetime of failure transactions. It can be observed that better result of T_{ALTF} can be achieved before the H_{Size} reaches 7. The proposed policy of taTHP helps the service provider find out "bad" transactions with small success probability and force them to withdraw

immediately. But in THP, most of these transactions keep the resource occupations until they get failed in the competition, which leads to the increase of T_{ALTF}.

In conclusion, the obvious improvement lies on the result of reasonable forecast of success probability of each transaction. The Transaction-Aware Tentative Hold Protocol enhances THP significantly in both success-rate and satisfactory quality of transactions.

5 Conclusions

To satisfy the special requirements of Web Service transaction, the transaction participant should be endued with more rights to expose its attitude and decide its actions in transaction coordination. However, the coordination interaction is still a passive mode because the service provider knows nothing about the transaction context. Therefore, we proposed a Transaction-Aware coordination protocol based on THP to solve these problems. Being aware of the success probability of a transaction, the service provider is able to maintain a small scope of transaction candidates and select "proper" ones to achieve higher resource utilization and better quality of transactional service. Finally, simulation results confirm that the proposed taTHP performs significantly better than THP.

References

1. Gray J. and Reuter A. Transaction Processing: Concepts and Techniques. Morgan Kaufmann, San Mateo, CA, USA. 1993.
2. Evans K., Klein J., Lyo J. Transaction Internet Protocol - requirements and supplemental information. http://www.landfield.com/rfcs/rfc2372.html, 1998.
3. OASIS Committee Specification, Business Transaction Protocol, version 1.0, http:// www.oasis-open.org/ , May 2002.
4. Cabrera F., et al. Web Services Transaction (WS-Transaction). http://www.ibm.com/ developerworks/library/ws-transpec/, August 2002.
5. Roberts J. and Krisnamurthy S. Tentative Hold Protocol. W3C Workshop on Web Services, http://www.w3.org/TR/2001/NOTE-tenthold-1-20011128/, November 2001.
6. Park J. and Choi K-S. Design of an Efficient Tentative Hold Protocol for Automated Coordination of Multi-Business Transactions. In Proceedings of the IEEE Conference on E-Commerce. June 2003.

Unstoppable Stateful PHP Web Services

German Shegalov*, Gerhard Weikum, and Klaus Berberich

Max-Planck Institute for Informatics, Saarbruecken, Germany
{shegalov. weikum, kberberi}@mpi-inf.mpg.de

Abstract. This paper presents the architecture and implementation of the EOS^2 failure-masking framework for composite Web Services. EOS^2 is based on the recently proposed notion of interaction contracts (IC), and provides exactly-once execution semantics for general, arbitrarily distributed Web Services in the presence of message losses and component crashes without requiring explicit coding effort by the application programmer. The EOS^2 implementation masks failures by adding a recovery layer to popular Web technology products: (i) the server-side script language PHP run on Apache Web server, and (ii) Internet browsers like IE to deliver recovery guarantees to the end-user.

1 Introduction

A growing number of Internet businesses deliver mission-critical applications (stock trading, auctions, etc.) to their customers as Web Services. These applications comprise heterogeneous components distributed over multiple layers. They pose strong requirements for service and consistent data availability from both legal and business standpoints. Since many systems count many millions of lines of code, some bugs pass quality assurance undetected which leads to unpredictable service outages at some point.

Recovery in transactional systems guarantees: i) that an operation sequence declared as a transaction is executed *atomically* (i.e., either completely or not at all when interrupted by a failure) and ii) that completed transactions *persist* all further failures. Atomicity and persistence do not suffice to guarantee correct behavior of the applications. It is the application program that needs to handle timeouts and other exceptions, retries failed requests to servers, handles message losses, and prepares itself with a full suite of failure-handling code.

Incorrect failure handling in applications often leads to incomplete or to unintentional *non-idempotent* request executions. This happens because many applications that are *stateful* by nature, i.e., with a state remembered between consecutive interactions, are rendered *stateless*, where all interactions are independent for easier manageability. Consequently, timeouts and resent messages among application servers or Web Services may lead to unintended duplication effects such as delivering two tickets for a single-ticket purchase request, resulting in severe customer irritation and business losses. The standard solution in the TP and DBMS world requires all state information of the application to be managed in the database

* Current affiliation: Oracle USA, Inc., Oracle Portland Development Center, Oregon, USA.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 132 – 143, 2006.

or in transactional queues [5, 16], but this entails a specific programming discipline that is often viewed as an unnatural burden by application developers.

The *interaction contract (IC) framework* [3] provides a generic solution by means of integrated data, process, and message recovery. It *masks* failures, and allows programmers to concentrate on the application logic, greatly simplifying and speeding up application development. A challenge in implementing this kind of comprehensive multi-tier application recovery is to minimize the overhead of synchronous disk writes *forced* by the recoverability criteria.

This paper presents the *E*xactly-*O*nce Web *S*ervice platform EOS^2 for composite Web Services with recovery guarantees. EOS^2 is a major advancement over its predecessor EOS, described in [3, 18], and differs significantly from this earlier work. EOS^2 is able to replay arbitrarily structured *n*-tier PHP applications with interleaved accesses to shared data, whereas EOS was limited to two-tier applications with a frontend browser and a single PHP backend node accessed without race conditions.

1.1 Related Work

Recovery for general systems of communicating processes has been extensively studied in the fault-tolerance community (e.g., [13, 20, 9, 1, 10]), where the main focus has been to avoid losing too much work in long-running computations (e.g., scientific applications), usually using distributed checkpointing. Most of this work does not mask failures. Methods that do mask failures exploit "pessimistic logging" (see, e.g., [12]), with forced log I/Os at both sender and receiver on every message exchange. Techniques that are even more expensive, such as process checkpointing (i.e., writing process state to disk) upon every interaction, were used in the fault-tolerant systems of the early eighties [4, 6, 14]. Thus, failure masking has been considered a luxury affordable only by mission-critical applications (e.g., stock exchanges). An exactly-once execution protocol limited to three-tier Web Services with *stateless* mid-tier servers has been developed by [11] as opposed to our solution for *n*-tier systems with arbitrarily distributed *stateful* and *stateless* components. The need for new recovery techniques has also been raised by [8], although the failure masking and OS crashes are not considered.

1.2 Review of Interaction Contracts

The framework considers a set of interacting components of three different types. (i) *Persistent* components (Pcom), e.g., clients and application servers, are able to recreate their state and messages as of the time of the last external interaction upon crashes, and eliminate duplicates; (ii) *Transactional* components (Tcom), e.g., database servers, provide the same guarantees only for the final interaction (commit request/reply); (iii) *eXternal* components (Xcom) without any recovery guarantees represent human users and legacy components that do not comply with the framework.

The Pcom is a central concept of the framework. Pcom's are assumed to be *piecewise deterministic*, i.e., their execution is deterministic up to nondeterministic input that is under control of other Xcom's, Pcom's, or Tcom's. A Pcom's state as of any particular time can be recreated via *deterministic replay* of the recovery log where the nondeterminism of the original execution is captured.

The exactly-once execution in the overall system is guaranteed when components obey the obligations defined in the IC's. Interactions (messages being passed from the receiver to the sender) between two Pcom's must comply with either the *Committed IC* (CIC) or the *Immediately Committed IC* (ICIC). Pcom's and Tcom's exchange messages using the *Transactional IC* (TIC). Xcom's are allowed to communicate only with Pcom's as determined by the *External IC* (XIC). The definitions of the CIC and the ICIC follow. Please see the original literature for details on the XIC and the TIC.

The sender part of the CIC consists of the following obligations. The sender promises: *(s1)* that its state as of the time of the message send or later is persistent; *(s2a)* to send the message repeatedly (driven by timeouts) until receiver releases it (perhaps implicitly) from this obligation; *(s2b)* to resend the message upon explicit receiver request until the receiver releases it from this obligation; *(s3)* that its messages have unique identifiers, e.g., *message sequence numbers (MSN)*.

The receiver promises: *(r1)* to eliminate duplicate messages (which sender may send to satisfy s2a); *(r2a)* that before releasing sender obligation S2a, its state as of the time of message receive or later is persistent without periodical resend by the sender; *(r2b)* that before releasing the sender from obligation S2b, its state as of the time of the message receive or later is persistent without the sender assistance.

After S2a release, the receiver must explicitly request the message from the sender; the interaction is *stable*, i.e., it persists (via recovery if needed) over subsequent crashes with the same state transition as originally. After s2b release, the interaction is *installed*, i.e., the sender and the receiver can both recover autonomously.

The *Immediately Committed IC* is a CIC where the receiver immediately installs the interaction (usually by adding the complete message to the stable log), such that the sender is released from the obligation s2a without entering the obligation s2b.

1.3 Contribution

This paper shows how to efficiently implement the IC framework in an open-source environment for composite Web Services with stateful components. The EOS^2 software enhances Web browsers and PHP [17] engines running server-side scripts with IC's and their strong recovery guarantees. In this setting, all components (e.g., application servers or Web Services) are potentially stateful, maintaining session state that spans multiple interactions among components. Efficiently masking failures in this advanced setting is a distinctive feature of the work presented here.

EOS^2 masks all transient failures like OS and component crashes and message losses to the application program, thus greatly relieving the programmer from writing exception handling code. EOS^2 server part is implemented on multithreaded Apache 1.3.20 within PHP 4.0.6 for Windows. It comprises less than 10,000 lines of *C* code including modifications to the PHP Zend engine, its Apache API, and the PHP session module along with ca. 500 lines of JavaScript code for browser recovery.

2 Persistent Browser

There are two major design goals of EOS enhancements to the browser. The first is to improve the user experience by saving as much of her input across a failure (such as a

browser crash or Web Service unavailability) as possible. This avoids the need for annoying repetition of long inputs, which may happen with lengthy forms such as e-government applications (tax declarations, visa applications, etc.) and e-business applications (e.g., insurance and credit approval requests). This is the task of the XIC implementation for the browser. The second is to guarantee to the end-user that all requests are executed exactly once, which is the task of the CIC stub of the browser.

Browser recovery is implemented for Microsoft Internet Explorer (IE), using JavaScript. The server adds the browser logging and recovery code as the last step of output processing. Original server scripts do not have to be changed. Logging is done by modifying a so-called *XML store*, an XML structure managed by IE on the client's disk similarly to persistent cookies. An *XML store* can keep up to 640 KB per HTTP session determined by the server name, which is sufficient, since we are going to keep just one copy of an HTML tree occupying typically less than 50 KB. The XML store feature is provided by IE as part of its default persistence behavior called "*userData Behavior*". The recovery code inserted by the server contains an invisible user-defined HTML element <*sdk:logger id=pagestate style="behavior:url (#default#userData);"*> with attached *userData* behavior. The XML object associated with the XML store can be accessed through a simple interface {*get*,*set*}*Attribute* or using the XML DOM that is natively supported by IE as well. When we need to force the XML store with the logging information to disk we call *pagestate. save*(*XMLStoreName*). For recovery, the content of the XML store is fetched by invoking *pagestate.load* (*XMLStoreName*). Since we had no access to the browser source code, we require the user to revisit the greeting page (e.g., http://servername/) of the Web Service manually after a crash, in order to have her session restored automatically.

3 Persistent PHP

PHP is a widely used scripting language for Web applications [17]; it is used with more than 50% of the Apache installations [2]. The PHP implementation is an open-source project consisting of many PHP modules and the Zend engine implementing the language interpreter [22]. PHP includes a *session module* for maintaining the *PHP application state* across subsequent HTTP requests. The session module supports various methods of storing the session state (e.g., in the file system, shared memory, central database, etc.). The state of a PHP application may be private (e.g., a shopping cart) or shared, concurrently accessed by multiple users (e.g., the highest bid in an electronic auction). PHP typically uses a cookie to propagate the session (state) id to the Web browser.

The session support is activated either explicitly by calling the function *session_start* somewhere in the script or it is started automatically, if appropriately configured, prior to executing the PHP code for the given request. The session module reads the state associated with the session id provided with the request from the session storage and makes it accessible to the PHP script as the session array. When the request does not contain a session id cookie, the session module generates a new session id and creates an empty session array. The PHP script may update the session array by changing or deleting the existing entries or by adding new ones. The new

state is made available for subsequent requests by either calling the function *session_write_close* or implicitly when the script terminates.

PHP offers several options to interact with (potentially PHP-enabled) *Web services*. One of the most popular and elegant methods provides the CURL module that allows PHP applications to access remote HTTP and many other resources in a uniform fashion. The complexity of the protocols is hidden behind a very simple interface that keeps the coding effort at minimum. This functionality is implemented by the CURL library for *C* developed as an open-source project [19]. Some Web Services are invoked via SOAP, the Simple Object Access Protocol layered on top of HTTP. When SOAP is involved, we would pass a SOAP message as a POST parameter. Note that by the EOS^2 implementation of a CIC for CURL and Session modules of PHP, recovery guarantees are provided at the HTTP layer. Thus, higher-level applications including PHP script libraries for SOAP and other protocols over HTTP are relieved from dealing with system errors at all.

3.1 EOS-PHP

EOS-PHP is the major part of our prototype for transparent Web Service recovery. It can serve as both an HTTP server and a middle-tier HTTP client at the same time. It transparently implements the (I)CIC stubs for incoming and outgoing HTTP interactions with other PHP applications and Web browsers. EOS-PHP is geared to provide the recovery guarantees for stateful PHP applications. The log is provided as *a universal storage* for log entries and the session state data. Log access is accelerated by LRU buffers. In addition, EOS-PHP delivers basic concurrency control in the form of latches.

When considering a single PHP Zend engine, we can distinguish three relevant system layers from the logging perspective. We observe HTTP requests at the highest level *L2*, individual PHP language statements at the middle level *L1*, and finally I/O calls to external resources such as the file system and TCP sockets (level *L0*). EOS-PHP does not support interactions with the file system, i.e., the PHP file system functions. Instead, EOS-PHP efficiently manages persistent application states stored as session variables. EOS-PHP does not deal with the PHP socket interface. Instead, EOS-PHP supports recoverable HTTP interactions through the CURL module. The purpose of this subsection is to describe HTTP request processing by EOS-PHP and logging that is necessary for correct PHP application recovery.

A request execution by EOS-PHP breaks down into the following stages: client identification (Stage 1), URI recovery (Stage 2 for browsers only), reply resend (Stage 3), request execution (Stage 4), output processing (Stage 5). Note that Stages 2 and 3 are EOS-PHP operations needed for browser recovery. Prior to the request execution, a shared *activity latch* is obtained for the duration of the request execution. It prevents the garbage collection mechanism that uses this latch in the exclusive mode from physical reorganization of the log file.

Stage 1: Client Identification. During request startup, EOS-PHP identifies the client id information submitted as cookies. If this information is missing the client is assigned a new id and is redirected to the first session URI (browsers only). A business-to-business (B2B) component (i.e., another EOS-PHP node for composite Web Services) autonomously generates its id by concatenating its host name and TCP

listen port (*socket*) number. Note that this does not incur any nondeterminism since the listen port number is fixed, and uniquely identifies a server application on the given host. Web servers typically listen to port 80. A Web application reachable through the URI *http://eosphp.com/* would introduce itself as "*eosphp.com:80*" when calling other Web services.

The following Stages 2, 3, and the state initialization part of Stage 4 are initiated on behalf of the function *session_start*. The request thread acquires an exclusive log latch because all these operations have to be performed atomically.

Stage 2: URI Recovery. Interactive clients (i.e., those whose user agent field submitted with the request header information is different from *EOS_CURL*) need an additional stage for assisting in recovering the last message sent to the EOS-PHP engine. EOS-PHP checks if the current URI coincides with the URI that started the session (i.e., the greeting page URI). If this is the case, we know that this is an interactive client revisiting the greeting page to restore the interrupted session. An empty page containing solely client recovery code is sent back to the browser without incrementing the MSN cookie. The volatile cookie *uri_recovery* is set to *true*.

Stage 3: Reply Message Resend. The log is consulted through the request message id lookup in the volatile *input message lookup table (IMLT)* in order to determine if the HTTP reply is already present. An IMLT table entry has the form *(client id, MSN, reply LSN)* where LSN is a log sequence number. In the positive case, the HTTP reply is served right away and the current request is terminated. When the *uri_recovery* cookie is provided, the server knows that the browser is solely restoring the message URI in the address bar without the need for message resend. To save the network latency, the server responds again with an empty HTML page with the browser recovery code. The cookie *uri_recovery* is set to *false*. When the IMLT contains an entry for the request with the reply LSN being invalid, EOS-PHP is dealing with a request message resend: this thread is paused until the reply LSN is set, and the reply can be served.

When the current request is not a duplicate, it is not terminated by this stage. It is important that we hold an exclusive latch for the log at least until the request is registered in the IMLT during the next stage, in order to prevent two identical messages (resends) from being handled both as original requests.

Stage 4: Request Execution. The request execution starts with fetching the PHP application state through the state buffer. The state buffer is latched in the shared mode to find the proper application state. If the entry for the current PHP application state could not be found (i.e., the request initializes a new PHP session), the request upgrades the state buffer latch to the exclusive mode and inserts a newly created empty state into the state buffer. At this point, the PHP application has a valid state. A new LSN is generated, and EOS-PHP adds an initial log entry to the log buffer that contains PHP representation of the HTTP request and the translated PHP file path. (In fact, a PHP script may depend on more than mere HTTP parameters, e.g., when it uses OS shell environment variables or Apache configuration parameters that change over time. An administrator should mark these variables for logging in the PHP configuration). An entry is also added to the IMLT containing the client id, the MSN of the message (both as submitted by the client cookies), and an invalid LSN. At this point, the request thread latches the PHP application state in the shared or exclusive

mode (as specified in the enhanced PHP function *session_start* that now accepts a Boolean flag *$read_only* as an optional argument) and releases the exclusive log latch as well as the shared state buffer latch. In contrast to the original PHP, the ability to access the application state in the shared mode is an appropriate response to the fact that the load of e-commerce sites is dominated by read-only catalog browsing requests. The application state remains latched until the script calls the function *session_close* replacing the original PHP function *session_write_close* to avoid irritation. If the request has been declared as a *write* by calling *session_start*(*false*), the application state is stamped with the request LSN for redo optimization before the state latch is released.

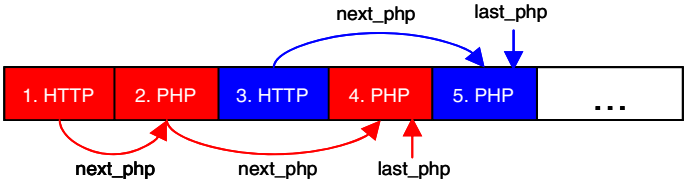


Fig. 1. Log buffer organization

The information logged during Stage 4 would suffice for deterministic replay of the HTTP request log entries one after another using the high-level routine *zend_execute_scripts*, if we had not to deal with nondeterministic calls throughout request execution. Nondeterministic calls generate further log entries. Since EOS-PHP currently can only replay HTTP requests sequentially as opposed to an arbitrary interleaving of PHP statements issued on behalf of distinct HTTP requests, we need to be able to find the needed log entry quickly. Thus, we link each entry in the log buffer to its successor using the *next_php* pointer of the buffer cell as depicted in Fig. 1. Each PHP-level log entry includes the LSN of its predecessor in the *next_php* chain, such that the *next_php* chain can be restored during recovery. We use a *last_php* pointer to keep track of script-internal nondeterminism. At the beginning of the request execution *last_php* refers to the very first (HTTP-level) log entry whose *next_php* field is NULL. The log is consulted upon every statement call that requires logging. If there is a successor of *last_php*, we deal with a replay and EOS-PHP assigns *last_php*→*next_php* to *last_php* and returns the logged return value without executing the statement. If there is no successor of the previously checked entry *last_php*, EOS-PHP either operates normally or completes request execution during the redo phase, and a new PHP level log entry is added to the log buffer. It is connected to the *next_php* chain and becomes new *last_php*. The new PHP log entry contains the LSN of the previous *last_php* that is used during analysis pass to restore the *next_php* chain. Note that log entries of different requests interleave in the log when they use shared latches or an updating request calls *session_close* before its completion.

We are aware that we would not have to deal with the *next_php* chain, if we implemented logging with PHP statement granularity. However, a finer-grained logging is more difficult to implement because different thread contexts (consisting of hierarchical function stacks each) would have to be recreated and applied when

replaying individual PHP actions (local and global variable accesses). On the other hand, the simpler recovery is, the more trust we have in its correctness.

Nondeterministic functions treated by EOS-PHP include system clock reads (e.g., *time()* returning the current time, random value generators such as *rand(min, max)* generating a random number in the interval between *min* and *max*, and last but not least *curl_exec* returning output of a different Web Service. The routines asking for the system clock and random values are not only interesting because of their potential direct usage in a PHP script, but also because their *C* prototypes (i.e., the underlying implementation in *C*) are used as input for generating PHP session ids that are pairwise distinct with a high probability. This avoids a potential bottleneck of having a single node in a Web farm assign sequence numbers as session ids to all clients.

The *C* prototype of the function *curl_exec*(*$handle*) implements the CIC transparently. One part of it is implementing periodic resend. The original code reporting failures to the user is replaced by a loop repeating requests on timeouts until the underlying *libcurl* function *curl_easy_perform* returns the success return code. When *curl_easy_perform* needs to be retried, we use a new copy of the previous CURL handle containing the same URI and the other original request settings while destroying the old one. Otherwise, *libcurl* would try to recycle its existing sockets, which saves resources. However, the socket that timed out is likely to belong to a dead TCP connection and therefore should not be used for retry to avoid blocking.

Stage 5: HTTP Output Processing. When the execution of the request is finished, EOS-PHP updates the reply LSN field of the request entry in the IMLT. In the current prototype solution, the log entries with HTTP output do not require immediate log forcing, since these messages are recreated during deterministic replay. In fact, for *curl_exec* requests EOS-PHP does not even create a log entry with the content of the outgoing message, just the reply is logged to resolve recovery dependency. The point is that EOS-PHP is able to send out the HTTP reply messages prior to forcing them to stable log. Therefore, EOS-PHP can lazily force output messages (from several KB to several MB) that are orders of magnitude larger than preceding log entries of the same request whose sizes range from less than 256 bytes to some KB.

In addition, the browser recovery code that is always cached in the main memory is inserted into original HTTP replies to interactive clients between the opening tag <*html*> and the successor opening tag that is typically <*head*>. This does not have to be logged of course.

3.2 Implementation and Efficiency Issues

Since *curl_exec* incurs sending a request message to a different EOS-PHP Web Service, we need to keep track of when the CIC interaction is installed by this counterpart, such that we can move on with the garbage collection at our own discretion. This is done via the volatile *output message lookup table (OMLT)* containing the URI invoked, our MSN, and the current CIC status. We currently assume that all PHP scripts belonging to the *same Web Application* are stored in *the same Web server directory*. In a B2B Web application we assume that a reply to calling */auctions/bid.php* on behalf of one end-user and */auctions/search.php* on behalf of another user stems from the same EOS-PHP instance. Thus, we store only URI paths without script file names in the OMLT.

In terms of a CIC, we need to answer the question of when we have to force the log other than for LRU buffer management. EOS-PHP communication with the outside world is currently limited to returning HTTP replies and sending CURL requests. Thus, the log is forced prior to sending out these messages to capture the nondeterministic interleaving. Since we log an entire HTTP request, we implemented only ICIC for EOS-PHP. When a reply arrives to an EOS-PHP CURL client, it can mark the interaction as *cic_installed* in its OMLT. Furthermore periodic resend of the *curl_exec* request message is stopped.

Now we need to remove unneeded IMLT and OMLT entries. When we deal with a request from a single-threaded browser, we know that as part of its XIC obligation, it has already installed *all* previous server replies. Thus, we can drop all entries stemming from the same client with MSN's lower than we see in the current request. Note that there are no OMLT records for browsers because we record only outgoing requests, but no replies. When we see a request from a multi-threaded B2B client with a particular MSN, it is not even guaranteed that we have already processed its previous requests. Thus, each B2B client includes into each request to a URI an additional cookie containing an *installed-MSN* with the following property: there is no OMLT entry with the same URI and MSN less than this installed-MSN whose CIC status differs from *cic_installed*. Thus, the B2B server is able to drop all entries for the given B2B client in its IMLT with MSN less or equal to installed-MSN of the B2B client. In turn, the B2B client can drop the OMLT entries with the B2B URI and the MSN less or equal to installed-MSN.

To implement this efficiently, we organize the OMLT as a hash table that maps URI's to the interaction lists containing (MSN, CIC status) pairs. When the function *curl_exec* is called, it finds the proper list associated with the URI being called, traverses the list while remembering the *maximum installed MSN* seen so far until it finds the first pair belonging to an uninstalled interaction. This pair becomes a new head of the interaction list for the given URI, whereas the other traversed pairs are garbage collected. The new interaction with the CIC status *unknown* is appended to the end of the shortened list. The maximum installed MSN will be inserted as a cookie *installed-MSN* into the HTTP request built by *curl_exec*. The IMLT is implemented similar to a PHP two-dimensional array indexed by client id and MSN with the LSN of the corresponding HTTP reply as the content, i.e., as a hash table that is used to find by client id the hash table mapping HTTP request MSN's to HTTP reply LSN's. For reply recovery in Stage 3, we perform a simple lookup IMLT[clien id][msn]. For garbage collection when a new request arrives (Stage 4), we find a pointer to the hash table mapping request MSN's to reply LSN's as referred to by IMLT[client id]. Then we traverse this hash table as a list in the ascending MSN order (buckets in a Zend hash table are linked together as in Java and C# implementations) and throw away all entries with MSN less or equal to the submitted cookie *installed-MSN*.

4 Experiments

To evaluate the run-time overhead of EOS-PHP, we performed measurements with Apache/1.3.20 and PHP/4.0.6 running on two PC's each with a 3 GHz Intel Pentium

IV processor and 1 GB main memory under Windows XP. The call structure of the evaluated application is shown in Fig. 2. The load on the frontend Web server was generated by the synthetic HTTP request generator *Apache JMeter/2.0.3*. The generator simulated conversations of *n* steps without involving human user interactions. Think times were not simulated. Both servers deploy the same PHP script that calls the nondeterministic function *time()* and reads the current application state, then increments a private state variable and a shared state variable on the backend server. The frontend server replies with a complete HTML page containing the shared and private *count* values, returning the HTML page to the load generator. In the following two experiments, we compare the three-tier system of two servers run by the original PHP engine against the equivalent system run by EOS-PHP.

Table 1 shows the total elapsed time, between the first request and the last reply as seen by the client, and the CPU time on the frontend and backend servers for *n = 1, 5, 10* steps, comparing the original PHP engine to EOS-PHP. The original PHP manages each session in a separate file. Changes to the session variables are made quasi persistent by the original PHP because the session file is written without being forced to disk. The function *_close(int filehandle)* called by the original PHP engine at the request end does not incur synchronous I/O on Windows. The response time overhead of 135-152% results also from the fact that the original PHP sends the reply before the disk write, the latter being performed during the request shutdown. The CPU time overhead on the frontend server is lower than on the backend server by a factor of approximately two due to the file (de)allocation activity of the original PHP when starting new sessions and terminating the old ones, whereas a single file is used all the time on the backend server. Nevertheless, we need to mention that the cost for forced I/O's dominates the overhead because inserting the function *_commit* before the call to *_close* in an additional test made the overhead shrink to less than 20%.

We also performed multi-user measurements by replicating the HTTP request driver on five different machines generating requests to the frontend server. Table 2 shows the measured average response and CPU times in terms of the simulated *n*-step

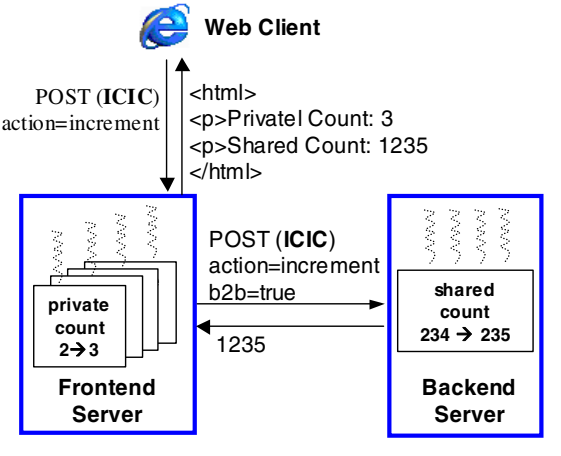


Fig. 2. Test application in the experiments

user sessions. The figures show that the response time overhead decreases in comparison to the one client measurement because concurrency becomes a more significant factor. Although the original PHP does not need concurrency control, the Apache Web server manages a number of shared data structures that are protected by internally implemented *semaphores* and *mutex locks* that increasingly suffer access contention. The overhead on the frontend server is larger than on the backend server due to a higher contention on the log latch which protects two synchronous log writes: one before calling *curl_exec* and another one prior to sending the output to the client.

Table 1. One-client experiment

Sessions	1 step	5 steps	10 steps
PHP elapsed [sec]	0.0480	0.2200	0.4500
EOS-PHP elapsed [sec]	0.1130	0.5550	1.1000
Overhead [%]	135%	152%	144%
PHP frontend CPU [sec]	0.0240	0.1625	0.3455
EOS-PHP frontend CPU [sec]	0.0305	0.2125	0.4636
Overhead [%]	27%	31%	34%
PHP backend CPU [sec]	0.0050	0.0300	0.0700
EOS-PHP backend CPU [sec]	0.0090	0.0550	0.1200
Overhead [%]	80%	83%	71%

Table 2. Five-clients experiment

Session	1 step	5 steps	10 steps
PHP elapsed [sec]	0.1560	0.7900	1.6100
EOS-PHP elapsed [sec]	0.3140	1.6850	3.1000
Overhead [%]	101%	113%	93%
PHP frontend CPU [sec]	0.0390	0.2708	0.5727
EOS-PHP frontend CPU [sec]	0.0815	0.6000	1.1545
Overhead [%]	109%	122%	102%
PHP backend CPU [sec]	0.0090	0.0550	0.1200
EOS-PHP backend CPU [sec]	0.0130	0.0750	0.1600
Overhead [%]	44%	36%	33%

The cost for the recovery guarantees is less than factor of two, which is an acceptable overhead. The price is worthwhile given the increased dependability and gained ease of programming.

5 Conclusion

The major accomplishment of this paper is the transparent integration of the IC support with real-world Web products: Internet Explorer and Zend's server-side scripting engine for the PHP language. Our prototype EOS^2 allows deploying arbitrarily distributed PHP application with the exactly-once execution guarantee. It is able to deterministically replay concurrent, interleaved requests operating on shared

application states. Developers are relieved from writing explicit code handling system failures. Experiments show that the rigorous end-to-end recovery guarantees are provided with acceptable overhead.

Performance of EOS-PHP can be improved by deeper integration of the log and recovery managers with the Zend engine, which would allow replaying the log at the PHP statement level. This would make the analysis pass in the current solution obsolete, and make the redo pass simpler. Moreover, such a solution would allow creating installation points in midst of request execution, which would accelerate replaying lengthy scripts during redo. Providing a distributed concurrency control protocol for PHP would be another interesting direction to pursue in future work.

References

1. Alvisi, L. and K. Marzullo: *Message Logging: Pessimistic, Optimistic, and Causal*, ICDCS 1995
2. *Apache Module Report*, http://www.securityspace.com/
3. Barga, R., D. Lomet, G. Shegalov and G. Weikum: *Recovery Guarantees for Internet Applications*, ACM Transactions on Internet Technologies 4(3), 2004
4. Bartlett, J: *A NonStop Kernel*, SOSP 1981
5. BEA Tuxedo, http://bea.com/
6. Borg, A., W. Blau, W. Graetsch, F. Herrmann, and W. Oberle: *Fault Tolerance Under UNIX*, ACM Transactions on Computer Systems 7(1), 1989
7. *Browser Trends Survey, Oct. 29th 2004*, http://www.entmag.com/ 8
8. Candea, G. et al:.: *Microreboot -- A Technique for Cheap Recovery*, OSDI 2004
9. Cristian, F.: *Understanding Fault-tolerant Distributed Systems*, Communications of the ACM, 34(2), 1991
10. Elnozahy, E., L. Alvisi, Y. Wang, and D. Johnson: *A Survey of Rollback-Recovery Protocols in Message-Passing Systems*, ACM Computing Surveys, 34(3), 2002
11. Frølund, S. and R. Guerraoui: *e-Transactions: End-to-End Reliability for Three-Tier Architectures*, IEEE Transactions on Software Engineering, 28(4), 2002
12. Huang, Y. and Y. Wang: *Why Optimistic Message Logging Has Not Been Used In Telecommunications Systems*, FTCS 1995
13. Johnson, D. and W. Zwaenepoel: *Sender-based Message Logging*, FTCS 1987
14. Kim, W.: *Highly Available Systems for Database Applications*, ACM Computing Surveys, 16(1), 1984
15. *Microsoft Developer Network*, http://msdn.microsoft.com/
16. *Oracle Advanced Queuing*, http://oracle.com/
17. *PHP: Hypertext Preprocessor*, http//www.php.net/
18. Shegalov, G., G. Weikum, R. Barga,, D. Lomet: *EOS: Exactly-Once E-Service Middleware*, Demo Paper, VLDB 2002.
19. Stenberg, D.: *cURL and libcurl*, http://curl.haxx.se/
20. Strom, R., D. Bacon, and S. Yemini: *Volatile Logging in n-Fault-Tolerant Distributed Systems*, FTCS 1988
21. *The World Wide Web Consortium*, http://w3.org/
22. Zend Technologies, Inc. *The PHP Company*, http://zend.com/

Quantified Matchmaking of Heterogeneous Services

Michael Pantazoglou, Aphrodite Tsalgatidou, and George Athanasopoulos

Department of Informatics & Telecommunications,
National & Kapodistrian University of Athens, 15784, Greece
{michaelp, atsalga, gathanas}@di.uoa.gr

Abstract. As the service-oriented computing paradigm and its related technologies mature, it is expected that electronic services will continue to grow in numbers. In such a setting, the course of service discovery could yield many alternative yet heterogeneous services which, by all means, may be of different type and moreover distinguished by their quality characteristics. To come through such situations and ease the task of service selection, service search engines need to be powered by an efficient matchmaking mechanism, which will abstract requesters from service heterogeneity and provide them with the means for choosing the service that best fits their requirements, among a wide set of services with similar functionally. In this paper, we present an efficient service matchmaking algorithm, which facilitates the task of heterogeneous service selection, whilst combining and exploiting the syntactic, semantic, and Quality-of-Service (QoS) properties contained in service advertisements.

Keywords: service discovery, service matchmaking, service ranking, heterogeneous services.

1 Introduction

Service discovery plays a fundamental role in service-oriented development, allowing developers to find and re-use existing units of software for rapidly building distributed applications. According to advocates of the *Service-Oriented Architecture* (SOA), electronic services will continue to grow in numbers as the related technologies mature. Accordingly, a query for a specific type of functionality could yield many alternative services, which may be of different type (e.g. web or peer-to-peer services) and moreover distinguished by different Quality-of-Service (QoS) characteristics. This case is better illustrated through the description of the following real-world scenario.

The IT department of a private clinic has decided to develop a service-oriented application, which will enable direct interactions between doctors, patients, as well as other partners (e.g. insurance companies, pharmacy companies, external doctors, etc). Fig. 1 depicts an excerpt of this application, where the doctor asks for a second opinion, based on anonymous medical information retrieved from the clinic's internal database. The clinic has already established partnerships with a number of external doctors as well as with other clinics and hospitals, which offer the appropriate services for the establishment of this type of communication. These services are potentially of different type, they may have been described with the use of different

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 144 – 155, 2006.

description protocols, and they may also have different quality-of-service characteristics. For example, hospitals may have exposed this type of functionality through a web service interface, while the external doctors may be contacted and asked for a second opinion through a specialized p2p service running on their PDAs or mobile phones. Consequently, the developer faces the problem of having to select among all these similar services the one that is most suitable for the specific task.

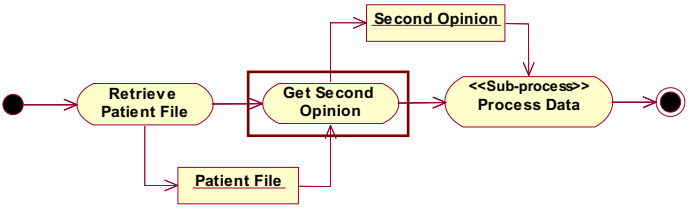


Fig. 1. A sample healthcare application requiring the use of a "get second opinion" service

To overcome such an arduous task, the course of service discovery needs to be leveraged by an efficient matchmaking mechanism, which will abstract requesters from the technical complexity and heterogeneity of the various service advertisements and provide them with a means for making the right selection. Naturally, the result of such a mechanism will be a list of appropriately ranked, similar services.

In this paper, we present a matchmaking algorithm, which aims at facilitating the task of service selection among a set of alternative services, by quantifying their similarity. Among the strong points of the proposed matchmaker are: 1) its ability to deal with the underlying heterogeneity of existing services, both in terms of their type and their description protocols, 2) the fact that it produces results by combining syntactic, semantic, and QoS service characteristics. The matchmaking algorithm has been fully implemented by the specification of the *Unified Service Query Language*, namely *USQL*, which is described in [1]. The detailed description of the language goes beyond the scope of this paper; however, USQL will be used as a means to express requirements, in order to validate the proposed algorithm.

Briefly, the rest of the paper is structured as follows: Section 2 introduces the basic concepts that form the basis for the definition of the matchmaking algorithm, which is described in Section 3. In Section 4, a preliminary experiment is conducted, with the use of USQL, so as to give some early results regarding the precision of the algorithm. Section 5 compares our work with related efforts and emphasizes its main contributions. Finally, Section 6 concludes with a small discussion on future work.

2 Basic Concepts

In this paragraph, we briefly present the conceptual model that has driven the definition of the matchmaking algorithm. Described by the W3C Body, the *Service-Oriented Model (SOM)* [3] defines that a service groups the message interactions it can be engaged in through an interface. Moving along this high-level assumption, we established in previous work a *Generic Service Model (GeSMO)* [2]. GeSMO moves

one step further by defining message interactions as operations, while also declaring that a service may expose multiple interfaces, which in turn offer one or more such operations. The following conceptual model (Fig. 2) depicts our perception on the notion of service and its fundamental parts.

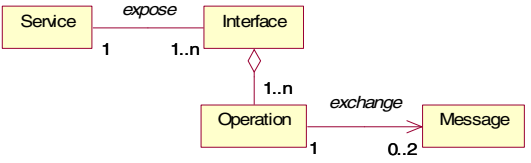


Fig. 2. High-level conceptual model for services

All concepts in the above model are given by default a syntactic description, but it is also possible that semantics and/or QoS properties are assigned to some of them. In our approach, we assume that services, service operations as well as their exchanged messages can be semantically described. In addition, service operations can be given QoS properties, such as availability, reliability, processing time, etc (Fig. 3). Consequently, when querying for a specific type of functionality in a service-oriented setting, it becomes natural that requirements can be expressed at the service level, the operation level, as well as the message level. Moreover, a service query may consist of syntactic, semantic, and QoS requirements.

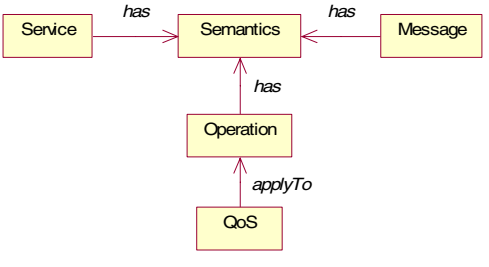


Fig. 3. Semantics and QoS as they are supported by our conceptual model

Having described our conceptual view on services, we proceed in the following with the definition of our matchmaking algorithm.

3 The Matchmaking Algorithm

The process of service matchmaking can be basically described as follows: *Given a service request comprising a set of requirements, check against a set of service advertisements and calculate the degree of match for each corresponding service.* The degree of match quantifies the suitability of each service with respect to the original request and it usually takes the form of a percentage amount.

A service request contains a series of requirements, which can be either simple (e.g. the name of the service provider, the type of input/output parameters, the processing time of a service operation, etc.), or complex. As the term implies, complex requirements consist of other requirements, which in turn may be either simple or complex. For example, a series of operation-level requirements, such as the operation name and semantics, the operation inputs and outputs, operation QoS, etc. are grouped as one complex requirement regarding the whole operation. In the next paragraphs, we define a set of formulas which are used to calculate the degree of match for simple and complex requirements.

3.1 Matching Simple Requirements

According to the conceptual study that was described in the previous section, the requester may express syntactic, semantic, and/or QoS requirements upon formulation of a service discovery request. To process such requests, a matchmaking algorithm should be primarily equipped with the appropriate mechanisms for the calculation of the degree of match of simple syntactic, semantic, and QoS requirements.

Similarity distance measure has been established as one of the most popular matchmaking techniques and is being used extensively in the areas of data mining and web information retrieval [16], [15]. The relative theory is basically evolved around the notion of *distance functions*, a definition of which can be found in [4]. Our proposed matchmaking algorithm utilizes this technique in order to calculate the degree of match of such requirements in a service discovery request.

Definition 1. Given a simple requirement q and the value a of the corresponding property in a service advertisement, their degree of match d is defined as follows:

$$d = 1 - dist(q, a) \ . \tag{1}$$

As we can see, the calculation of the degree of match employs *dist*, a normalized similarity distance measure function. More specifically, *dist* will return 1 in cases where q and a do not match at all, 0 in cases where q and a perfectly match, and an appropriate value between 0 and 1, if where there is a partial match between q and a. In other words, the more a matches q, the more *dist* leans towards 0 and, reversely, the less a matches q, the more *dist* leans towards 1. Consequently, according to equation (1), the degree of match d equals to 1, if the advertised property exactly matches the requested one, while it takes the value of 0, if there is no match.

3.1.1 Syntactic Requirements

The expression of a syntactic requirement implicitly states a set of accepted values for the corresponding advertised property. Consider an example, where the requester is looking for services being offered by Microsoft. The implied set of accepted values for the service provider property comprises any phrase or word containing the keyword *'Microsoft'*. The example reveals that, the result of applying syntactic matchmaking has a binary nature, that is, either the advertised property's value belongs to the set of accepted values implied by the syntactic requirement, or it does not. The following definition captures this assumption:

Definition 2. Given a simple syntactic requirement q_{syn}, its implied set of accepted values S_q, and the value a_{syn} of the corresponding syntactic property in a service advertisement, their similarity distance measure function is defined as:

$$dist_{syn}(q_{syn}, a_{syn}) = \begin{cases} 0 & , a_{syn} \in S_q \\ 1 & , a_{syn} \notin S_q \end{cases}. \quad (2)$$

Going back to the above example, $dist_{syn}(q_{syn}$, *'Microsoft Corporation'*$) = 0$, while $dist_{syn}(q_{syn}$, *'IBM'*$) = 1$. Respectively, the degree of match in the first case would be equal to 1 (i.e. the simple syntactic requirement was fully met) while in the second case it would be equal to 0 (i.e. the simple syntactic requirement wasn't met).

3.1.2 Semantic Requirements

In measuring the distance measure between two semantic concepts we take into consideration their hierarchical relation in the ontology graph. Particularly, the semantic relations *exact*, *plug-in*, *subsume*, and *fail*, between two semantic concepts, which have been described in [5], can be quantified with the definition of the following semantic distance measure function:

Definition 3. Given a simple semantic requirement q_{sem} and the value a_{sem} of the corresponding semantic property in a service advertisement, their distance is measured as follows:

$$dist_{sem}(q_{sem}, a_{sem}) = \begin{cases} 0 & , \text{if } q_{sem} \text{ and } a_{sem} \textit{ exactly} \text{ match} \\ 1/3 & , \text{if } q_{sem} \text{ is a } \textit{plug-in} \text{ of } a_{sem} \\ 2/3 & , \text{if } q_{sem} \textit{ subsumes } a_{sem} \\ 1 & , \text{if } q_{sem} \text{ and } a_{sem} \text{ don't have an immediate relation } (\textit{fail}) \end{cases} \quad (3)$$

The values assigned to each of the identified semantic relations reflect their semantic order, as it has been explained in [5].

3.1.3 QoS Requirements

As in the case of syntactic search criteria, a QoS requirement defines a set of accepted values for a given QoS property, from the requester's point of view. For example, if the requester requires that the operation must be at least 99.9% available, given the fact that the operation availability cannot be validated more than 100%, the implied set of accepted values for this specific QoS property will be [0.999, 1], where percentages have been expressed as real numbers between 0 and 1.

Due to the fact that most QoS properties are numerical, we can refine the definition of the similarity distance measure function so that it provides us with more realistic results. Such definition is given as follows:

Definition 4. Given a simple QoS requirement q_{qos}, its implied set of accepted values S_q, and the value a_{qos} of the corresponding QoS property in a service advertisement, their distance measure, $dist_{qos}$, is defined as:

$$dist_{qos}(q_{qos}, a_{qos}) = \begin{cases} 0 & , a_{qos} \in S_q \\ \dfrac{|q_{qos} - a_{qos}|}{\max(q_{qos}, a_{qos})} & , a_{qos} \notin S_q \end{cases} \quad (4)$$

Note that, the above function is applicable only to QoS properties which take the form of positive numbers. Such properties include the operation availability, reliability, processing time etc. Another appropriate similarity distance measure function should be defined for non-numerical QoS properties (e.g. security).

3.2 Matching Complex Requirements

While the matchmaking process of simple requirements is straightforward and depends primarily on their type (i.e. whether these refer to syntactic, semantic, or QoS properties of the service), matching complex requirements requires an advanced formula which must be able to also satisfy a number of conditions imposed by intuition and practice.

Upon formulation of a service request, some requirements are often given higher priority than others. Prioritized requirements are particularly useful when the task of service discovery is performed at design-time, where the ultimate selection of a service is human-driven. In such cases, requesters may assign different priorities to their needs, so as to distinguish the real important requirements from the less important or optional ones. For instance, during the development of a service-oriented application, the satisfaction of the requested capability of an operation (i.e. semantics of the operation) would be considered more important than matching its actual signature, since the developer has the ability to adapt his/her application accordingly, for the operation to fit in. It is therefore imperative for a matchmaking algorithm to take into consideration the likely different priorities of requirements in a service request. Intuitively, the existence of requirements with different priority levels should have an impact on the calculation of the degree of match: the higher the priority level of a requirement is, the more the requirement's degree of match should affect the calculation of the total degree of match. Moreover, it makes sense to claim that, if the most important requirements in a query are not satisfied, the resulting degree of match should be leaning towards zero.

In the definitions that follow, we capture the essence of the aforementioned conditions in order to render the matchmaker of complex requirements intuitive.

Definition 5. Let $P = \{p_1, p_2, ..., p_k\}$ be an ordered set of k priority levels p_i, $1 \le i \le k$, where $p_1 \prec p_2 \prec ... \prec p_k$ and $Q = \{q_1, q_2, ..., q_n\}$ be an unordered set of n requirements, which have been assigned different priority levels, $k \le n$. Then, Q can be expressed as

$$Q = Q_1 \cup Q_2 \cup ... \cup Q_k . \quad (5)$$

In equation (5), Q_i, $i = 1..k$ are the un-ordered sets of n_i requirements with p_i priority level, where $Q_i \cap Q_j = \emptyset \quad , \forall i, j \in [1,k], i \ne j$, $n_i > 0$ and $\sum_{i=1}^{k} n_i = n$.

It should be noted that, the actual type and values of the priority levels do not affect the matchmaker definition. In our approach, we are only interested in the defined order between the different priority levels, which we use to group requirements according to equation (5). In this way, the matchmaking algorithm is rendered independent from the actual matchmaker implementation. For instance, in the USQL language we have specified two values for the priority level of the requirements, namely *'low'* and *'high'* [1].

In what follows, we define the formula for the calculation of the degree of match for complex requirements.

Definition 6. Let Q be an un-ordered set of n requirements, $n > 0$, with k different priority levels (see Definition 5) and *adv* be an advertisement which Q is checked against. Then, the degree of match d_m for Q is calculated with the use of the following formula:

$$d_m = \frac{1}{\sum_{i=1}^{k} n_i \cdot i} \cdot \left[k \cdot D_k + \sum_{i=1}^{k-1} \left(\frac{i \cdot D_i}{k-i+1} \cdot \left(1 + \sum_{j=i+1}^{k} \frac{D_j}{n_j} \right) \right) \right]. \tag{6}$$

In the above definition, we have $D_i = \sum_{j=1}^{n_i} d_j$, where $n_i > 0$ is the number of requirements having i priority level and d_j is the degree of match of the j^{th} requirement in subset Q_i. It can be proven that $0 \leq d_m \leq 1$, however, due to lack of space we omit the proof in this paper.

A closer look at equation (6) reveals that the defined formula abides by the intuitive conditions which were set previously. Indeed, if all requirements with the highest priority are not met (i.e. $D_k = 0$), then the resulting degree of match for Q diminishes towards zero. Also, the significance of the degrees of match of lower-prioritized requirements in calculating the overall matching degree is affected the degrees of match of the higher-prioritized ones. In other words, the higher the priority of a requirement is, the more its degree of match determines the overall calculation.

3.3 Matchmaking Against Multiple Advertisements

Up to now, we have provided definitions for the calculation of the matching degrees of simple and complex requirements against a given advertisement. Since a service request is basically a set of requirements, it can also be considered as a complex requirement and, thereby, equation (6) is used for the calculation of the overall degree of match of a service. Nevertheless, in practice, service requests are commonly checked against a number of service advertisements. To accommodate this fact, we need to reshape equation (6) with the use of matrices. First, we provide a generic definition of what a matchmaker is:

Definition 7. Let $Q = \{q_1, q_2, ..., q_n\}$ be an unordered set of n requirements, and $A = \{a_1, a_2, ..., a_m\}$ be an unordered set of m advertisements. We define a matchmaker M as

$$M : Q \times A \mapsto D. \tag{7}$$

where $D = [d_{j1}]$, $0 \le d_{j1} \le 1$, $j = 1..m$ is a $m \times 1$ matrix containing the resulting degrees of match of the advertisements.

In general, when n_k requirements with the same priority level k are checked against m advertisements, we can make use of an $m \times n_k$ matrix $R_k = [d_{ij}]$, $i=1..m$, $j=1..n$, for displaying the resulting degrees of match. Each element d_{ij} in this matrix is the resulting degree of match of the jth requirement, checked against the ith advertisement, and has been produced with the use of the appropriate formula, from the ones defined in equations (2), (3), (4) and (6). Then, the summaries of the degrees of match of all n_k requirements per advertisement ($\sum_{x=1}^{n_k} d_x$) are collectively calculated as follows:

$$S_k = R_k \cdot U_{nk} . \tag{8}$$

where U_{nk} is a $n_k \times 1$ matrix, whose elements are all equal to 1, and S_{nk} is the resulting $m \times 1$ matrix.

Definition 8. Let Q be an un-ordered set of n requirements, $n > 0$, with k different priority levels (see Definition 5). Given the equations (6), (7) and (8), the degrees of match for a number of m advertisements contained in an unordered set $A = \{a_1, a_2, ..., a_m\}$ are calculated as

$$M(Q,A) = D_m = \frac{1}{\sum_{i=1}^{k} n_i \cdot i} \cdot \left[k \cdot S_k + \sum_{i=1}^{k-1} \left(\frac{i \cdot S_i + DS_i \cdot \sum_{j=i+1}^{k} \frac{S_j}{n_j}}{k-i+1} \right) \right]. \tag{9}$$

where:

- D_m is the resulting $m \times 1$ matrix containing the matching degrees of the m service advertisements
- DS_x , $x=1..k$, is an $m \times m$ diagonal matrix, immediately produced by S_x as follows:

$$S_x = \begin{bmatrix} d_1 \\ d_2 \\ . \\ d_m \end{bmatrix} \Rightarrow DS_x = \begin{bmatrix} d_1 & 0 & ... & 0 \\ 0 & d_2 & ... & 0 \\ . & . & ... & . \\ 0 & 0 & ... & d_m \end{bmatrix}. \tag{10}$$

This concludes the definition of the matchmaker.

4 Evaluation

The experiment described in this section provides some preliminary results regarding the precision of the proposed matchmaking algorithm. Going back to the scenario

described in Section 1, we will apply matchmaking against a number of alternative services for the *"get second opinion"* task (see Fig. 1). For the purposes of this early experiment, we implemented two web services (ws1 and ws2) and a JXTA p2p service (ps3) with similar functionality, yet distinguished by certain syntactic, semantic and QoS characteristics. Both web services have been described with the use of WSDL-S (see http://www.w3.org/Submission/WSDL-S/), regarding their interface and semantics, and WS-QoS (see http://www.wsqos.net), with respect to their QoS offers. For the p2p service, we used WSDL and OWL-S (see http://www.w3.org/Submission/OWL-S/) to describe it syntactically and semantically. Finally, the same ontology that was used for semantically annotating all services was also employed in the formulation of our query. The following table summarizes the three services and their properties:

Table 1. Services used for the evaluation of the matchmaking algorithm

Service	*Provider*	*Domain*	*Description*	*Operation*
ws1	Medisystem	Healthcare	%second opinion%	Capability: GetSecondOpinion Input: EpisodeId:string Output: Diagnosis:string Availability: 99.95%
ws2	eHealth	PrivateClinicServices	%second opinion%	Capability: GetSecondOpinion Input: EpisodeFile:file Output: SecondOpinion:file Availability: 99.96%
ps3	Medisystem	CardiologyServices	%second opinion%	Capability: GetSecondOpinion Input: EpisodeDesc:string Output: Diagnosis:string Availability: 99.899%

Precision. We used USQL to formulate a query for a *"get second opinion"* service in the domain of *Cardiology*, which is a sub-domain of *Healthcare*, provided by *Medisystem*, offering an operation with capability semantically described as *"GetSecondOpinion"*. The requested operation should accept an *episode file* as input, and return the *diagnosis* as output. Moreover, its availability should be equal or greater than *99.9%*. From the above requirements, the operation and domain were given *high* priority. The USQL request was produced according to these requirements and can be found in http://cgi.di.uoa.gr/~michaelp/usql-request-wise06.usql.

The application of the matchmaker formula, defined in equation (9), along with the use of formulas (2), (3), (4), and (6), produced the following degrees of match:

Table 2. Results of the matchmaking algorithm

Service	*Degree of Match*
ws1	77.42 %
ws2	41.14 %
ps3	91.31 %

According to the results, the p2p service (ps3) was found to be closer to our requirements. However, this should not come as a surprise: a closer look at Table 1 shows that, only the operation input and availability were not fully compliant with the requested ones.

5 Comparison with Related Work

A large number of research efforts have been conducted over the years, dealing with the problem of similarity measuring. Many of these approaches were adapted to service matchmaking, with special attention paid at semantic similarity between ontology classes and/or their properties [6], [7]. In [8], a novel service retrieval approach was proposed, that captures service semantics via process models, and applies a pattern-matching algorithm to locate desired services. An ontological approach was proposed in [11] which, like our matchmaking algorithm, also considers user assigned priorities in matchmaking. Following a different direction, a set of algorithms were developed in [9] which enable searching for web service operations that are similar to a given one. The underlying idea of their search engine is the grouping of inputs and outputs into semantically meaningful concepts. Thus, syntactic information in service advertisements attains semantics and can be exploited in a more fruitful manner. Many proposals have also emerged to deal with QoS matchmaking. In [10], a matchmaking framework was described, which maps QoS requirements of consumers with the published QoS information of providers, also accommodating QoS-Constraints.

As opposed to the above mentioned approaches, which are restricted to performing matchmaking against either syntactic, semantic, or QoS search criteria, our matchmaker is capable of blending all these types of criteria in calculating the service degree of match, thus supporting the specification of syntactic, semantic, as well as QoS requirements within a service query. The calculation of the degree of match of a service remains independent of the way in which the degree of match of each one of the constituent requirements is calculated. Thus, the matchmaker can be extended with as many types of requirements and their related matchmaking mechanisms as needed. In this regard, existing advanced matchmaking mechanisms like the ones mentioned above can be seamlessly embedded in our approach. The LARKS framework [12] also combines syntactic and semantic matchmaking, yet its main drawback lies in that it supports a rather static service description schema. Our matchmaker overcomes this shortcoming as it abides by a high-level conceptual model which complies with most service-oriented technologies and standards, such as WSDL, WSDL-S, OWL-S, WS-QoS, etc. Due to the abstraction of the underlying model, it is expected that the matchmaking algorithm can be applied to a wide range of heterogeneous service-oriented environments, ranging from UDDI and ebXML registry lookups to service discovery within p2p networks and/or grid virtual organizations. Other important features of our proposed matchmaking algorithm are its intuitiveness and scalability provided by its own definition (see equation (9)). Hence, it becomes possible to apply the matchmaking algorithm to any number of service advertisements in a parallel manner.

The matchmaking algorithm presented here has been fully implemented by the specification of USQL [1] and a USQL-enacting service search engine prototype [13]. The matchmaker, the USQL language, and its associated search engine have been developed in the context of the SODIUM project [14] and form part of its provided platform [13]. Within the context of SODIUM, the presented matchmaking algorithm has been applied in a number of use cases accruing from applications in the domains of *Healthcare* and *Crisis Management*, where the ultimate selection of the most appropriate service for a given task was significantly facilitated.

6 Discussion

In this paper, we presented a service matchmaking algorithm capable of assessing complex service requests against a number of service advertisements and ranking the returned results. Furthermore, we evaluated the matchmaking algorithm's precision with the conduction of a preliminary experiment. Among the matchmaker's novelties is its applicability to any type of services, provided that their descriptions are compliant with the matchmaker's underlying service model. Also, as opposed to most of the related work that we have looked at, our matchmaking algorithm considers syntactic, semantic, and quality requirements in calculating the overall matching degree of a service. All in all, we believe that, the main contribution of our approach is the provision of an intuitive, unified matchmaking approach, in terms of service and requirements heterogeneity, which is capable of dealing with complex service requests. Our matchmaker alleviates users from the cumbersome task of separately matching syntactic, semantic, and QoS requirements and manually combining the results.

In the future, we aim at selectively accommodating in our matchmaking algorithm the most promising among the aforementioned efforts, by exploiting the fact that its definition remains independent from the way individual degrees of match are produced. This will allow us to conduct more extended experiments, which we expect to give interesting results. Finally, our immediate plans include extending the USQL language to support more than two priority levels for service requirements, so that we have the opportunity to better evaluate the matchmaking algorithm.

Acknowledgement. This work is partially supported by the Special Account of Research Funds of the National and Kapodistrian University of Athens under contract 70/4/5829 and by the European Commission under contract IST-FP6-004559 for the SODIUM project [14].

References

1. Tsalgatidou, A., Pantazoglou, M., Athanasopoulos, G. (2006) *Specification of the Unified Service Query Language*. Technical Report, http://cgi.di.uoa.gr/~michaelp/TR/usql-1.0-spec.pdf
2. Tsalgatidou, A., Athanasopoulos, G., Pantazoglou, M., et al. (2006) *Generic Service Model Specification*. Technical Report, http://cgi.di.uoa.gr/~gathanas/TR/gesmo-1.0-report.pdf

3. W3C Working Group (2004) *Web Services Architecture.* Note 11 February 2004, http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/
4. Eiter, T. et al. (2001) *Matchmaking for Structured Objects*. In Proc. of the Third International Conference on Data Warehousing and Knowledge Discovery, Lecture Notes In Computer Science, Vol. 2114, 186-194
5. Srinivasan, N., Paolucci, M., and Sycara, K. (2004) *An Efficient Algorithm for OWL-S Based Semantic Search in UDDI*. Semantic Web Services and Web Process Composition: First International Workshop (SWSWPC), San Diego, CA, USA
6. Li Kuang, Shuiguang Deng et al. (2005) *Exploring Semantic Technologies in Service Matchmaking*, Third European Conference on Web Services (ECOWS'05), 226-234
7. Bramantoro, A. et al. (2005) *A Semantic Distance Measure for Matching Web Services*. In Proc. of the Web Information Systems Engineering – WISE 2005 Workshops: WISE 2005 International Workshops, New York, NY, USA, 217-226
8. Klein, M., Bernstein, A. (2004) *Towards High-Precision Service Retrieval*. IEEE Internet Computing, 8 (1), 30-36, Jan/Feb, 2004.
9. Xin Dong, Alon Halevy, et al. (2004) *Similarity Search for Web Services*. In Proc. of VLDB, Canada
10. Taher, L. et al. (2005) *Establishing Association between QoS Properties in Service Oriented Architecture*. In Proc. of the International Conference on Next Generation Web Services Practices (NWeSP'05), 163-168,
11. Ribeiro, C. et al. (2006) *An Ontological Approach for Personalized Services*. 20th International Conference on Advanced Information Networking and Applications, Vol. 2, 729-733,
12. Sycara, K. et al. (1999) *Dynamic service matchmaking among agents in open information environments*. ACM SIGMOD Record 28 (1), Special Issue on Semantic Interoperability in Global Information Systems, 47–53
13. Tsalgatidou, A. et al. (2006) *Developing Scientific Workflows from Heterogeneous Services*. SIGMOD Record, Vol. 35 (2), 22-28
14. SODIUM Project, http://www.atc.gr/sodium
15. Mehran Sahami, Vibhu Mittal et al. (2004) *The Happy Searcher: Challenges in Web Information Retrieval*. Trends in Artificial Intelligence, 8th Pacific Rim International Conference on Artificial Intelligence (PRICAI)
16. Lin, D. (1998) *An information-theoretic definition of similarity*, in International Conference on Machine Learning

Pattern Based Property Specification and Verification for Service Composition

Jian Yu[1, 2], Tan Phan Manh[1], Jun Han[1], Yan Jin[1], Yanbo Han[2], and Jianwu Wang[2]

[1] Faculty of ICT, Swinburne University of Technology, 3122 Hawthorn, Australia
{jyu, phan_tan, jhan, yjin}@ict.swin.edu.au
[2] Grid and Service Computing Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China 100080
{yujian, yhan, wjw}@software.ict.ac.cn

Abstract. Service composition is becoming the dominant paradigm for developing Web service applications. It is important to ensure that a service composition complies with the requirements for the application. A rigorous compliance checking approach usually needs the requirements being specified in property specification formalisms such as temporal logics, which are difficult for ordinary software practitioners to comprehend. In this paper, we propose a property pattern based specification language, named PROPOLS, and use it to verify BPEL service composition schemas. PROPOLS is easy to understand and use, yet is formally based. It builds on Dwyer et al.'s property pattern system and extends it with the logical composition of patterns to accommodate the specification of complex requirements. PROPOLS is encoded in an ontology language, OWL, to facilitate the sharing and reuse of domain knowledge. A Finite State Automata based framework for verifying BPEL schemas against PROPOLS properties is also discussed.

1 Introduction

Web service composition is emerging as a promising technology for the effective integration of applications across globally distributed organizations [1, 2]. When encapsulating modular business functions as standard Web services, cross-organizational business processes can be built with a service composition language like BPEL [3] or BPML [4].

It is important to ensure the behavioral compliance between a service composition application and the requirements. Unexpected application behaviors may not only lead to mission failure, but also may bring negative impact on all the participants of this process. Model checking [5] is a formal approach to software behavioral compliance checking. In this approach, a software application is abstracted as a formal model like Labeled Transition Systems (LTS), Finite State Automata (FSA), Petri nets, or process algebra. The behavioral requirements are specified as properties in formalisms such as Linear Temporal Logic (LTL), Computation Tree Logic (CTL), or Quantified Regular Expressions (QRE). Then the formal model can be verified against the specified requirements/properties through exhaustive state space

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 156 – 168, 2006.

exploration. A serious problem, however, prevents the wide adoption of this approach. That is, the formal properties are surprisingly difficult to write for practitioners, who usually don't have solid mathematical backgrounds [6, 7].

In this paper, we present a lightweight specification language called PROPOLS (Property Specification Pattern Ontology Language for Service Composition), and an associated approach to the verification of BPEL schemas. PROPOLS is an OWL-based high-level pattern language for specifying the behavioral properties of service composition applications. PROPOLS is based on Dwyer et al.'s property patterns [6], which are high-level abstractions of frequently used temporal logic formulae. The property patterns enable people who are not experts in temporal logics to read and write formal specifications with ease and thus make model checking tools more accessible to common software practitioners [8]. Although it is claimed in [6] that patterns can be nested, no further work has been done on how to define composite patterns and what are their semantics. PROPOLS refines/extends the original pattern system in [6] by introducing the logical composition of patterns. This mechanism enables the definition of complex requirements in terms of property patterns, which is previously difficult or even impossible. PROPOLS uses the Web Ontology Language (OWL) as its base language. This makes PROPOLS properties sharable and reusable within/across application domains.

In addition to the PROPOLS language, we present a verification framework for checking the compliance of BPEL schemas against PROPOLS properties. The key techniques used include representing the semantics of PROPOLS properties as FSAs, representing the semantics of a BPEL schema as a LTS/FSA using Foster's BPEL2LTS tool [9], and checking the language inclusion between these two FSAs. The verification approach is illustrated using a non-trivial example.

This paper is structured as follows. In the next section, a motivating example scenario is presented. Section 3 gives a brief introduction to the original property specification patterns. Section 4 presents PROPOLS, including its syntax and semantics. Section 5 introduces the BPEL verification framework and illustrates the verification approach based on the example scenario. Finally, we discuss the related work in section 6 and conclude the paper in section 7.

2 A Motivating Scenario

Let's assume a computer manufacturing company AdvantWise is to provide an online purchasing service. The key requirements to this service are sketched as follows:

1) Customers can place purchase orders online. Any order should be checked before being processed. For an invalid order, the customer will get a rejection notification. For a valid one, the customer will get a confirmation.
2) The transactions between customers and AdvantWise follow a hard credit rule. That is, on the one hand, the customer pays to AdvantWise only when she has received the ordered products. On the other hand, AdvantWise processes the order only when it is confirmed that the customer will pay if the order is fulfilled. For this to be possible, a third party, bank, is introduced. To let AdvantWise start processing order, the customer must deposit the payment to the bank first. Then the bank will notify AdvantWise that the payment for the order has been already

kept in the bank. Anytime when the order is canceled, the payment will be refunded by the bank. If the order is fulfilled, the bank ensures that the payment is transferred to AdvantWise.

3) To fulfill an order, AdvantWise first checks its inventory. If there are enough products in the inventory, the products of the ordered quantity are packaged for shipping. If not, AdvantWise will initiate an emergency production schedule.

Clearly, the above-stated requirements express certain business constraints that the system implementation must follow. On the one hand, it is not easy to translate the embedded behavioral constraints into temporal logics in a straightforward manner. On the other hand, Dwyler's patterns are not expressive enough to state all the constraints, for example, the mutual exclusion between rejection and confirmation. After introducing PROPOLS in section 4, we will formulate all these requirements in PROPOLS, against which the BPEL service composition being designed in section 5 will be checked.

3 Property Specification Patterns

Property specification patterns were first proposed by Dwyer et al in [6]. These patterns include a set of commonly occurring high-level specification abstractions for formalisms like LTL, CTL or QRE. They enable people who are not experts in such formalisms to read and write formal specifications. According to [10], 92% of 555 property specifications collected from different sources matched one of the patterns.

A pattern property specification consists of a *pattern* and a *scope*. The pattern specifies *what* must occur and the scope specifies *when* the pattern must hold.

Patterns are classified into occurrence patterns and order patterns. We briefly describe the meaning of the important patterns below (the symbol P or Q represents a given state/event). Details of these patterns can be found in [11].

- **Absence:** P does not occur within a scope.
- **Universality:** P occurs throughout a scope.
- **Existence:** P must occur within a scope.
- **Bounded Existence:** P must occur at least/exactly/at most k times within a scope.
- **Precedence:** P must always be preceded by Q within a scope.
- **Response:** P must always be followed by Q within a scope.

A scope defines a starting and an ending state/event for a pattern. There are five basic kinds of scopes:

- **Globally:** the pattern must hold during the entire system execution.
- **Before:** the pattern must hold up to the first occurrence of a given P.
- **After:** the pattern must hold after the first occurrence of a given P.
- **Between…And:** the pattern must hold from an occurrence of a given P to an occurrence of a given Q.
- **After…Until:** the same as "between…and", but the pattern must hold even if Q never occurs.

The semantics of pattern properties may be given in LTL, CTL, QRE, or FSA [6, 7, 12, 13]. Fig. 1 illustrates the FSA semantics by three pattern properties. There, the

symbol 'O' denotes any other state/event than P and Q. Fig. 1(a) indicates that, before P occurs, an occurrence of Q is not accepted by the FSA. Fig. 1(b) states that if Q has occurred, an occurrence of P is necessary to drive the FSA to a final state. Finally, Fig. 1(c) says that only the occurrence of P can make the FSA reach a final state.

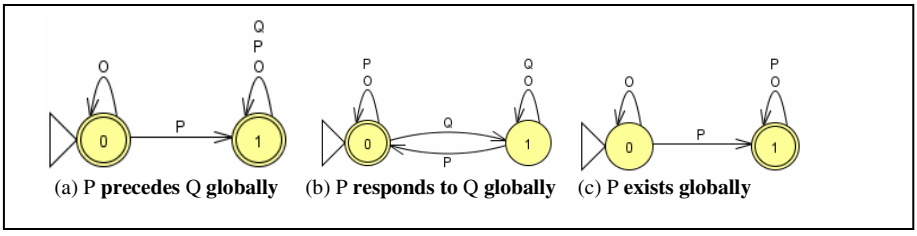


Fig. 1. FSA Semantics of Example Pattern Properties

4 The PROPOLS Language

In this section, we first illustrate the structure and main components of the PROPOLS language. We then explain the syntax and semantics of the composite pattern, which are refinements/extensions to the original pattern system. A composite pattern is built by connecting pattern properties using logic operators. The main benefit of the composite pattern mechanism lies in that it enables the specification of complex requirements. Finally, the PROPOLS properties for the example scenario are given.

4.1 Language Structure

We designed PROPOLS as an ontology language for two purposes: First, ontology can be used to define standard terminology for the pattern system. Second, practitioners like business experts and system analysts can use concepts from existing domain ontologies to define pattern properties at a high level of abstraction. The benefits include: The pattern properties themselves are also become part of the shared formal domain knowledge, so they can be reused in a wide scope, and off-the-shelf semantic Web techniques can be used to map properties to related Web services automatically. At present, OWL is the most widely used Web ontology language. So we choose OWL as the base language of PROPOLS.

Fig. 2 shows a graphical overview of the PROPOLS ontology. It is generated from Ontoviz [14], an ontology visualization plug-in for an OWL editor tool Protégé. Its major elements are explained below.

OrderPattern: This class defines the set of order patterns we discussed in section 3. We use a more natural name *LeadsTo* to replace the original name *RespondsTo*. Here, P *LeadsTo* Q is the same as Q *RespondsTo* P.

OccurrencePattern: This class defines the set of occurrence patterns we discussed in section 3.

Scope: As discussed in section 3, every elementary pattern will be associated with a scope coming from *globally*, *before*, *after*, *between…and* or *after…until*.

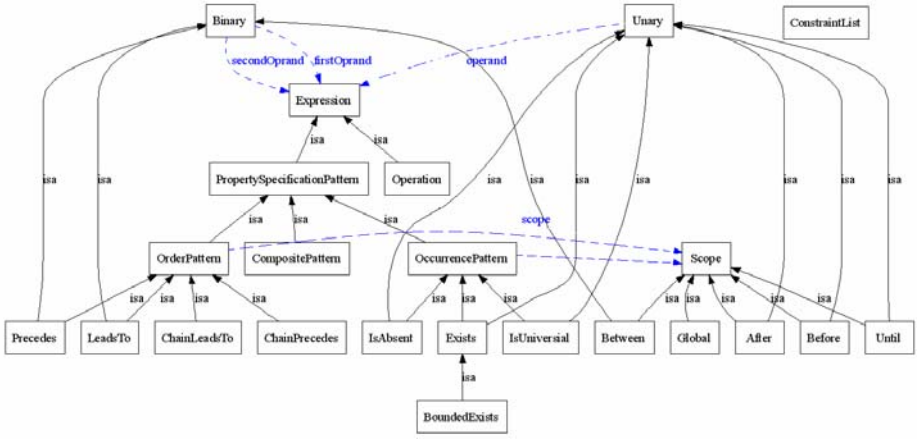


Fig. 2. PROPOLS Ontology

Operation: This is the class for service operations. An operation is considered as an event in the pattern properties. It has a *provider* property linking to its provider and a *conceptReference* property linking to the corresponding concept in some ontology. Later, the *conceptReference* can be used to semantically map an event in pattern properties to a Web service operation and vice versa.

Expression, Unary and Binary: *Expression* is a general class for patterns and operations. For the succinctness of the PROPOLS ontology, all the patterns and scopes are the subclasses of *Unary* or *Binary*. So they share the properties defined in Unary or Binary, either has an *operand* property pointing to *Expression* or has a *firstOperand* property and a *secondOperand* property. For example, *Exists* is a unary pattern and *Between* is a binary scope. Further restrictions are set for different kinds of expressions. For example, a scope only accepts operations as its operands.
ConstriantList:. This class is a container for pattern properties/constraints.

4.2 Composite Patterns

As shown in Fig. 3, a composite pattern is the composition of pattern properties using Boolean logic operators including *Not*, *And*, *Or*, *Xor* and *Imply*.

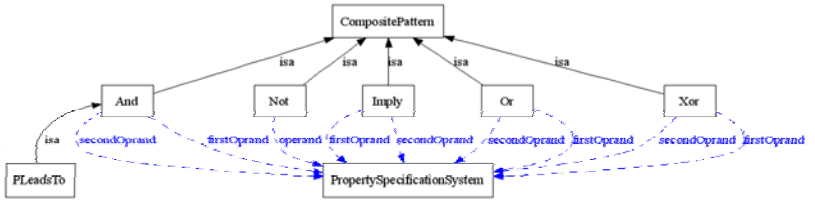


Fig. 3. Composite Pattern Structure

With composite patterns, complex properties can be stated straightforward. For example, if we want to define a property according to "If the order is fulfilled, the bank ensures that the payment transfers to AdvantWise". We may write a composite pattern connected with the 'And' operator:

```
Customer.GetOrderFulfilled Precedes Bank.Transfer Globally
And
Customer.GetOrderFulfilled LeadsTo Bank.Transfer Globally
```
(1)

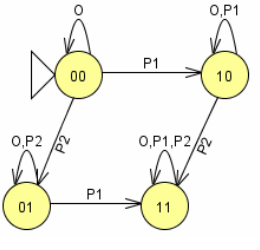
In this situation, both 'GetOrderFulfilled' and 'Transfer' should occur and must occur sequentially in the program. This composite pattern is used frequently, so we add a stereotyped composite pattern: *PLeadsTo,* to our pattern system.

Another example of complex properties is the first requirement of the example scenario. This requirement claims an 'exclusive or' relation between order confirmation and rejection. It also demands a precedent occurrence of the "check order" activity. With composite patterns, we may specify this requirement as follows:

```
(Customer.ConfirmOrder Exists Globally
Xor Customer.RejectOrder Exists Globally)
And Manufacturer.CheckOrder Precedes Customer.ConfirmOrder
 Globally
And Manufacturer.checkOrder Precedes Customer.RejectOrder Globally
```
(2)

Next, we briefly explain the semantics of composite patterns. A formal semantics definition can be found in an accompanying technical report [15].

As stated earlier, every elementary pattern property has a corresponding FSA semantics. We thus define the semantics of a composite pattern property from the logical composition of FSAs of its component pattern properties. For example, Fig. 4 shows the resultant FSAs of 4 logical compositions between two properties: "P1 **exists globally**" and "P2 **exists globally**". The states are the Cartesian product of the two property FSAs' states. The first number in a state label represents the state of the first property FSA, while the second represents the state of the second property FSA. The final states of the composite pattern are determined by the logic operator used. For example, the pairing of one final state 'And' one non-final state is a non-final state, and one final state 'Xor' one non-final state is a final state. The final states for different compositions are also described in Fig. 4.



Logic Operator	Final States
And	11
Or	01,10,11
Xor	01,10
Imply	00,01,11

Fig. 4. Logical Compositions of Two 'Exists' Properties

With such definitions, we have proved in [15] a theorem: the language set that can be accepted by a composite FSA is the composition of the language set of the component elementary FSAs. For example, if we have a composite pattern 'pr1 And pr2', then Language(pr1 **And** pr2) = Language(pr1) **And** Language(pr2). Here, the '**And**' between two sets Language(pr1) and Language(pr2) is the set-intersection operator. This theorem proves the correctness of our semantics definition to composite patterns.

4.3 PROPOLS Example

In this subsection, we use PROPOLS to describe the example requirements introduced in section 2. For easy reading, we first write the pattern properties in a simple text format in Fig. 5.

In Fig. 5, the first property is for requirement 1. The second, third, and fourth constraints are for requirement 2. The last property is for requirement 3. These pattern properties are quite intuitive and self-explanatory.

```
Constraint List: Hard Credit Rule
1.   (See property (2) in section 4.2)
2.   Customer.ConfirmOrder PLeadsto
      Bank.Deposit Globally
3.   Bank.Depoist PLeadsto
      Manufacturer.StartOrderProcessing Globally
4.   Customer.GetOrderFulfilled PLeadsto
      Bank.Transfer Globally
5.   Manufacturer.CheckInventory Precedes
      Manufacturer.ScheduleProducing Globally
```

Fig. 5. Pattern Properties for the Example Requirements

To showcase the real definition of properties, we present part of the PROPOLS code for the first property in Fig. 6. A complete version can be found in [15].

```
<And rdf:ID="And_Property">
   <secondOprand rdf:resource="#Pre_Canc"/>
   <firstOprand rdf:resource="#And_inst_1"/></And>
<Precedes rdf:ID="Pre_Canc">
  <scope><Global rdf:ID="Global_Inst"/></scope>
     <firstOprand>
        <Operation rdf:ID="checkOrder">
         <provider rdf:datatype="&XMLSchema;anyURI">
          #Manufacturer</provider>
        <conceptReference rdf:datatype="&XMLSchema;anyURI">
           #CheckOrder</conceptReference>
        </Operation></firstOprand>
    <secondOprand rdf:resource="#RejectOrder"/>
</Precedes>
……
```

Fig. 6. Excerpt of an Example PROPOLS Property Definition

5 Verification of BPEL

In this section, we present an approach to the compliance checking of BPEL schemas against PROPOLS properties. The presentation starts with a high-level description of a BPEL schema to implement AdvantWise's online purchase process in the aforementioned example scenario, and then explains the verification approach using the example BPEL schema. A snapshot of our developed verification tool is also presented.

5.1 An Example BPEL Schema

Fig. 7 shows the main structure of the BPEL schema as an implementation of the online purchase process for AdvantWise. This diagram keeps the message exchange primitives in the actual BPEL schema. It uses black bars to represent the other participants of the process, including the Customer and the Bank.

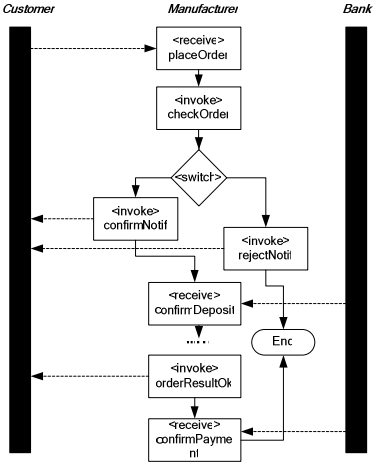


Fig. 7. Example BPEL Schema: Main Structure

5.2 Verification Approach

Fig. 8 illustrates our approach to verifying the compliance of BPEL schemas to PROPOLS properties. The main steps of the verification process are explained below.

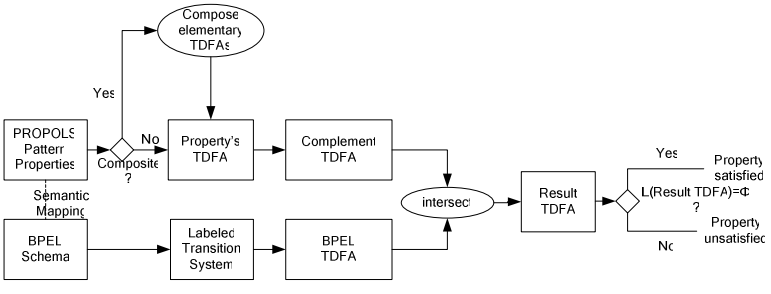


Fig. 8. Principle of PROPOLS Verification Approach

Semantic Mapping. First of all, there is a semantic mapping between the operations defined in PROPOLS and Web service operations. The mapping is based on the condition that both of these operations refer to the same concept in the domain ontology. The PROPOLS operations can tell their semantics via the *conceptReference* property. There are two typical ways in which one can add semantic annotations to Web service operations: by using an external annotation file or directly appending semantic annotations to the WSDL file. In this context, we use the latter approach. As exemplified in Fig. 9, we extend the *Customer* Web service's operation definition with WSDL-S semantic extension element *wssem:modelReference*.

```
<portType name="Customer>
  <operation name="confirmNotif"
    wssem:modelReference="
    http://www.purl.org/onto/operationOnto#ConfirmOrder>
    …
   <operation name="rejectNotif"
     wssem:modelReference="
     http://www.purl.org/onto/operationOnto#RejectOrder>
…
```

Fig. 9. Example Semantic Annotations to Web Service Operations in WSDL

In the above example, the Web service operation *confirmNotif* and PROPOLS operation *ConfirmOrder* refer to the same ontology concept, so we can use *confirmNotif* to replace *ConfirmOrder* in the compliance checking process. A complete treatment of the mapping between operations can be found in [15].

Verification Process. As shown in Fig. 8, the verification is conducted in 3 steps. (1) For every pattern property, a semantic equivalent Total and Deterministic FSA (TDFA) is built. If the property is a composite one, the corresponding TDFA is constructed by composing the TDFAs of its sub-properties according to the composition semantics definition in section 4.2. (2) For the BPEL schema, a finite and deterministic LTS model is generated, from which a TDFA is built by introducing the set of final states and an error state to collect all the unacceptable events (or operation invocations) of each state. (3) The compliance of the BPEL schema to the PROPOLS properties is then checked as a verification problem of whether the accepting event sequences of the BPEL TDFA are all present in the accepting event

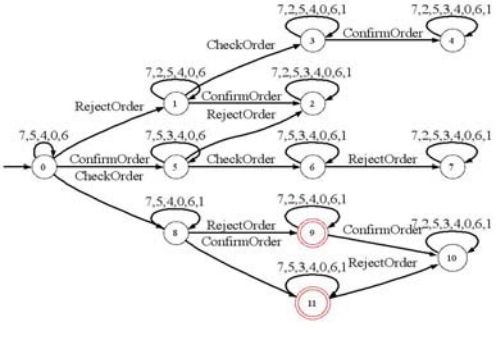
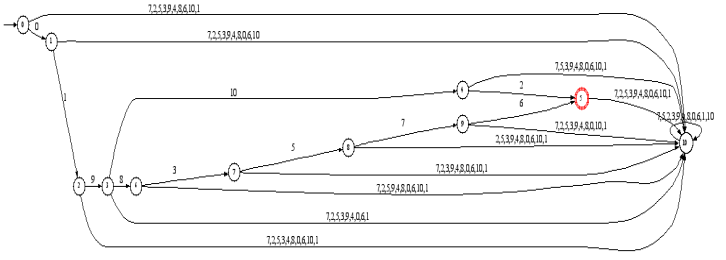


Fig. 10. TDFA of the Example Property

sequence set of the property TDFA. This is done by testing the emptiness of the intersection of the BPEL TDFA and the complement of the property TDFA. A detailed explanation of this approach can be found in [15].

Next, we illustrate the verification process using property 1 in Fig. 9. Fig. 10 shows property 1's TDFA. Fig. 11 shows the TFDA of our example BPEL schema. This TFDA was obtained in two steps. First, Foster's BPEL2LTS tool [9] is used to obtain a LTS for the BPEL schema. Then, the LTS is transformed to a FSA (See [15] for detail).



Id	Event
0	rejectNotif
1	placeOrder
2	checkOrder
3	confirmNotif
4	confirmDeposit
5	fulfilOrder
6	confirmPayment
7	issueInvoice
8	checkOrderReply
9	orderResultFail
10	orderResultOk

Fig. 11. TDFA of the Example BPEL Schema

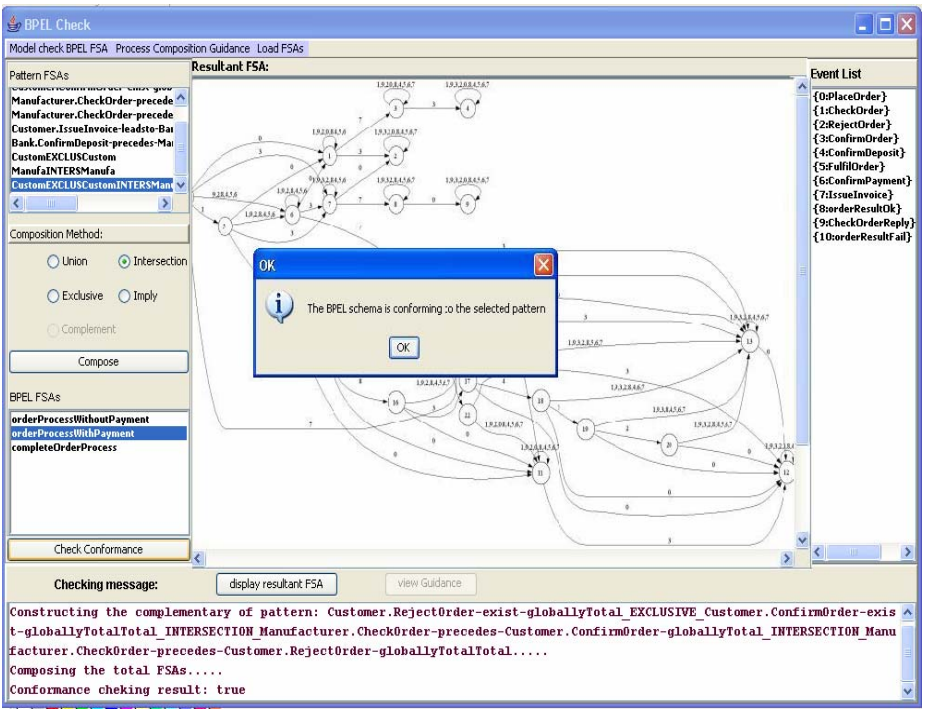


Fig. 12. Snapshot of the Verification Example

After obtaining the complement of the property TDFA, it is intersected with the BPEL TDFA. It turned out that the resultant FSA has no final state. This means that the BPEL schema conforms to the example pattern property. A snapshot of the final result is shown in Fig. 12.

We have implemented a prototype tool for verifying BPEL Schemas against PROPOLS properties. Fig. 12 is the snapshot of this tool. The left panel contains a list of properties, the logical operators, and the BPEL schema waiting for verification. While composing a property and a BPEL schema according to the above-stated approach, the resultant TDFA is shown in the right panel.

6 Related Work

Particularly relevant to the work presented in this paper are two streams of work: formal verification of BPEL schemas and property specification patterns.

A body of work has been reported in the area of formal property specification and verification of BPEL schemas. The main differences among them lie in the property specification language and the formal semantic model for BPEL. In [16], Foster relies on Finite State Processes (FSPs) to both semantically represent a BPEL schema and specify the properties. In [17], Stahl maps BPEL schemas into Petri nets and utilises a verification tool LORA (Low Level Petri net Analyzer) to verify CTL properties. In [18], the authors map BPEL into Promela, the input language of the model checker SPIN, and then use SPIN to check LTL properties. The most significant difference between these approaches and our work is that we focus on a practitioner-oriented approach to property specification. Compared to their heavy-weighted formal property specification languages, our pattern-based PROPOLS language is easier to understand and use. Also, its ontology-based nature helps in establishing a natural connection between pattern-based properties and the domain knowledge.

The property specification patterns are originally proposed by Dwyer et al. in [6]. Since then they have been applied and extended in many ways. Smith et al. [7] proposed a specification approach which enables fine-tuning patterns to achieve more precise meanings, based on a combined use of a “disciplined” natural language and a FSA template language. For example, six different templates were identified to fine-tune the response pattern. Gruhn et al. [8] extended the patterns with time for describing real-time related properties. Paun et al. [19] extended the patterns to deal with events in a state-based formalism. Furthermore, the patterns are the foundation for the extensible specification language in the Bandera system [20].

7 Conclusion

In this paper, we proposed a verification approach to BPEL schemas, which employs an ontology language PROPOLS for the property specification. PROPOLS builds on and extends Dwyer et al.’s pattern system. Its pattern-based nature enables software practitioners to write formal behavioral properties more easily. Its logical composite pattern mechanism allows one to state complex requirements. The ontology encoding also facilitates the sharing and reuse of PROPOLS constraints.

In the future, we intend to develop a graphical interface for the PROPOLS language. We also intend to apply the pattern properties in PROPOLS to provide just-in-time guidance to the BPEL designer during the service composition process.

Acknowledgments. We would like to thank Dr. Howard Foster at Imperial College, London, for his help in using his tool to translate BPEL schemas to Labeled Transition Systems.

References

1. Papazoglou, M.P., Georgakopoulos, D.: Special Issue on Service Oriented Computing. Communications of ACM 46 (10) (2003) 24 – 28
2. Alonso, G., Casati, F., Grigori, Kuno H., Machiraju, V.: Web Services Concepts, Architectures and Applications. Springer-Verlag (2004)
3. Arkin, A., Askary, S., Bloch, B., Curbera, F., Goland, Y., Kartha, N., Liu, C.K., Thatte, S., Yendluri, P., Yiu, A.: Web Services Business Process Execution Language Version 2.0 Working Draft. http://www.oasis-open.org/committees/download.php/10347/wsbpel-specification-draft-120204.htm (2004)
4. BPMI: Business Process Modeling Language. http://www.bpmi.org/ (2002)
5. Clarke, E.M., Moon, I., Powers, G.J., Burch, J.R.: Automatic Verification of Sequential Control Systems using Temporal Logic. American Institute of Chemical Engineers Journal, 38 (1) (1992) 67 – 75
6. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Property Specification Patterns for Finite-State Verification. In 2nd Workshop on Formal Methods in Software Practice (1998) 7 - 15, Clearwater Beach, FL, USA
7. Smith, R.L., Avrunin, G.S., Clarke L.A., Osterweil L.J.: PROPEL: An Approach Supporting Property Elucidation. In Proc. 24th International Conference on Software Engineering (2002) 11-21, Orlando, FL, USA
8. Gruhn V. Laue R.: Specification Patterns for Time-Related Properties. In 12th International Symposium on Temporal Representation and Reasoning (2005) 189 - 191, Burlington, Vermont, USA
9. Foster, H.: LTSA WS-Engineering. http://www.doc.ic.ac.uk/ltsa/bpel4ws/ (2006)
10. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Patterns in Property Specifications for Finite state Verification. In Proc. International Conference on Software Engineering (1999) 411-420, Los Angeles, CA, USA
11. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: A System of Specification Patterns. http://www.cis.ksu.edu/santos/spec-patterns, (1997)
12. Jin, Y., Han, J.: Consistency and Interoperability Checking for Component Interaction Rules. In Proc. 12th Asia-Pacific Software Engineering Conference (2005), Taipei, Taiwan
13. Li, Z., Han, J., Jin, Y.: Pattern-Based Specification and Validation of Web Services Interaction Properties. In Proc. 3rd International Conference on Service Oriented Computing (2005) Amsterdam, Netherland
14. OntoViz Tab: Visualizing Protégé Ontologies http://protege.stanford.edu/plugins/ontoviz/ontoviz.html (2005)
15. Yu, J., Phan, M.T., Han, J., Jin, Y.: Pattern based Property Specification and Verification for Service Composition. Technical Report SUT.CeCSES-TR010. CeCSES, Swinburne University of Technology, http://www.it.swin.edu.au/centres/cecses/trs.htm (2006)

16. Foster, H.: A Rigorous Approach to Engineering Web Services Compositions. PhD thesis, Imperial College London. http://www.doc.ict.ac.uk/~hf1 (2006)
17. Stahl C.: A Petri Net Semantics for BPEL. Informatik-Berichte 188, Humboldt-Universitat zu Berlin, June 2005 (2005)
18. Fu, X., Bultan T., Su J.: Analysis of Interacting BPEL Web Services. In Proc. 13th World Wide Web Conference (2004) 621-630, New York, NY, USA
19. Paun, D.O., Chechik, M: Events in Linear-Time Properties. In Proc. 4th International Conference on Requirements Engineering (1999) Limerick, Ireland
20. Corbett, J.C., Dwyer, M.B., Hatcliff, J., Laubach', S., Pasareanu, C.S., Zheng, R., Zheng, H: Bandera: Extracting finite-state models from Java source code. In Proc. 22nd International Conference on Software Engineering (2000) 439-448, Limerick, Irelan

Detecting the Web Services Feature Interactions*

Jianyin Zhang, Fangchun Yang, and Sen Su

State Key Laboratory of Networking and Switching Technology
Beijing University of Posts & Telecommunications (BUPT), China
marcozjy@gmail.com, {fcyang, susen}@bupt.edu.cn

Abstract. Feature interaction has been first identified as a problem in the telecom. But it is not limited in this domain, it can also occur in any software system that is subject to changes. With the introduction of service composition technique in the Web services area, a variety of message interactions are resulted among the composed services. As a result, Web Services Feature Interactions problem becomes inevitable. This paper investigates the detection of this problem and proposes a novel multilayer detection system (WSFIDS) by taking inspiration from the immune system. With the application of some immune principles, the system provides an effective method to detect both known and previously unknown feature interactions.

1 Introduction

Feature interaction problem [1, 2, 3], firstly coined in the telecom by Bellcore, is one of the important bottlenecks for the development of new services. It mainly includes interactions that occur because the requirements of multiple features are not compatible, and interactions that occur when a feature behaves differently in the presence of other features. But this phenomenon is not unique to the telecom. It can occur in any software system that is subject to changes, such as component-based systems beyond telecom.

In the Web services area, the introduction of service composition in the service creation brings the potential of similar problem as feature interaction in the telecom. The message interactions among the atomic services resulting from the service composition bring two types of "feature interactions" in Web services. One type leads to the normal service composition, and the other will negatively impact the service quality and robustness, reduce the service scalability, and even result in the failure of service composition in practice. In this paper, we only pay attention to the latter type of "feature interactions" and define its related problem as Web Services Feature Interactions (WSFI) problem. The detection and resolution of WSFI problem will play an important role to increase both the flexibility of introducing new services and the robustness of the composite services.

Although the WSFI problem was first introduced in the sixth International Workshop on Feature Interactions in Telecommunication and Software Systems

* Supported by the National Basic Research and Development Program (973 program) of China under Grant No.2003CB314806; the Program for New Century Excellent Talents in University (No: NCET-05-0114); the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT).

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 169 – 174, 2006.

(FIW'00), it has not received enough attention as yet. Michael Weiss [4] first addresses this field. He presents a goal-oriented approach for detecting feature interactions among Web services. He also distinguishes explicitly between functional and non-functional feature interactions. In [5] he extends his work with emphasis on the classification of feature interactions. He presents a classification on the basis of the classical classification of feature interaction problems in the telecom. Moreover, he analyzes the different types of the potential WSFI in a "Virtual Bookstore" scenario. Kenneth J. Turner [6] extends feature notation CRESS to graphically and formally describe Web services and service composition. Moreover, he briefly discusses the WSFI detection by using CRESS and scenario notation MUSTARD together. We [7] propose a Petri net-based method to detect the race conditions, one type of functional feature interactions. On modeling the composite Web services, Petri net analysis methods are employed in the detection algorithm.

In this paper, we propose a novel multi-layer WSFI detection system (WSFIDS) by taking inspiration from the immune system and the application of immune principles in the areas of network security [8, 9], NGN service interaction [10], etc. The rest of this paper is organized as follows: We first describe the system architecture of WSFIDS and then illustrate the WSFI detection procedure in detail. After discussing the properties of WSFIDS, we conclude this paper with the future work finally.

2 WSFI Detection

The rapid development of Web services and its distributed nature lead to the great diversity of service providers. Because of the privacy issue, the internal service logic of one Web service is hardly obtained by others except its provider. Thus those detection methods, in which the internal service logic is needed, are not applicable to the WSFI detection. So an effective WSFI detection system should perform the detection task by analyzing the interaction information among the composed services, rather than by utilizing the internal information of service logic in the atomic services.

2.1 Comparison Between the Immune System and WSFIDS

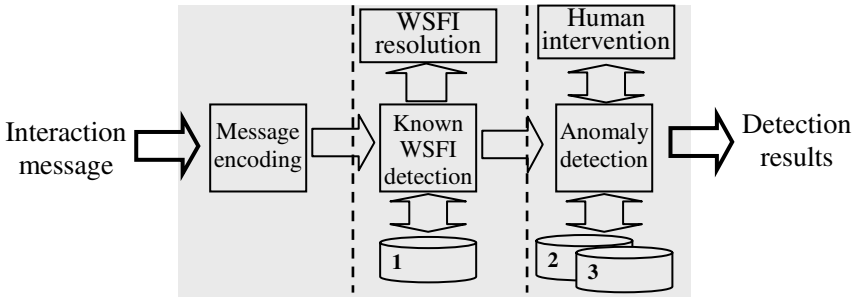
The WSFI detection system aims at detecting the abnormal feature interactions in order to maintain the normal service composition, which is similar in functionality to the immune system protecting the body from the nonself cells and ensuring the normal system operations. The other similarities are summarized thorough the comparison between the immune system and WSFIDS (see Fig. 1).

Immune System	WSFIDS
Nonself cells/antigens	WSFI
Self cells	Normal message interactions
Antigen presentation	Message analysis and encoding
Antigen recognition/ Negative selection	String matching
Co-stimulation	Human intervention
Immune memory	Anomaly detector storage

Fig. 1. Comparison between the immune system and WSFIDS

2.2 Functional Layers of WSFIDS

WSFIDS acts in a centralized fashion because none of the decentralized mechanism is found to be suitable for the WSFI detection. We assume that all the interaction messages among the composed services can be got and fully understood by WSFIDS. In the system implementation, we may deploy the WSFIDS on the Web services composition engine, the entity to control the compositional logic. WSFIDS is a multilayer defense system, and its functional layers are shown in Fig.2. Below we will describe each layer in detail.



1 DIB I : stores the known WSFI information
2 DIB II : stores the information of new WSFI detector
3 DIB III: stores the normal interaction information

Fig. 2. Functional layers of WSFIDS. DIB is the abbreviation of Detection Information Base.

2.2.1 Layer I: Message Encoding

This is the first defense layer in WSFIDS. It performs the similar function as antigen presentation in the immune system. When the incoming message is received, the detection-related information is extracted from the message and encoded as a string to prepare for the message matching.

Segment I	Message Type	Atomic Service ID	Operation ID
Segment II	Data Type	Data Value	Data Range
Segment III	Time Constraint	Data Constraint	State Constraint

Fig. 3. The basic message encoding structure

According to the known WSFI and service composition language [11, 12], we present the basic message encoding structure (see Fig.3), where segment I, II and III correspond to the message information, data Information and constraint information, respectively. We adopt a variable structure consisting of a cluster of segments (or strings). More segments may be added to meet the detection requirements in practice.

2.2.2 Layer II: Known WSFI Detection

The purpose of the layer is to detect whether one type of known WSFI happens by string matching between the incoming information and the known WSFI information

from DIB I (see Fig.2). Once a match is made, it means one type of known WSFI happens and the corresponding WSFI resolution will be activated to resolve the WSFI. The mechanism and methods of WSFI resolution is out of the scope of this paper and will not be discussed here.

The DIB I strings (or detectors) act as the "memory cells" in the immune system. They are long-lived until the service composition is completed so that the WSFI information stored in these strings will not be lost in the detection process. They are designed using the similar method as message encoding.

2.2.3 Layer III: Anomaly Detection

In this layer, those immune principles, including negative selection, co-stimulation, and immune memory, are applied to the anomaly detection, where anomaly is restricted to those WSFI previously unknown. Anomaly detection begins with string matching between the incoming information string and the anomaly detector string from DIB II (see Fig.2). If the two strings match, it is likely that WSFI happens. Then human intervention will be performed to determine whether WSFI really happens. Here human intervention performs the similar function as "co-stimulation" in the immune process. The "co-stimulation" will be positive if the incoming information is confirmed to be the WSFI information. Then WSFI resolution works.

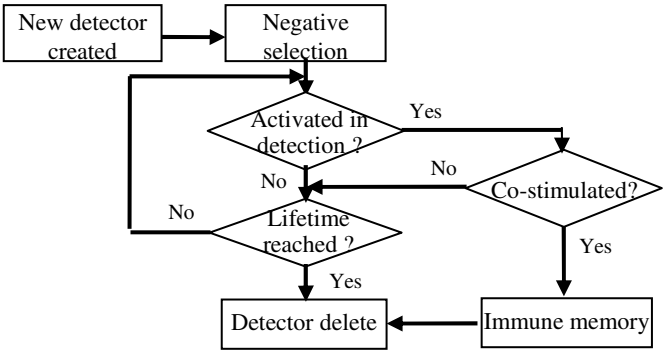


Fig. 4. Anomaly detector string lifecycle model

The lifecycle of a DIB II string is illustrated in Fig.4. At first, the new detector string is generated according to the DIB I string (see Fig.5), rather than randomly created. Then it is stored in the DIB II. After being censored against the self string from DIB III in the "negative selection" process (see Fig.6), the new string will be used for the detection. When a string match is made, the string will be activated. If an activated string receive a positive "co-stimulation", it is copied to DIB I and deleted from DIB II. Whenever the string reaches the lifetime in the detection process, it will be deleted immediately.

Decreasing the string numbers in DIB II generally decrease the numbers of match and the overall detection time in the anomaly detection. But it increases the false-negative error rate, the error rate of treating WSFI information as the normal interaction information. Therefore a tradeoff between the detection efficiency and false-negative error rate should be made in the DIB II design.

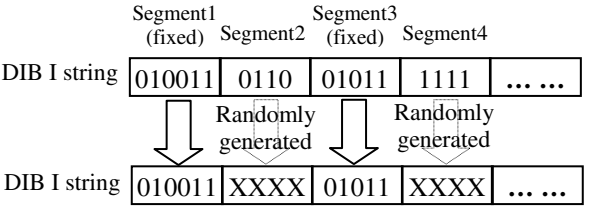


Fig. 5. DIB II string generation. Some segments of DIB I string will be fixed and copied to the corresponding segments of the new DIB II string directly. And other segments of the new string will be randomly generated.

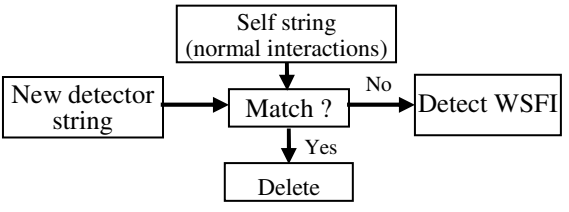


Fig. 6. Negative selection in the anomaly detection

2.4 Analysis

In our point of view, a robust system for detecting the WSFI should be capable of detecting dynamically not only all kinds of specific known feature interactions, but also unknown feature interactions, in a uniform manner. According to this criterion, we present a multilayer detection system -- WSFIDS. The system overcomes the drawbacks of static detection method mostly employed in the current research and the limitation of classification-based approach. It mainly has the following properties:

1) *Multilayer defense*: WSFIDS employs a multilayer defense mechanism for the reliable and in-depth detection. Although each layer operates independently, the integrity of all layers will take effect on improving the detection performance.
2) *Online mode*: WSFIDS employs an online method for the WSFI detection. Compared with static detection methods, it is more effective in the practical scenarios of Web service composition.
3) *Learning ability*: WSFIDS has the ability to "learn" and "remember" the WSFI previously unseen, then improves the response upon later encounter of the same type of WSFI. This self-perfection characteristic of WSFIDS greatly improves the detection ability to distinguish "nonself" from "self".
4) *Anomaly detection*: Anomaly detection is essential to the WSFI detection because composite service will definitely encounter new "pathogens" in the runtime. Using ideas from the immune principles such as immune memory, WSFIDS can detect not only the known WSFI, but also those novel ones.

3 Conclusion

With the rapid development of Web services, WSFI will inevitably become an important obstacle for the new service creation. Based on the immune system, we

propose a novel multilayer system to perform online detection of WSFI. With the immune system being an adequate source of inspiration, we believe that our immunity-inspired approach is promising in the WSFI detection and well worth continuing investigation.

In terms of future work, we intend to investigate a decentralized mechanism applying to WSFIDS. Moreover, testing our system against a common benchmark is desired in order to compare the efficiency and accuracy of the proposed method against other solutions. Unfortunately, there does not exist such a baseline. So we expect to supplement our work with those contents in the future.

References

1. Keck D. O. and Kuehn P.J., The Feature and Service Interaction Problem in Telecommunications Systems: A Survey, IEEE Transactions on Software Engineering, October 1998, 24(10): 779-796
2. Calder M, Kolberg M, Magill E, S. Reiff-Marganiec, Feature interaction: a critical review and considered forecast, Computer Networks, 2003, 41 (1): 115-141
3. Elke Pulvermueller, Andreas Speck, James O. Coplien, Maja D'Hondt, Wolfgang DeMeuter, Feature interaction in composed system, In: Proceeding of Feature Interaction in Composed System (ECOOP 2001),2001: 1-6
4. Weiss, M. and Esfandiari, B., On Feature Interactions among Web Services, In: Proceedings of the International Conference on Web Services (ICWS), USA, 2004
5. Michael Weiss, Babak Esfandiari, and Yun Luo, Towards a Classification of Web Service Feature Interactions, In: Proceedings of the Third International Conference on Service Oriented Computing (ICSOC05), Amsterdam, Netherlands, 2005
6. Kenneth J. Turner. Formalising Web Services. In: Proceeding of Formal Techniques for Networked and Distributed Systems (FORTE XVIII), LNCS 3731, October 2005: 473-488
7. Jianyin Zhang, Sen Su, Fangchun Yang, Detecting Race Conditions in Web Services, In: Proceedings of the International Conference on Internet and Web Applications and Services (ICIW'06), Guadeloupe, French, February 2006.
8. Harmer, P.K.; Williams, P.D.; Gunsch, G.H.; Lamont, G.B., An artificial immune system architecture for computer security applications, IEEE Transactions on Evolutionary Computation, June 2002, 6(3):252-280
9. Dasgupta, D., Gonzalez, F., An immunity-based technique to characterize intrusions in computer networks, IEEE Transactions on Evolutionary Computation, June 2002, 6(3): 281-291
10. Wenjian Xiong, An online NGN service interaction detection method based on immunology theory (Ph.D. dissertation), Beijing: School of Computer Science and Technology, Beijing University of Posts and Telecommunications, 2005 (in Chinese)
11. T. Andrews et al., editors. Business Process Execution Language for Web Services. Version 1.1. BEA, IBM, Microsoft, SAP, Siebel, May 2003.
12. A. Arkin et al., editors. Web Services Business Process Execution Language. Version 2.0.OASIS, Billerica, Massachusetts, Feb. 2005.

Exploiting Rating Behaviors for Effective Collaborative Filtering

Dingyi Han, Yong Yu, and Gui-Rong Xue

Shanghai Jiao Tong University, No. 800, Dongchuan Road, Shanghai, 200240, China
{handy, yyu, grxue}@apex.sjtu.edu.cn

Abstract. Collaborative Filtering (CF) is important in the e-business era as it can help business companies to predict customer preferences. However, Sparsity is still a major problem preventing it from achieving better effectiveness. Lots of ratings in the training matrix are unknown. Few current CF methods try to fill in those blanks before predicting the ratings of an active user. In this work, we have validated the effectiveness of matrix filling methods for the collaborative filtering. Moreover, we have tried three different matrix filling methods based on the whole training dataset and their clustered subsets with different weights to show the different effects. By comparison, we have analyzed the characteristics of those methods and have found that the mainstream method, Personality diagnosis (PD), can work better with most matrix filling method. Its MAE can reach 0.935 on a 2%-density EachMovie training dataset by item based matrix filling method, which is a 10.1% improvement. Similar improvements can be found both on EachMovie and MovieLens datasets. Our experiments also show that there is no need to do cluster-based matrix filling but the filled values should be assigned with a lower weight during the prediction process.

1 Introduction

Collaborative Filtering (CF) plays an important role in the e-business era as it can help business companies to predict customer preferences by analyzing a small set of training rating data.

However, the sparsity of the training rating data is still a major problem preventing it from achieving better effectiveness, on account of the cost consuming factors in time and effort of the data collection process. i.e. Lots of ratings in the training matrix are given as blank. Take the popular data set, EachMovie, as an example. It is an explicit voting example set using data from the EachMovie collaborative filtering site [1] deployed by Digital Equipment Research Center from 1995 through 1997 [2]. It includes over 2.1 million ratings (ranged in value from 0 to 5) of 1,623 movies by 61,265 users. i.e. Only about 2.1% ratings are collected during the three-year period. The rests are all blanks.

[1] http://www.eachmovie.com/

[2] For more information, visit http://research.compaq.com/SRC/eachmovie/

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 175–186, 2006.

Few current CF methods try to fill in those blanks before predicting the ratings of an active user. Most of them analyze the training data directly to find similar user/item or to establish a model. They do not exploit the known user ratings. Therefore, to find some effective matrix filling methods becomes meaningful to CF systems as they may improve the prediction accuracy. Meanwhile, matrix filling methods can also lower the training data collection cost. Since they will fill in the blank elements in the rating matrix, the training data may not be needed as much as before. i.e. the CF systems may work with less ratings to retain the same prediction accuracy.

In this work, we use a mainstream method, Personality diagnosis (PD), to see whether the matrix filling methods can work. We have compared several different matrix filling methods and have validated their effectiveness for CF. The experiments show that with proper matrix filling methods, PD results can still be improved. We have then tried the matrix filling methods based on clustered subsets, little improvements have appeared. Besides, we have also set different weights to the filled values to achieve higher effectiveness.

The rest of the paper is organized as follows: We first brief the related work in Section 2. Section 3 introduces some background knowledge, including our notation, problem definition and the PD Algorithm. It is in Section 4 that the matrix filling methods are described. The experimental results and analysis are given in Section 5. Section 6 is the conclusion part.

2 Related Work

In this section, we will review the CF algorithms and current sparsity problem resolutions in CF.

2.1 CF Algorithms

There are three classes of prediction algorithms: memory-based algorithms, model-based algorithms and the hybrid algorithms.

Memory-based algorithms maintain a table of all user ratings for all items in the computer memory. Based on the table, they perform some computation to make predictions. The notable ones include the Pearson-Correlation based approach [1], the vector similarity based approach [2] and the extended generalized vector-space model [3]. These methods usually have significant memory cost and poor scalability by the memory capacity limitation.

Model-based algorithms first learn a descriptive model of users rating behaviors and then they generate recommendations according to the model. Compared to memory-based algorithms, they are memory economical. Algorithms in this class include Bayesian network approach [2], clustering approach[4,5] and the aspect models[6]. However, they are often time-consuming to build and update the models, and can not cover as diverse a user range as the memory-based approaches do.

A hybrid memory-and-model-based approach named personality diagnosis (PD)[7] is proposed by Pennock et al. It retains some advantages of both memory-based and model-based algorithms, namely simplicity, extensibility, normative grounding and explanatory power. According to the experiments in [7], PD can outperform both Pearson-Correlation based approach and vector similarity based approach in accuracy.

2.2 Sparsity Problem Resolutions in CF

There are currently three kinds of methods to deal with the sparsity problem. The first one is to do dimension-reduction. The second one is to acquire additional information. The last but newest one is to fill the unprovided rating values.

Dimension-reduction methods aims at reducing the dimensionality of the user-item matrix directly. Principle Component Analysis (PCA) [8] and information retrieval techniques such as Latent Semantic Indexing (LSI) [9,10] are used. Zeng proposed to compute the user similarities by a matrix conversion method[11]. These approaches address the sparsity problem by removing unrepresentative or insignificant users of items to condense the user-item matrix. However, some potentially useful information might also be removed during the reduction process.

Content-boosted CF [12,13] appproaches require additional information regarding items as well as a metric to compute meaningful similarities among them. In [14], Popescul et al. also proposed a unified probabilistic model for integrating content information to solve the sparse-data problem. Most previous studies have demonstrated significant improvement in recommendation quality. However, such item information may be difficult or expansive to acquire in practice.

Xue et al. proposed a novel cluster-based smoothing method to solve the sparsity problem as well as the scalability problem [15]. They try to use smoothing methods to solve the sparsity problem and achieve a higher accuracy when using KNN algorithm to predict ratings. However, whether the method works on other CF algorithms, such as PD, are left unknown.

3 Background

3.1 Filling Method Definition

Let U be the user set, n the user number, m the item number. Denote the $n \times m$ matrix of all the ratings as $\mathbf{R}$. The rating of user i for item j is R_{ij}. $\mathbf{R}_i$ is user i's rating vector. The value of R_{ij} maybe an provided integer R_{ij}^{true} or a blank (marked as $\perp$).

The CF problem thus can be described as the follows. Given a rating matrix $\mathbf{R}$ and some rating $\mathbf{R}_i^{\text{true}}$ of a certain user i. CF is a function cf that can predict a blank rating $R_{ij} = \perp$ of user i. i.e. $R_{ij} = cf(\mathbf{R}, \mathbf{R}_i^{\text{true}})$. The closer to R_{ij}^{true} the predicted R_{ij} is, the better the CF algorithm will be.

A matrix filling method f is to assign new values to blank value R_{ij} $(= \perp)$ in the matrix $\mathbf{R}$. We use superscript '$^{\text{f}}$' to mark the value as a filled one.

$$R_{ij}^{\text{f}} = \begin{cases} R_{ij} & \text{if } R_{ij} \neq \perp, \\ f(\mathbf{R}^{\text{true}}, i, j) & \text{else.} \end{cases}$$

Thus, a matrix filling method is a function to replace the $\perp$'s in a sparse matrix $\mathbf{R}$ with new values $f(\mathbf{R}^{\text{true}}, i, j)$. Figure 1 shows the matrix filling process in detail.

$$\mathbf{R}=\begin{pmatrix} R_{11} & \perp & \dots & R_{1m} \\ \perp & R_{22} & \dots & \perp \\ \vdots & \vdots & \ddots & \vdots \\ R_{n1} & \perp & \dots & \perp \end{pmatrix} \xrightarrow{\text{fill}} \begin{pmatrix} R_{11} & f(\mathbf{R}^{\text{true}},1,2) & \dots & R_{1m} \\ f(\mathbf{R}^{\text{true}},2,1) & R_{22} & \dots & f(\mathbf{R}^{\text{true}},2,m) \\ \vdots & \vdots & \ddots & \vdots \\ R_{n1} & f(\mathbf{R}^{\text{true}},n,2) & \dots & f(\mathbf{R}^{\text{true}},n,m) \end{pmatrix} = \mathbf{R}^{\text{f}}$$

Fig. 1. The Filling Method in Collaborative Filtering

3.2 PD Algorithm

PD algorithm [7] assumes that users report ratings for movies with Gaussian noise. i.e. user i's reported rating for movie j is drawn from an independent normal distribution with mean R_{ij}^{true}.

$$\Pr(R_{ij} = x | R_{ij}^{\text{true}} = y) \propto e^{-(x-y)^2/2\sigma^2}$$

where σ is a free parameter. Thus, a same user may report different ratings on different occasions. If $y = \perp$, it assigns a uniform distribution over ratings, which is a simple matrix filling method adopted.

The algorithm further assumes the distribution of personality types or rating vectors in the matrix is representative of the distribution of personalities in the target population of users. Thus,

$$\Pr(\mathbf{R}_a^{\text{true}} = \mathbf{R}_i) = \frac{1}{n}$$

By applying Bayes rule, the probability that the active user is of the same personality type as any other can be computed.

$$\begin{aligned} &\Pr(\mathbf{R}_a^{\text{true}} = \mathbf{R}_i | R_{a1} = x_1, \cdots, R_{am} = x_m) \\ &\propto \Pr(R_{a1} = x_1 | R_{a1}^{\text{true}} = x_1) \\ &\quad \cdots \Pr(R_{am} = x_m | R_{am}^{\text{true}} = x_m) \\ &\quad \cdot \Pr(\mathbf{R}_a^{\text{true}} = \mathbf{R}_i) \end{aligned}$$

Once this quantity computed for each user i, the probability distribution for the active user's rating of an unvoted movie j is:

$$\begin{aligned} &\Pr(R_{aj} = x_j | R_{a1} = x_1, \cdots, R_{am} = x_m) \\ &= \sum_{i \in U} \Pr(R_{aj} = x_j | \mathbf{R}_a^{\text{true}} = \mathbf{R}_i) \\ &\quad \cdot \Pr(\mathbf{R}_a^{\text{true}} = \mathbf{R}_i | R_{a1} = x_1, \cdots, R_{am} = x_m) \end{aligned}$$

The highest $\Pr(R_{aj} = x_j)$ means the active user a may probably vote movie j with value x_j. Thus, x_j is predicted.

4 Filling Methods

There are many underlying scenarios of matrix filling methods. One user may probably vote the value similar to the most of others. Therefore, we may use the mean rating value of the item to replace the blanks. Some users may vote high rates to most movies and others may vote low rates according to their tastes. Hence, we may use the mean rating value of the user to replace those blanks. Besides, we may also cluster users/items into groups to get better matrix filling results.

Accordingly, matrix filling methods can be divided into groups from different perspective. From the dimensional perspective, they are divided into three groups: item-based, user-based and user-item-based matrix filling. From the granulated perspective, they are divided into matrix-based and cluster-based matrix filling. Thus, we get six methods listed in Table 1.

Table 1. The Matrix Filling Methods

Perspective	Item-Based	User-Based	User-Item-Based
Matrix-Based	IM	UM	UIM
Cluster-Based	IC	UC	UIC

4.1 Dimensional Perspective

Item-Based Matrix Filling. Item-based matrix filling method is a function like $f(\{R_{ij_0}\}, i_0, j_0)$. It replaces the blanks with other users' rating values to the same item. We may use the mean value of the other users' rating.

$$f_{\mathrm{IM}}(\mathbf{R}^{\mathrm{true}}, i_0, j_0) = f(\{R_{ij}|j = j_0\}, i_0, j_0) = \underset{i, R_{ij_0} \neq \perp}{average}(R_{ij_0}) \triangleq \overline{R_{ij_0}}$$

User-Based Matrix Filling. User-based matrix filling method is a function like
$f(\{R_{i_0 j}\}, i_0, j_0)$. It replaces the blanks with the same user's rating value to the other items. We may use the mean value of the user's ratings to other items.

$$f_{\mathrm{UM}}(\mathbf{R}^{\mathrm{true}}, i_0, j_0) = f(\{R_{ij}|i = i_0\}, i_0, j_0) = \underset{j, R_{i_0 j} \neq \perp}{average}(R_{i_0 j}) \triangleq \overline{R_{i_0 j}}$$

User-Item-Based Matrix Filling. User-item-based matrix filling method takes both of the above two scenarios into consideration. We may first calculate each user's average rating values to the movies, and then do item-based matrix filling on the deviation rating matrix. Denote $\Delta R_{i_0 j_0} = R_{i_0 j_0} - \overline{R_{i_0 j}}$ as the deviation rating. The matrix filling function can be described as:

$$f_{\mathrm{UIM}}(\mathbf{R}^{\mathrm{true}}, i_0, j_0) = \overline{R_{i_0 j}} + \underset{i, R_{ij_0} \neq \perp}{average}(\Delta R_{ij_0}) \triangleq \overline{R_{i_0 j}} + \overline{\Delta R_{ij_0}}$$

The whole matrix filling process can be described as Figure 2.

$$\left(\begin{array}{cccc} \Delta R_{11} & \perp & \dots & \Delta R_{1m} \\ \perp & \Delta R_{22} & \dots & \perp \\ \vdots & \vdots & \ddots & \vdots \\ \Delta R_{n1} & \perp & \dots & \perp \end{array}\right) \left| \left(\begin{array}{c} \overline{R_{1j}} \\ \overline{R_{2j}} \\ \vdots \\ \overline{R_{nj}} \end{array}\right)\right.$$

Step 1 ↗ ↘ Step 2

$$\left(\begin{array}{cccc} R_{11} & \perp & \dots & R_{1m} \\ \perp & R_{22} & \dots & \perp \\ \vdots & \vdots & \ddots & \vdots \\ R_{n1} & \perp & \dots & \perp \end{array}\right) \underset{\longrightarrow}{\text{filling}} \left(\begin{array}{cccc} R_{11} & \overline{R_{1j}} + \overline{\Delta R_{i2}} & \dots & R_{1m} \\ \overline{R_{2j}} + \overline{\Delta R_{i1}} & R_{22} & \dots & \overline{R_{2j}} + \overline{\Delta R_{im}} \\ \vdots & \vdots & \ddots & \vdots \\ R_{n1} & \overline{R_{nj}} + \overline{\Delta R_{i2}} & \dots & \overline{R_{nj}} + \overline{\Delta R_{im}} \end{array}\right)$$

Fig. 2. User-Item-Based Matrix Filling Method

4.2 Granulated Perspective

Matrix-Based Matrix Filling. Matrix-based matrix filling methods are large granulated methods. They reference to all the rating values in the matrix just as f_{IM}, f_{UM} and f_{UIM} discussed before.

Cluster-Based Matrix Filling. Cluster-based matrix filling methods hypothesize that user preferences can be implicitly clustered into groups and matrix filling methods can get better effectiveness with similar users. So these methods first cluster users or items into groups by some clustering algorithm. Then, they perform matrix filling methods in each cluster.

IC method. IC method first clusters users into groups. Thus the whole rating matrix is divided into several small matrixes vertically. For each small matrix, it does filling like what IM does to the whole matrix.

$$f_{\text{IC}}(\mathbf{R}^{\text{true}}, i_0, j_0) = \underset{i \in \mathcal{C}(i_0), R_{ij_0} \neq \perp}{average} (R_{ij_0})$$

Where $\mathcal{C}(i_0)$ is the user cluster that i_0 belongs to.

UC method. US method first clusters items into groups. Thus the whole rating matrix is divided into several small matrixes horizontally. For each small matrix, it does filling like what US does to the whole matrix.

$$f_{\text{UC}}(\mathbf{R}^{\text{true}}, i_0, j_0) = \underset{j \in \mathcal{C}(j_0), R_{i_0 j} \neq \perp}{average} (R_{i_0 j})$$

Where $\mathcal{C}(j_0)$ is the item cluster that j_0 belongs to.

UIC method. UIC method first clusters users into groups. For each user cluster, it does filling like what UIM does to the original matrix.

$$f_{\text{UIC}}(\mathbf{R}^{\text{true}}, i_0, j_0) = \underset{R_{i_0 j} \neq \perp}{average}(R_{i_0 j}) + \underset{i \in \mathcal{C}(i_0), R_{ij_0} \neq \perp}{average} (\Delta R_{ij_0})$$

4.3 Weighted Matrix Filling for PD

Since the filled values are not real user ratings, give them different weights to real user ratings may also improve the prediction accuracy. With this consideration, PD algorithm can be modified slightly by adding a weight factor $\lambda \in [0, 1]$ to the report rating probablity equation.

$$Pr(R_{ij} = x | R_{ij}^{filled} = y) = \lambda Pr(R_{ij} = x | R_{ij}^{true} = y)$$

5 Experimental Results

5.1 Metrics and Datasets

Mean Absolute Error(MAE) [16] is used here to measure the prediction quality.

$$\text{MAE} = \frac{\sum_{u \in T} |R_{ui} - \tilde{R}_{ui}|}{|T|}$$

where R_{ui} is the rating given to item i by user u, $\tilde{R}_{ui}$ is the predicted value of user u on item i, T is the test set, and $|T|$ is the size of the test set.

To test the matrix filling methods with different data distributions, we have taken EachMovie and MovieLens [3] as two sources of our test-beds. For each data source, we have extracted two subsets with no intersection and a limitation of more than 20 rates per user. One is for training, containing 4,000 users. The other is for testing, containing 1,500 users. Some characteristics of the data sets are listed in Table 2.

Table 2. Training Dataset Characteristics

Dataset Name	Data Souce	Rate Number (All)	Movie Number	Matrix Density	Rates Level
EM-Train	EachMovie	293,081	1,628	4.50%	0,1,2,3,4,5
ML-Train	MovieLens	670,346	3,900	4.30%	1,2,3,4,5
EM-Test	EachMovie	107,379	1,628	4.40%	0,1,2,3,4,5
ML-Test	MovieLens	253,918	3,900	4.34%	1,2,3,4,5

By predicting each rate as the most appearing rating value, 3, we can also get an MAE lower bound of each test set. They are 1.20 on EM-Test and 1.07 on ML-Train. If we get an MAE larger than the lower bound, we may regard the prediction as failed for some reasons, for example, not enough training data.

[3] 1 Million MovieLens Dataset, http://www.cs.umn.edu/research/GroupLens/

5.2 Does Matrix Filling Work for PD?

To confirm matrix filling methods can help PD algorithm to get better performance, we have performed PD on some different density training matrixes. Both randomly deleting rating data from training data and randomly choose a certain number of rates per user have been tried to make different density. MAE values of given 20 (i.e. we are given 20 know ratings to predict the 21st item of each test user.) are shown in Figure 3 and Figure 4.

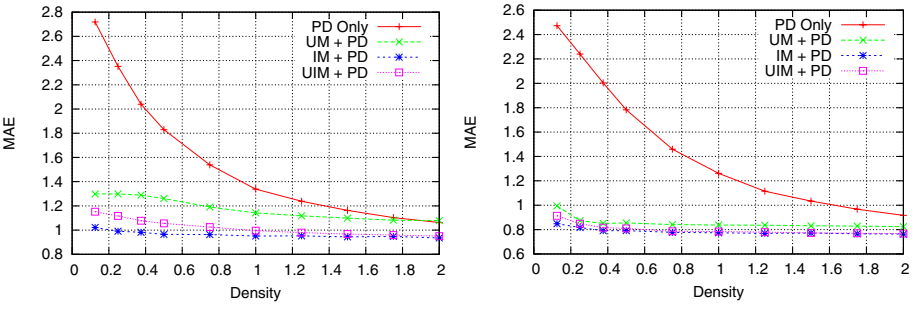


Fig. 3. PD MAEs on Different Sparsity Data Sets With Different Smoothing Methods (Randomly Deleting on Training Data, Given 20)

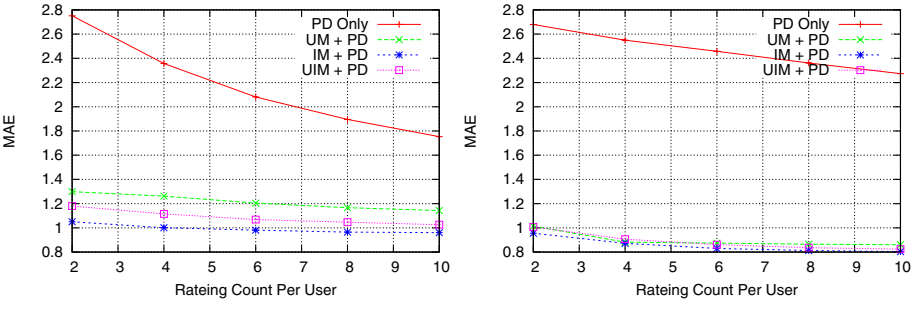


Fig. 4. PD MAEs on Different Sparsity Data Sets With Different Smoothing methods (Randomly Choose a Certain Number of Ratings Per User, Given 20)

PD algorithm works better when given more training data, either with training matrix filled or not. Its prediction succeeds when the density of training matrix is larger than 1.5% without training matrix filled.

User-based matrix filling method (UM) can improve the recommendation on datasets whose density is lower than 1.5% on both datasets. When the densities are larger than 2.0% on EM-Train , PD can achieve higher accuracy without

such filling. The MAE of PD only is 1.063 while the MAE of UM+PD is 1.078. The same happens when on the whole training set of 4.30%-density ML-Train. The MAE of PD only is 0.764 while the MAE of UM+PD is 0.796. This reflects the fact that recommendation systems encourage users to show their different opinions to different items. If every user rate each movie with an average rating, the system is meaningless.

Item-based matrix filling method (IM) can greatly improve the accuracy. On the 2%-density dataset, its MAE get a 10.6% improvement on EM-Train (0.935 vs. 1.063) and a 10.1% improvement on ML-Train (0.763 vs. 0.917). Even when given only 0.5% ratings, it can still make PD attain better recommendation accuracy than PD can get with 2% ratings without matrix filling on both datasets, which confirms that most rating distributions of items are similar to normal distribution. i.e. most users vote one movie with similar ratings. This result also indicates that with proper matrix filling methods, we may not need as many training data to predict user rating as we need without them.

MAEs get by PD algorithm after UIM matrix filling are better than those get after UM matrix filling, but worse than those get after IM matrix filling.

5.3 Does Clustering Help Matrix Filling?

On the 2%-density datasets, we have used the Repeated Bisections method of cluto[4] [17] to cluster the ratings into different number of groups and have done matrix filling with UC, IC and UIC. The MAEs are shown in Figure 5.

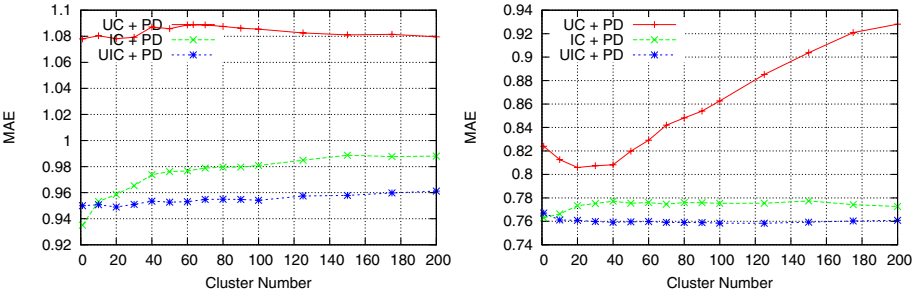


Fig. 5. PD MAEs With Different Cluster-Number-Based Matrix Filling Methods (Training Matrix Density = 2%, Given 20)

To UC on ML-Train dataset, cluster numbers around 30 can help to improve the prediction accuracy, but larger cluster numbers just worsen the accuracy. On EM-Train dataset, despite a small undulation around 40-100 clusters, no clear improvement or deterioration has been found. IC MAEs get larger when the cluster number grows to 40 and they almost keep unchanged with larger cluster

[4] Cluto is a high-performance clustering tool. For more information, visit http://glaros.dtc.umn.edu/gkhome/cluto/cluto/

numbers on both datasets. UIC MAEs just keep little changes with all cluster numbers on the two datasets.

Compared with the improvement that UM, IM and UIM methods have contributed to PD, the effect of clustering is small. Considering the additional efforts brought by clustering and some deterioration, our conclusion is that clustering may sometimes bring a little improvement to PD, but it is not a necessary step.

5.4 Should Filled Values Assigned with Different Weights?

Still on the 2%-density datasets, we have conducted a set of experiments to test whether assign different weights to filled values could improve the prediction accuracy. We have used The method introduced in Section 4.3. The MAEs are shown in Figure 6.

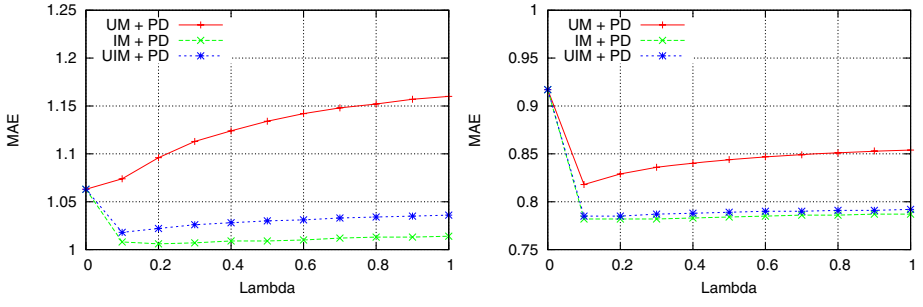


Fig. 6. PD MAEs on Different Sparsity Data Sets With Different Smoothing Methods (Randomly Deleting on Training Data, Given 20)

It is clear that when assigned a small weight around 0.1 to 0.2, All three matrix filling methods can achieve a higher prediction accuracy on both datasets.

6 Conclusion

In this paper, we have compared three matrix filling methods to deal with the sparsity problem in recommendation systems among which item-based matrix filling is the best. PD MAE can reach 0.935 on a 2%-density EachMovie training dataset by IM method, which is a 10.1% improvement. This also indicates that with proper matrix filling methods, we may not need as many training data to predict user rating as we need without them. Our experiments on clustering-based matrix filled shows little improvement. Considering the extra effort brought by clustering and the occasional deterioration, there is no need to do matrix filling based on clustering results. By assigning different weights to filled values, we also get some improvement on MAEs. All of

these findings have been validated on both EachMovie dataset and MovieLens dataset.

Our future work may include running CF algorithms other than PD to test the matrix filling effectiveness and testing other matrix filling methods.

References

1. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Grouplens: applying collaborative filtering to usenet news. Communications of the ACM **40** (1997) 77–87
2. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Conference on Uncertainty in Artifical Intelligence (UAI-1998). (1998) 43–52
3. Soboroff, I., Nicholas, C.: Collaborative filtering and the generalized vector space model (poster session). In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2000) 351–353
4. Kohrs, A., Merialdo, B.: Clustering for clooaborative filtering applications, IOS Press (1999)
5. L.H. Ungar, D.P.F.: Clustering methods for collaborative filtering. In: Proceedings of the Workshop on Recommendation Systems, AAAI Press (1998)
6. Hofmann, T.: Latent semantic models for collaborative filtering. ACM Transactions on Information System **22** (2004) 89–115
7. Penmnock, D.M., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filtering by personality diagnosis: A hybrid memory-and-model-based approach. In: Proceedings of the Sixteenth Conference on Uncertainty in Artifical Intelligence (UAI-2000). (2000) 473–480
8. Goldberg, K.Y., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval **4** (2001) 133–151
9. Fisher, D., Hildrum, K., Hong, J., Newman, M., Thomas, M., Vuduc, R.: Swami: a framework for collaborative filtering algorithm development and evaluation. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2000) 366–368
10. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.T.: Application of dimensionality reduction in recommender system – a case study. In: ACM WebKDD 2000 Web Mining for E-Commerce Workshop. (2000)
11. Zeng, C., Xing, C.X., Zhou, L.Z.: Similarity measure and instance selection for collaborative filtering. In: WWW '03: Proceedings of the 12th international conference on World Wide Web, New York, NY, USA, ACM Press (2003) 652–658
12. Balabanovic, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Communication of the ACM **40** (1997) 66–72
13. Claypool, M., Gokhale, A., Mirands, T., Murnikov, P., Netes, D., Sartin, M.: Combining content-based and collaborative filters in an online newspaper. In: ACM SIGIR Workshop on Recommender Systems - Implementation and Evaluation. (1999)
14. Popescul, A., Ungar, L.H., Pennock, D.M., Lawrence, S.: Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: Proceedings of the Seventeenth Conference on Uncertainty in Artifical Intelligence (UAI-2001). (2001) 437–444

15. Xue, G.R., Lin, C., Yang, Q., Xi, W., Zeng, H.J., Yu, Y., Chen, Z.: Scalable collaborative filtering using cluster-based smoothing. In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2005) 114–121
16. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Transactions on Information System **22** (2004) 5–53
17. Zhao, Y., Karypis, G., Fayyad, U.: Hierarchical clustering algorithms for document datasets. Data Mining and Knowledge Discovery **10** (2005) 141 – 168

Exploiting Link Analysis with a Three-Layer Web Structure Model

Qiang Wang, Yan Liu, and JunYong Luo

Information Engineering Institute, Information Engineering University, Zhengzhou, 450002, P.R. China
wongqqq@gmail.com

Abstract. Link analysis is one of the most effective methods of web structure mining. Traditional link analysis methods only consider the flat structure of the web with hyperlinks. It may affect the precision of the analysis result inevitably. It is observed that the web could be treated as an entity with a three-layer structure: host layer, page layer and block layer. Considering this three-layer structure is expected to improve the precision significantly when performing link analysis. A novel algorithm, three-layer based ranking is proposed to complete this task. In this algorithm, the important hosts and blocks are found within adaptations of the traditional link analysis methods. Based on these hosts and blocks, the web pages both belonged to important hosts and containing important blocks could be retrieved. These web pages are just the authoritative web pages that link analysis is looking for. We experimentally evaluate the precision of our three-layer based ranking algorithm. It is concluded from extensive experiments that this method outperforms other traditional algorithms significantly.

Keywords: Information Retrieval, Web structure mining and Link analysis.

1 Introduction

Recent development of WWW has been growing dramatically, enabling us to obtain more information of various fields than before. In the meanwhile, it makes us more difficult to get what we really want from the web. Therefore, it becomes quite necessary to extract web structure. Link analysis is one of the most effective methods of performing web structure mining. Google's Pagerank [1] is a successful applied algorithm, which uses the random surf model to wander the web link graph constructed by the link relationship between pages. Another successful link analysis algorithm—HITS [2] proposes the idea of authorities and hubs in the web link graph, of which authorities are web pages that have good content to the topic and hubs are web pages point to authorities. Salsa [3] combines the idea of PageRank and HITS and performs random walk in a bipartite graph. (([4] has a good conclusion of them).

However, most existing link analysis algorithms regard the hyperlinks as the only standard of measuring web pages' importance. In fact, semantic, content

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 187–198, 2006.

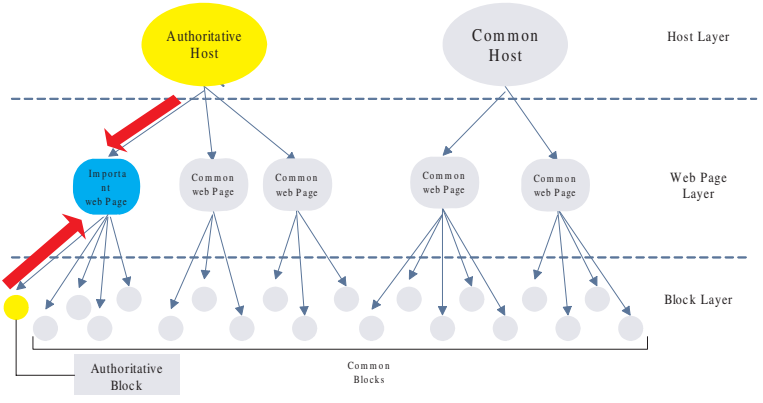


Fig. 1. The Basic concept of three-layer based ranking algorithm: The yellow host is an authoritative host, and the yellow block is a block with high authoritative value. As mentioned in our basic concept, the web page between them would probably be an authoritative web page.

and structural factors may also affect the determination of whether a web page is authoritative. We wonder that the precision of link analysis would be improved significantly if we can take these factors into consideration. It is also observed that WWW is organized with a hierarchical structure from macro viewpoint, while from micro viewpoint the web page could be segmented into several semantic blocks. Based on this consideration, WWW could be described as Figure 1. The top layer of the web structure consists of many hosts. Under these hosts, there are web pages belonged to them. Each web page is segmented into blocks according to semantic, content and other factors. Also we find that an authoritative web page usually belongs to an authoritative host, meanwhile, this web page might carry at least an authoritative block. A novel ranking algorithm called three-layer based ranking is proposed to merge these three layers so as to improve the precision of link analysis in this paper. The basic concept behind this algorithm is: an authoritative page should belong to an authoritative host; meanwhile, this authoritative page should also include one or several authoritative blocks. Thus, we conduct our research according to these steps: Firstly, the relationships between adjacent layers (e.g. host-page relationship) could be obtained easily. After that, we extract the relationship between the host layer and the block layer. Then, several traditional algorithms (three-layer based PageRank, HITS and Salsa) are adapted to discover the authoritative hosts and blocks. Finally, the web pages with high rank could be obtained based on the basic concept of three-layer based ranking. Several experiments are performed to test the effectiveness of this concept. The results indicate that our three-layer based ranking algorithm could enhance the link analysis precision significantly.

The remainder of this paper is organized as follows. We will have an overview of previous link analysis approaches in section 2. In section 3, the three-layer based ranking model would be put out. Based on the ranking model, we will

describe the ranking algorithm in detail in section 4. Then experiment results will be presented in section 5. Finally, in section 6 we will have a conclusion of our work and discuss future works.

2 Background Knowledge

So far the methods researching link analysis from structure view could be classified into two classes: referring the structure of WWW and considering web page's inner content. In this section, both of the methods will be explained separately.

Link analysis methods referring WWW structure treats a web site as a hierarchical organization. The directory structure of the website corresponds to this hierarchical structure. Such a hierarchical structure is a tree structure; each node in the tree corresponds to a web page. With global view, WWW is organized by many websites with this kind of hierarchical structure. Therefore, WWW could be regarded as a forest containing large amount of trees (websites). Based on this concept, a lot of research has been performed in recent years. G.R. Xue et al proposed a novel algorithm Hierarchical Rank [5]. In their algorithm, they divide the WWW into two layers: host layer and web page layer, then they use a DHC (Dissipative Heat Conductance) model to calculate the web page's weight iteratively from host node. [6] extracts a website's topic hierarchy from its link structure considering link structure, content and directory structure. However, they only research these factors from the hierarchical structure view, not referring to the content of the web page, which might carry more information.

Link analysis considering web page's inner content believes that web pages could be segmented into blocks according to semantic, content, size, location and other factors. The organization of web page together with these blocks is also a tree structure, where each node in the tree corresponds to a block. Considering it when performing link analysis could help us discovering the intrinsic blocks and improving the performance. [8] has done a good job to learn the block's importance through analyzing the spatial features (such as position and size) and content features (such as the number of images and links). In [9], Deng Cai et al show the block level link analysis, which segment the web page into semantic blocks using VIPS (VIsion-based Page Segmentation)[10]. After that, they propose block level Pagerank and HITS algorithm, which pays more attention to the semantic factor. [11] presents an automatic method to measure the topic coherence between web page and its blocks. This method is quite helpful to our work. There are also other block level link analysis methods [7], [12].

3 Three-Layer Based Web Model

A sketch of three-layer based web model is shown in Figure 2. Let us use $G(H, P, B, E_{H-P}, E_{P-B}, E_{H-B})$ to name this graph, in which:

$H = \{h_1, h_2, \cdots, h_u\}$ is the collection of host nodes, yellow nodes in the top layer are just these host nodes in Figure 2;

$P = \{p_1, p_2, \cdots, p_v\}$ is the collection of web pages. Each page contains several blocks. Purple nodes in the middle layer of the graph denote the web page nodes;

$B = \{b_1, b_2, \cdots, b_w\}$ is the collection of blocks. Each block only belongs to one web page. Green nodes in the bottom correspond to these blocks;

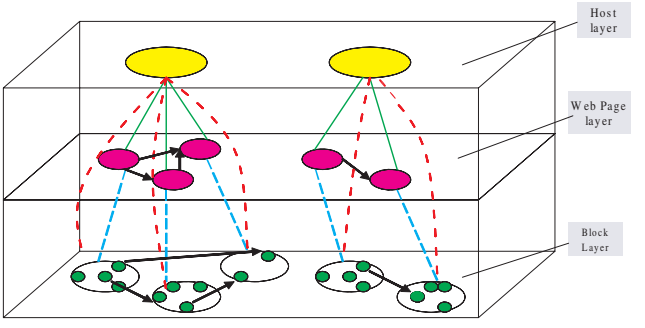


Fig. 2. The three-layer based ranking model

E_{H-P} is the collection of edges between host nodes and web page nodes. These edges denote the relationships between host layer and web page layer. A $u * v$ matrix W_{hp} is used to measure this relationship, where u is the host number, v is the page number, and an arbitrary element a_{ij} in matrix W_{hp} denotes whether page j belongs to host i and the coherence between them. In Figure 2, green lines correspond to this host-page relationship;

E_{P-B} is the collection of edges between webpage nodes and blocks. These edges denote the relationships between the web pages and blocks. A $v * w$ matrix W_{pb} measures the relationships between page nodes and block nodes. v denotes the page number, w corresponds to the block number. An arbitrary element b_{ij} in W_{pb} tells us whether block j belongs to web page i and the coherence between them. Blue line is used to denote the page-block relationship in Figure 2;

E_{H-B} is the collection of the pseudo edges between hosts and blocks. These edges denote the pseudo relationship between hosts and blocks. Actually this kind of relationship does not exist. We propose it to help us find the important hosts and blocks later. In Figure 2, red lines correspond to these pseudo relationships. A $u * w$ matrix W_{hb} is used to measure this indirect relationship, where u denotes host number, w corresponds block number and an arbitrary element a_{ij} in W_{hb} measures the pseudo relationship.

As mentioned before, what Matrix W_{hb} represents is the pseudo relationship between hosts and blocks. This kind of relationship is different from the other two relationships. Host-page relationship could be measured through hyperlinks, and page-block relationship could be extracted through content, location or other methods. However, there is no direct connection between host layer and block layer. Thus, host-block relationship couldn't be measured using the similar methods. Fortunately, from linear algebra view, we could get matrix W_{hb} through multiplying W_{hp} and W_{pb} using this formula:

$$W_{hb} = W_{hp} * W_{pb} \tag{1}$$

Particularly, we could understand Formula 1 in this way: Suppose hb_{ij} is an arbitrary element in W_{hb}, hp_{ij} and pb_{ij} correspond to the host-page and page-block relationship. Formula 1 could be understood as $hb_{ij} = \sum_{k=1}^{s} hp_{ik} * pb_{kj}$ (s is the total number of web pages), which means that the weight of any host-block relationship equals to the multiplication of all its host-page relationship and all the page-block relationship.

4 Three-Layer Based Ranking Algorithm

As mentioned in section 1, the basic concept behind the three-layer based algorithm is: an authoritative page should belong to an authoritative host; meanwhile, this authoritative page should also include one or several authoritative blocks. Based on this concept, section 4.1 would adapt three traditional link analysis algorithms (PageRank, HITS and Salsa) to discover the authoritative hosts and blocks. After that, we will merge host and block's factor together and find out the authoritative web pages in section 4.2.

4.1 Finding the Authoritative Hosts and Blocks

Three-Layer Based PageRank. The original idea of PageRank algorithm is that a page is important if it is pointed to by other important pages. That is, the importance of a page (its PageRank score) is determined by summing the PageRank scores of all pages that point to it.

The three-layer based PageRank is quite simple. The key difference is that it performs PageRank algorithm separately in host layer and block layer, after that, we could obtain the important hosts and blocks. With these important hosts and blocks, the authoritative web pages between them could be extracted.

In the host layer, vertex in the host layer graph is the host in the web, and citation relationship between hosts forms the edges between these vertexes. In the similar way, blocks in web pages become the nodes in block layer graph, and citation relationship constitutes the edges. With the important hosts and blocks found separately in each layer, the web pages between them could be obtained. These web pages are just the authoritative web pages we are looking for. This process is illustrated in Figure 3.

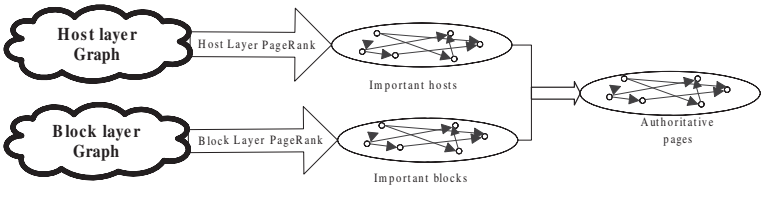


Fig. 3. The process of Three-layer based PageRank

Three-Layer Based HITS. Different from PageRank Algorithm, HITS propose the idea of authority and hub. An authority is a node with valuable information on a topic and a hub is a node with a list of citations to nodes having valuable information on the topic. Compared with original HITS algorithm, hosts play the role of hubs, and blocks are authorities in our three-layer based HITS.

The main process is as below: The mutually reinforcing idea of this merging method is similar to that of the original HITS algorithm as described in Formula 2. A_b corresponds to the vector of authority values for blocks, H_h denotes the vector of hub values for hosts. W_{hb} is the pseudo relationship matrix mentioned in section 3.1.

$$A_b = W_{hb}^T H_h \quad H_h = W_{hb} A_b \tag{2}$$

According to Kleinberg's verification, the two weighting vectors would be obtained when the iteration comes to convergence. In the process of recurrence, authority hosts and blocks would have a mutual enhancement to each other. Finally, the authority weight of hosts and blocks would be obtained after certain recurrence. The first several nodes would be the authority hosts and blocks.

Three-layer Based Salsa. Salsa algorithm combines the idea of PageRank and HITS. It performs a random walk on the bipartite hubs and authorities graph, alternating between the hub and authority sides. The main idea of three-layer based Salsa is: hosts and blocks could constitute a bipartite graph. A random walk is performed on this bipartite host and block graph, alternating between hosts and blocks. The random walk starts from a host, then proceeding to a block uniformly. After that, it will walk back to another host uniformly. Thus, the authority weights could be defined to be the stationary distribution of this random walk. In order to have a formal definition of host and host nodes' authority weight, we could have this formula:

$$b_i = \sum_{j:j \in B(i)} \frac{\omega_{ji}}{|F(j)|} h_j \quad h_i = \sum_{j:j \in F(i)} \frac{\omega_{ij}}{|B(j)|} b_j \tag{3}$$

In Formula 3, b_i denotes the ith block's authority weight, h_i denotes the ith host's authority weight. And $|F(i)|$ corresponds to the ith node's out-degree, $|B(i)|$ represents the ith node's in-degree, and ω_{ij} is the element measuring the pseudo relationship between ith host and jth block in matrix W_{hb}. When the recurrence goes to convergence, the authority weight vector of hosts and blocks would be obtained. The hosts and blocks with higher authority value in these two vectors would be the authoritative hosts and blocks.

4.2 Merging Hosts and Blocks to Find Out the Authoritative Web Pages

As mentioned before, after obtaining the authority hosts and blocks, we have to merge them so as to discover the authority web pages. However, the vector of hosts and blocks are two vectors with different length. Hence, they have to be normalized to have the same lengths equal to the number of web pages.

For the authority weight vector of hosts, we expand it to vector $\overrightarrow{host}$,whose length equals to the number of web pages. Consider the example described in Figure 4. 5 web pages are under 3 hosts. The vector of host equals to (0.7573, 0.4257, 0.6537). After expanding, we could obtain $\overrightarrow{host}$=(0.7573, 0.7573, 0.4257, 0.6537, 0.6537), whose length is 5, just the same as the web pages' number.

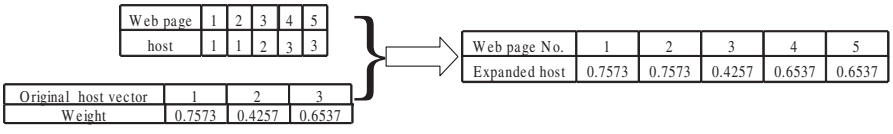


Fig. 4. The example host's vector expanding

For the vector of blocks, we have to shrink it to have the same length as the web page number. Suppose the shrunk vector is $\overrightarrow{block}$. Assuming that web page is constructed by many blocks, and each block has different weight value, we decide to merge all the blocks in a web page using Formula 4. In Formula 4,$weight$ is an arbitrary element of $\overrightarrow{block}$, bw_i corresponds to the block's weight in A_b. n denotes the blocks' number in that web page. Choosing n as the base logarithm could normalize $weight$ to the range (0, 1).

$$weight = \log_n \sum_{i=1}^{n} bw_i \tag{4}$$

Finally each web page's authority value could be obtained through this formula:

$$\overrightarrow{page} = (1-\alpha)\overrightarrow{host} + \alpha\overrightarrow{block} \tag{5}$$

In Formula 5, $\overrightarrow{page}$ is a vector denoting web pages' weight value. An arbitrary element of the vector corresponds to the score the web page has. A web page which is much closer to the query topic would have higher weight in $\overrightarrow{page}$. α is used to give different weight to $\overrightarrow{host}$ and $\overrightarrow{block}$.

5 Experiments and Evaluation

In this section, several experiments are performed based on the web pages downloaded from ODP[1]. The following problems are addressed by our experiments:

- How much should the weight parameter α be set to let the result better? After experiments, we find the best-performed α changes with different queries and different algorithms.
- What's the superiority of three layer based link analysis compared with other common link analysis methods, such as original PageRank, HITS and Salsa? It is observed that this method outperforms more than 12% on average.

[1] http://dmoz.org/

- Three-layer based PageRank, HITS and Salsa, which algorithm performs better in the practical query? It is observed that all these three method performs similarly. However, Salsa seems to work better than the other two methods.
- How does the amount of dataset affect our three-layer method? The experiment result shows that when more dataset is used, algorithms will perform better.

5.1 Experiment Methods

The experiment is mainly conducted according to these steps:

Step1: Web page retrieval. For example, suppose we choose "Turin Winter Olympic Games" as our researching topic. With the help of ODP, we download all the web pages on this topic. These web pages form the *Root Set* of our experiments. This set is then expanded to *Base Set* through including web pages that point to, or are pointed to, by the pages in the *Root Set*. For each page in *Root Set*, we include only the first 50 pages that point to this page, in the order returned by ODP. We then extract the links between the pages of *Base Set*, and construct the hyperlink graph.

Step2: Host Retrieval. Within the web pages found in ***Step 1***, we extract the hosts containing these web pages. For example, suppose we get a web page's URL `http://sports.yahoo.com/bascketball/default.html`, then its host is `sports.yahoo.com`. These hosts form *Host Set*. After obtaining *Host Set* and *Base Set*, we could calculate the value of W_{hp}.

Step3: Block Segmentation. With the help of VIPS, we divide all the web pages in *Base Set* into blocks. Based on the method referred before, the relationship between web pages and their blocks could be extracted. After that, matrix W_{pb} is obtained.

Step4: Perform three-layer based PageRank, HITS and Salsa separately and merging $\overrightarrow{host}$ and $\overrightarrow{block}$ vector with different α to see when it performs best.

Step5: Have three existing algorithms compared with our methods. PageRank, HITS and Salsa are three traditional algorithms. Based on the same *Base Set*, these three algorithms are conducted separately. Finally, we have their results compared with our methods'.

Step6: Analyze the impact of dataset amount. Dataset amount may influence on the query result. We perform our experiment with different amount of *Base Set*, and observe the performance of our method.

5.2 Experiment Dataset and Preparation

We setup our experiments on the following different queries:

Turin Winter Games, Bird flu, FIFA 2006, Computer Science, Wal-Mart, Stanford and Search Engine.

These queries range different areas, from society to sports. Our objectivity is to observe how the different algorithms represent these different (usually

unrelated) communities in the top positions of the ranking. We are also interested in observing the behavior of the algorithms when query a broad topic ("Computer Science") .

In order to have a comparison of our research result with the human judgment, a pre-experiment survey is designed. In this survey, we ask 10 volunteers (including 3 experts and 7 graduate students) to figure out the authoritative hosts, pages and blocks of the dataset based on the query words we have mentioned before. The survey result will be used in P@10 precision calculation.

5.3 Experiment Results and Evaluations

Determination of α. In order to determine, host factor and block factor, which is more important for our three-layer based ranking algorithm, and when the algorithm performs best, we decide to perform it with different α. 20 values from 0 to 1 by step 0.05 are chosen for α to test when the precision is highest. The precision at 10 (P@10) is the main evaluation metric here. We wonder different query might have different value of best performed α. Thus, we conduct all the 7 queries of our dataset with different value of α. Based on the results, we have a fitting curve for each query. Two $\alpha - precision$ relationships with their fitting curves are shown in Figure 5. The top three sub graphs are based on the query "Turin Winter Games" and the bottom sub graphs are based on "Bird flu". In light of Figure 5, we could infer that the best-performed α changes with different query words as well as different ranking methods. However, almost for all the queries, we observe that when α is set to a value in the range [0.57,0.65], the performance will be better than other conditions. It means that when merging host factor and block factor, more weight should be given to block factor.

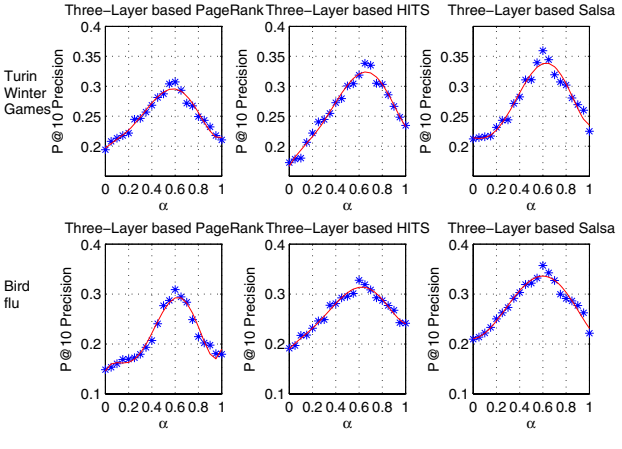


Fig. 5. The fitting curves of $\alpha - precision$ relationship with four different queries

Another observation of this experiment is that three-layer based Salsa performs better than the other two methods even with different query words. Moreover, it seems that the three-layer based HITS could obtain a little higher precision than three-layer based PageRank. These hypotheses as well as the comparison with traditional link analysis algorithms will be researched in the next sub section.

Comparison with Other Traditional Algorithms. In Figure 6, we have a comparison of the precision between three-layer based algorithms and original link analysis algorithms with different query words. The horizontal axis denotes different query words we have mentioned in 5.2, each color bar corresponds to one query word shown on the right. And the vertical axis represents P@10 precision.

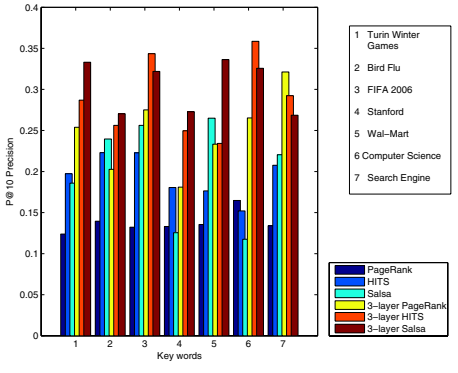


Fig. 6. The precision comparison among different algorithms with different query words

From Figure 6, it is observed that our three-layer based algorithm outperforms the three traditional link analysis (PageRank, HITS and Salsa) significantly. Particularly, the highest precision is 0.3586, whose query topic is "Computer Science", obtained through three-layer based HITS. The average P@10 precision of all the three-layer based algorithms is 0.2801, while that of the traditional algorithms is only 0.1778. Among the three-layer based algorithms, the average P@10 precision of PageRank, HITS and Salsa is 0.2475, 0.2888 and 0.3041 separately. It is also found that most of the algorithms perform not so ideal with the query word "Computer Science", which is mainly because that this query word is quite abroad and volunteers may have different understanding on it.

By now, we could have a summary of the comparison. Among the three three-layer based methods, three-layer based Salsa performs better than the other two. Three-layer based HITS and PageRank performs quite similarly. Also, the query result is influenced by different query words.

The Influence of Data Amount to Our Algorithm. The amount of dataset could also have impact on the performance of link analysis. Generally, large

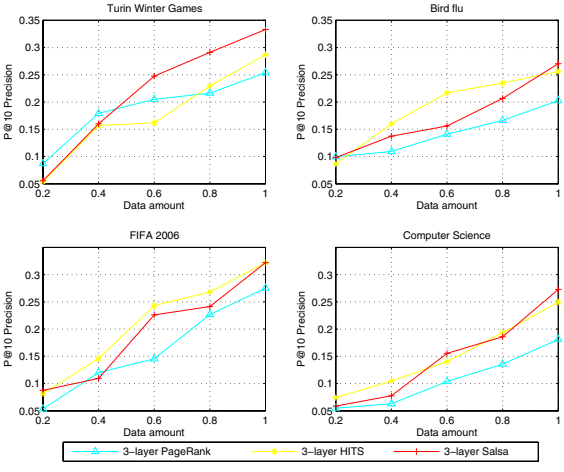


Fig. 7. The relationship between dataset amount and P@10 of Three-layer based algorithms

dataset might provide enough environments for link analysis; however, it could also drift analysis to the wrong direction because of redundant data. In order to test our method, we take different amount of dataset for experiment. The experiment results of four queries are shown in Figure 7.

In Figure 7, horizontal axis corresponds to the percentage of the dataset we used in our experiments. For example, 0.2 means we use only 20% of our base dataset for the experiments, and 1 means we use all the dataset. Vertical axis denotes the P@10 precision. As can be seen from the diagrams, all the algorithms perform better when the data amount gets larger. When all of the dataset is used, all the three algorithms perform best. It indicates that, for all the algorithms, enough dataset is necessary.

6 Conclusion and Future Works

In this paper, we propose a new three-layer based web model, which divide the web into host layer, web page layer and block layer. Based on this model, we improve link analysis through these steps: Firstly, authority hosts and blocks are discovered in each layer. In order to deal with it, we adapt the original PageRank, HITS and Salsa to fit our need. After that, these authority hosts and blocks are merged to discover the authority web pages between them. Experiments show that our three-layer based method outperforms the original link analysis algorithms (PageRank, HITS and Salsa) significantly. However, merging the authority hosts and blocks to discover the authority web pages only through giving different weights to them is not so scientific. We will continue to research some better methods.

References

1. L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford University, Stanford, CA, 1998.
2. J.M.Kleinberg, Authoritative Sources in a Hyperlinked Environment, In Proc. ACM-SIAM Symposium on Discrete Algorithms, 1998.
3. R. LEMPEL and S. MORAN, SALSA: The Stochastic Approach for Link-Structure Analysis, ACM Transactions on Information Systems, Vol. 19, No. 2, April 2001, Pages 131–160.
4. Borodin, J. S. Rosenthal, G. O. Roberts, and P. Tsaparas, Link Analysis Ranking: Algorithms, Theory, and Experiments. ACM Transactions on Internet Technology, Vol. 5, No. 1, February 2005, Pages 231–297.
5. G.R. Xue, et al. , Exploiting the Hierarchical Structure for Link Analysis. SIGIR'05, August 15–19, 2005, Salvador, Brazil.
6. Nan Liu, Christopher C. Yang, Extracting a Website's Content Structure From its Link Structure, CIKM'05, October 31-November 5, 2005, Bremen, Germany.
7. SD Kamvar, TH Haveliwala, CD Manning, and GH. Golub. Exploiting the block structure of the web for. computating pagerank. Technical Report Technical Report,. Stanford University, March 2003.
8. Ruihua Song, Haifeng Liu, Ji-Rong Wen, Wei-Ying Ma, Learning Block Importance Models for Web Pages, WWW 2004, May 17-22, 2004, New York, NY USA.
9. D. Cai, X.-F He, J.-R. Wen, and W.-Y. Ma, Block-level Link Analysis, SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK.
10. D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, VIPS: a visionbased page segmentation algorithm, Microsoft Technical Report, MSR-TR-2003-79, 2003.
11. Shen Huang et al, A study on combination of block importance and relevance to estimate page relevance, WWW 2005, May 10-14, 2005, Chiba, Japan.
12. Soumen Chakrabarti, Integrating the Document Object Model with Hyperlinks for Enhanced Topic Distillation and Information Extraction, In the Proceedings of the 10th World Wide Web Conference (WWW 10), 2001.

Concept Hierarchy Construction by Combining Spectral Clustering and Subsumption Estimation

Jing Chen and Qing Li

Department of Computer Science, City University of Hong Kong
83 Tat Chee Avenue, Kowloon, Hong Kong
{jerryjin, itqli}@cityu.edu.hk

Abstract. With the rapid development of the Web, how to add structural guidance (in the form of concept hierarchies) for Web document navigation becomes a hot research topic. In this paper, we present a method for the automatic acquisition of concept hierarchies. Given a set of concepts, each concept is regarded as a vertex in an undirected, weighted graph. The problem of concept hierarchy construction is then transformed into a modified graph partitioning problem and solved by spectral methods. As the undirected graph cannot accurately depict the hyponymy information regarding the concepts, subsumption estimation is introduced to guide the spectral clustering algorithm. Experiments on real data show very encouraging results.

1 Introduction

Deriving a concept hierarchy for a set of documents has always been a concern in the information retrieval area. With the rapid development of the Web, how to add structural guidance for Web document navigation stands out as a fundamental problem, and concept hierarchies are a promising solution. Many Web applications and information systems will be able to benefit from such a technology. The Yahoo Directory [1] and the Open Directory [2] are manually constructed based on human knowledge and perception of the world. Automating the hierarchy construction process can dramatically reduce the workload for creating new hierarchies or refining existing ones. Ontologies are the basis of the Semantic Web, and the taxonomic backbone of an ontology is an essential part of ontology engineering. Concept hierarchies are also crucial to the construction and navigation of e-textbooks compiled on the Web. Such a hierarchy can serve as the table of content for an automatically constructed book, and is the heart of the e-textbook generation process [22].

In this paper, our definition of concept hierarchies is similar to the explanation 'hierarchical topic structures of text segments' in [3], but the text segments are restricted to concepts instead of queries or short questions as in [3]. Organizing concepts in the form of short text segments into a hierarchical structure is a non-trivial task, given that the information contained in the concepts is very limited and can hardly be used to form a reasonable and meaningful hierarchy. There are several ways to automatically construct concept hierarchies. Some [5] are based on existing knowledge

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 199 – 209, 2006.

such as WordNet [4], where the taxonomic relation is determined by the hyponymy information stated in WordNet. While the evidence in WordNet is reliable, the information contained in such knowledge sources is much too general to be extended to domain-specific concepts that are most commonly used in information systems. Other methods include rule-based methods (e.g. [6][7][8]). Usually, a set of pre-defined patterns or rules are used to extract concept relations, hypernym/hyponym relations for example. The patterns or rules involve expert knowledge and can be time-consuming to define. Furthermore, they often result in a very low recall because patterns rarely appear in the corpus. Still some other statistics methods (e.g. [13]) try to calculate the probability of one concept subsuming another, and extract the subsumption pairs that exceed a threshold. However, this approach is error-prone since it totally relies on the computed subsumption, and not all hyponymy information can be found in this manner. For a given set of concepts, the above mentioned methods can hardly generate a complete and meaningful hierarchy due to their common low recall.

Another popular approach to generating concept hierarchies is clustering (e.g. [3][6][12]). However, calculating the semantic similarity between concepts for clustering can be difficult because the information contained in the short natural language segments is generally inadequate. In some previous work [3], a concept is extended to a feature vector from snippet results of Web search engines, then the similarity can be calculated between the vectors. The situation is more complicated when concepts in different levels are involved. Clustering can group closely related concepts into one cluster, but cannot identify which concept in a cluster is more generic. In other words, clustering cannot generate hierarchies of common taxonomic relations.

In this paper, we address the problem of concept hierarchy construction with clustering, while also combining the notion of hypernym/hyponym relations. The log-likelihood ratio measure is used as the similarity between concepts. And a spectral clustering algorithm makes graph partitioning decisions with the guidance of the subsumption estimation, thus bridging the gap between clustering algorithms that generate unnaturally organized hierarchies and low-recall approaches that cannot accurately cover all related concepts in their results.

In the next section, we review on some existing works related to our research. In section 3, we provide some background on subsumption, and propose a modified subsumption measure and the tree subsumption measure. Following that, in section 4, a spectral clustering algorithm combining subsumption estimation is presented. Section 5 provides some experimental results on both synthetic and real data. The conclusion is given in Section 6.

2 Related Works and Background

The best known source that has been widely used in hierarchy construction is WordNet [4]. The major benefit of WordNet to hierarchy construction is that WordNet provides hypernym relations summarized by human. Some studies (e.g., [5]) have been taking advantage of these human justified relations to aid hypernym-based concept hierarchy construction.

2.1 Rule-Based Methods

Pattern-based methods are a typical type of rule-based systems. Of all the pattern-based methods, six lexico-syntactic patterns defined by Hearst [8] have been repeatedly used by a variety of researchers ([5][6]). These patterns, shown in Figure 1, are capable of extracting hypernyms from natural language text. Although the patterns are humanly defined and have a relatively high precision, they suffer from a very low recall because the patterns rarely appear in a corpus.

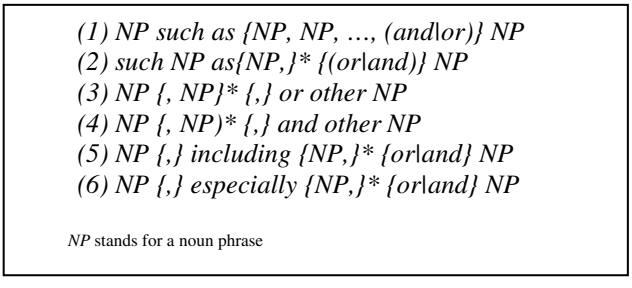


Fig. 1. Hearst patterns

In [5], Cimiano et al. targeted at learning taxonomic relations from diverse sources of evidence, and combining them with two simple strategies that experimentally outperform single sources. The taxonomic relations are extracted by natural language patterns such as Heart-patterns, WordNet hypernym relations and the 'vertical relations' heuristic [7]. Cimiano et al. also reported the precision and recall of each individual approach, with the recall never exceeding 10% for the selected corpus. Although Web search engine results were introduced to overcome the data sparseness problems and better recall was gained, the cost was a lower precision. The authors then combined the heterogeneous evidence with the mean and maximum of the value obtained from each source. The reported performance was improved but the recall was still below 30% with a less than 20% precision.

2.2 Clustering Algorithms

Clustering is an important branch of the information retrieval and data mining research. Recently, a lot of interest (e.g., [9][10][11]) has been on clustering the returned Web pages from a search engine and naming the clusters with the most related concepts. The generated concept hierarchy is data-driven and normally cannot be used as a representation of domain knowledge. Our work is not so much on giving a uniform solution for clustering a list of related Web pages which can be of any topic. Instead, we concentrate on providing a rational backbone hierarchy for any given domain. Such a hierarchy may not be associated with a document repository.

Gao et al. [12] make use of the reinforcement relationship among three different substances - category, document and term - to achieve the task of automatically generating a hierarchical taxonomy. Categories/documents and document/term construct two separate bipartite graphs. When the two bipartite graphs are merged together, the traditional spectral clustering methods fail. And the authors use the

consistency on the document partitioning to bridge categories and terms, and their algorithm, called consistent bipartite spectral graph co-partitioning, recursively partitions the category set until each resulted subset contains only one category. However, the hierarchy constructed in this manner cannot display the hypernym relation common to human knowledge.

Caraballo's work [6] is a combination of pattern-based hypernyms detection and clustering algorithms. A basic problem with unsupervised clustering algorithms is that the internal nodes of the hierarchy tree are usually unlabeled. Thus, Caraballo tackled this problem by naming internal nodes with hypernyms found in the same document set. First, nouns were extracted from conjunction and appositive data from the Wall Street Journal corpus. An unlabeled noun hierarchy was built using a bottom-up clustering algorithm. Then the third and forth of Hearst's patterns were adopted to identify hypernyms from the corpus. For each internal node of the hierarchy tree, a vector of hypernyms was constructed by accumulating the vectors of its children. The hypernym with the largest number in the vector was selected as the label of the node. However, if concepts from different levels participate in the clustering process, it will be unlikely to find a reasonable hypernym vector since some of the hypernyms are already included in the cluster.

The labeling problem faced by unsupervised clustering was also pointed out in Cimiano and Staab's work [5]. They overcame this problem by guiding the clustering process with a hypernym oracle. Their hypernym oracle recorded all sets of terms and their hypernyms. The terms and their hypernyms were collected from three distinct sources: WordNet, Hearst patterns matched in a corpus, and the Web. The hierarchical clustering algorithm consulted the oracle for each term pair, and made sure that only those with a common hypernym are clustered together. However, the information in the hypernym oracle they used is still limited, and cannot handle domain-specific concepts very well. In our approach, we make use of the more general method, subsumption estimation, which is discussed in detail in the following section.

3 Subsumption

Other than rule-based methods and clustering algorithms, Sanderson and Croft [13] focused on a type of co-occurrence called subsumption to identify concept relations. It first selects terms either by query expansion or an occurrence ratio. Then the selected terms are checked against each other. The concept hierarchy is naturally built by observing all the terms that satisfy the subsumption. This approach is simple yet effective, and it has shown some positive results provided by the users. However, in many cases, the concept pairs whose subsumption value exceeds the threshold are rare, and it is unlikely that a concept hierarchy can be built out of all concepts without lowering the threshold, and thus sacrificing precision.

For our purpose, we propose a modified subsumption value as below:

$$S(u,v)=\begin{cases} P(u\mid v)-P(v\mid u), \text{when } P(u\mid v)>P(v\mid u) \\ 0 \quad , \text{ otherwise} \end{cases}$$

where $P(u|v)$ is the conditional probability that concept u appears given concept v. The subsumption value in Sanderson and Croft's method depends on $P(u|v)$ only, and is more prior to error when both $P(u|v)$ and $P(v|u)$ are high. The higher our subsumption

value $S(u|v)$ is, the more probable it is that concept u subsumes v. Since we do not set a threshold to $S(u|v)$, we use another strategy as described immediately below.

3.1 Subsumption of a Tree

We can define a subsumption value for a concept hierarchy in the form of a tree. Given a tree $T(V)=<V,E>$, where $V=v_1,v_2,\ldots,|V|$ is the vertex set and $E=\{<v_i,v_j>,$ *if* v_i *is the parent node of* $v_j\}$ is the edge set, M is the adjacency matrix for tree T where $M_{ij}=1$ if $<v_i, v_j> \in E$ and otherwise $M_{ij}=0$. The subsumption estimation $S(V)$ is defined as follows:

$$S(V)=\frac{\sum_{i,j} M_{ij} \times S(v_i, v_j)}{\sum_{i,j} M_{ij}}$$

The subsumption value is the mean of all subsumption values within a tree. By searching for a tree structure $T(V)$ with the maximum $S(V)$, we can obtain a reasonable hierarchy with more confidence. The subsumption estimation of a tree is used as guidance for the later clustering algorithm to be presented in section 4.

3.2 Estimating the Quality of the Hierarchy

The quality of the resulting hierarchy strongly depends on the input concept set. It is only possible to construct a meaningful hierarchy if the concept set is coherent and all related to a certain domain. Otherwise, it would be even difficult for humans to determine the relations among concepts and build a corresponding hierarchy. In our approach, each hierarchy is a tree structure that characterizes the hypernym/hyponym relations in the input concept set. So it is important to estimate the quality of the hierarchy generated in the middle stages of our clustering algorithm as well as the input concept set to obtain a reasonable result. If there is no improvement, the clustering algorithm will not further partition the graph.

We use the subsumption value of a tree as the estimation measure. The initial concept set does not readily constitute a tree, so a root node must be selected first, and all other concepts are regarded as its children. Given a set of vertexes $V=v_1,v_2,\ldots,v_{|V|}$, $T_i(V)$ denotes the tree generated in which v_i is the root node, and all other nodes are its children. The root node selected for V, denoted as $r(V)$, is the node with the largest subsumption estimation given :

$$r(V)=\underset{v_i \in V}{\operatorname{argmax}} S(T_i(V)), T_i=(V, E_i=\{<v_i, v_j> \mid j \neq i, 1 \leq j \leq |V|\})$$

The quality of the vertex set V, represented by $S(V)$, is estimated by the subsumption value $S(T_{max}(V))$ corresponding to v_{max}.

4 The Concept Hierarchy Construction Algorithm

In our approach, given a set of concepts, we first represent the concepts and their relations as an undirected, weighted graph according to the statistical information in a document set. The document set can be collected from search engine results with the

concept set submitted as queries. Based on this graph, our approach combines the spectral clustering algorithm (or spectral graph partitioning) and subsumption estimation to generate the concept hierarchy.

4.1 Spectral Graph Partitioning

Spectral methods [23] recently emerge as effective ways for data clustering, image segmentation, and Web ranking analysis (e.g. [12][14]). Given a graph $G=(V,E)$, the weight of an edge, $E(i,j)$, is the similarity between the two vertexes, v_i and v_j. The adjacency matrix W of graph G can be defined as:

$$W_{ij} = \begin{cases} E_{ij}, if < v_i, v_j > \in E \\ 0, \quad \text{otherwise} \end{cases}$$

The goal of graph partitioning is to cut graph G into several disjointed vertex sets such that the intra-set density is high with the lowest cutting cost. A partition results in two disjointed vertex sets, A and B, where $A \cap B = \phi$, and $A \cup B = V$. To obtain $m(m \geq 2)$ sub-graphs, the partition is performed recursively on sub-graphs. The criterion measure for a good bi-partition is the *cut* value:

$$cut(A,B) = \sum_{v_i \in A, v_j \in B} W_{ij}$$

where $w(u,v)$ is the weight of any edge that was removed by the partition. Minimizing the *cut* value will result in an optimal partitioning of the graph. Shi et al. [14] proposed the normalized cut criterion measure that 'computes the cut cost as a fraction of the total edge connections to all the nodes in the graph' instead of 'looking at the value of total edge weight connecting the two partitions'. The authors further proved that by calculating the second smallest eigenvalue of the Laplacian matrix (the Fiedler value, [21]), the eigenvector corresponding to the Fiedler value (the Fiedler vector) can be used to minimize the normalized cut.

4.2 Choice of Semantic Similarity

When we represent the concepts and their relations as a graph, one of the key issues is the measure of relations between vertexes, viz. the weight of the edges. As the vertexes stand for concepts, the weight can be viewed as the correlation between concepts. We use an association measure commonly adopted in statistical linguistics to measure the correlation between concepts. Association measures estimate the compositionality of two or more words that may form a collocation in natural language, and can also be used to measure words or phrases that tend to co-occur in the same context. In our scenario, we define the co-occurrence as the number of times two concepts appear in the same document, and if the strength of co-occurrence is large then the two concepts are strongly correlated to each other.

Several association measures have been proposed in previous research, such as the log-likelihood ratio, the t-test, the chi-squared test and mutual information. We first tried the point-wise mutual information [24] as it is commonly used to measure correlation in previous efforts. However, mutual information is particularly weak in low-frequency cases. In our experiment, we have found that some of the columns of the

adjacency matrix contain relatively large mutual information values while the concept corresponding to the column appears only a few times in the corpus. Such a weakness is also reported in [16][17]. We have then tried the chi-squared test [25] and log-likelihood ratio [15], and found that the latter handles sparseness better than the other two. Due to this observation, the similarity measure between vertexes is computed by log-likelihood ratio in the rest of this paper.

4.3 Guidance from Subsumption Estimation

As previously discussed, spectral clustering methods can group closely related concepts into the same cluster. But it can not identify the hypernym/hyponym relations between the concepts. To consider the unsymmetric hierarchical relations in the clustering process, we impose subsumption estimation onto the spectral clustering algorithm.

The Fiedler vector of the Laplacian matrix of the graph is the real-valued solution to the normalized cut minimization problem. The process of spectral graph partitioning can be viewed as a mapping from $|V|$-dimensional space onto a one-dimensional space. Each concept corresponds to a value in the Fiedler vector, and concepts with close values are likely to belong to the same sub-cluster. However, the solution needs to be converted into a binary form, and a splitting point among the values must be chosen to indicate the partition. Some approaches cut the vector by taking the median value in the vector as the splitting point (the median cut) [26]. Others choose the splitting point that results in a partition with the best normalized cut value (the ratio cut) [14]. In our approach, we choose the splitting point corresponding to the largest subsumption value.

Let the original vertex set $OV=< w_1,w_2,\ldots,w_{|V|} >$. The Laplacian matrix is defined by $L=D-W$, where D is a diagonal matrix with $D_{ii}=\sum_j W_{ij}$. The value in the Fiedler vector corresponding to v_i is denoted by $f(v_i)$. Let $V=< v_1,v_2,\ldots,v_{|V|}>$ be a sorted vertex set of OV such that $r(V)= v_1$ and $f(v_i)\leq f(v_j)$, if $1<i<j$ and $v_i,v_j \in V$. The tree $T(V)$ constructed for V then has v_1 as the root node and all other nodes $v_i(i>1)$ the children of v_1. $S(V)$ is the maximum subsumption value corresponding to v_1. For a given $2\leq k\leq |V|$, $V_k=\{v_i|1<i\leq k\}$ and $V_k' =\{w_j|k<i\leq |V|\}$ stands for two disjoint sets such that $V_k \cup V_k'=V-v_1$, $V_k \cap V_k'=\varphi$.

A node in V_k is picked as the root node, and all other nodes are its child node. The tree generated in this means is a subtree of the T(V), with the root node of V_k being the child node of $r(V)$. The root node $r(V_k)$ corresponds to the maximum subsumption value $S(V_k)$, i.e., :

$$r(V_k) = \arg\max_{v_i \in V_k} S(T_i(V_k)), T_i(V_k) = (V_k, E_i = \{< v_i, v_j > | \, v_i \neq v_j \in V_k\}) \cdot$$

A root node $r(V_k')$ and its corresponding maximum subsumption value $S(V_k')$ can be calculated in the same manner for V_k'. Thus, V_k is the vertex subset of V with the k-1 smallest values in the Fiedler vector, and V_k' is the one with the largest $|V|$-k values. The subset with the larger subsumption estimation results in the better improvement in quality.

Initially, any node $v_j(j\geq 2)$ is a child node of $r(V)$. Without losing generality, we can assume that $S(V_k)$ is always larger than $S(V_k')$, so in the rest of our paper, we will only consider V_k since it is estimated with a better quality. Let V_{max} denote the vertex subset that corresponds to the largest subsumption value $S(V_{max})$ selected in all $r(V_k)$ for $2\leq k\leq |V|$. $r(V_{max})$ is the root node of the subtree $T(V_{max})$, as well as the child node of $r(V)$.

The depth of tree *T(V)* thus grows, and the initial graph is partitioned into two subsets V_{max} and $V_{max}' \cup r(V)$ with the largest subsumption gain. The process of the subsumption estimation guided spectral clustering is provided in the next subsection.

4.4 The SCSE Algorithm

Based on the previous discussions and deliberation, we now present our spectral clustering algorithm guided by subsumption estimation. The algorithm is of an acronym SCSE, whose partitioning process is as summarized in Figure 2.

Algorithm SCSE

1	Given a weighted graph G=(V,E), compute adjacency matrix W and diagonal matrix D.
2	Compute r(V) and S(V) for V.
3	Solve $(D-W)x=\lambda Dx$ for the two smallest eigenvalues/vectors.
4	Use the second smallest eigenvector to sort vertex set V, and calculate the $r(V_k)$ and $r(V_k')$ for all possible k values based on the sorted vertex set.
5	Choose the V_{max} resulting in the largest subsumption estimation $S(V_{max})$, and partition the graph into sub-clusters V_{max} and $V_{max}' \cup r(v)$.
6	Recursively partition the segmented parts if $S(V_{max})>S(V)$.

Fig. 2. Subsumption estimation guided spectral clustering algorithm

5 Experimental Results

In this section, we conduct our experiments on a human-labeled concept hierarchy. The output hierarchical structure is then compared with the original one to show the performance of our algorithm.

5.1 Data Preparations

We asked a computer science majored graduate student to create a concept hierarchy about 'data mining'. The generated hierarchy contained 18 concepts, which is shown in Figure 3. The concept set of the hierarchy was used as input to our algorithm, and the auto-constructed hierarchy is compared to the human-labeled one in our experiment.

5.1.1 Web Data Collection

In order to calculate the adjacency matrix, we need a data set to measure the correlation between concepts. We resort to Web search engines such as Google [18] for this task. Each concept in the handcrafted hierarchy was used as a query and submitted to the search engine. The top 50 results of each query were crawled and constituted the basic data collection. Web pages in the data collection were pre-processed to obtain a more accurate estimation of the correlation. Hub pages were first filtered out, and noisy Web pages cleansed according the approach in [19].

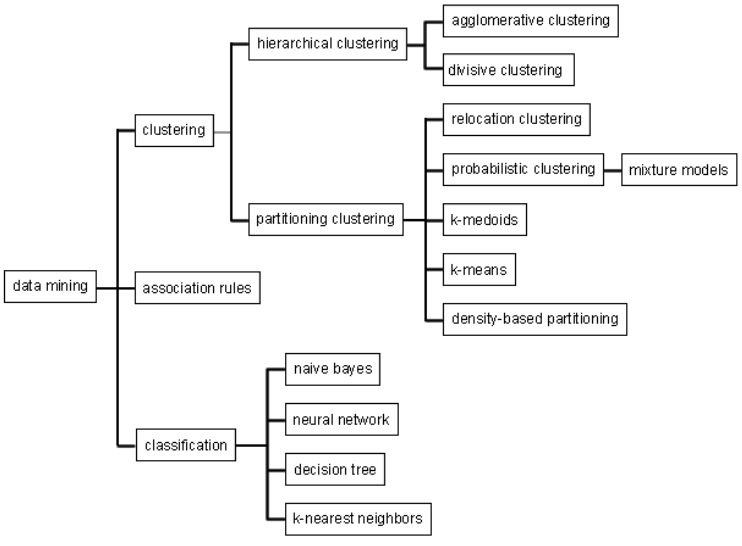


Fig. 3. A concept hierarchy generated by human experts

On the pre-processed data collection that contains documents mostly related to the specific domain, we examine the co-occurrence of each concept pair, and compute the likelihood ratio reflecting their correlation. We tested sentence and document as the context of a co-occurrence and adopted the document level in our further experiments, because the sentence-level adjacency matrix was too sparse. The weighted graph characterized by the document-level adjacency matrix is then partitioned by our algorithm.

5.1.2 Experimental Environment and Tools

The spectral graph partitioning algorithm we used was implemented in a MATLAB software package called the Graph Analysis Toolbox [20]. It was developed by Boston University, and we made modifications to tailor the implementation to our guided algorithm. Our experiments were run on a Fedora Core 4 system with a double processor (Intel Xeon(TM) CPU 2.40GHz) and a 2GB memory.

5.2 The Output Hierarchy

The result of our algorithm (SCSE) on the 'data mining' concept hierarchy is shown in Figure 4(a). Figure 4(b) depicts the results of the input concept set partitioned by the spectral clustering algorithm (SC) only. It can be observed that in our auto-generated hierarchy, almost all hypernym/hyponym relations are correctly discovered except for 'probabilistic clustering' and 'k-means'. While some of the relations were not detected by the algorithm, the shape of the hierarchy is reasonable and natural. The results of the SC algorithm can only group related concepts together, but cannot distinguish between generic and specific concepts. Resorting to a knowledge base such as WordNet or a pattern-based approach and labeling the internal nodes will not generate a reasonable result for the hierarchy shown in Figure 4(b), because the concepts clustered are from different levels. It will be difficult to find a concept or concept vector to label the clusters since generic concepts 'data mining' and 'clustering' are already included in the hierarchy.

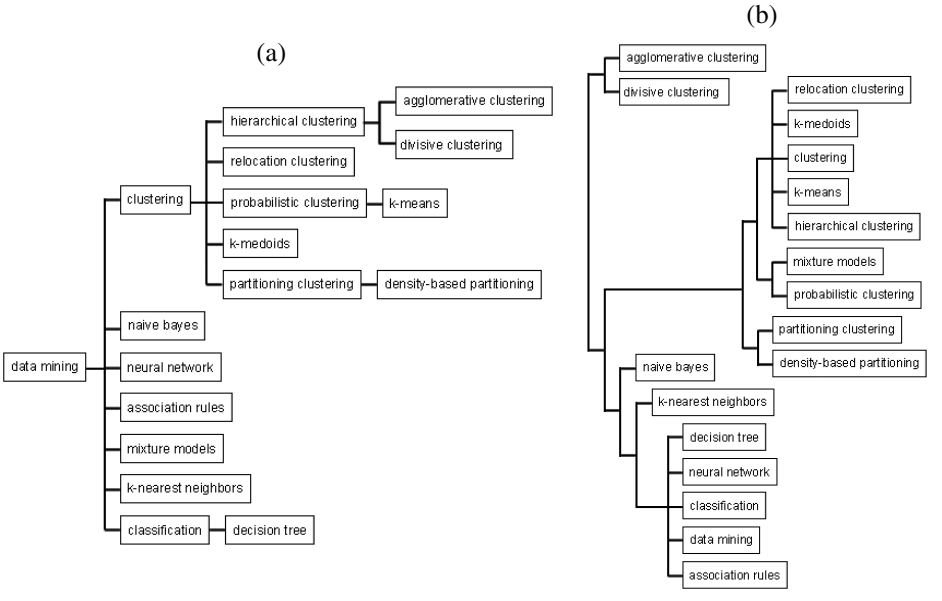


Fig. 4. The concept hierarchy generated by SCSE (a) and SC (b)

6 Conclusion and Future Work

Concept hierarchies have wide applications in many areas of computer science. Yet automatically constructing a concept hierarchy is a challenging task. In this paper, we have presented a method that treats concept hierarchy construction as a graph partitioning problem. A modified spectral graph partitioning algorithm (SCSE) has been proposed to make proper partition decisions, with the guidance of subsumption values for a concept hierarchy. Experimental results from real data show the encouraging potential of the algorithm. In the future, more extensive experiments and formal analysis will be performed to further evaluate and refine the algorithm.

Acknowledgments. The research described here is supported by a Strategic Research Grant of CityU under grant no. 7001815, and by the National Basic Research Fund of China ("973" Program) under Grant No.2003CB317006.

References

1. Yahoo Directory, http://www.yahoo.com
2. Open Directory, http://www.dmoz.org
3. Chuang, S.-L., Chien, L.-F.: A Practical Web-based Approach to Generating Topic Hierarchy for Text Segments. Proceedings of the 2004 ACM CIKM International Conference on Information and Knowledge Management, Washington (2004) 127-136
4. Fellbaum, C.: WordNet, An Electronic Lexical Database. MIT Press (1998)

5. Cimiano, P., Pivk, A., Schmidt-Thieme, L., Staab, S.: Learning Taxonomic Relations from Heterogeneous Sources of Evidence. In: Ontology Learning from Text: Methods, Evaluation and Applications, IOS Press (2005) 59-73
6. Caraballo, S.A.: Automatic Construction of a Hypernym-labeled Noun Hierarchy from Text. Proceedings of 27th Annual Meeting of the Association for Computational Linguistics, Univeristy of Maryland, College Park, Maryland (1999)
7. Velardi, P., Fabriani, P., Missikoff, M.: Using Text Processing Techniques to Automatically Enrich a Domain Ontology. Proceedings of 2nd International Conference on Formal Ontology in Information Systems, (2001) 270-284
8. Hearst, M.A.: Automatic Acquisition of Hyponyms from Large Text Corpora. Proceedings of the 14th International Conference on Computational Linguistics (1992) 539-545
9. Zamir, O., Etzioni, O.: Grouper: A Dynamic Clustering Interface to Web Search Results. Computer Networks 31(11-16) (1999) 1361-1374
10. Zeng, H.-J., He, Q.-C., Chen, Z., Ma, W.-Y.: Learning To Cluster Web Search Results. Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval, Sheffield, United Kingdom (2004) 210-217
11. Vivisimo, http://vivisimo.com/html/index
12. Gao, B., Liu, T.-Y., Feng, G., Qin, T., Cheng, Q.S., Ma, W.-Y.: Hierarchical Taxonomy Preparation for Text Categorization Using Consistent Bipartite Spectral Graph Copartitioning. IEEE Trans. Knowl. Data Eng. 17(9) (2005) 1263-1273
13. Sanderson, M., Croft, B.: Deriving Concept Hierarchies from Text. Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (1999) 206-213
14. Shi, J., Malik, J., Normalized Cuts and Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8) (2000) 888-905
15. Dunning, T.: Accurate Methods for the Statistics of Surprise and Coincidence. Computational Linguistics 19(1) (1993) 61-74
16. Manning C., Schutze, H.: Foundations of Statistical Natural Language Processing, MIT Press (1999)
17. Yang, Y., Pedersen, J.O.: A Comparative Study on Feature Selection in Text Categorization. Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee (1997) 412-420
18. Brin, S., Page, L.: The Anatomy of a Large-scale Hypertextual Web Search Engine. Computer Networks 30(1-7) (1998) 107-117
19. Zhang, Z., Chen, J., Li, X.: A Preprocessing Framework and Approach for Web Applications. Journal of Web Engineering Vol.2, No.3 (2004) 175-191
20. Grady, L., Schwartz, E.L.: The Graph Analysis Toolbox: Image Processing on Arbitrary Graphs. Technical Report, Boston University, Boston, MA (2003)
21. Golub, G.H., Van Loan, C.F.: Matrix Computations, John Hopkins Press (1989)
22. Chen, J., Li, Q., Jia, W.: Automatically Generating an *E*-textbook on the Web. World Wide Web 8(4) (2005) 377-394
23. Chung, F.: Spectral Graph Theory. American Mathematical Society (1997)
24. Church, K.W., Hanks, P. Word Association Norms, Mutual Information and Lexicography. Computational Linguistics 16(1) (1990) 22-29
25. Snedecor, G.W., Cochran, W.G.: Statistical Methods, Eighth Edition, Iowa State University Press (1989)
26. Grady, L., Schwartz, E.L.: Isoperimetric Graph Partitioning for Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2005)

Automatic Hierarchical Classification of Structured Deep Web Databases

Weifeng Su[1], Jiying Wang[2], and Frederick Lochovsky[1]

[1] Hong Kong University of Science & Technology, Hong Kong
{weifeng,fred}@cse.ust.hk
[2] City University, Hong Kong
wangjy@cityu.edu.hk

Abstract. We present a method that automatically classifies structured deep Web databases according to a pre-defined topic hierarchy. We assume that there are some manually classified databases, i.e., training databases, in every node of the topic hierarchy. Each training database is probed using queries constructed from the node titles of the topic hierarchy and the query result counts reported by the database are used to represent the content of the database. Hence, when adding a new database it can be probed by the same set of queries and classified to a node whose training databases are most similar to the new one. Specifically, a support vector machine classifier is trained on each internal node of the topic hierarchy with these training databases and the new database can be classified into the hierarchy top-down level by level. A feature extension method is proposed to create discriminant features. Experiments run on real structured Web databases collected from the Internet show that this classification method is quite accurate.

1 Introduction

Today on the Web, more and more online databases, which compose the *deep Web*, are available. These Web databases can be classified into *structured* and *unstructured* databases. Structured databases provide data objects as structured records with multiple fields, e.g., a query against the Web database of the book vendor www.Amazon.com returns the title, authors, ISBN, etc. of qualifying books. In contrast, unstructured databases provide unstructured data objects, e.g., text or image, in response to user queries. In [4] it is estimated that nearly 80% of current Web databases are structured databases. To locate information from structured Web databases, the user needs to find the relevant Web sites of the Web databases, go to the query interface of those Web databases, submit queries through the interfaces, extract the relevant information from the result Web pages, and compare or integrate the results from the multiple sources. This process, which is referred to as *information integration* [14], is obviously tedious and burdensome for users and automation of the process is desirable.

In this paper, we concentrate on the first step of this process, i.e., the identification of relevant Web databases. Our proposed solution is to organize structured Web databases into a topic hierarchy, similar to that of the Yahoo! Web directory for static Web pages. For example, given the directory in Figure 1, to help a user get configuration and price information about notebook computers, an information integration system can collect

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 210–221, 2006.

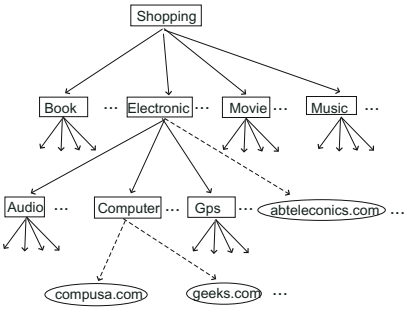


Fig. 1. Shopping sub-directory in Google directory

relevant information first from compusa.com and geeks.com, and then from abteleconics.com, which may also sell computers. There actually exist some commercial Web sites that perform manual classification to organize thousands of Web databases into a topic hierarchy. Examples of such directories include CompletePlanet [1], InvisibleWeb [2] and Librarians' Index to the Internet [3]. While manual classification produces a good quality directory, it is obviously time-consuming, expensive and non-scalable. For example, although CompletePlanet is currently the largest deep Web directory comprising over 70,000 Web databases, it nevertheless covers less than 16% of the deep Web databases according to [4]. Furthermore, since the number of deep Web sites is growing at an exponential rate, manual classification becomes problematic.

In this paper, we propose a hierarchical classification method to automatically classify structured Web databases into a pre-defined topic hierarchy using a combination of database probing and machine learning techniques. We assume that there are a certain number of human-classified databases, i.e., *training databases*, at each node of the topic hierarchy. This assumption is reasonable considering that there are some commercial directories, such as CompletePlanet, already available. For each training database, we submit a set of probing queries constructed from the associated titles of hierarchy nodes. Subsequently, the query result counts returned by the training databases are used to train and construct a support vector machine (SVM) classifier for each internal node of the topic hierarchy. When a new database is added, the same set of probing queries are submitted to the database and the SVM classifiers are utilized to classify the new database into the hierarchy. The classification method we use in this paper is a top-down process that starts at the root of the hierarchy and pushes the new database as low as possible in the hierarchy.

There are two main contributions in this paper:

1. To the best of our knowledge, this paper is the first to address the structured Web database classification problem. We believe structured Web database classification is necessary and critical to the emergence of a one-stop search service that can help users locate relevant information in the deep Web.
2. We develop a top-down, level-based hierarchical classification method using database probing techniques and support vector machine classifiers. Our experiments on real Web databases demonstrate that probing-based Web database classification is both feasible and highly accurate.

The rest of this paper is organized as follows. Section 2 provides some background on the techniques used in our approach. Section 3 first formally defines a topic hierarchy and then outlines our probing-based classification method. Details of the classification method are described in section 4. Section 5 reports our experiments and the experimental results. Section 6 concludes the paper.

2 Background

Our work draws upon previous work on support vector machines, text database classification and structured Web source clustering.

Support Vector Machines (SVM). Vapnik [13] proposed the theory and algorithm of support vector machines in 1979. Since then, this method has been gaining popularity in the machine learning community, becoming the state-of-the-art method for classification, clustering and regression. It has been successfully applied in many other applications [5,9,10]. The SVM classification method tries to learn a separating hyperplane that maximizes the margins between different classes (positive or negative) of records. Of course, not all problems are linearly separable. This can be solved by a modification of the optimization formulation that allows, but penalizes, examples that fall on the wrong side of the decision boundary.

Text Database Hierarchical Classification. Text database hierarchical classification organizes text databases into a topic hierarchy. In [6,7], a text database is assigned to a category according to its specificity and coverage measure to this category. The specificity and coverage measure are calculated based on the query result counts the databases returned for a set of queries, which are generated from a set of pre-classified documents. In [15], both test databases and categories in the topic hierarchy are represented as document vectors, between which the similarity can be calculated. A category vector is formed by its text description and the description of its subcategories. After a set of keywords, selected from the category description, are submitted to probe each test database, the database vector is formed by the documents returned for the probing queries. A new database is assigned to the category with which it has the biggest cosine similarity.

Structured Web Source Clustering. In [8], a clustering model is proposed to organize structured Web databases according to the schema of their query interfaces, where the topic hierarchy is formed automatically instead of being predefined. Although this work is effective for organizing structured Web databases into subject domains, such as Book and Music, it is incapable of classifying the databases into a finer topic hierarchy because the query interfaces of the databases in a given subject domain are semantically similar to each other. For example, the online book vendor Amazon.com sells all kinds of books while computerbooksonline.com focuses on only computer books. Although books from these two sites cover different topics, the two web sites have similar query interfaces. As a result, using only the query-interface-based approach, the two databases cannot be differentiated. In fact, given a topic hierarchy tree about books, Amazon.com should be put into the root node of the tree while computerbooksonline.com should be put into a child node that is about computers of the tree.

3 Classification of Structured Databases

In this section, we discuss how structured databases can be organized into a topic hierarchy. We first give a formal definition of the topic hierarchy and then present a probing-based method for structured Web database classification.

3.1 Topic Hierarchy

As mentioned in Section 1, there exist several commercial directories on the Web that organize Web databases into a topic hierarchy. Figure 1 shows a portion of the hierarchical topic directory of google.com related to shopping. In this figure, the square nodes form the topic hierarchy and the oval nodes represent the structured Web databases that are classified into the hierarchy. Provided with such a topic hierarchy, users can follow its arcs to find relevant databases for any specific topic.

Definition 1. *A **topic hierarchy** is a rooted directed tree consisting of a set of nodes N and a set of edges $E : N \rightarrow N$, where*

- *each node $n \in N$ represents a topic category and each node has a title $tl(n)$ that describes the topic of the node.*
- *each directed edge $e \in E$ represents a subcategory relationship between the end nodes of e, where the target node $t(e)$ is a subcategory of the source node $s(e)$.*

If a database d belongs to a node n in the topic hierarchy, then the major content of d is about the topic of node n.

In a topic hierarchy, a node n is an *internal node* if there exists another node n' such that n' is a subcategory of n. Otherwise, n is a *leaf node*. In the topic hierarchy in Figure 1, the node with title Electronic is an internal node while nodes with title Computer and Gps are leaf nodes.

3.2 Probing-Based Classification

Given a topic hierarchy, the *structured database hierarchical classification task* is to correctly place topic-unknown structured databases into the nodes of the hierarchy, where databases in the same node share similar topics and databases in different nodes are about different topics. Note, however, that in the topic hierarchy, topics of different nodes are allowed to overlap. Consequently, given a set of pre-classified databases, a new database should be placed into the node that has the *most similar* pre-classified databases with the new database.

Example 1. *Consider the topic hierarchy in Figure 1. Online computer vendors compusa.com and geeks.com, which are in the Computer node, mainly selling computers; while abteleconics.com, which is in the Electronic node, sells all kinds of electronics, including computers. Suppose a new online computer vendor, clubmac.com, needs to be classified. As it only sells computers, its content is more similar to the content of compusa.com and geeks.com than to the content of abteleconics.com. Hence, clubmac.com should be place in the Computer node instead of the Electronic node.*

In this paper, we assume that there are some existing structured Web databases, i.e., training databases, that have been manually placed into the topic hierarchy. The training databases under a node can be used to represent the node. Hence, the problem now is how to represent a database. It is natural to represent a database according to its content. However, as mentioned in Section 1, most of the content of Web databases is hidden behind their query interfaces. To tackle this problem, we believe a probing-based approach that submits a set of constructed queries to a Web database and analyzes the returned results is able to reveal the content of the Web database. If we somehow associate each probing query with a node in the topic hierarchy and submit the query to a target database, then the number of results returned by this database would answer the questions of whether and how much the database's content is similar to the topic(s) represented by the node.

The queries that are used to probe the Web databases are critical for the success of a probing-based classification method. We found that the node titles in the topic hierarchy are ideal to be used as probing-queries because they are labeled by human experts such that they are the most representative of the topics of their nodes. Moreover, the titles of a node's subcategories may also be good sources for the probing queries, as they often cover different aspects of the topic of their "parent" node. Therefore, in this paper we employ the title of a topic node and its subcategories' titles to construct the set of probing-queries that are associated with the topic. As leaf nodes do not have subcategories of their own, one additional keyword level is needed for leaf nodes of the topic hierarchy. For simplicity, we call the additional set of keywords for the leaf nodes the subcategory of the leaf nodes.

4 Using SVM to Classify Structured Databases

In the structured database classification task, the goal is to find content differences among different topic nodes, where a topic node is represented by the content of the structured databases that belong to the node, and the content of a structured database is represented by the number of results that it returns for the set of probing-queries. There are two classic approaches to hierarchical text classification [12], the big-bang and the top-down, level-based approach. Comparing these two, we adopt the top-down approach because it has the following advantages:

1. The hierarchical information in the topic hierarchy can be fully utilized. The big-bang approach uses a single flat classifier that takes all nodes in the hierarchy into consideration at the same time, in which the hierarchical information is ignored.
2. A classification that involves too many classes, which may degenerate the classification accuracy, is avoided. Again, since the big-bang approach considers all nodes in the hierarchy in a single flat classifier, it may involve too many classes in the classifier for a complex topic hierarchy.

4.1 Probing Process

In this subsection, we describe how to perform the query probing process, in which queries are constructed from titles of the nodes in a topic hierarchy.

Given the query interface of a structured Web database d and a query t, the task of probing d using t is performed in the following steps:

1. t is submitted exhaustively to every editbox or textbox of the query interface of d.
2. For each submission of t, the count of relevant results reported by d is extracted. For example, if a database reports "Your search for Art matches 159 records", the wrapper extracts the reported count, i.e., 159.

For a structured Web database whose query interface has more than one editbox, we select the editbox that has the biggest total number of reported counts for all submitted queries.

4.2 Structuring the Input Space

After obtaining the reported counts for the probing queries, we can build the feature space for Web database classification. The query result counts are transformed to feature values, which are used as input to the classifier. Hence, "feature" and "query" are used interchangeably, when appropriate, in the following. The feature space construction consists of three steps:

1. Query result count normalization.
2. Feature extension.
3. Feature selection.

Query Result Count Normalization

Different structured databases in the deep Web may contain a different number of relational records in their back-end databases. For example, the biggest online book vendor Amazon.com contains millions of books, while pabook.com, which focuses on art books, only has about 20,000 books. Intuitively, if for the same probing query both Amazon.com and pabook.com report 5,000 matched results, then we should consider pabook.com to be more similar to the topic of the query. Consequently, the normalized query result counts are more suitable than their absolute values to measure the database content's distribution over the topics. A normalized query result count for the ith probe of Web database d_m is calculated as

$$f_{mi} = \frac{r_{mi}}{\sum_{j=1}^{l} r_{mj}}, \tag{1}$$

where r_{mi} represents the query result count reported by Web database d_m for query q_i, and l represents the total number of probing queries. A normalized query result is a real value between 0..1 and is referred to as a *feature value*.

Feature Extension

We observe that in general, the higher the level of a topic node, the smaller the number of training databases that are available to the node. Insufficient training examples may degenerate the performance of classification, which is verified by the experimental results in Section 5.4. We alleviate this problem by introducing a new feature for each node of the topic hierarchy. For a node n that has subcategories $n_i, i = 1..k$ with feature values $f_i, i = 1..k$, a new feature f_{cn} is created for node n whose value is the average of its children nodes' feature values:

$$f_{cn} = \frac{\sum_{i=1}^{k} f_i}{k}. \quad (2)$$

Intuitively, the feature f_{cn} for node n is a summary of its subcategories. Our experiments show that this artificial feature helps to increase the classification accuracy.

Feature Selection

Consider the topic hierarchy in Figure 1. A query "comedy" under Movie can be very effective when classifying databases as either Electronic or Movie, but it will be a noisy or useless feature if we want to build a classifier on node Electronic to decide whether a database belongs to node Electronic itself or any subcategory of node Electronic. Hence, when constructing a classifier for node n, we only consider those features whose nodes are descendants of n plus the extended feature generated for n in the previous step and n's own features. For example, in Figure 1, when building a classifier for node Electronic, the feature values of the node's training databases for domain Book, Movie and Music are ignored. Each feature is related to one of the classifier's classes whose node is the smallest ancestor of the feature in the topic hierarchy. For example, suppose, given the topic hierarchy in Figure 1, that we want to build a classifier for node Electronic whose output classes include Electronic, Audio. Computer and Gps. Then, features "Laptop" and "Computer" would belong to class Computer and feature "Electronic" would belong to class Electronic.

Different features of n may have different discrimination ability. We calculate the discrimination ability ρ_j of a feature f_j according to the ratio between its average value in the class to which it belongs and its overall average. The criterion is defined as

$$\rho_j = \frac{\overline{x}_{jk}}{\overline{x}_j}, \quad (3)$$

where $\overline{x}_{jk}$ is the average of feature x_j for databases of class k to which x_j belongs and $\overline{x}_j$ is the overall average of feature f_j over all databases. We consider a query with small discriminating ability as a noise feature that should be removed. When building a classifier for a node n in the topic hierarchy, we define the *selected feature ratio* r_f to be the proportion of the features under node n that are used for classification according to their discrimination ability ρ. The effect of the selected feature ratio r_f on the classification performance is presented in Section 5.4.

4.3 Training Process

After each database is represented as a vector with each component in the vector calculated using equation 1, an SVM classifier is trained for each internal node of the topic hierarchy. The method of constructing a classifier hierarchically for each internal node of the topic hierarchy is shown in Algorithm 1.

First, for each training database d_i, the constructed title queries are submitted to d_i and the returned query result counts are normalized to get the feature vector v_i of d_i (Lines 3-6). Then we build a classifier c_j for each internal node n_j in the following steps:

Algorithm 1. Hierarchical Training

Input:
D - a set of training databases
Q - a set of constructed queries
T - a topic hierarchy containing databases in D
r_f - the ratio of features that will be selected
Output:
$\mathcal{C} = \{c_1, ..., c_n\}$ - a set of classifiers for all internal nodes of T

```
1: begin
2: C ← ∅
3: for each database d_i ∈ D do
4:     probe d_i with all queries in Q
5:     normalize the query result counts by Equation (1)
6: end for
7: for each internal node n_j of T do
8:     for each node that is a descendant of n_j do
9:         create a feature and set its feature value by Equation (2)
10:    end for
11:    for each feature f_k under node n_j do
12:        Calculate ρ_k by Equation (3)
13:    end for
14:    rank features by ρ_k and select features that are in the top r_f percent
15:    c_j ← BuildSVMClassifier(n_j)
16:    C ← C + c_j
17: end for
18: end
```

1. Each node that is a descendent of n_j is created for a feature (Lines 8-10).
2. The feature selection process is performed (Lines 11-14), in which features are ranked by their differentiation ability and a certain percentage of the features are selected to form the feature space for the classifier c_j.
3. The SVM classifier c_j is trained (Line 16), during which each child n_{jk} of n_j is represented as all training databases under n_{jk} and n_j is represented by the databases in itself (Line 15).

4.4 Classification Process

When there is a classifier available for each node of the topic hierarchy, new databases can be placed into the topic hierarchy using a hierarchical classification algorithm. First, for the new database d, the constructed title queries are submitted to d, and the returned query result counts are normalized to get the feature value for each query so that d is represented as a vector v_d. Then, the classification is performed top-down level by level. During the process, feature extension and feature selection are necessary as required by the classifier. The classification stops going down the topic hierarchy when d cannot be pushed down anymore or it reaches a leaf node.

5 Experiments

In this section, we present our experiments on classifying some real structured Web database. We describe first the collected data and the previous work to which we compare our method, then the evaluation metric, and finally the experimental results. In our experiments, we use Platt's sequential minimal optimization (SMO) algorithm [11] to train a support vector machine classifier. A linear kernel is used because [9] concluded different kernels have similar performance for high-dimensional data.

5.1 Data Collection

For the experiments, we collected real Web databases in four representative domains in E-commerce: Book, Movie, Music, and Electronic. We then expanded these domains down one more level by selecting the ones with the most number of databases as listed in the Google directory. The topic hierarchy used in the experiments contains 22 leaf nodes. Each leaf node of the topic hierarchy contains at least three structured Web databases and each leaf node contains 3-14 keywords collected from the Google directory that are actually the subcategories of the leaf nodes in the Google topic hierarchy. We collected 243 titles in all. We manually collected 169 structured Web databases from the Google directory and Yahoo! directory and manually checked each database to put it into the corresponding node in the topic hierarchy. Each domain contains 30-70 databases. We ran a ten-fold cross validation on the collected data set to take full advantage of the data. During data set division, the databases within each node are distributed to each subset as evenly as possible.

5.2 Related Work for Comparison

To the best of our knowledge, this is the first work to automatically classify structured Web databases using query-probing. Since there is no other method directly available for comparison, we adapt two probing-based text database classification methods, introduced in Section 2, for structured database classification and compare our method with them.

Specificity & Coverage (S&C): In [6,7], text databases are classified into a topic hierarchy according to their specificity and coverage, which is evaluated by query probing. Similar to our method, queries based on titles are used as probes in the implementation of the Specificity & Coverage method. In [6], there are two threshold parameters for specificity and coverage, τ_s and τ_c, respectively. A database can be classified into a topic node only if its estimated specificity and coverage are higher than these thresholds. We favor the S&C algorithm by trying multiple pairs of carefully selected τ_s and τ_c. The pair that has the best classification results is selected (Section 5.4).

Cosine Distance Metric (CDM): Wang et al. present three algorithms [15] for classifying a text database to a topic hierarchy among which the High Similarity with Database Centroid (HSDC) algorithm has the best performance. In HSDC, both the text database and the topic nodes are represented as vectors and the cosine distance metric is used to evaluate the similarity. A new database is classified to the node whose vector has the biggest cosine distance metric with the new database vector.

5.3 Evaluation Metrics

To measure a classifier's performance, the precision-recall measure in [7] is adopted. In particular, given a node n in the topic hierarchy, we expand it by including all its subcategories, i.e., $Expanded(n) = \{c | c \textit{ is } n \textit{ or } c \textit{ is a descendant of } n\}$. Given a database d with its ideal category $Ideal(d)$ that was classified as $Classified(d)$, we define $Correct(d) = Expanded(Ideal(d))$. The precision and recall of the classification are measured as

$$precision = \frac{|Correct \bigcap Classified|}{|Classified|},$$

$$recall = \frac{|Correct \bigcap Classified|}{|Correct|}.$$

In traditional IR, the two metrics are combined into a single metric, F-measure, to facilitate the classification result comparison:

$$F - measure = \frac{2 \times precision \times recall}{precision + recall}.$$

5.4 Experimental Results

In this section, we report and analyze the experimental results of the three different classification methods. Specifically, we use "HSVM" to represent our method, "S&C" to represent the Specificity & Coverage method, and "CDM" to represent the cosine distance metric method.

Accuracy of Different Classification Methods

Figure 2 shows the performance of these three methods with varying selected feature ratio r_f, where r_f is the proportion of the features selected from a feature list ranked by feature discrimination ability. S&C is implemented with the specificity threshold $\tau_s = 0.6$ and the coverage threshold $\tau_c = 6$ since these two threshold values generate the best performance among the many pairs of valid values that were tested. It can be seen that HSVM consistently outperforms CDM and CDM outperforms S&C for any r_f. In fact, HSVM is the only one that can achieve an F-measure over 0.8.

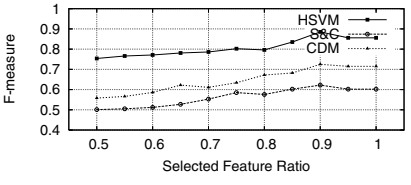


Fig. 2. The average accuracies of HSVM, S&C and CDM with different selected feature ratios

Effect of Feature Selection

To remove the noisy features, we rank the features by their discrimination ability ρ and discard those features that are not in the top r_f percent. In Figure 2, we can see the effect of the feature selection ratio over the classification accuracy. All these methods reach their performance peak when r_f is set to 90%. Since the topic hierarchy is built by experts manually, the node titles are generally the best representatives of their content. Consequently, few of the probing queries constructed from the node titles are noisy features. Thus, if too many features are removed, information that is useful for classification is lost, which makes the accuracy decrease.

Effect of Feature Extension

Recall that a new feature f_{cn} is created for each internal node to alleviate the problem of a lack of sufficient training databases. In Figure 3(a), we compare the classification accuracy of HSVM with this extended feature and without it. We can see that the extended feature does help to improve the classification accuracy for any selected feature ratio r_f.

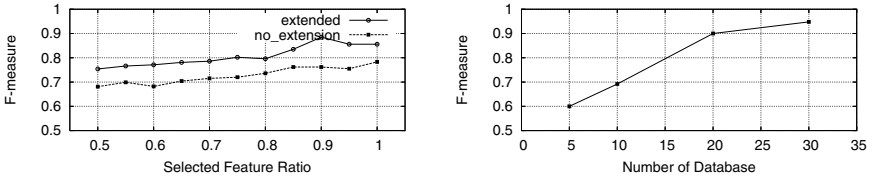


Fig. 3. (a) The F-measures with the extended features and without the extended features for varying selected feature ratio. (b) The F-measures for nodes with a different number of databases when the selected feature ratio $r_f = 0.9$.

Effect of the Number of Training Databases

Almost all machine-learning techniques are sensitive to the number of training instances. In most cases, the more training instances there are available, the more accurate is a trained classifier. Figure 3(b) shows the F-measure for nodes that have different numbers of databases in the topic hierarchy. We classify the nodes into four buckets according to the number of databases they contain: 0-4, 5-9, 10-19 and 20-35. We use 5, 10, 20 and 30 to represent the 4 buckets. It can be seen that the F-measure increases steadily when more databases are available. The F-measure reaches 0.9 when there are more than 10 databases available. This shows that our approach is accurate if a reasonable number of training databases are available. Considering that there are some fairly large scale directories available now for structured Web databases, our method is feasible in practice to classify structured Web databases in the deep Web.

6 Conclusions

In this paper, we present an approach that classifies structured Web databases into a predefined topic hierarchy using a combination of query-probing and SVM machine

learning techniques. In this approach, we assume a set of training databases are available for each node of the hierarchy. The node titles of the hierarchy are used to constructed a set of probing queries submitted to structured Web databases. After collecting the query result counts reported by the training databases, SVM classifiers are trained to classify new topic-unknown databases. To the best of our knowledge, our work is the first to address the automatic structured Web database classification problem. Experiments show that this is an accurate method and can be applied on a large scale.

Acknowledgment. This research was supported by the Research Grants Council of Hong Kong under grant HKUST6172/04E.

References

1. CompletePlanet. http://www.completeplanet.com.
2. InvisibleWeb. http://www.invisibleweb.com.
3. Librarians' Index to the Internet. http://www.lii.org.
4. K. C.-C. Chang, B. He, C. Li, and Z. Zhang. Structured databases on the Web: Observations and implications. Technical Report UIUCDCS-R-2003-2321, CS Department, University of Illinois at Urbana-Champaign, February 2003.
5. C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
6. L. Gravano, P. G. Ipeirotis, and M. Sahami. Probe, count, and classify: Categorizing hidden Web databases. In *ACM SIGMOD Conference*, pages 363–374, 2001.
7. L. Gravano, P. G. Ipeirotis, and M. Sahami. Qprober: A system for automatic classification of hidden-Web databases. *ACM Transactions on Information Systems*, 21(1):1–41, January 2003.
8. B. He, T. Tao, and K. C.-C. Chang. Organizing structured Web sources by query schemas: A clustering approach. In *Proceedings of the 13th Conference on Information and Knowledge Management*, pages 22–31, 2004.
9. T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings 10th European Conference on Machine Learning*, pages 137–142, 1998.
10. H. Kriegel, P. Kroeger, A. Pryakhin, and M. Schubert. Using support vector machines for classifying large sets of multi-represented objects. In *SIAM International Conference on Data Mining*.
11. J. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods*, 2000.
12. A. Sun and E. Lim. Hierarchical text classification and evaluation. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 521–528, 2001.
13. V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
14. J. Wang and F. Lochovsky. Data extraction and label assignment for web databases. In *Proceedings of the 12th International Conference on World Wide Web*, 2003.
15. W. Wang, W. Meng, and C. Yu. Concept hierarchy based text database categorization in a metasearch engine environment. In *Proceedings of the First International Conference on Web Information Systems Engineering*, pages 283–290, June 2000.

A Robust Web-Based Approach for Broadcasting Downward Messages in a Large-Scaled Company

Chih-Chin Liang[1], Chia-Hung Wang[2], Hsing Luh[2], and Ping-Yu Hsu[3]

[1] Telecommunication Laboratories, ChungHwa Telecom Co., Ltd No.21-3, Sec. 1, Sinyi Rd., Jhongjheng District, Taipei City, Taiwan 100, R.O.C.
Department of Management, National Central University, No. 300, Jung-da Rd., Jung-li City, Taoyuan, Taiwan 320, R.O.C.
lgcwow@gmail.com
http://www.chttl.com.tw/eindex.htm
[2] Department of Mathematical Sciences, National Chengchi University, No.64, Sec. 2, Jhihnan Rd., Wen-Shan District, Taipei City, Taiwan 116, R.O.C.
{93751502, slu}@nccu.edu.tw
[3] Department of Management, National Central University, No. 300, Jung-da Rd., Jung-li City, Taoyuan, Taiwan 320, R.O.C.
pyhsu@mgt.ncu.edu.tw

Abstract. Downward communication is a popular push-based scheme to forward messages from headquarters to front-line staff in a large-scaled company. With the maturing intranet and web technology, broadcasting algorithms, including pull-based and push-based broadcasting algorithms, it is feasible to send downward messages through web-based design by sending packets on a network. To avoid losing messages due to the traditional push-based method, companies adopt a pull-based algorithm to build up the broadcasting system. However, although the pull-based method can ensure that a message is received, it has a critical problem, the network is always congested. The push-based method can avoid congesting the network, but it needs a specific robust design to ensure that the message reaches its destination. Hence, adopting only a pull-based or a push-based broadcasting algorithm is no longer feasible especially not for a large-scaled company with complex network architecture. To ensure that every receiver will read downward messages thereby reducing the consumption of network bandwidth, this work proposes a robust web-based push- and pull-based broadcasting system for sending downward messages. This proposed system was successfully applied to a large-scaled company for a one-year period.

Keywords: web-based enterprise systems, e-commerce, downward communication.

1 Introduction

Information management for front-line staff such as front-desk, customer service, call center, and others, is difficult to handle adequately in a large-scaled company. This is due to the fact that the needs of customers change frequently and new services are being generated by a company at a rapid rate in order to meet various demands, e.g. the

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 222 – 233, 2006.

promotion for a new product, discounts, etc. A decision maker must disseminate strategic downward messages to the front-line staff through a push-based approach in order to satisfy various demands [1]. Traditionally, the push-based approaches for sending downward messages included official documents, e-mails and telephone messages. However, these methods have defects, such as, messages cannot be ensure to be received, poor efficiency due to the multiple organizational hierarchies of a large-scaled company, and messages passing through too many intermediaries, resulting in misunderstandings [2]. Traditional push-based approaches for sending downward messages are not suitable in a large service-oriented company, because they cannot effectively send downward messages to each receiver.

A broadcasting algorithm used for sending unidirectional packets from senders to numerous receivers through a network is similar to that for sending downward messages. Hence, a company should be able to utilize a broadcasting algorithm to disseminate downward messages, although this has received little attention to-date.. Broadcasting algorithms can be classified into two categories: push-based broadcasting algorithms and pull-based broadcasting algorithms [3]. The push-based broadcasting algorithms are similar to the traditional push-based approaches: a sender actively disseminates a packet to receivers, but has the potential of losing the packet. Repeatedly retrieving a packet from a sender, a pull-based broadcasting algorithm can ensure a packet is retrieved, but it causes a congested network [3].

In recent years, adopting push- and pull-based broadcasting algorithm for sending packets has become a trend for designing a broadcasting system [4], [5], [6], [7]. The present study proposes a robust push- and pull-based broadcasting system (UBS, united broadcasting system) to send downward messages in a large-scaled company with a hierarchical organization. In addition, with the advances made in web-based technology, a front-end staff member can access a server not only through a personal computer but also through various other devices, since UBS is a web-based design [8]. The UBS uses servers as senders and client devices as receivers. Each server allows a client's device to retrieving retrieve messages through a pull-based design, and each message can be replicated among servers through a push-based design. To reduce network traffic for the pull-based design, the UBS client devices retrieve messages mainly from a server in the same local area network (LAN). In addition, to avoid losing messages, a robust push-based design is used to help replicate messages among servers.

To ensure the UBS design is applicable, starting in January 2005 we applied the UBS to the largest telecom company in Taiwan, ChungHwa Telecom (NYSE: CHT), This work simulates the model of UBS to understand the robustness and the efficiency of broadcasting messages to help CHT to decide the scalability of this broadcasting framework. Finally, CHT choose 40 servers as senders to serve more than 10 000 client devices successfully.

The rest of this paper is organized as follows. In Section 2 the broadcasting algorithms and the problems for a robust design to send downward messages are illustrated. In Section 3, the UBS design is described. In Section 4, the derived model of UBS is presented. In Section 5, the managers' utility is discussed. Finally, conclusions are drawn in Section 6.

2 Literature Review

2.1 Broadcasting Algorithm

Broadcasting algorithms can be classified as two methods: the push-based broadcasting algorithm and the pull-based broadcasting algorithm [3]. A push-based broadcasting algorithm is used for the scheduling principle of push-based systems. A sender actively sends items to receivers using scheduling principles. The pull-based broadcasting algorithm indicates that the receiver sends requests to a sender for retrieving items [7].

However, when adopting to send downward messages, it must be noted that each algorithm has its limitation. Adopting the push-based broadcasting algorithm to send downward messages through a network is more effective than traditional approaches and consumes little network resource. However, without a robust design, a push-based broadcasting method potentially has the same drawback as the traditional push-based approaches: the chance of losing messages. In addition, adopting the pull-based broadcasting algorithm can avoid losing messages through repeated request for retrieving messages. However, it places a heavy drain on the network resources [3]. In addition, the network environment of a large-scaled company is a complex internetwork connecting each LAN through a wide-area network (WAN) with restricted network bandwidth. All client devices in the same LAN are connected to each other with unrestricted network bandwidth [4]. Therefore, adopting only a pull-based design or a push-based design cannot fit the requirements of a large-scaled company [5]. Therefore, how to correctly combine the pull-based algorithm and the push-based algorithm becomes key to assist a company sending downward messages.

2.2 The Robust Design of Push-Based Approaches

In the push-based design, all messages must reach a destination. Hence, precise timing and the correct order are the major concerns for broadcasting downward messages, in addition to receiving the correct text of each message [6].

Existing robust push-based broadcasting systems typically utilize three techniques: token-passing approach, discrete acknowledgement approach, and, two-phase approach [10]. The above robust mechanisms can be divided into two kinds of design: adding information to the messages being sent to verify their accuracy and status, e.g. the list of visited stations, and utilizing the signal, and an acknowledgement, to let senders and receivers decide the next action [6].

These robust approaches are focused on ensuring that a destination can receive the completed messages. However, the goals of downward communication are that each active receiver must be able to successfully receive every downward message efficiently, and it must be feasible for the hierarchical network architecture of a large-scale company. Hence, an adjusted robust design is needed.

3 The Design of the UBS

The UBS is designed as a hierarchical architecture, where a sender connects its parent sender and its child senders through WAN. A receiver mainly connects a sender within

the LAN. Fig. 1 shows the architecture of the UBS. The chief sender (CS), the senders, and the receivers have a message storage in which each broadcast messages can be stored.

The design of the UBS makes the following assumptions: at least one sender is in operational mode, the clocks of all senders are synchronized, and a downward message is equally distributed throughout a balance tree in which the depth and the number of layers in each branch are the same. The operational mode indicates that a sender, the CS, or a client is functioning well, and that the message storage can be accessed successfully.

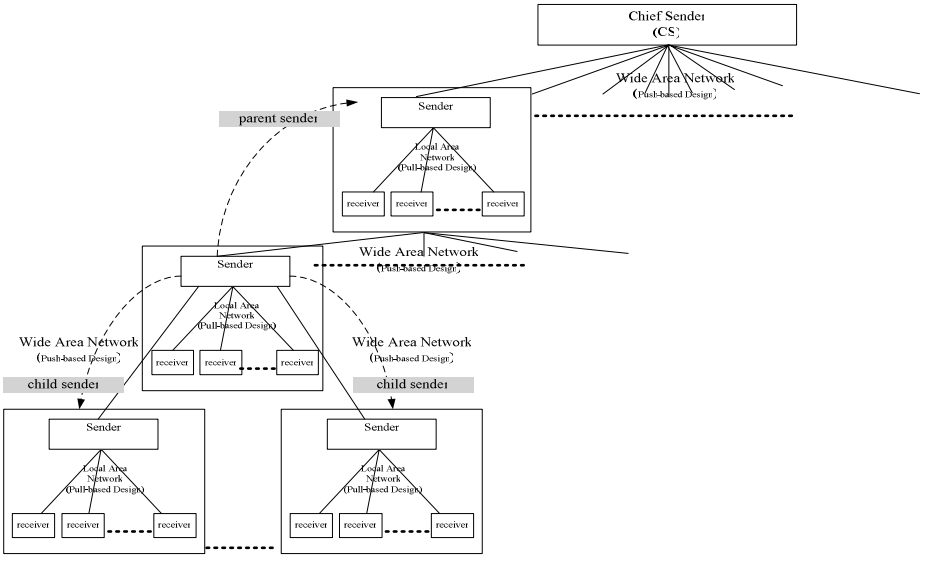


Fig. 1. The architecture of the UBS

To avoid the frequent transfer of messages between each sender through WAN, transmitting messages among senders using a robust push-based broadcasting design. Utilizing a pull-based broadcasting design to transfer messages between a sender and a receiver in the same LAN can reduce the consumption of network resources. In addition, the UBS utilizes the designs of robust push-based broadcasting mechanisms to ensure a message can be replicated among servers: hiding information in a downward message, and sending a specified signal or acknowledgement regarding the status of a downward message.

The data structure of a downward message with hidden information consists of the text, the timestamp, and the message length. The timestamp records the time point in a transaction maintained by the computer in fractions of a second, and is used for various synchronization goals. The message length is the number of characters of the text for verifying the accuracy of a received message.

Additionally, to ensure each child sender receives unread messages, a sender sends a signal to inform all its child senders about a new message that it is ready to send before

it sends a message to its child senders. Hence, if a child sender receives a signal about a new message being sent, but receives no message within a certain time interval, then the sender sends a signal requesting that another sender sends the unread message to the sender. Moreover, if a sender is unable to send a new message to a child sender, the sender then sends a signal about the message it cannot send to the sender to the rest of the child senders. Therefore, the rest of the child senders can send the message to the child senders of the sender without the message.

3.1 General Procedure

An operator writes a message and sends it to a sender A. Timestamp and message length are generated whenever the text of the message is written to the message storage of sender A. Next, sender A sends a signal asking to send the downward message to the CS and to all child senders of sender A. After the request is received, the CS and each child sender of sender A are locked in sequence. Then, sender A sends the downward message to the CS and to each of its child senders. Next, the downward message is recorded in the message storages of the CS and every child sender of sender A, based on the timestamp sequence. Next, the CS and all child senders of sender A broadcast the downward message until the CS and all senders in operational mode receive the complete message. The process of broadcasting a downward message is similar to the above steps when a broadcast operator writes a message and sends it directly to the CS.

Each receiver frequently sends a request for a new downward message to the sender in the same LAN, and retrieves the unread downward message from the sender within the same LAN. To compare the most recent timestamp of the stored downward message of a receiver and the timestamp of the new downward message, the new downward message is stored in the message storage of a receiver in timestamp sequence and displays the message text to a front-line staff member.

3.2 Recovery Procedure

Potential process and communication failures are as follows: crash [12], general omission [12], and timing failure [13], [14]. However, since the robust design of UBS assumes synchronized time, timing failure is not discussed in this paper. In addition, all handling phases of computer errors are classified as follows: error detection; damage assessment; error recovery; and, fault treatment and continuing service [15].

In the error detection phase, a sender and a receiver have two possible error situations: crash and general omission. In the damage assessment phase, a crash indicates that a sender or a receiver is not working properly and transmits no message. Additionally, if a sender general omission is detected, all child senders of the sender will incorrectly receive a downward message. A receiver retrieves the incomplete downward message from a sender when a general omission occurs.

In the error recovery phase, to run an error recovery, a sender or a receiver adopts the forward recovery for a crash and general omission. The sender reserves its current status and cancels the request for retrieval messages from receivers until the error situation is eliminated. When the sender recovers from a crash, it sends a signal of request consisting of a timestamp and an IP address to the parent sender. Unread downward messages are sent to the recovered sender after the parent sender compares

the received timestamp with the latest timestamp for the stored downward message in its message storage. When a general omission occurs, storing an uncompleted downward message in the message storage is deemed incorrect. Furthermore, no receiver in the same LAN as the sender in which the general omission occurred can retrieve downward messages from the sender. Finally, to ensure that unread downward messages can be received, the recovered sender sends a request to its parent sender to ensure that the parent sender resends the unread downward messages. Additionally, in the error recovery phase, the receiver retains its status and retrieves new downward messages when it recovers from a crash. To recover from a general omission, the receiver retrieves the downward message again since the received message is incomplete.

In the fault-treatment and continuing service phase, when a sender crashes, the downward message cannot be stored in the message storage within a certain time interval. A sender resends the lost downward message whenever the crashed parent sender recovers. However, the broadcast service must continue even when a crash occurs. A parent sender sends a signal to ask that a new downward message be inserted into its child senders, before sending the new message. For example, sending no downward message to sender A, the parent sender of sender A instead sends the downward message to the other child senders. Moreover, the notice, "sender A has malfunctioned" is also sent to the rest of the child senders. Hence, the rest of the child senders are in charge of broadcasting the downward message to all child senders of sender A. In addition, if a sender cannot send the new inserted message to the CS, the sender sends the message to the senders on the second layer and continues the broadcasting service. Whenever the CS recovers, it compares the stored messages with the senders on the second layer and requests the resending of the unread messages to the CS.

However, a sender may crash before the downward message is completely sent to its child senders. That is, not all child senders may receive the downward message. A robust solution to this problem is as follows: whenever the downward message is stored in the message storage of a sender, the signal, "the new downward message is broadcasted," is broadcasted to all child senders. Within a certain time interval, each child sender of the sender in which the downward message is not received then sends a request to the sender to resend the downward message. If a child sender of the crash sender still does not receive the downward message within the certain time interval, it sends a request to another sender, until the downward message has been successfully received. At least one sender is then in operational mode, and the child senders of the crashed sender can find at least one sender to send the downward message. Hence, the procedure of broadcasting downward messages continues to work properly.

Moreover, when a general omission occurs, the downward message cannot be stored in the message storage of a sender. The sender then sends a request for the downward message to be resent to its parent sender. In addition, if a sender then still does not receive a downward message from its parent sender, then the sender sends the request to other senders to retrieve the downward message.

The robust design of the UBS can also deal with mixed errors, crashes and general omissions. If the parent sender crashes, it will produce a general omission. If the parent sender is first omitted and then crashes, then the incomplete downward message is not stored in the message storage of the parent sender. Hence, the child senders will not receive an incomplete downward message. If a child sender crashes, then it cannot generate a general omission. If a child sender is first omitted and then crashes, the

incomplete downward message is not stored in its message storage. Hence, no incomplete downward message is delivered to any sender. However, if a sender receives a new downward message that is incorrect and the parent sender crashes before the request to resend the downward message is received, then this robust design can accommodate the scenario. For example, a sender sends a request to send a downward message to another sender. It waits for a certain time interval and receives no downward message. If a sender has already sent the request to send downward message to its parent sender, then the parent sender crashes immediately before a new downward message is sent, the sender then sends a request to send a downward message to another sender after waiting for a specific time interval and receiving no downward message.

In the fault-tolerant and continuing service phase, a receiver retrieves no new message until it recovers from the crash. Whenever a receiver is in operational mode and cannot find a parent sender to connect to, it will try to connect to one of the grand senders instead.

In summary, this design is robust and efficient since a receiver can correctly receive messages and the push- and pull-based broadcasting technology in this design can work with acceptable bandwidth consumption.

4 A Push- and Pull-Based Model

In this section, the steady state probability will be studied to learn the mean waiting time and the probability of no server in operational mode. Additionally, adopting extra servers, the senders, can improve the performance of sending downward messages but the cost is arising. Hence, deciding the proper number of servers under budget constraint is a major problem of the broadcasting system. The time interval of sending each message is a random variable. That is, the arrival time is a random variable. In this system, clients, the receivers, ask to retrieve messages from a server repeatedly and the period of a message retrieved by every client is assumed continuity.

4.1 Problem Formulation

Clients can retrieve new downward messages one by one. The broadcasting process is finished after the last client gets the message. For example, 10,000 clients try to get a downward message means 10,000 jobs arrive this queueing system and wait for service. Assuming the arrival stream has k jobs, k client computers, the actual number of jobs in an arriving module is k.

There are s machines (servers) in the system to send downward messages, and they have i.i.d. exponential lifetimes. Let $1/\alpha$ be the mean time to the failure, include crash and general omission, of a machine. When a machine fails, it needs mean repair time $1/\beta$.

We assume that both inter arrival and service times follow exponential distribution with means λ^{-1} and μ^{-1} respectively. A state of system is denoted by (n,i), where n is the number of jobs in the service or in the waiting room, $n \geqq 0$, and i is the number of servers in operational mode in the system, $0 \leqq i \leqq s$. The stationary probability is denoted by $\boldsymbol{\pi}$ as

$$\boldsymbol{\pi} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots),$$

where $\boldsymbol{\pi}_{\mathbf{n}}$ is a $(s+1)$-dimensional row vector of stationary probability when there are n jobs in the system. For this system, a steady state will exist if the utilization factor $\rho < 1$. It is clear that the queueing problem is still Markovian in the sense that future behavior is a function only for present not for the past. Let $W(s)$ be the mean waiting time in the system given s servers initially. We want to find the steady-state probability, P_n, when n jobs in the system.This work arranges the states (n, i) in lexicographic order and partition of the state space according to the number of customers, n, i.e.

$$S_n = \{(n,i) \mid 0 \le i \le s\},\ n = 0,1,2,\ldots$$

In this model, we have the transition rate matrix $\boldsymbol{Q}$ which is of the block form. For the state balance equations $\boldsymbol{\pi Q} = \mathbf{0}$ and the normalization condition $\boldsymbol{\pi 1} = \mathbf{1}$, we can compute $\boldsymbol{\pi}_n$, $n = 0,1,\ldots$. Then we have the steady state probability

$$P_n = \boldsymbol{\pi}_n \cdot \mathbf{1} \tag{1}$$

and the limiting probability that there are i servers in operational mode

$$P((i-1) \text{ servers in operational mode}) = \left(\sum_{n=0}^{\infty} \boldsymbol{\pi}_n \right) \cdot e_i, \tag{2}$$

for $i=1,\ldots,s+1$, where e_i is a $(s+1)$-dimensional column vector with one at the i-th component and zero elsewhere. In specific, we have the limiting probability that there is no server in operational mode in the system,

$$P(\text{no server in operational mode}) = \left(\sum_{n=0}^{\infty} \boldsymbol{\pi}_n \right) \cdot e_1, \tag{3}$$

and this probability is small enough to ignore for $\alpha \ll \beta$. From equation (1), we can compute the mean number of jobs in the system,

$$L = \sum_{n=0}^{\infty} n P_n. \tag{4}$$

By using the Little's formula, we also have mean waiting time

$$W(s) = \frac{L}{\lambda}. \tag{5}$$

With mean waiting time W, our objective is to determine the increase in investment of service for a corresponding decrease in system delay.

4.2 Queueing Design Problems

This work considers a system with unknown number of servers s, where s belongs to the set of positive integers Z^+. Assuming a set-up cost M is for each server. Suppose the company has the limited budget B for adding servers. Our objective is to determine the optimal value of s under the budget constraint. If there are s servers in the system, then

k jobs are shared by servers. This work assumes k jobs are divided equally by s servers for simplicity, From equation (5), we have the average waiting time in the system $W(s)$.

Using $U(W(s))$ to denote the manager's utility function for expected waiting time $W(s)$. The expected waiting time $W(s)$ will decrease when the number of servers s increases. Assume the manager's utility $U(W(s))$ increases as decreasing $W(s)$. The objective is to maximize the objective function $f(s)$ with respect to s, where

$$f(s) = U(W(s)) - M \cdot s. \tag{6}$$

We assume $f(s)$ is unimodal of s on $[0,B/M]$.This work uses this representation to describe the manager's choice. The manager will choose so as to maximize his or her utility function on the feasible set. The problem of maximizing a given function on a given set is a mathematical problem. This gives rise to the following optimization problem

$$\begin{aligned} \max\ & f(s) = U(W(s)) - M \cdot s \\ \text{st.}\ & M \cdot s \le B \\ & W(s) \le D \\ & s \in Z^{+} \end{aligned} \tag{7}$$

where D is the delay constraint. If s^* is the optimal number of servers under the budget constraint, then the basic geometric properties of this objective function $f(s)$ are that it is increasing as $0<s<s^*$ and it becomes decreasing as $s \geqq s^*+1$.

The efficiency of the computations relies not only on the implicit enumeration scheme but also on the efficiency of calculations of the probabilities. Due to the complexity of the probabilities and the requirement of integer values for the decision variables, a search technique such as implicit enumeration is usually required. A set of decision variables must first be specified, and then the constraints and objective function are evaluated. This optimization problem gives rise to a trade-off of better customer service versus the expense of providing more service capability.

5 Results

5.1 Implementation

The best way to prove that this design is applicable is by successfully applying it to a large-scaled company. The largest service-oriented company in Taiwan, ChungHwa Telecom Company, CHT, has more than 10,000 front-line staff members spread out over Taiwan. As of 2003, CHT had built up a pull-based broadcasting system where one server connects to 10,000 client devices. However, this had two drawbacks: a congested network and the ability to send only one downward message at once.

Finally, CHT decided to adopt a self-developed web-based broadcasting system to replace the original design. The present study proposes a novel design with push- and pull-based broadcasting algorithms. To study the acceptable number of servers for CHT, this work simulates the model proposed in section 4 for a period of one year in order to determine the average waiting time for broadcasting messages. The following

set of parameters is derived from the experience of the original design of CHT. The mean time of sending one message from a server to a destination is 0.0082 second, 217.4 downward messages need to be broadcast each day, the mean time between machine failure is $1/\alpha$= 1,094.4 hours, and the average repair time for each malfunction is $1/\beta$= 6.4 hours. That is, the probability of no server being in operational mode, equation (3), is small enough that it can be ignored, since $\alpha << \beta$. Moreover, the working time of the system is 24 hours a day, seven working days a week, and 52 weeks a year. To fit the organizational hierarchy, this work assumes that one server is set up at each center and at headquarters. The organization hierarchy of CHT has five layers, the first layer is the headquarters, the second layer has three regional administrative centers, the third layer has 36 regional centers, the fourth layer has 144 service centers, and the fifth layer has 10,000 front-end staff members. The present study simulates the model from two layers (the first layer and the fifth layer, making for one server for 10,000 clients) to five layers (including the servers from all five layers, resulting in 184 servers for10,000 clients) for a period of one year.

CHT's budget for this project was $1,100,000 US dollars. Experience tells us that the cost for installing an extra server is $5,882 US dollars per server and the cost of setting up the network environment for operating the UBS is $8,825 US dollars. This work proposes a simple survey to learn the users' utility. It presents four sets of simulated results and investment cost, from one server to 184 servers, to 184 executive managers for all of the units of CHT. In the present study, the managers are given 10 points each and they are asked to assign these 10 points among the users' utility. Table 1 shows the simulated average waiting time, real investment cost, and evaluated users' utility score.

Table 1. The table of average waiting time, investment cost and users' utility

Number of servers	The average waiting time to broadcast a message (seconds)	Investing cost (US dollars)	Users' Utility
2-tier architecture with 1 server and 10 000 clients	85.23	$ 14,707	1.31
3-tier architecture with 4 servers and 10,000 clients	35.34	$ 32,353	2.69
4-tier architecture with 40 servers and 10 000 clients	16.79	$ 244,105	4.44
5-tier architecture with 184 servers and 10 000 clients	11.90	$ 1,091,113	1.56

As the described by the above results, CHT decided to set up a four-tier broadcasting system with 40 servers and 10,000 client devices allocated based on the organizational hierarchy. Finally, the UBS successfully helped CHT send downward messages to front-line staff members for the "2005 Spring Multi-media Computer Exposition." The salespersons of CHT used different types of equipment to retrieve messages through the web-based application [17].

6 Conclusion and Remarks

Downward communication is an important way to send strategic messages to front-line staff members. However, to do this in an efficient manner is rather difficult to do, especially for a large-scaled company.

Although network technology can be utilized for broadcasting downward messages, simply using pull-based or push-based broadcasting algorithms is not adequate for a complex internetwork. A large-scaled company adopts a pull-based broadcasting algorithm to send downward messages to numerous client devices through one server only. However, the accompanying problems are a congested network traffic and an inefficient broadcasting mechanism.

This work proposed UBS, a robust web-based push- and pull-based broadcasting system. The largest telecom company in Taiwan, CHT, decided to utilize UBS as the broadcasting system and they adopted a four-layer architecture consisting of 40 servers serving over 10,000 client devices into 40 groups since January 2005.

However, it should be noted that our design has one issue that needs to be solved and requires further discussion. That is, if a new message is successfully sent to a sender, and if that sender then crashes immediately before the signal of "new message received" can be broadcast to its child senders, then that causes the problem that the new message can not be broadcast to the child senders. Because child senders are unable to be notified that a new message is coming, they also can not send any request to other senders to send the message again. Fortunately, the above situation has never happened since the system was implemented. The possible reason for this may be that the time interval is extremely short between the message being received completely to the signal being sent out to every child sender.

Acknowledgement. This study is supported by National Science Council, Taiwan, Republic of China, through the Project No. NSC 95-2416-H-008-028.

References

1. Agrawal, M., Rao, H. R., Sanders, G. L.: Impact of Mobile Computing Terminals in Police Work**.** J Organ Comput El Commer, 13(2003) 73-89.
2. Greenberg , J., Baron, R. A.: Behavior in Organizations, Allyn and Bacon Inc, MA (1993).
3. Défago, X., Schiper, A., and Urbán, P.: Total Order Broadcast and Multicast Algorithms; Taxonomy and Survey. ACM Comput Surv, 36 (2004) 372-421.
4. Herrería-Alonso, S., Suárez-González, A., Fernández-Veiga, M., Rubio, R. F. R., López-García, C.: Improving aggregate flow control in differentiated services networks. Comput Net, 44 (2004) 499-512.
5. Wang, W. M., Liang, C. C., Lu, H. Z., Chow, W. S., Chang, K. Y.: Research of Testing Process: The Case of TOPS-System Delivery Process. TL Tech J, 34(2004) 7-34.
6. Saxena, N., Pinotti, C. M., Das, S. K.: A Probabilistic Push-Pull Hybrid Scheduling Algorithm for Asymmetric Wireless Environment. IEEE conference GLOBALCOM (2004) 5-9.
7. Bhide, M., Deolasee, P., Katkar, A., Panchbudhe, A., Ramamritham, K., and Shenoy, P.: Adaptive Push-Pull: Disseminating Dynamic Web Data. IEEE Trans Comput 51 (2002) 652-668.

8. Blake, M. B., Fado, D. H., Mack, G. A.: A publicsh and subscribe collaborateon architecture for web based information. ACM conference WWW (2005) 1164-1165.
9. Rodeh, O.,Birman, K. P., Dolev, D.: Using AVL trees for fault-tolerant group key management. J Inform Sec, 1 (2002) 84-99.
10. Birman, K. P. and Renesse, R.: Reliable Distributed Computing with the Isis Toolkit. Wiley: IEEE Computer Society Press (1994).
11. Lu, H. Z., Liang, C. C., Chuan, C. C., Wang, W. M.: Discussion of TOPS/Order Software Dissemination, news publish and operation mechanism. TL Tech J, 42 (2005) 719-733.
12. Fich, F., Rparentt, E.: Hundreds of impossibility results for distributed computing. Dist Comput, 16 (2003) 121-163.
13. Davies, D. W.: An Historical Study of the Beginnings of Packet Switching. Comput J, 44 (2001) 152.
14. Androutsellis-Theotokis, S., Spinellis, D., A survey of peer-to-peer content distribution technologies. ACM Comput Surv, 36 (2004) 335-371.
15. Castro, M., Liskov, B.: Practical Byzantine Fault Tolerance and Proactive Recovery. ACM Trans Comput Syst, 20 (2002) 398-461.
16. Kamilla, K., Håkan, L., Lundberg, L., Svahnberg, C.,: Optimal recovery schemes in fault tolerant distributed computing, Acta Inf, 41(2005) 341-365.
17. Pascal, C. L.: Enabling Chance Interaction Through Instant Messaging**.** IEEE Trans Prof Commun, 46 (2003) 138-141.
18. Liang, C. C., Hsu, P. Y., Leu, J. D., Luh, H.,: Industrial Track: An effective approach for content delivery in an evolving intranet environment- a case study of the largest telecom company in Taiwan, Lect Notes Comput Sci, 3806 (2005) 740- 749.

Buffer-Preposed QoS Adaptation Framework and Load Shedding Techniques over Streams

Rui Zhou, Guoren Wang, Donghong Han,
Pizhen Gong, Chuan Xiao, and Hongru Li

Institute of Computer System, Northeastern University, Shenyang, China
wanggr@mail.neu.edu.cn

Abstract. Maintaining the quality of queries over streaming data is often thought to be of tremendous challenge since data arrival rate and average per-tuple CPU processing cost are highly unpredictable. In this paper, we address a novel buffer-preposed QoS adaptation framework on the basis of control theory and present several load shedding techniques and scheduling strategies in order to guarantee the QoS of processing streaming data. As the most significant part of our framework, buffer manager consisting of scheduler, adaptor and cleaner, is deliberately introduced and analyzed. The experiments on both synthetic data and real life data show that our system, which is built by adding several concrete strategies on the framework, outperforms existing works on both resource utilization and QoS assurance.

1 Introduction

Data stream applications such as network monitoring, on-line transaction flow analysis, intrusion detection and sensor networks pose tremendous challenges to traditional Database Management Systems (DBMSs). To Meet requirements of such scenarios, Data stream Management Systems (DSMSs) such as Aurora [7]/Borealis [8], STREAM [4], NiagaraCQ [10] and TelegraphCQ [11] are built for data management and query processing upon multiple, unbounded, continuous, time-varying input streams, which desire specific processing techniques different from fixed-size stored data sets.

One of the most significant characteristics of DSMSs is to provide assured QoS (such as tuple processing delay [1]) by load shedding [9] in order to cope with excessive incoming tuples and keep up with high-speed streams. In most scenarios, data arrival rates are fluctuating and unpredictable, and average per-tuple processing cost also varies due to inherent uncertainty of processing cost for different tuples, changes of queries (submitting new queries or cancelling old ones) or other urgent incoming tasks taking up part of CPU cycles. Consequently adaptivity is strongly demanded on building a DSMS to deal with these uncertainties and provide stable service for clients.

To achieve adaptivity, Aurora dynamically adjusts its shedding ratio (the fraction of tuples to be dropped) by plus or minus a step value. This solution may lead to overshoot or undershoot and also reacts slowly to bursty traffic. Tu [1]

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 234–246, 2006.

addressed a control-based adaptation framework and solved the above problem. However, drawbacks still exist when the stream arrival rate fluctuates frequently around CPU's top processing ability, for the reason that if those dropped tuples can be stored temporarily rather than discarded during high stream speed cycles and taken out to be processed when speed falls down, shedding ratio will be reduced and more tuples can be evaluated. Thus we propose a novel buffer-preposed framework to offer a better DSMS adaptation and elaborate the strategies required in constructing a buffer manager. Section 2 investigates related works. The framework and buffer manager are introduced in Section 3 and 4, respectively. Experiments and conclusions are given in Section 5 and 6.

2 Related Work

There has been considerable work on data stream processing. The survey in [12] gives an overview of stream work, and has summarized the issues of building a Data stream Management System. Specialized systems have been mentioned in Section 1. The existing load shedding works can be classified into two categories.

The first one focuses on specific operators (or queries). In [2,3,5,13], approximate joins were extensively studied to give MAX-subset [5] outputs of unbounded streams by dropping the tuples, which produce less join results. The work [6] mainly discussed about minimize the degree of inaccuracy while shedding load on aggregation queries. Our work considers about the other category [1,8,9] that integrates the queries as a whole forming a query network. Data stream Manage Systems monitor the QoS of the outputs, and make corresponding decisions on when, where and how much load to shed. Aurora/Borealis takes advantage of its LSRM to determine the target and amount for adaptation and inserts drop operators into its query network to reduce resource usage. In TelegraphCQ, synopses are built to capture the properties of the tuples, which are dropped from triage queues [15] in case of excessive stream load. Tu et al. [1] proposed a novel framework that regards DSMS as a plant and adjusts incoming flow rate according to current system status by leveraging control theory. Loadstar [14] introduces load shedding techniques to facilitate classifying multiple data streams of large volume and high speed. There are also other topics related to stream applications, such as search for moving object trajectories [16], comparison and contrast between Lp-Norm and edit distance [17], and location-aware topology matching in P2P systems [18].

3 Buffer-Preposed QoS Adaptation Framework

In this section, we propose a novel data stream processing system framework, which consists of two parts: upstream part and downstream part. The architecture is depicted in Figure 1. As for downstream part, query processor and CPU scheduler act as the fundamental evaluation components of DSMS, the same as operator network in Aurora/Borealis and STREAM. Monitor sends the QoS of output results to PI controller so that the controller could determine

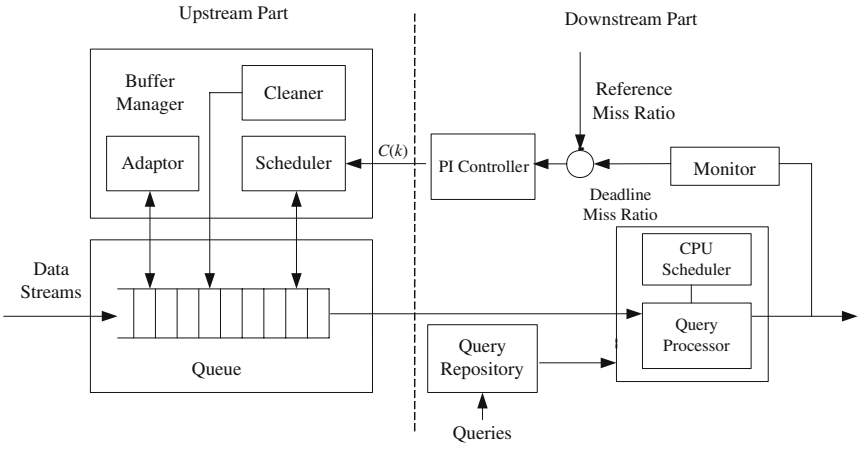


Fig. 1. Buffer-preposed QoS Adaptation Framework

the number of tuples ($C(k)$) to be injected into the query processor during the next monitoring cycle.

Applying control theory results in a better adaptation, and details can be found in [1]. However, drawbacks still exist when the stream arrival rate fluctuates frequently around CPU's top processing ability as is mentioned in Section 1. Hence, we prepose a buffer regulated by a buffer manager at the upstream part in front of former classic DSMS (the downstream part) to eliminate the problem. Three modules are built in buffer manager: scheduler allocates memory resources and dispatches the tuples in and out of their corresponding queues; adaptor will shed load if the queues are about to overflow; and cleaner is utilized to purge of those QoS-violated tuples. Detailed illustrations are given in Section 4. Note that all the strategies applied in the three modules are orthogonal to our framework, they may be replaced by better ones in the future.

In this paper, we take deadline miss ratio [1] as an example of QoS, note that other QoS metrics also work in our framework. Firstly, *tuple delay* is defined as time elapsed between the generation of the tuple at source and the end of its processing in query processor. A violation of tuple delay requirement is called *deadline miss*, and *deadline miss ratio* is the fraction of missing tuples of the entire data stream. Concerning that, tuple delay is differently defined in [26], whereas both of them could make sense and one can be simply transformed to the other.

4 Buffer Manager

4.1 Scheduler

Scheduler acts not only as a resource manager, but also a communicator between buffer and query processor. Its utility and functions are described as follows:

1. Dynamically allocate memory resource for the incoming streams according to their arrival rates and average per-tuple processing cost. It is believed that higher arrival rate, lower processing cost and higher tuple priority will result in a longer queue in memory. We preset a proper queue length limitation for each stream, and reallocate resources only when remarkable changes of system state take place, such as joining of a new stream, leaving of an old one, or tremendous speed varying of an existing stream, while on the common circumstance, we shed some load to deal with temporary stream speed fluctuation, which will be mentioned in Section 4.2.
2. Receive incoming tuples, dispatch them to their corresponding queue according to some strategies, such as FIFO(First In First Out), EDF [19](Earliest Deadline First), MUF [20](Maximum Urgent First), and pass the number of tuples required by DSMS from the queues to downstream part in terms of a suitable ratio at each sampling cycle. Our work mainly focuses on the situation where tuple priority is not explicit or difficult to determine, which accords with most application scenarios. Therefore FIFO, EDF are studied and compared as our candidate scheduling strategies, noting that other strategies also work in our framework.

4.2 Buffer Adaptor

It may be lack of resources for system to store and process every tuple of the incoming streams, especially when the streams have high arrival rates. We provide a queue with a proper maximum length to each stream. If the queue is about to overflow, we drop some tuples to reduce load in order to make sure DSMS is under a normal state. In this section, we mainly focus on single stream adaptation, and it is easy to be applied to multi-streams. There are many adaptation strategies to maintain the queue. In this section, we will discuss some of them.

Tail Drop, Drop Front, Random Drop. If queue length reaches its predefined maximum limitation, the queue will not permit entering of future tuples, and load shedding will be set up. One of the following strategies will be adopted. Tail drop [21] (referred to as TD) means dropping those new incoming tuples, i.e. dropping from the end of the queue, while drop front [23] means dropping from the front of the queue. Random drop [22] is also easy to understand (dropping a tuple in the queue randomly without regarding its position).

Note that the above three strategies are performed only when the queue is full, we consider this condition to be obviously deficient as there is no preserved room for future important tuples. Moreover if stream rate arises and lasts for a period of time, late arriving tuples will most likely be dropped, which will lead to unfairness to some extent. RED′ and PID can avoid this problem.

RED′. RED(Random Early Detection) [24] is dramatically studied for Active Queue Management(AQM)in the field of computer network. Since strategies in each module are orthogonal with the framework, adaptor can be implemented with any strategy. Now we will simply introduce RED′ adaptation scheme, a variation of RED. The average queue length is calculated by:

$$\tilde{q} = (1 - w_q) \cdot \tilde{q} + w_q \cdot q \tag{1}$$

where q is the current queue length and $\tilde{q}$ is the historical average length. w_q is weight factor, $w_q \in [0,1]$. If $\tilde{q} < min_{th}$ (min_{th}: predefined minimum length of queue), all arriving tuples will enter into the queue; if $\tilde{q} > max_{th}$ (max_{th}: predefined maximum length of queue), all these incoming ones will be dropped ; and if $min_{th} \leq \tilde{q} \leq max_{th}$, the shedding ratio p will be given in Equation 2, where max_p is a maximum shedding ratio.

$$p = max_p \cdot (\tilde{q} - min_{th})/(max_{th} - min_{th}) \tag{2}$$

PID. PID [25] (Proportional, Integral and Differential) control is widely used in automatic control and system engineering. First, we define *occupying ratio* as *the number of tuples in queue divided by maximum queue length*. In order to reserve some space for future tuples, we assign an expected *occupying ratio* (denoted as r) as the reference and try to control the queue length to be $r \cdot L$. Symbols of this section are listed in Table 1.

Table 1. Symbols used in PID Adaptation

L	maximum queue length
Y	output signal: occupying ratio of queue
X_{in}	incoming rate of tuples entering the queue
X_{out}	output rate of tuples (i.e. CPU processing ability)
X	queue length increment (i.e. X_{in}-X_{out})
S	stream rate
U	load shedding ratio

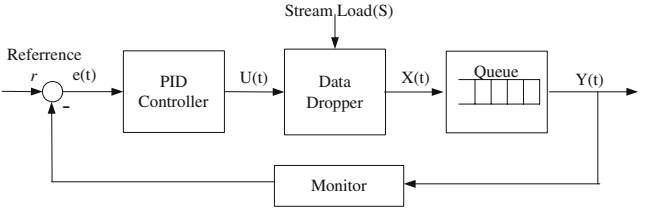


Fig. 2. The Feedback Control Loop

The blocks of closed-loop feedback control are given in Figure 2. As to queue, the plant, it is easy to obtain the following difference equation, which is utilized to model the variation of occupying ratio of the queue,

$$Y(k) = Y(k-1) + \frac{X(k)}{L} \tag{3}$$

and the transfer function in Z-domain is:

$$G(z) = \frac{Y(z)}{X(z)} = \frac{1}{L(1 - z^{-1})} \tag{4}$$

With respect to PID controller, the input signal is calculated as the following:

$$X(k) = K_P \cdot e(k) + K_I \sum_{j=0}^{k} e(j) + K_D \left[e(k) - e(k-1)\right] \tag{5}$$

and we can get controller transfer function:

$$C(z) = \frac{X(z)}{E(z)} = K_P + K_I \frac{1}{1 - z^{-1}} + K_D(1 - z^{-1}) \tag{6}$$

After analyzing the system's closed-loop transfer function (see Equation 7) by means of Root Locus or Frequency Response (available in MATLAB), we conclude that the system is marginally stable, and thus controllable. Parameters K_P, K_I, K_D can be determined through real experiments.

$$T(z) = \frac{C(z)G(z)}{1 + C(z)G(z)} \tag{7}$$

Considering the data dropper and the queue as a whole, we have:

$$X(k) = X_{in}(k) - X_{out}(k) = (1 - U(k)) \cdot S(k) - X_{out}(k) \tag{8}$$

and thus the controller passes the shedding ratio $U(k)$ to data dropper. $U(k)$ can be deduced from Equation 8: $U(k) = 1 - \frac{X(k)+C(k)}{S(k)}$, where $C(k)$ equals to $X_{out}(k)$, and $S(k)$, $C(k)$ can be predicted by $S(k-1)$ and $C(k-1)$ obtained in the last sampling period.

4.3 Cleaner

Cleaner detects the tuples in queues that will not satisfy the QoS of the queries submitted in DSMS if passed to CPU to be processed. For instance, if possible out-of-date tuples which may miss its deadline can be removed from queues by cleaner, CPU cycles will be saved to produce useful results. As is mentioned in previous sections, we adopt deadline miss ratio as the expected QoS of DSMS, and then the cleaning strategy should be invalidating (the same as removing) the tuples whose permitted tuple delay will be probably violated. Now we will introduce the rule of removing a possible outdated tuple. Table 2 gives out the variables used in the following subsections, here t stands for timestamp, and T stands for time period.

Cleaning Strategy. At the kth monitoring cycle, the following condition should be satisfied if a tuple is supposed to be purged.

$$T_t^i + T_w^i(k) + T_p^i(k) > T_d \tag{9}$$

Here, T_t^i can be determined by $T_t^i = t_a^i - t_g^i$, note that t_a and t_g can be obtained when the corresponding tuple arrives. As average per-tuple processing cost may be variable in different monitoring cycles, we use $T_w^i(k)$ to associate T_w^i with cycle information k (also the same as to $T_p^i(k)$ and T_p^i) to make a more accurate estimation of waiting time. Details of estimating of $T_w^i(k)$ and $T_p^i(k)$ are presented in the next subsections.

Table 2. Variables used in Cleaning Strategies

t_g	generation timestamp of a tuple (at source)
t_a	arrival timestamp of a tuple (at destination)
T_t	transmission time of a tuple
T_w	waiting time of a tuple before it can be processed
T_p	processing time of a tuple in query networks
T_d	tuple delay predefined by query properties or users
$T_p^i(k)$	processing time of the ith tuple in queue at the kth monitoring cycle
$ET_w^i(k)$	estimated waiting time of the ith tuple in queue at the kth monitoring cycle

Estimation of $T_p^i(k)$. Several reasons that may result in fluctuations of average per-tuple processing cost have been given in Section 1. Therefore, we are supposed to formulate an estimation of $T_p^i(k)$, the predicted cost of a tuple in the kth monitoring cycle, to help judge if the tuple can be invalidated.

We utilize Single Exponential Smoothing to estimate the processing cost, and the formula is given below:

$$ET_p^i(k) = \alpha T_p^i(k-1) + (1-\alpha) ET_p^i(k-1) \tag{10}$$

where $0 < \alpha \leq 1$ and $k \geq 2$. $T_p^i(k-1)$ is the real average per-tuple processing cost in $(k-1)$th monitoring cycle, while $ET_p^i(k-1)$ is the estimated one. $ET_p^i(k)$ is the weighted sum of $T_p^i(k-1)$ and $ET_p^i(k-1)$. Weight α is called smoothing constant. When α is close to 1, latest processing ability dominates the estimation of $ET_p^i(k)$, while when α is close to 0, historical processing cost plays a more important role. We can set α according to specific application scenarios and a proper value can be found through a series of extensive experiments.

Estimation of $T_w^i(k)$. Note that, as for a certain tuple, there exists a long period of time during which the tuple is waiting in the queue after arrival in advance of the time being processed, it is crucial to perform an estimation of the waiting time. First we have:

$$T_w^i(k) = T_w^{i-1}(k) + T_p^{i-1}(k) \tag{11}$$

The waiting time of the ith tuple equals to the sum of waiting time and processing time of the $(i-1)$th tuple (suppose the ith and $(i-1)$th tuple are in the same processing cycle). Substitute $ET_w^i(k)$ and $ET_w^{i-1}(k)$ for $T_w^i(k)$ and $T_w^{i-1}(k)$ in Equation 11, we get:

$$ET_w^i(k) = ET_w^{i-1}(k) + T_p^{i-1}(k) \tag{12}$$

After recurrence substitution, we have:

$$ET_w^i(k) = \sum_{j=1}^{i-1} T_p^j(k) \tag{13}$$

Assume that estimated per-tuple processing cost in the kth monitoring cycle are the same, denoted as $ET_p(k)$, Equation 13 can be reduced to Equation 14.

$$ET_w^i(k) = (i-1)ET_p(k) \tag{14}$$

Summary of Estimation. Replacing $T_w^i(k)$ and $T_p^i(k)$ with $ET_w^i(k)$ and $ET_p^i(k)$ in Equation 9, and also combining with Equation 10 and Equation 14, we can get the final conclusion, i.e. the formula of cleaning strategy:

$$T_t^i + i \cdot [\alpha T_p(k-1) + (1-\alpha)ET_p(k-1)] > T_d \tag{15}$$

Cleaning is performed from queue front to queue end, as a result, if a tuple is removed, its processing cost will not add in the estimation of waiting time of its successive tuple. Hence the estimation is probably accurate, and cleaner can reduce excessive stream load gracefully and effectively.

5 Experimental Results

To assess the practical performance of our model, we perform several sets of experiments on both synthetic and real life datasets. First we test our framework with various buffer scheduling strategies, adapting schemes, and cost estimation precisions. After certifying the validity of our framework, we pick a set of concrete strategies and build an adaptation system, referred to as CWB (Control With Buffer), and compare its performance with that of the system mentioned in [1], referred to as COB (Control withOut Buffer). The results show that CWB outperforms COB, achieving higher CPU utilization and lower deadline miss ratio, especially when stream arrival rate fluctuates wildly.

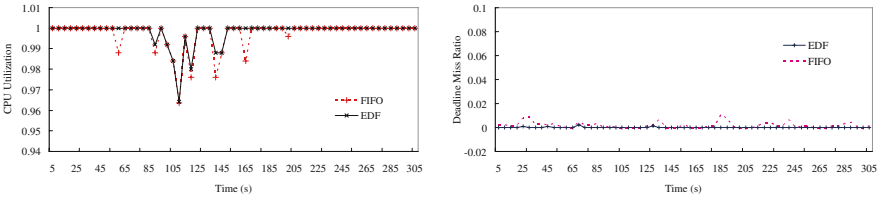
For synthetic data, we generate the tuples with arrival time following exponential distribution, i.e. in each time interval, the number of arriving tuples follows Poisson distribution. For real life data, we use the LBL-PKT-4 dataset from Internet Traffic Archive [27]. Monitoring cycle of downstream part (CPU) and upstream part (buffer) are 5s, 1s, respectively. Tuple delay is randomly chosen from 250ms, 500ms, 1s, 2s. Maximum queue length of incoming stream is set to be 150 tuples. We construct the downstream part according to [1] as stated in Section 3. Our discussion is mainly about the strategies in buffer manager (the upstream part) and comparison between CWB and COB.

5.1 Experiments on Buffer Manager

Buffer manager includes scheduler, adaptor and cleaner. For cleaner, we determine the smoothing constant α as 0.1 through experiments, which obtains a nearly optimal prediction of average per-tuple processing cost.

Scheduling Strategies. As for scheduler, FIFO and EDF as tuple transmission strategies are thoroughly studied and compared. Here, we use synthetic dataset, permitted deadline miss ratio 0.01, and the simplest TD strategy for adaptor.

Figure 3 shows that EDF leads to higher CPU utilization and lower deadline miss ratio. That is because some miss-prone tuples can be laid at front and processed earlier by EDF, and thus deadline misses are reduced. Meanwhile this optimization could result in cleaning less tuples, which can reduce the load



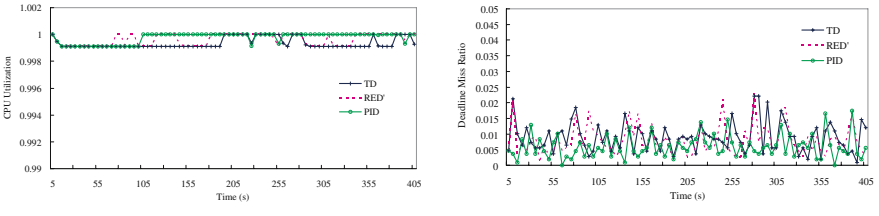
(a) CPU Utilization (b) Deadline Miss Ratio

Fig. 3. Performance on FIFO and EDF

shedding ratio, i.e. increase the CPU utilization. Therefor EDF rather than FIFO is used in the following experiments.

Adaptation Strategies. As for adaptor, we perform the comparison among TD, RED′ and PID. We set the parameters in Equation 1 and 2 as: $w_p = 0.15$, $min_{th} = 30$, $max_{th} = 120$, $max_p = 0.1$ and the parameters in Equation 5 as: $K_P = 30$, $K_I = 18$, $K_D = 7.5$, the reference occupying ratio 0.5. System load is l.2 times of CPU processing ability.

As is shown in Figure 4, PID performs the best while TD the worst, and RED′ is also acceptable but exhibits only a second choice. Figure 5 gives the frequencies of queue occupying ratio under three adapting strategies during 1000s



(a) CPU Utilization (b) Deadline Miss Ratio

Fig. 4. Performance on TD, RED′ and PID

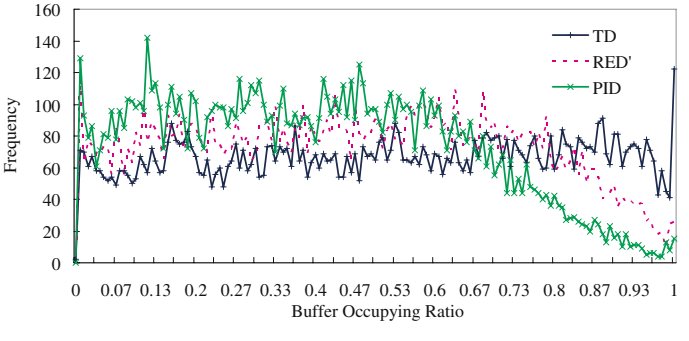
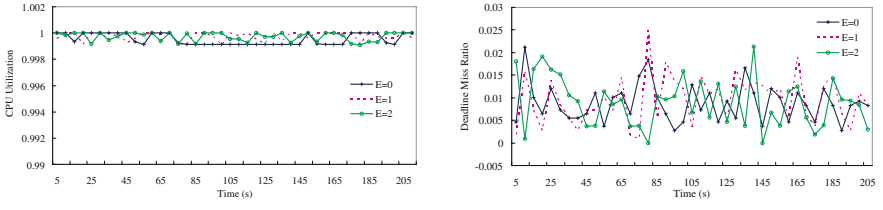


Fig. 5. Frequency of Buffer Occupying Ratio

execution. Here sampling cycle is 0.1s. The experimental result shows PID effectively reduces the occupying ratio of queue and provides reserved space for future tuples. Like EDF, PID is used exclusively in the following experiments.

Precisions of Cost Estimation. We also investigate the performance of our framework with respect to different precisions of cost estimation. Let E stand for the precision level. E=0, $T_p = ET_p$; E=1, $T_p \in [0.3ET_p, 1.7ET_p]$; E=2, $T_p \in [0.1ET_p, 4.1ET_p]$. T_p, ET_p are real tuple processing cost and average tuple processing cost, respectively.



(a) CPU Utilization (b) Deadline Miss Ratio

Fig. 6. Different Precisions of Cost Estimation

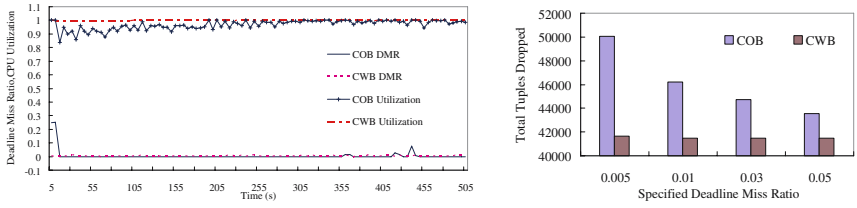
From figure 6, we can draw the conclusion that, though the fluctuation of CPU utilization and deadline miss ratio arises as E increases, the affects are not prominent and system is capable to learn the variation and works stably and robustly. E is set to 0 unless particularly specified in other experiments.

5.2 Performance of CWB vs. COB

After discussing about buffer manager, we compare CWB and COB with parameters set according to Section 5.1. From figure 7(a), we know that CWB reacts faster, while COB converges to a steady state after 15 seconds. Although CWB undulates frequently, its fluctuating amplitude is trivial and can be ignored, whereas COB exhibits the opposite. As to CPU utilization, CWB can make full use of CPU resources for the reason that buffered data can be passed to the downstream part if CPU becomes idle. Figure 7(b) illustrates that dropped tuples of CWB is much fewer than COB, which means CWB has processed more tuples. This is accorded with higher CPU utilization of CWB. As the expected deadline miss ratio arises, fewer tuples are dropped. The decrease of CWB is not obvious due to effectiveness of cleaning strategy, for only those tuples prone to miss their deadlines are doomed to be removed.

5.3 Experiments on Real Life Datasets

As is shown in Figure 8, the performance of CWB is much better than that of COB contrasting to the result on synthetic dataset. That is because the arrival rate of LBL-PKT-4 TCP Traffic is more fluctuating than Poisson distribution data. Our framework provides a more adaptive solution in real life applications.



(a) CPU Utilization and Deadline Miss Ratio (b) Total Tuples Dropped

Fig. 7. Performance on CWB and COB

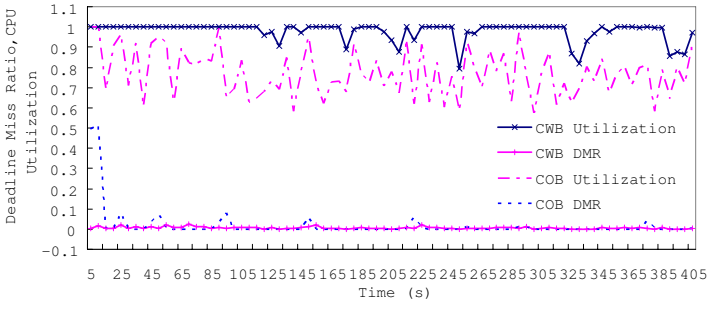


Fig. 8. Experimental Result on Real Life Dataset

6 Conclusions and Future Work

In this paper, we propose a novel QoS adaptation framework including upstream part and downstream part. In the upstream part, buffer manager including scheduler, adaptor and cleaner is deliberately introduced and analyzed. The experiments on both synthetic data and real life data show that our system, which is built by adding several concrete strategies on the framework, outperforms existing works on both resource utilization and shedding ratio. A promising direction for future work is to consider the priority of the tuples, i.e. incoming data are of different importance. On the circumstance, we are supposed to find out a set of specific strategies for scheduler, adaptor and cleaner.

Acknowledgement. This work is partially supported by National Natural Science Foundation of China under grant No. 60573089 and 60473074 and supported by Natural Science Foundation of Liaoning Province under grant no. 20052031.

References

1. Yi-Cheng Tu, Mohamed Hefeeda, Yuni Xia, and Sunil Prabhakar. Control-based Quality Adaptation in Data Stream Management Systems. In *Proceedings of DEXA, pages 746-755*, August 2005.
2. J. Kang, J. F. Naughton, and S. D. Viglas. Evaluating window joins over unbounded streams. In *Proc. of ICDE*, Bangalore, India, March 2003.

3. J.Xie, J. Yang, and Y. Chen. On joining and caching stochastic streams. In *Proc. 2005 ACM SIGMOD Conf.*, Baltimore, Maryland, USA, June 2005.
4. The STREAM Group. STREAM: The Stanford Stream Data Manager. *IEEE Data Engineering Bulletin* ,26(1):19-26,March 2003.
5. A. Das, J. Gehrke, and M. Riedewald. Approximate Join Processing Over Data Streams. In *Proc. 2003 ACM SIGMOD Conf.*, June 2003.
6. B. Babcock, M. Datar, and R. Motwani. Load Shedding for Aggregation Queries over Data Streams. In *Proc. 2004 Int. Conf. on Data Engineering*, Feb. 2004.
7. D. Abadi, D. Carney, et al. Aurora: a new model and architecture for data stream management. *VLDB Journal*, Vol.12(2),pp.120-139,2003.
8. D. Abadi, Y. Ahmad, M. Balazinska, U. Cetintemel, M. Cherniack, J. Hwang, W. Lindner, A. Maskey, A. Rasin, E. Ryvkina, N. Tatbul, and S. Zdonik. The Design of the Borealis Stream Processing Engine. In *Procs. of CIDR*, Jan. 2005.
9. N. Tatbul, U. Cetintemel, S. Zdonik. M. Cherniack, M. Stonebraker. Load Shedding in a Data Stream Manager. In *Proc. 29th int. Conf. on VLDB*, Sep. 2003.
10. J. Chen, D. J. DeWitt, F. Tian, and Y. Wang. NiagaraCQ: A scalable continous query system for internet databasses. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 379-390, 2000.
11. S. Chandrasekaran, A. Deshpande, M. Franklin, et al. TelegraphCQ: Continuous Dataflow Processing for an Uncertain World. In *Proc. of CIDR*, Jan. 2003.
12. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proc. Principles of Database Systems (PODS)*, June 2002.
13. D. Han, R. Zhou, C. Xiao. G. Wang. *et al.* Load shedding for Window Joins over Data Streams. In *Proc. Int. Conf. on WAIM*, Hongkong, June 2006,
14. Yun Chi, Haixun Wang, and Philip S. Yu. LoadStar: Load Shedding in Data Stream Mining. In *Proc. Of the 31st VLDB Conf.*, pages 1302-1305, August 2005.
15. Frederick Reiss and Joseph M. Hellerstein. Data Triage: An Adaptive Architecture for Load Shedding in TelegraphCQ. In *Proc. of ICDE*, pages 155-156, April 2005.
16. L. Chen, M. T. Ösu, and V. Oria, Robust and Fast Similarity Search for Moving Object Trajectories. In *Proceedings of 24th ACM International Conference on Management of Data (SIGMOD'05)*, Baltimore, June 2005, pages 491-502.
17. L. Chen, R. Ng. On the Marriage of Lp-Norm and Edit Distance. In *Proc. of VLDB*, Toronto, Canada, August 2004, pages 792-803.
18. Y. Liu , X.Liu , L. Xiao , Li. Ni, and X. Zhang. Location-Aware Topology Matching in P2P Systems. *IEEE INFOCOM*, 2004
19. Abdelzaher, T., Sharma, V., Lu, C. A Utilization Bound for Aperiodic Tasks and Priority Driven Scheduling. *IEEE Trans. on Computers*, 53, 2004, 334-350.
20. D. B. Stewart and P. K. Khosla, Real-Time Scheduling of Sensor-Based Control Systems, Real-Time Programming, ed. by W. Halang and K. Ramamritham, (Tarrytown, New York: Pergamon Press Inc.), 1992.
21. S. Floyd, V. Jacobson. Traffic phase effects in packet-switched gateways. ACM Comp. Commun. Rev., Vol.21, No.2, pp.26-42, Apr. 1991.
22. Hashem E S. Analysis of random drop for gateway congestion control. MIT Lab for Computer Science : Technical Report, MIT 1989.
23. T Lakshman, A Neidhardt, and T Ott. The drop from front strategy in TCP and in TCP over ATM. In *IEEE INFOCOM*, San Francisco, CA, 1996. pp.1242-1250
24. S. Floyd and V. Jacobson. Random Early Detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, August 1997.

25. G. F. Franklin, J. D. Powell, and A. Emami-Naeini. Feedback Control of Dynamic Systems. Prentice Hall, Massachusetts, 2002.
26. Yi-Cheng Tu, Liu Song, Sunil Prabhakar. Load Shedding in Stream Databases: A Control-Based Approach. March 2006. Technical report, Purdue University.
27. V. Paxson and S. Floyd. Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, 3(3), pp. 226-244, June 1995. http://ita.ee.lbl.gov/html/contrib/LBL-PKT.html

Cardinality Computing: A New Step Towards Fully Representing Multi-sets by Bloom Filters*

Jiakui Zhao, Dongqing Yang, Lijun Chen, Jun Gao, and Tengjiao Wang

School of EECS, Peking University, Beijing 100871, China
{jkzhao, dqyang, ljchen, gaojun, tjwang}@pku.edu.cn

Abstract. Bloom Filters are space and time efficient randomized data structures for representing (multi-)sets with certain allowable errors, and are widely used in many applications. Previous works on Bloom Filters considered how to support insertions, deletions, membership queries, and multiplicity queries over (multi-)sets. In this paper, we introduce two novel algorithms for computing cardinalities of multi-sets represented by Bloom Filters, which extend the functionality of the Bloom Filter and thus make it usable in a variety of new applications. The Bloom structure presented in the previous work is used without any modification, and our algorithms have no influence to previous functionality. For Bloom Filters support cardinality computing in addition to insertions, deletions, membership queries, and multiplicity queries simultaneously, our work is a new step towards fully representing multi-sets by Bloom Filters. Performance analysis and experimental results show the difference of the two algorithms and show that our algorithms perform well in most cases.

1 Introduction

Insertions, deletions, membership queries, multiplicity queries, and cardinality computing are five important operations and queries over multi-sets. In order to represent multi-sets as fully as possible by sketches, the sketches must support the five operations and queries simultaneously. Bloom Filters are space and time efficient randomized data structures for representing (multi)-sets with certain allowable errors, and previous works [1], [2], [3] considered how to support insertions, deletions, membership queries, and multiplicity queries. For fully representing multi-sets, Bloom Filters must support cardinality computing in addition to previous considerations. In this paper, we consider cardinality computing and introduce two algorithms (*FCount* and *BFCDist*) to compute the cardinalities of multi-sets represented by Bloom Filters. The *FCount* algorithm uses membership queries to count the distinct elements over multi-sets, and the *BFCDist* algorithm uses pure probabilistic method to compute the cardinalities. Our work extends the functionality of the Bloom Filter and thus make it usable in a variety of new applications. From functionality extension's point of view, our work can be seen as a new step towards fully representing multi-sets by Bloom Filters.

* Supported by State Key Laboratory of Networking and Switching Technology, NSFC Grant 60473051 and 60503037, and NSFBC Grant 4062018.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 247–258, 2006.

1.1 Previous Bloom Filters

Standard Bloom Filters. B. H. Bloom [1] introduced the Bloom Filter in 1970, which is called the *Standard Bloom Filter* in this paper. It uses k hash functions $h_1, h_2, \ldots, h_k$ to hash elements into a bit vector V of size m, and supports insertions and membership queries over (multi-)sets. Initially, all bits are set to 0; for each inserted element e, the bits at positions $h_1(e), h_2(e), \ldots, h_k(e)$ are set to 1. For an element e, its membership can be checked by examining the bits at positions $h_1(e), h_2(e), \ldots, h_k(e)$. An element e is reported to be contained in the set iff all the k bits had been set to 1. This method allows a small probability of producing false positive error (a positive result may be returned for an element e which actually is not contained in the set), but no false negative error.

Counting Bloom Filters. L. Fan et al. [2] introduced an extension of the Standard Bloom Filter for supporting deletions in 1998, which is called the *Counting Bloom Filter* in this paper. It replaces the bit vector V by a counter vector C. Initially, all counters are set to 0; an insertion (deletion) of an element e will increase (decrease) the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ by 1. Membership checking is the same as that in the Standard Bloom Filter, except that an element e is reported to be contained in the set iff the k counters are all not zero-valued. To maintain the compactness of the Counting Bloom Filter, the counters are limited to 4 bits, which is shown to be statistically enough to encode the number of the elements hashed to the same position. However, Counting Bloom Filters are not adequate when dealing with multi-sets in which identical elements may appear hundreds and thousands of times.

Spectral Bloom Filters. Saar and Matias [3] introduced the *Spectral Bloom Filter* for supporting multiplicity queries over multi-sets in 2003. The algorithms for evaluating multiplicity queries are *Minimum Selection*, *Minimal Increase*, and *Recurring Minimum*. In *Minimum Selection*, things are the same as that in the Counting Bloom Filter; for an element e, the minimum of the values of the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ is selected as the multiplicity. In *Minimal Increase*, when performing an insertion of an element e, only increasing the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ whose values are the minimum of the k counters. *Minimal Increase* has lower error rate and lower error size; unfortunately, it does not support deletions. *Recurring Minimum* is similar to *Minimum Selection* except that it uses a two-level Spectral Bloom Filter to reduce the error rate. After performing an insertion (deletion) of an element in the Primary Spectral Bloom Filter, if it has single minimum among the k counters, performing the insertion (deletion) in the Secondary Spectral Bloom Filter. When evaluating a multiplicity query for an element, if it has recurring minimum in the Primary Spectral Bloom Filter, return the minimum; otherwise, performing the evaluation in the Secondary Spectral Bloom Filter. Since through observations, the elements which have single minimum in the Primary Spectral Bloom Filter have high error rate in evaluating multiplicity queries. In [3], a data structure called *string-array index* is used to maintain the counter vector, which has good space, access, and query performance, and supports multi-sets in which identical elements may appear hundreds and thousands of times.

1.2 Paper Outline

The rest of this paper is structured as follows. In Section 2, we introduce the preliminary knowledge which is important for the introduction of the cardinality computing algorithms. In Section 3, we introduce the algorithm *FCount* and *BFCDist* for computing cardinalities over multi-sets. Section 4 presents the applications of the cardinality computing algorithms. Experimental results are presented in Section 5, followed by our discussions and conclusions in Section 6.

2 Preliminary Knowledge

In order to represent multi-sets by Bloom Filters and support insertions, deletions, membership queries, and multiplicity queries, we must maintain counter vectors. In case of using k hash functions to hash elements into a counter vector of m counters $C_1, C_2, \ldots, C_m$, where C_i is the number of the elements hashed into the ith counter, the goal is to have a compact encoding of the counter vector whose size is as close to $\sum_{i=1}^{m} \lceil \log_2 C_i \rceil$ bits as possible. The *string-array index* presented in [3] is a good choice for maintaining the counter vector. Let $M = \sum_{i=1}^{m} \lceil \log_2 C_i \rceil$, we can encode the counter vector into a *string-array index* of $M + o(M) + O(m)$ bits, and insertions, deletions, membership queries, and multiplicity queries take each $O(1)$ expected amortized time. In the rest of this paper, "counter vector" refers to a counter vector maintained by the *string-array index*, and we will not consider the detailed data structure of the counter vector anymore during the introduction of the cardinality computing algorithms. Insertions, deletions, membership queries, and multiplicity queries can be executed over *string-array index* maintained counter vectors as described in Section 1.1.

The probability for a false positive error of membership queries is dependent on the number of the counters and the number of the hash functions. If n distinct elements were uniformly hashed into a counter vector of size m by k hash functions, the probability that a particular counter is still zero-valued is $(1 - \frac{1}{m})^{kn}$. Equation 1 characterizes the false positive error rate of the membership queries.

$$E_{rror}R_{ate} = (1 - (1 - \frac{1}{m})^{kn})^k \approx (1 - e^{-\frac{kn}{m}})^k \tag{1}$$

The value of $(1 - e^{-\frac{kn}{m}})^k$ is minimized for $k = \ln(2)\frac{m}{n}$, in which case the error rate is $(\frac{1}{2})^k \approx (0.6185)^{\frac{m}{n}}$, and each counter has a $\frac{1}{2}$ probability of being zero-valued. Thus, Bloom Filters are highly effective even for $m = \psi n$ using a small constant ψ. Let $\gamma = \frac{nk}{m}$, note that $\gamma = \ln(2) \approx 0.7$ in the optimal case. Table 1 shows the false positive error rate under optimal $\frac{m}{n}$ and k combinations.

3 Cardinality Computing Algorithms

In this section, we introduce two algorithms for counting the distinct elements over multi-sets. *BFCDist* (*B*loom *F*ilter *C*ount *Dist*inct) algorithm uses pure probabilistic method to count the distinct elements. *FCount* (*F*ilter and *Count*) algorithm uses membership queries to count the distinct elements.

Table 1. False positive error rate of membership queries in case of hashing n distinct elements into m counters by k hash functions under the optimal $\frac{m}{n}$ and k combinations

$\frac{m}{n}$	02	03	04	05	06	07
optimal k	01	02	03	04	04	05
error rate	$3.93 * 10^{-1}$	$2.37 * 10^{-1}$	$1.47 * 10^{-1}$	$9.20 * 10^{-2}$	$5.61 * 10^{-2}$	$3.47 * 10^{-2}$
$\frac{m}{n}$	08	09	10	11	12	13
optimal k	06	06	07	08	08	09
error rate	$2.16 * 10^{-2}$	$1.33 * 10^{-2}$	$8.19 * 10^{-3}$	$5.09 * 10^{-3}$	$3.14 * 10^{-3}$	$1.94 * 10^{-3}$

3.1 BFCDist Algorithm

Probabilistic Model. We use k hash functions $h_1, h_2, \ldots, h_k$ to hash elements of a multi-set $\hbar$ into a counter vector of size m. Initially, all the m counters are set to zero; an insertion (deletion) of an element e will increase (decrease) the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ by 1. The model assumes that each hash function uniformly distributes the hashed elements over the m counters.

Theorem 1. *If m_0 denotes the number of the zero-valued counters among the m counters of the counter vector, $|\hbar|$ can be characterized by Equation 2.*

$$|\hbar| = \frac{1}{k} \cdot \frac{\ln m - \ln E(m_0)}{\ln m - \ln(m-1)} \tag{2}$$

Proof. If $\hbar$ contains N elements currently, the probability that a random selected counter c of the counter vector is zero-valued, $\mathbb{P}(c = 0)$ can be characterized by Equation 3; the expected value of m_0, $E(m_0)$ can be characterized by Equation 4. The reason why the exponential value in Equation 3 and Equation 4 is not kN but $k|\hbar|$ is that identical elements will be hashed into identical counters, and they contribute to whether a given counter is zero-valued or not "only once". Deducing Equation 2 from Equation 4 completes the proof of the theorem.

$$\mathbb{P}(c = 0) = \left(1 - \frac{1}{m}\right)^{k|\hbar|} \tag{3}$$

$$E(m_0) = m\left(1 - \frac{1}{m}\right)^{k|\hbar|} \tag{4}$$

Algorithm Description. Equation 5 can be easily deduced from Equation 2 by replacing $E(m_0)$ by m_0, where $\widetilde{|\hbar|}$ is the approximate value of $|\hbar|$. For the value of m and k are known, if we trace the value of m_0 during the insertions and deletions of the elements (the *online BFCDist* algorithm), computing the value of $\widetilde{|\hbar|}$ takes $O(1)$ expected amortized time; otherwise (the *offline BFCDist* algorithm), computing the value of $\widetilde{|\hbar|}$ takes $O(m)$ expected amortized time.

$$\widetilde{|\hbar|} = \lceil \frac{1}{k} \cdot \frac{\ln m - \ln m_0}{\ln m - \ln(m-1)} \rfloor \tag{5}$$

Error Estimation. Equation 6 characterizes the mean variance of m_0, $\sigma^2(m_0)$. The standard deviation of m_0, $\sigma(m_0)$ characterizes the error of approximating $E(m_0)$ by m_0. In order to compute the value of $\sigma(m_0)$, the probability that there are exactly μ $(m - \min(m, k|\hbar|) < \mu < m)$ zero-valued counters among the m counters of the counter vector, $\mathbb{P}(m_0 = \mu)$ must be computed in advance.

$$\sigma^2(m_0) = \sum_{\mu=1}^{m}(\mu - E(m_0))^2\mathbb{P}(m_0 = \mu) = \sum_{\mu=1}^{m}\mu^2\mathbb{P}(m_0 = \mu) - E^2(m_0) \tag{6}$$

Lemma 1. *Equation 7 characterizes the number of the ways of distributing ξ distinct balls into τ distinct boxes with no box empty, $\Psi(\xi, \tau)$ $(\xi \geq \tau \geq 1)$.*

$$\Psi(\xi, \tau) = \tau^{\xi} + \sum_{i=1}^{\tau-1}(-1)^i\binom{\tau}{i}(\tau - i)^{\xi} \tag{7}$$

Proof. Let Λ_i $(1 \leq i \leq \tau)$ denote the set of the ways of distributing ξ distinct balls into τ distinct boxes and the ith box must be empty after the distribution. $|\bigcup_{i=1}^{\tau}\Lambda_i|$ denotes the number of the ways of distributing ξ distinct balls into τ distinct boxes with at least one box empty, which can be characterized by the *Inclusion-Exclusion Principle* as $|\bigcup_{i=1}^{\tau}\Lambda_i| = \sum_{i=1}^{\tau-1}(-1)^{i-1}\binom{\tau}{i}(\tau - i)^{\xi}$.[1] The number of the ways of distributing ξ distinct balls into τ distinct boxes is τ^{ξ}. Therefore, the number of the ways of distributing ξ distinct balls into τ distinct boxes with no box empty, $\Psi(\xi, \tau)$ $(\xi \geq \tau \geq 1)$ can be characterized as $\Psi(\xi, \tau) = \tau^{\xi} - |\bigcup_{i=1}^{\tau}\Lambda_i| = \tau^{\xi} - \sum_{i=1}^{\tau-1}(-1)^{i-1}\binom{\tau}{i}(\tau - i)^{\xi} = \tau^{\xi} + \sum_{i=1}^{\tau-1}(-1)^{i}\binom{\tau}{i}(\tau - i)^{\xi}$.

Theorem 2. *Equation 8 characterizes the probability that there are exactly μ $(m - \min(m, k|\hbar|) < \mu < m)$ zero-valued counters among the m counters of the counter vector, $\mathbb{P}(m_0 = \mu)$.*

$$\mathbb{P}(m_0 = \mu) = \binom{m}{\mu}\left(\left(1 - \frac{\mu}{m}\right)^{k|\hbar|} + (-1)^{m-\mu}\sum_{i=1}^{m-\mu-1}(-1)^i\binom{m-\mu}{i}\left(\frac{i}{m}\right)^{k|\hbar|}\right) \tag{8}$$

Proof. The number of the ways of distributing $k|\hbar|$ distinct hashes into m distinct counters is $m^{k|\hbar|}$. The number of the ways of selecting $m - \mu$ counters from m distinct counters is $\binom{m}{\mu}$, and the number of the ways of distributing the $k|\hbar|$ distinct hashes into the selected $m - \mu$ distinct counters with no counter "empty" is $\Psi(k|\hbar|, m - \mu)$. Therefore, the probability that there are exactly μ zero-valued counters among the m counters of the counter vector, $\mathbb{P}(m_0 = \mu)$ can be characterized by $\frac{\binom{m}{\mu}\Psi(k|\hbar|, m-\mu)}{m^{k|\hbar|}}$, which equals to the right hand of Equation 8. Table 2 shows an example of the distribution of the number of the zero-valued counters.

Claim 1. *The value of $\mathbb{P}(m_0 = \mu)$ is maximized for $\mu = \lceil E(m_0) \rfloor$.*

[1] Detailed deduction is presented in the Appendix.

Table 2. Distribution of m_0 $(11 \leq m_0 < 40)$ for $m = 40,\ k = 6$, and $|\hbar| = 5$, in which case $\lceil E(m_0) \rfloor = 19$, $\sigma(m_0) \approx 1.8$, and the value of $\mathbb{P}(m_0 = \mu)$ is maximized for $\mu = 19$; for the three values of m_0 marked by '•', $\widetilde{|\hbar|} = |\hbar| = 5$, which has a probability of 0.59

m_0	$= 11$	$= 12$	$= 13$	$= 14$	$= 15$	$= 16$
probability	$7.52 * 10^{-6}$	$1.28 * 10^{-4}$	$1.18 * 10^{-3}$	$6.85 * 10^{-3}$	$2.67 * 10^{-2}$	$7.30 * 10^{-2}$
m_0	$= 17$	$= 18^{\bullet}$	$= 19^{\bullet}$	$= 20^{\bullet}$	$= 21$	$= 22$
probability	$1.43 * 10^{-1}$	$2.04 * 10^{-1}$	$2.16 * 10^{-1}$	$1.69 * 10^{-1}$	$9.90 * 10^{-2}$	$4.32 * 10^{-2}$
m_0	$= 23$	$= 24$	$= 25$	$= 26$	$= 27$	≥ 28
probability	$1.40 * 10^{-2}$	$3.35 * 10^{-3}$	$5.88 * 10^{-4}$	$7.43 * 10^{-5}$	$6.66 * 10^{-6}$	$< 10^{-6}$

3.2 FCount Algorithm

Probabilistic Model and Algorithm Description. We use k hash functions $h_1, h_2, \ldots, h_k$ to hash elements of a multi-set $\hbar$ into a counter vector of size m. Initially, the approximate value of $|\hbar|$, $\widetilde{|\hbar|}$ is set to 0, and all the m counters are set to 0. An insertion (deletion) of an element e will increase (decrease) the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ by 1. Before an insertion of an element e, if one of the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ is zero-valued (i.e. membership checking returns *false*), $\widetilde{|\hbar|}$ is increased by 1. After a deletion of an element e, if one of the counters at positions $h_1(e), h_2(e), \ldots, h_k(e)$ is zero-valued, $\widetilde{|\hbar|}$ is decreased by 1. The model assumes that each hash function uniformly distributes the hashed elements over the m counters of the counter vector.

Error Estimation. Suppose we insert a sequence of elements $e_1, e_2, ..., e_N$ into a multi-set $\hbar$ represented by a Bloom Filter which is empty before insertions. $e_{i_1}, e_{i_2}, ..., e_{i_n}$ $(i_1 < i_2 < ... < i_n)$ is a (maybe non-continuous) subsequence of $e_1, e_2, ..., e_N$, and is consisting of all the elements e_t which satisfies the condition that not exists an element e_l of $e_1, e_2, ..., e_N$, $l < t$ and $e_l = e_t$; n is the number of the distinct elements in $e_1, e_2, ..., e_N$. Since the same elements will be hashed into the same counters of the counter vector, the error size of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ after the insertions of $e_1, e_2, ..., e_N$ equals to the error size of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ after the insertions of $e_{i_1}, e_{i_2}, ..., e_{i_n}$. In case of inserting $e_{i_1}, e_{i_2}, ..., e_{i_n}$ into the multi-set, before the insertion of e_{i_j} $(2 \leq j \leq n)$, the false positive error probability of the membership checking is $(1-(1-\frac{1}{m})^{k(j-1)})^k$. Equation 9 shows the expected value of the error size of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ after the insertions of $e_1, e_2, ..., e_N$. For the false positive property of the membership checking, $\widetilde{|\hbar|}$ is equal or lesser than $|\hbar|$ at all times during the insertions. In the case of $m = 8000,\ k = 6$, and $n = 1000$, the expected value of the error size is 4.

$$E(error\ size) = \sum_{j=1}^{n-1} \left(1 - \left(1 - \frac{1}{m}\right)^{kj}\right)^k \tag{9}$$

Now, we delete the sequence of elements $e_1, e_2, ..., e_N$ from the Bloom Filter. $e_{f_1}, e_{f_2}, ..., e_{f_n}$ $(f_1 < f_2 < ... < f_n)$ is a (maybe non-continuous) subsequence of

$e_1, e_2, ..., e_N$, and is consisting of all the elements e_g which satisfies the condition that not exists an element e_r of $e_1, e_2, ..., e_N$, $r > g$ and $e_r = e_g$. Since the same elements will be hashed into the same counters of the counter vector, the error size of deleting $e_1, e_2, ..., e_N$ (after the insertions of $e_1, e_2, ..., e_N$) equals to the error size of deleting $e_{f_1}, e_{f_2}, ..., e_{f_n}$ (after the insertions of $e_{i_1}, e_{i_2}, ..., e_{i_n}$). For the false positive property of the membership checking, there is potential negative (positive) error in case of insertions (deletions). In case of deleting $e_{f_1}, e_{f_2}, ..., e_{f_n}$ after the insertions of $e_{i_1}, e_{i_2}, ..., e_{i_n}$, after the deletion of element e_{f_j} $(1 \leq j \leq n-1)$, the false positive error probability of the membership checking is $(1-(1-\frac{1}{m})^{k(n-j)})^k$. The expected value of the error size of the deletions of $e_1, e_2, ..., e_N$ is $\sum_{j=1}^{n-1}(1-(1-\frac{1}{m})^{k(n-j)})^k$, which equals to the right hand of Equation 9. Although the expected value of the negative error size of the insertions of $e_1, e_2, ..., e_N$ equals to the expected value of the positive error size of the deletions of $e_1, e_2, ..., e_N$, the value of $\widetilde{|\hbar|}$ may still be non-zero-valued (maybe a small positive/negative integer) after the deletions of $e_1, e_2, ..., e_N$. In case of both insertions and deletions over the multi-set $\hbar$ may be emitted at any times, the expected value of the error size of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ is $\sum_{j=1}^{|\hbar|-1}(1-(1-\frac{1}{m})^{kj})^k$, and the error is negative; but the negative expected error does not mean that $\widetilde{|\hbar|}$ is equal or lesser than $|\hbar|$ at all times (except the case that only insertions can be emitted), $\widetilde{|\hbar|}$ may greater than $|\hbar|$ in some rare cases. Using $\sum_{j=1}^{\widetilde{|\hbar|}-1}(1-(1-\frac{1}{m})^{kj})^k$ to compensate the expected negative error of $\widetilde{|\hbar|}$ is a good optimization, and is called *OFCount* (*O*ptimized *FCount*) algorithm.

3.3 Algorithms Comparison

Positive/Negative Error. If $|\hbar| < \widetilde{|\hbar|}$ ($|\hbar| > \widetilde{|\hbar|}$), the error of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ is positive (negative). In case of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ using the *BFCDist* algorithm, the error has about the same probability of being positive or negative. In case of approximating $|\hbar|$ by $\widetilde{|\hbar|}$ using the *FCount* algorithm, if only insertions can be emitted, the error is negative; otherwise, the error may be positive or negative, but the error has a very small probability of being positive. Although the *OFCount* algorithm compensates part of the negative error of the *FCount* algorithm, the error has still a very small probability of being positive.

Online/Offline Computing. If a cardinality computing algorithm overlaps with the insertions and deletions, the algorithm is an online algorithm; otherwise, the algorithm is an offline algorithm. Since the *FCount* algorithm computes the cardinality of a multi-set during the insertion and deletion, *FCount* algorithm is an online algorithm. *BFCDist* algorithm has an online version which counts the number of the zero-valued counters during the insertions and deletions, and an offline version which counts the number of the zero-valued counters seperately.

Error Size. The error size of the *FCount* algorithm has close relation with the false positive error rate of the membership checking. As shown in Section 2, the

false positive error rate of the membership checking is very low under optimal $\frac{m}{n}$ and k combinations. For example, in the optimal case of $m = 8000$, $n = 1000$, and $k = 6$, the expected value of the error size of the *FCount* algorithm is about 4; but, int the case of $m = 8000$, $n = 2000$, and $k = 6$, the expected value of the error size of the *FCount* algorithm is about 108. The *OFCount* algorithm can reduce the error size to some extent, but in the non-optimal case, the error may still be very large. Compared with the *FCount* algorithm and the *OFCount* algorithm, the *BFCDist* algorithm is a pure probabilistic algorithm, and the $\frac{m}{n}$ and k combinations have a lesser influence to the error size of the *BFCDist* algorithm.

Distributed Cardinality Computing. The *offline BFCDist* algorithm can be easily extended to distributed environment. For example, if queries are required for computing the cardinality of the unions of several multi-sets, we can build the Bloom Filters for each multi-set using the same hash functions and counter vector size. When a cardinality computing query is emitted, we transform each counter vector into a bit vector (if a counter is zero-valued, the corresponding bit is set to 1; otherwise, the corresponding bit is set to 0) and send all bit vectors to a central site. Then we logic OR all the bit vectors in the central site and count the number of the zero-valued counters in the result bit vector. Finally, we compute the cardinality using Equation 5. The *online BFCDist* algorithm and the *FCount* algorithm can not be easily extended to distributed environment.

3.4 Compared with the FM Sketch

Flajolet and Martin [4] introduced the FM sketch which is designed specifically for counting distinct elements. There are two important extensions which use the FM sketch to estimate the set-expression cardinalities [5] and the join result cardinalities [6]. As shown in [4], [5], [6], if a single sketch is used, the percentage relative error is about 50%, which is certainly too high for most applications. In order to remedy this situation, a set of ξ sketches with different hash functions are used, and averaging all the sketches to generate the estimated cardinality. As shown in [4], when $\xi = 64$ (256), the expected percentage relative error can be reduced to about 10% (5%). But this remedy has the disadvantage of requiring the calculation of many hash functions and the store of many sketches; if we want the expected percentage relative error can be reduced to 1%, the storage and calculation overhead may be unacceptable. Compared with the FM sketch, Bloom Filers are designed originally for supporting membership checking, and functionality extension makes it support multiplicity queries (Bloom Filters can be seen as histograms of single element level), which are all not supported by the FM sketch. If a small number of sketches can reach the performance requirement, the storage and calculation overhead of FM sketches is low. The space complexity of Bloom Filters is linear, but as shown in Section 5, the percentage relative error is below 1% in most cases, and the calculation overhead is very low all the time.

4 Applications

Bloom Filters have been used in database applications since the 1970s, and have become popular in the networking literature in recent years. A good survey of the network applications and historical database applications is presented in [7], in which membership checking is used by all the applications. Saar and Matias [3] introduced the Spectral Bloom Filter, in which multiplicity queries are supported, and applications which use the multiplicity queries are presented. Important applications of the Bloom Filters presented recently include [8], [9], in which Bloom Filters are used to detect the duplications in streaming data such as the clicking stream (which is an important application in the web monitoring and management area). Since Bloom Filters support both insertions and deletions, it can be used to evaluate the online continuous queries over both sliding and landmark stream windows [10]. Our algorithms in this paper can be used to approximately evaluate the *holistic* aggregate COUNT-DISTINCT over both (distributed) streaming data and (distributed) offline data with low space/time overhead and low error size, which make Bloom Filters usable in a variety of new applications. For example, along with the duplication detection in the clicking stream, we can use the Bloom Filter to approximate the number of the distinct clicks, which is very important to the internet publisher. We can also use the Bloom Filter to approximate the number of the distinct IPs from web usage logs distributed over all mirrored web servers; at the same time, membership queries and multiplicity queries over the IPs of the web usage logs can also be supported.

5 Experimental Results

We have tested the accuracy of the algorithms described in Section 3 on a 1.4 GHz Pentium IV CPU with 256M of memory running RedHat Linux, and the algorithms are implemented by the C++ programming language. As in [3], the hash functions we used is of the modulo/multiply type: given a value v, its hash value $h(v)$ $(0 \leq h(v) < m)$ is computed by $h(v) = \lceil m(\alpha v \bmod 1) \rceil$, where α is taken uniformly at random from $[0, 1]$. The random number is generated by the *GNU Scientific Library* [11], and the random number generating algorithm is *gsl_rng_taus2* [12]. In addition to the *string-array index* described in Section 2, the *Elias encoding method* [13] is also a good encoding method which can makes the counter vector more compact while sacrificing some lookup performance. Saar and Matias [3] tested the storage needed by the encoding methods; the experimental results can also be applied to this paper, so we no longer consider the storage overhead in this section. We use two sets of experiments to test the accuracy of the algorithms. The error of the cardinality is characterized by the difference between the approximated cardinality and the accurate cardinality, $|\widetilde{\hbar}| - |\hbar|$. The average cardinality error is characterized by the mean value of the absolute values of $|\widetilde{\hbar}| - |\hbar|$. Using our cardinality computing algorithms, the error of the approximated cardinality is independent of the distribution of the elements in the multi-set, so the distribution of the elements is not considered.

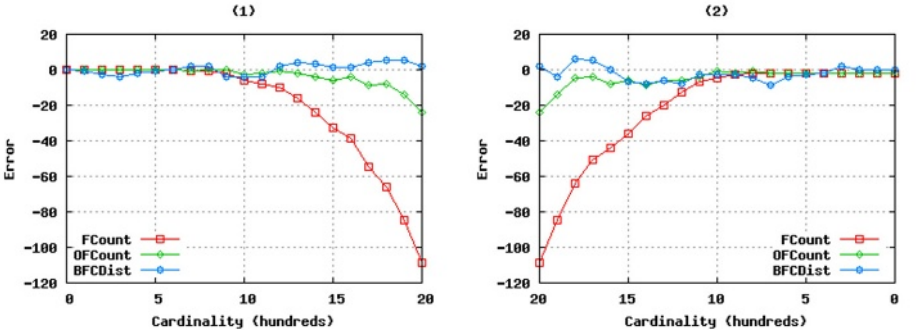


Fig. 1. Error of approximating the cardinality of a multi-set by using 6 hash functions to hash elements into a counter vector of size 8000. (1) The error during the insertion of an element sequence which contains 2000 distinct elements into the empty multi-set. (2) The error during the deletion of the element sequence inserted by the previous step.

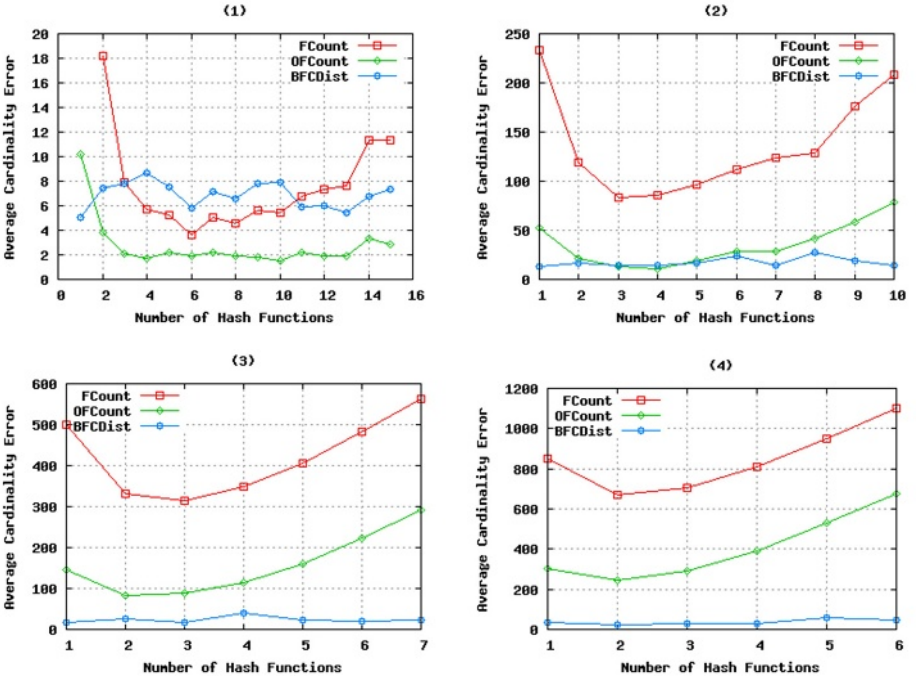


Fig. 2. Average cardinality error of approximating the cardinality of a multi-set under different $\frac{m}{n}$ and k combinations. The average is of 10 times when the cardinality equals to n during the insertions and deletions. $m = 8000$. (1) $n = 1000$ and $k \in [01, 16]$. (2) $n = 2000$ and $k \in [01, 10]$. (3) $n = 3000$ and $k \in [01, 07]$. (4) $n = 4000$ and $k \in [01, 06]$.

In the first set of experiments, we use 6 hash functions to hash elements into a counter vector of size 8000. We first insert a sequence of elements which contains 2000 distinct elements, and then delete the elements, during which we test the error of the cardinality with the change of the accurate cardinality. Fig. 1.1 (1.2) shows the error during the insertions (deletions). We can see from Fig. 1 that when the number of the distinct elements is lesser than 1000, the error is very small; when the number of the distinct elements is greater than 1000, for the increasing of the false positive error rate of the membership checking, the error of the *OFCount* algorithm, especially the *FCount* algorithm grows very fast.

In the second set of experiments, we test the average cardinality error under different $\frac{m}{n}$ and k combinations, where m is the counter vector size, n is the number of the distinct elements, and k is the hash function number. Fig. 2 shows the average cardinality error, in which m is fixed to 8000, n increases from 1000 to 4000 stepped by 1000 from Fig. 2.1 to 2.4, and each point indicates the average cardinality error under a specific $\frac{m}{n}$ and k combination. The average cardinality error is of 10 times when the accurate cardinality equals to n during randomized insertions and deletions of elements. We can see from Fig. 2 that with the value of $\frac{m}{n}$ decreases from 8 to 2, the average error of all the three algorithms increases at the same time, and the increasing rate of the *FCount* (*BFCDist*) algorithm is the highest (lowest); the $\frac{m}{n}$ and k combinations have a lesser influence to the average cardinality error of the *BFCDist* algorithm, but the *FCount* and *OFCount* algorithm have the lowest average cardinality error near the optimal $\frac{m}{n}$ and k combinations shown in Table 1. We also can see that with the value of $\frac{m}{n}$ decreases from 8 to 2, the relation between the average error of the *OFCount* algorithm and the average error of the *BFCDist* algorithm changes from lesser than to greater than, and the dividing line is about $\frac{m}{n} = 4$.

6 Discussions and Conclusions

In this paper, we extend the functionality of Bloom Filters by introducing novel algorithms for computing the cardinality of multi-sets represented by Bloom Filters. According to performance analysis and experimental results, if online cardinality computing is allowed, we can use the *OFCount* algorithm in case of the ratio between the counter vector size and the approximated cardinality is high, and use the *BFCDist* algorithm in case of the ratio is low; if online cardinality computing is not allowed, we can only use the *BFCDist* algorithm. The dividing line between the high and the low ratio is about $\frac{m}{n} = 4$, where m is the counter vector size, and n is the (approximated) cardinality. But no matter online computing is allowed or not, the ratio between the average cardinality error and the accurate cardinality is below 1% in most of the cases. For Bloom Filters can support cardinality computing with low space/time overhead and low error size in addition to insertions, deletions, membership queries, and multiplicity queries, it can be used in a variety of new applications, and our work can be seen as a new step towards fully representing multi-sets by Bloom Filters.

References

1. B. H. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. Communication of the ACM. 1970, 13(7), 422-426.
2. L. Fan, P. Cao, J. Almeida, and A. Z. Border. Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol. ACM SIGCOMM Computer Communication Review. 1998, 28(4), 254-265.
3. Saar Cohen and Yossi Matias. Spectral Bloom Filters. Proceedings of SIGMOD 2003, 241-252.
4. Philippe Flajolet and Nigel Martin. Probabilistic Counting Algorithms for Data Base Applications. Journal of Computer and System Sciences. 1985, 31(2), 182-209.
5. Sumit Ganguly, Minos N. Garofalakis, and Rajeev Rastogi. Tracking Set-Expression Cardinalities over Continuous Update Streams. VLDB Journal. 2004, 13(4), 354-369.
6. Minos N. Garofalakis, Sumit Ganguly, Amit Kumar, and Rajeev Rastogi. Join-Distinct Aggregate Estimation over Update Streams. Proceedings of PODS 2005, 259-270.
7. Andrei Broder and Michael Mitzenmacher. Network Applications of Bloom Filters: A Survey. Internet Mathematics. 2004, 1(4), 485-509.
8. Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Duplicate Detection in Click Streams. Proceedings of WWW 2005, 12-21.
9. Fan Deng and Davood Rafiei. Approximately Detecting Duplicates for Streaming Data using Stable Bloom Filters. Proceedings of SIGMOD 2006, 25-36.
10. Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and Issues in Data Stream Systems. Proceedings of PODS 2002, 1-16.
11. http://www.gnu.org/software/gsl/
12. P. L'Ecuyer. Tables of Maximally Equidistributed Combined LFSR Generators. Mathematics of Computation. 1999, 68(225), 261-269.
13. P. Elias. Universal Codeword Sets and Representations of the Integers. IEEE Transactions on Information Theory. 1975, 21(2), 194-202.

Appendix

Lemma 2. $|\bigcup_{i=1}^{\tau} \Lambda_i| = \sum_{i=1}^{\tau-1} (-1)^{i-1} \binom{\tau}{i} (\tau - i)^{\xi}$.

Proof. Using *Inclusion-Exclusion Principle*, $|\bigcup_{i=1}^{\tau} \Lambda_i| = \sum_{i=1}^{\tau} |\Lambda_i| - \sum_{i<j} |\Lambda_i \bigcap \Lambda_j| + \sum_{i<j<k} |\Lambda_i \bigcap \Lambda_j \bigcap \Lambda_k| - \cdots + (-1)^{\tau-1} |\Lambda_1 \bigcap \Lambda_2 \bigcap \cdots \bigcap \Lambda_\tau| = \binom{\tau}{1}(\tau-1)^{\xi} - \binom{\tau}{2}(\tau - 2)^{\xi} + \binom{\tau}{3}(\tau-3)^{\xi} - \cdots + (-1)^{\tau-2}\binom{\tau}{\tau-1}(\tau - (\tau-1))^{\xi} = \sum_{i=1}^{\tau-1} (-1)^{i-1} \binom{\tau}{i} (\tau - i)^{\xi}$.

An Efficient Scheme to Completely Avoid Re-labeling in XML Updates

Hye-Kyeong Ko and SangKeun Lee*

Department of Computer Science and Engineering,
Korea University, Seoul, South Korea
{ellefgt, yalphy}@korea.ac.kr

Abstract. A number of labeling schemes have been designed to label element nodes such that the relationship between nodes can easily be determined by comparing their labels. However, the label update cost is high in present labeling schemes. These schemes must re-label existing nodes or re-calculate certain values when inserting an order-sensitive element. In this paper, a labeling scheme is proposed to support order-sensitive update, without re-labeling or re-calculation. In experimental results, the proposed scheme efficiently processes order-sensitive queries and updates.

1 Introduction

The growing number of XML repositories on the World Wide Web has provided the momentum for the development of systems that can store and query XML data efficiently [10]. In order to facilitate the determination of relationships among nodes, nodes in the XML tree are typically labeled in such a way that ancestor-descendant relationships between any two nodes can be established quickly. Hence, a good and compact labeling scheme is crucial to efficiently process XML queries.

Early work on labeling schemes is typically interval-based [5], [11]. A depth-first traversal of the XML tree is carried out to assign to each node a pair of values that cover the range of values in the labels of its descendant nodes. However, XML documents on the Web are subjected to frequent changes. As a result, such static interval-based labeling schemes require re-labeling of the entire XML tree when frequent insertions and deletions of nodes occur. The prefix-based labeling scheme is designed to handle dynamic XML trees [1], [3], [4]. New nodes can be inserted without affecting the labels of existing nodes. Tatarinov et al. [9] introduces a labeling scheme that can be used to support order-sensitive queries. However, no existing labeling scheme is able to support dynamic updates when order is a concern.

In this paper, a novel labeling scheme for XML documents, which is based on the property of binary strings, is proposed. In the proposed labeling scheme,

* Corresponding author. This work was supported in part by Seoul R&BD Program from Seoul Metropolitan Government, South Korea.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 259–264, 2006.

dynamic updates and answer queries with ordered XQuery and XPath functions are able to be supported.

The subsequent sections of the paper are organized as follows. Section 2 reviews related work and provides the motivation for this paper. Section 3 describes the properties of the proposed labeling scheme. Section 4 presents the results of the experiments. Section 5 concludes this paper.

2 Related Work and Motivation

This section presents the families of labeling schemes, namely containment [2], [12], prefix [1], [3], [6], [9] and prime [10]. In the containment schemes [2], [12], every node is assigned two variables: "start" and "end". These two variables represent an interval [start, end]. Although the containment scheme can determine the ancestor-descendant relationship quickly, it does not work well when inserting a node into the XML tree. In the prefix labeling schemes [3], [9], the label of a node is that its parent's label (*prefix*) concatenates its own (*self*) label. Compared to the containment scheme, the prefix scheme only needs to re-label sibling nodes after the inserted node and descendants of these siblings, which are more dynamic when updating. Xiaodong et al. [10] proposed a novel approach to label XML trees with prime numbers. Prime scheme utilizes the *Chinese Remainder Theorem* (CRT) [10] for the document order. When using the *Simultaneous Congruence* (SC) value in CRT to mod the self label of each node, the document order for each node can be calculated.

Motivation. The binary [3] and DeweyID prefix schemes [9] both need to re-label existing nodes when inserting an order-sensitive node. Although the prime scheme supports order-sensitive updates without any re-labeling of the existing nodes, the SC values are very large numbers and re-calculation is very time consuming. Therefore, the main objective of this paper is to dramatically decrease the update costs and completely avoid re-labeling existing nodes when inserting an order-sensitive node.

3 Update Friendly Labeling Scheme

3.1 XML Tree Labeling and Order-Sensitive Update

In the following discussion, updates on XML documents are described by commonly used tree operations, namely, *INSERT* an element, text, or attribute, and *DELETE* an element, text, or attribute. In fact, deletion can be realized more easily. The deletion of a node will not affect the ordering of the nodes in the XML tree. However, *INSERT* operation is slightly more complicated than *DELETE* operation.

Definition 3.1 (Label for node N)**.** *The first child of N is labeled with label(N)."10", the second child of N is labeled with label(N)."110", and the* i^{th} *child is labeled with label(N)."*1^i0*".*

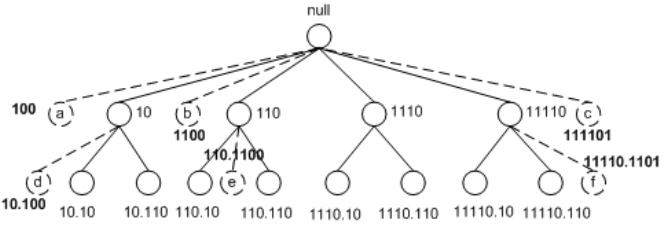


Fig. 1. Order-sensitive update of proposed scheme

Fig. 1 shows the labeled XML tree. The root node is labeled with an empty string. Then, the four child nodes of the root are labeled. The self label of the first child node is "10", the second child is "110", the third is "1110", and the fourth is "11110". In Fig. 1, the label of the node concatenates its parent's label and own label. For the two nodes "10" and "10.110", "10" is an ancestor of "10.110" if "10" is a prefix of "10.110".

Definition 3.2 (Insert string order)**.** *For any two existing adjacent labeling, N_{left} and N_{right}, a new labeling N_{new} can always be allocated between them without modifying already allocated labeling by the following Insert string order, where len(N) is defined as the bit length:*

1. *$len(N_{left}) \leq len(N_{right})$ then, $N_{new} = N_{right} \cdot 0$*
2. *$len(N_{left}) > len(N_{right})$ then, $N_{new} = N_{left} \cdot 1$*
3. *Insert a node before the first sibling node, $N_{new} = N_{left} \cdot 0$*
4. *Insert a node after the last sibling node, $N_{new} = N_{right} \cdot 1$*

Definition 3.3 (Lexicographical order $\prec$)**.** *Given two labels N_{left} and N_{right}, N_{left} is lexicographically equal to N_{right} iff they are exactly the same. N_{left} is said to be lexicographically smaller than N_{right} ($N_{left} \prec N_{right}$) iff*

1. *N_{left} is a prefix of N_{right}, or*
2. *"10" $\prec$ "110" $\prec$ "1110" $\prec$ "11110", or compare N_{left} and N_{right} by string. If the current string $N_{current}$ of N_{left} and the current string $N_{current}$ of N_{right} satisfy condition (1), then $N_{left} \prec N_{right}$ and the comparison is discontinued.*

Theorem 3.1. *The sibling self labels of the proposed scheme are lexicographically ordered but not numerically ordered. The labels (prefix label $\oplus$ delimiter $\oplus$ self label) of the proposed scheme are lexically ordered when comparing the labels component by component.*

Algorithm 1 shows the labeling of the XML document. The labeling algorithm which inserts the new self label is shown when the left self label and the right self label are known. If the size of the left self label is smaller than the size of the right self label of the new inserted node, an additional "0" is concatenated with the label. Otherwise, "1" is concatenated with the self label and the new inserted node is the left self label. Due to space limitations, size analysis is omitted.

Algorithm 1. Labeling Algorithm

XML labeling
Input: XML document
Output: label of each node
1. for all the sibling child nodes of each node of the XML document
2. for the first sibling child node, *selflabel*[1] = "10"
3. if the number of sibling nodes *Snum* > 1
   then *selflabel*[*Snum*] = "110"
   selflabel = labeling("1" ⊕ *selflabel*)
4. label = *prefixlabel* ⊕ delimiter ⊕ each element of *selflabel*

Update labeling
Input: $selflabel_{left}$, $selflabel_{right}$
Output: label of new inserted node
1. get the sizes of $selflabel_{left}$ and $selflabel_{right}$
2. if size($selflabel_{left}$) ≤ size($selflabel_{right}$)
   then $selflabel_{new} = selflabel_{right} \oplus$ "0"
3. else if size($selflabel_{left}$) > size($selflabel_{right}$)
   then $selflabel_{new} = selflabel_{left} \oplus$ "1"

4 Performance Analysis

The four labeling schemes were implemented in Java [8]: interval-based labeling scheme (interval) [5], DeweyID (Dewey) [9], the prime scheme (Prime) [10] and the proposed scheme. The three sets of experiments (storage, query and update) were demonstrated to evaluate and compare the performance of the four labeling schemes. We used the real-world XML data available in [7] to test the four schemes. Table 1 shows the characteristics of the various datasets used and shows the maximum number of nodes of XML. In the experiments, the labels of the XML are stored in a relational database with a table structure similar to that in [9].

4.1 Space Requirements

The label size in Fig. 2 refers to the total label size for all the nodes in each dataset. In Fig. 2, the proposed scheme has a large label size for each of the six

Table 1. Characteristics of datasets

Dataset	Topic	♯nodes
D1	Sigmod record	41
D2	Club	340
D3	Actor	1,110
D4	Department	2,686
D5	NASA	4,834
D6	Shakespeare's plays	6,636

Table 2. Test queries on the scaled D6

	Query	♯nodes
Q1	/paly//act[4]	185
Q2	/play//act[3]//following::act	370
Q3	/play//act/personal	969
Q4	/speech[4]//preceding::line	66,946
Q5	/act//following-sibling::speech[3]	143,725
Q6	/play/speech	154,755

datasets because it adds the label of the parent node as a prefix to the label of the child node. The total label sizes of all the six datasets for Interval, Prime and the proposed scheme are 0.76, 2.64 and 3.9 times that of DeweyID.

4.2 Query and Update Performance

In this experiment, the query performance of the four schemes was tested based on all the XML files in the Shakespeare′s plays dataset (D6). In order to test a large data workload, D6 was replicated 5 times, as shown in [9]. The ordered and un-ordered queries and the number of nodes returned from this scaled dataset are shown in Table 2. Except Q3 and Q6 (un-ordered queries), the proposed scheme worked fastest for the other 4 ordered queries in Fig. 3. The elements in the D6 are order-sensitive. Here, the update performance of the *Hamlet* XML file is demonstrated in D6. Fig. 4 shows the number of nodes for re-labeling when applying different schemes. The *Hamlet* XML file has a total of 6,636 nodes, but DeweyID and Interval need to re-label 6,605 nodes when inserting an act before act[1].

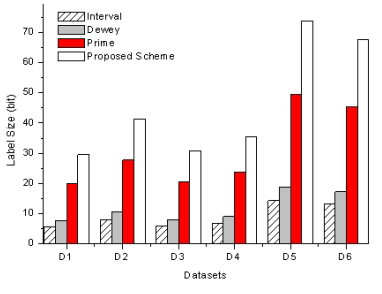


Fig. 2. Space for dataset

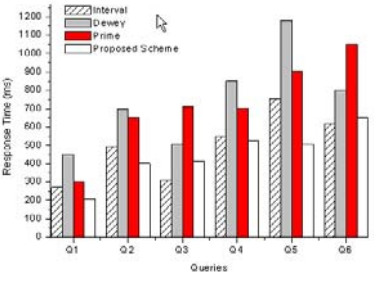


Fig. 3. Processing time of query

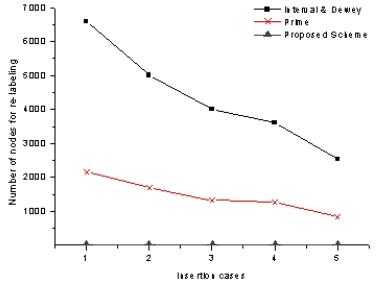


Fig. 4. Number of re-labeling node

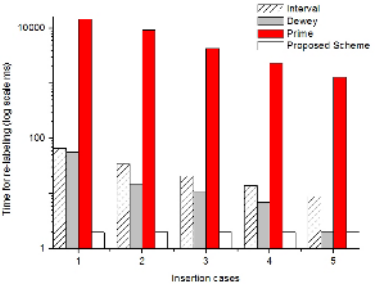


Fig. 5. Processing time for re-labeling

For Prime, the number of SC values required to be re-calculated is counted in Fig. 4. Because the prime each used the SC value for every label, the number of SC values required to be re-calculated is 1/3 of the number of nodes required

to be relabeled by Interval and DeweyID. In all five cases, the proposed scheme does not need to re-label existing nodes or re-calculate any values.

5 Conclusion and Future Work

The objective for designing labeling scheme for XML trees is to allow quick determination of the relationships among the element nodes without actually accessing the XML file. A new binary string based labeling scheme is developed, motivated by the need to efficiently support queries and updates in ordered XML trees. The proposed scheme can allocate any number of new labels without modifying already allocated labels, and does not need to re-label existing nodes or re-calculate any values when inserting order-sensitive nodes into the XML tree. Experiment results indicate that the proposed scheme is able to efficiently process ordered queries and maintain low cost of update. We will further discuss how to efficiently decrease the label size.

References

1. S. Abiteboul, H. Kaplan, and T. Milo. Compact labeling schemes for ancestor queries. In *Proceedings of ACM SODA*, pages 547–556, 2001.
2. R. Agrawal, A. Borgida, and H. V. Jagadish. Efficient management of transitive relationships in large data and knowledge bases. In *Proceeding of ACM SIGMOD*, pages 253–262, 1989.
3. E. Cohen, H. kaplan, and T. Milo. Labeling dynamic xml trees. In *Proceeding of PODS*, pages 271–281, 2002.
4. H. Kaplan, T. Milo, and R. Shabo. A comparison of labeling schemes for ancestor queries. In *Proceeding of ACM SODA*, pages 954-963, 2002.
5. Q. Li and B. Moon. Indexing and querying xml data for regular path expressions. In *Proceeding of VLDB*, pages 361–370, 2001.
6. J. McHugh, S. Abiteboul, R. Goloman, D. Quass, and J. Widom. Lore: A database management system for semistructured data. In *ACM SIGMOD Record*, 26(3), pages 54–66, 1997.
7. Niagara project. http://www.cs.wisc.edu/niagara/.
8. Sun. Java. http://java.sun.com/.
9. Tatarinov, S. D. Viglas, K. Beyer, J. Shanmugasundaram, E. Shekita, and C. Zhang. Storing and querying ordered xml using a relational database system. In *Proceeding of ACM SIGMOD*, pages 204–215, 2002.
10. X. Wu, M.-L. Lee, and W. Hsu. A prime number labeling scheme for dynamic ordered xml trees. In *Proceeding of ICDE*, pages 66–78, 2004.
11. M. Yoshikawa and T. Amagasa. Xrel: A path-based approach to storage and retrieval of xml documents using relational databaases. In *Proceeding of ACM TOIT*, pages 110-141, 2001.
12. C. Zhang, J. F. Naughton, D. J. DeWitt, Q. Luo, and G. M. Lohman. On supporting containment queries in relational database management systems. In *Proceeding of ACM SIGMOD*, pages 425–436, 2001.

Modeling Portlet Aggregation Through Statecharts

Oscar Díaz, Arantza Irastorza, Maider Azanza, and Felipe M. Villoria

Department of Computer Languages and Systems
University of the Basque Country
San Sebastian (Spain)
oscar.diaz@ehu.es, arantza.irastorza@ehu.es,
maider_azanza@ikasle.ehu.es, felipe.martin@ehu.es

Abstract. A portal is a key component of an enterprise integration strategy. It provides integration at the user interface level, whereas other integration technologies support business process, functional or data integration. To this end, portlet syndication is the next wave following the successful use of content syndication in current portals. A portlet is a front-end application which is rendered within the portal framework. From this perspective, portlets can be regarded as Web components, and the portal as the component container where portlets are aggregated to provide higher order applications. Unlike back-end integration approaches (e.g. workflow systems), portlet aggregation demands front-end solutions that permit users navigate freely among portlets in a hypertext way. To this end, the Hypermedia Model Based on Statecharts is used. This model uses the structure and execution semantics of statecharts to specify both the structural organization and the browsing semantics of portlet aggregation. Besides familiarity, statecharts bring formal validation to portal design, helping portal designers in the development of structured portals. As a prove of concept, this model has been realized in the *eXo* portal platform.

1 Introduction

The significance of portal applications stems not only from being a handy way to access data but also from being the means of facilitating the integration with third-party applications. Different standardization efforts (e.g. WSRP, JSR168) have provided a common way to include applications within the portal realm: the portlets [3]. Portlets are applications within a portal in much the same way as servlets are applications within a Web server. The difference stems from portlets being full-fledge, multi-step, user-facing applications. They are very much like Windows applications in a user desktop, in the sense that a portlet renders markup *fragments* that are surrounded by a decoration containing controls.

However, the added-value of a portal does not stop at providing a single access point for invoking heterogeneous applications but it should also include means to integrate these applications. A portal can contain several portlets, and portlet aggregation refers to the combination of a set of portlets to achieve a common goal within the portal realm.

So far, this aggregation is totally unconstrained. A portal page can contains a number of portlets whose fragments can be arranged into columns and rows, and minimized,

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 265–276, 2006.

maximized, or arranged to suit the user's needs. This is known as "side-by-side" aggregation. The value rests on providing a unified access point to the user, and rendering together content from diverse sources.

In the current approach, however, portlet navigation (i.e. browsing along the portlet fragments) is completely detached from portal navigation (i.e. browsing along the portal pages). And all portlets are readily rendered when entering in the container page.

This situation prevents the portal from becoming a proper workplace where tasks tend to have some order. Portlets, as task enablers, should also be engaged in some workflow that guides the user in accomplishing some goal. However, traditional workflows impose a rigid control on the activities and data available where tasks are predetermined. By contrast, a "deskflow" should be more easy-going where users can freely browse along the available tasks while some general guidelines can be prescribed. In other words, integration at the back-end can be better served by a workflow, predetermined approach where user intervention is minimized and rigidly controlled. By contrast, integration at the front-end aligns better with an hypertextual approach where the user is less constrained.

Therefore, the challenge is to find a formalism ductile enough to specify both constraint and unconstrained flows. This paper argues about the benefits of using statecharts for this purpose.

However, statecharts have been traditionally used for control specification and presentation concerns fall outside. But portlets are front-end applications that conform the portal interface. Therefore, an extension to statecharts is adopted that uses the structure and execution semantics of statecharts to specify both the structural organization and the browsing semantics of portlet aggregation [5]. The paper provides a sample case for a six-portlet portal which has been implemented in the eXo platform. Those interested in seeing the *Browsing* portal working can visit *http://www.onekin.org/academicBrowsing/*.

The rest of the paper is structured as follows. Section 2 introduces the notion of portlet aggregation and its issues. Section 3 presents the portal deskflow, formalized by a statechart. Section 4 explains the presentation model, complementary to the statechart model. Section 5 is about related work and finally, conclusions are given in section 6.

2 Selection of the Modeling Approach

Broadly speaking, aggregation can be defined as the purposeful combination of a set of artifacts to achieve a common goal. Literature can then be classified according to the type of artifacts being aggregated. If artifacts are text then, there is an extensive literature on hypertext approaches where the model helps to cope with the complexities of the structure and navigation of the application.

If the artifacts are Web services, aggregation models have been proposed to either orchestrate or compose Web services into higher-level services while preserving some formal properties such as availability, reachability and the like. Also, an hybrid approach is possible where artifacts include both text and Web services. WebML is a case in point [10]. Here, the focus is on Web service composition scenarios that involve human interaction on the Web. Other kind of artifacts are software components.

Component-based development aims at building systems by assembling components. Finally, this work focuses on portlets as the artifact to be aggregated.

The peculiarities of the artifact (i.e. text, Web services, portlets) influence the aggregation model. For instance, Web services have input/output parameters. Hence, Web service aggregation needs to address the role of parameters during aggregation (e.g. using semantic approaches [11][13]). By contrast, portlets do not have input/output parameters. Instead of delivering a data-based XML document, portlets deliver markup fragments (e.g. XHTML) ready to be rendered by the portal. Moreover, portlets tend to be stateful, and the interaction lasts for a whole session, rather than the simple request that characterizes the one-shot interaction of Web Services.

Based on the peculiarities of portlets and their most common containers, the portals, we identify two main requirements for the portlet aggregation model, namely:

- *hypertext-like navigation style.* Portlet aggregation should permit users to explore the applications freely, by skipping among applications if required. This does not preclude that a more conducted process *à la workflow* could need to be enforced in some cases, but the free-surfing style should be predominant.
- *front-end aggregation*. Portlets do not return data structures but markup (i.e. semi-structured documents where content and presentation are mixed together). Therefore, aggregation should be achieved in how markups from distinct portlets are arranged and sequentially presented. As an example, consider two portlets: *purcharseOrder* permits to buy some product whereas *creditRequest* allows to apply for a credit. The portal designer needs to enforce a precedence rule so that a credit can not be requested till a product is bought during the same session. This rule can be enforced in the back-end through some pre-conditions attached to the *creditRequest* service or some workflow engine enforcing the flow dependency. By contrast, a "front-end" approach can rely on disabling the *"creditRequest"* button until the *"purchaseOrder"* button has been clicked on. The integration is not achieved at the back-end but via presentation-based mechanisms.

To accomplish these requirements, the Hypermedia Model Based on Statecharts (HMBS) [5] has been adapted for our purposes. The use of statecharts [7,6] or their predecessors, state-transition diagrams, is common for modeling hypermedia applications, Web service composition [1,2] and reactive systems. Statecharts provide a concise and intuitive visual notation as well as being rigorously defined with a formal syntax and operational semantics. A hypermedia system, and also a portal, may be considered as a reactive system, since it must interactively attend to external events given in the form of user requests during browsing.

For the purpose of this paper, portal modeling implies describing:

- the structure of the portal, i.e how content is distributed both among pages and within a page
- the navigation of the portal, i.e. the order in which content is being made available

Next sections address these issues using statecharts and the HMBS model.

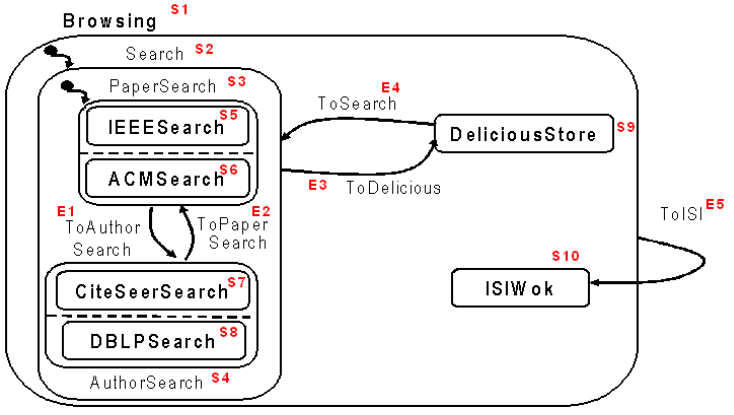


Fig. 1. Statechart of the *Browsing* portal

3 The Portal Deskflow

A portal defines a workspace, i.e. a compendium of front-end tasks. From a portal perspective, a task is a unit of behaviour: you are either visualising the task or you are not. How the task itself achieves its duty is outside the portal realm. The portal responsibility is to define a semi-structured flow among the contained tasks, i.e. the "deskflow".

Statecharts are used to define the "deskflow". Statecharts constitute a visual formalism for describing states and transitions in a modular fashion. It extends the classical formalism of state transition diagrams by incorporating the notions of hierarchy, orthogonality (concurrency), a broadcast mechanism for communication between concurrent components, composition, and refinement of states. Specifically, an OR-type decomposition is used when a state is to be decomposed into a set of exclusive substates, whereas an AND-type decomposition is used to decompose a state into parallel, or orthogonal substates. Each concurrent region in an AND state is delimited by a dashed row.

As an example, consider that the following tasks can be achieved within a portal: the *IEEESearch* task, which comprises a subset of the functionality of the IEEE portal; the *ACMSearch* task, which covers part of the offering of the portal.acm.org for the ACM organization; the *CiteSeerSearch* task, which includes the functionality for author searching at citeseer; the *DBLPSearch* task, which embodies the functionality for author searching at Ley's site; the *DeliciousStore* task, which provides the functionality available at *del.icio.us* for keeping track of references found in the Web; the *ISIWoK* task, which permits to obtain distinct quality parameters of a journal (e.g. impact factor) or paper through the ISI Web of Knowledge portal.

From the portal perspective, tasks are basic states. These tasks can be arranged along a deskflow as illustrated in figure 1. Here the portal designer initially considers two states: you are either searching for something on the Web, or you are storing your finding in *del.icio.us*. At any time, you can move to the ISIWoK task to consult the impact. *Search* is an abstract state which contemplates two situations: searching for a paper or searching for an author. Papers can be looked for at either IEEE or ACM. These

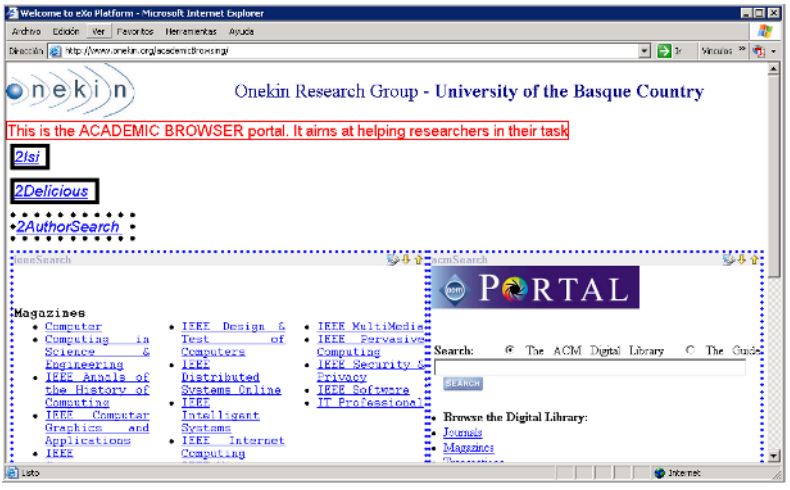


Fig. 2. "Presentation configuration" counterpart of the state configuration *{Browsing, Search, PaperSearch, IEEESearch, ACMSearch}*

options are simultaneously available, and hence, they are represented as an AND state (i.e. dotted line). On the other hand, author information can be obtained through the *CiteSeerSearch* task or *DBLPSearch* task. Both tasks can be simultaneously available, hence, they are also modeled as another AND state.

At a given moment, the system is in a certain *configuration state*, i.e. the set of currently active states of the statechart [7]. Basically, a configuration state comprises a basic state (e.g. *ACMSearch*), and its container states (i.e. *PaperSearch* and *Search*). The substates of an OR decomposition are not allowed to be active simultaneously, whereas the substates of an AND decomposition are simultaneously active as long as their container states remains active.

For our sample problem four state configurations are possible:

- configuration1: *{Browsing, ISIWoK}*
- configuration2: *{Browsing, DeliciousStore}*
- configuration3: *{Browsing, Search, PaperSearch, IEEESearch, ACMSearch}*
- configuration4: *{Browsing, Search, AuthorSearch, CiteSeerSearch, DBLPSearch}*

By mapping each state to its widget counterpart, state configurations are mapped into "presentation configurations". Figure 2 shows the result for the first state configuration. Next section addresses how this mapping is realized.

4 The Portal Structure

Portal structure refers to the presentation counterpart of the *deskflow*. To this end, we adapt the extensions proposed by the HMBS model to the portal setting. Specifically, HMBS [5] extends statecharts with mappings to those primitives found in hypertext

applications. Hence, we first introduce the presentation model (e.g. rendering concerns in a portal setting), and next, the mapping from a state configuration to a presentation configuration.

4.1 The Presentation Model

Basically, the presentation model refers to the GUI widgets and layouts used for supporting portal pages. For our case, GUI widgets are portlets, anchors and texts. Style and layout parameters are used to govern the aesthetic and layout of these widgets. Next paragraphs describe the main parameters which, with distinct variations, can be found in IDE for portal development such as eXo, Oracle Portal or IBM's Web Sphere[1].

Style parameters include background-color, border-style (values include none, dotted, dashed, and so on), border-color, border-width, font-family, color (often referred to as the foreground color), font-size, font-style (values include normal italic and oblique), text-align (i.e. how text is aligned within the element) and transition. The latter indicates how transitions are realized. The options include *anchor* (e.g. a button) or helping *text* where the transition is achieved by clicking on the underlined text.

As for the **layout**, it indicates how the artifacts are arranged along the presentation workspace. This is specified through a template function that takes a state configuration and produces a template where each corresponding artifact is placed in a cell of the template. This template is then ready to be rendered as an HTML <table> tag. Therefore, for the purpose of this paper, *a table-based page is the rendering counterpart of a state configuration*. However, to avoid cluttered pages in dense configuration states, other template functions can be envisaged where frames or tabs could be used to distribute the presentation artifacts along distinct presentation areas.

The description of how artifacts are distributed along the table's cell is achieved through the following parameters:

- **distribution.** It describes the policy to locate the set of anchors within the scope of the policy. Options include *together* and *detached*. The *together* option places all anchors together regardless of the associated task, whereas *detached* locates anchors beside the related task.
- **position.** Regardless of whether the distribution is *together* or *detached*, anchors can be on the top, bottom, left or right of the page or the related task.
- **alignment.** The previous parameters described the relationships among the portal pages, anchors and helping text, but the designer must make another decision: the distribution of different functionalities shown in the same page. Will they be one below the other or one next to the other? In the former case, the functionalities set a column, and in the latter, they set a row of functionalities. Thus, the values of this parameter are column and row.
- **presentationDef.** Another important decision in the appearance description will be the look-and-feel of the portal. Here, again, we can have different perspectives depending on the designer, the type of portal users, or the main aim of the portal. Thus, in some contexts, the objective could be that absolutely all the portal pages had the

[1] http://www.exoplatform.org/ ; http:/www.oracle.com/appserver/portal_home.html ; http://www.ibm.com/websphere/

Table 1. States and their presentation counterpart

State	Style guidelines
Browsing	transition=both; distribution=detached; position=top; textPosition=top; presentationDef=free; alignment=row **PortalAppDescriptor** {background=white; borderStyle=none; borderWidth=0px; borderColor=transparent} **WindowAppDescriptor** {borderStyle=solid; borderColor=orange; borderWidth=4px; background=white; fontFamily=times; fontSize=12pt; fontStyle=normal; color=black; textAlign=justify;} **AnchorAppDescriptor** {borderStyle=solid; borderColor=black; borderWidth=5px; background=white; fontFamily=arial; fontSize=14pt; fontStyle=italic; color=black; textAlign=justify;} **TextAppDescriptor** {background=white; borderStyle=solid; borderWidth=1px; borderColor=red; fontFamily=arial; color=red; fontSize=14pt; fontStyle=normal; textAlign=justify}
PaperSearch	**WindowAppDescriptor**{alignment=column; borderStyle=dotted; borderColor=blue;} **AnchorAppDescriptor** {borderStyle=dotted; borderWidth=7px}
IEEESearch	**WindowAppDescriptor** {fontFamily=courier; fontSize=14pt}
AuthorSearch	**WindowAppDescriptor** {alignment=column; position=bottom; }
ISIWok	**WindowAppDescriptor** { fontFamily=arial; fontSize=16pt;}

same look-and-feel, in other contexts it could be enough that the style inside a page was coherent (e.g. all the anchors with the same color and font-type), but in the same portal the style could change from one page to another (e.g. in one page all the anchors are red, and in another, blue), and at last, in other contexts, perhaps, an unified style is not important or the designers would like to use different styles to stress different points (e.g. assuming that in a portal page we can distinguish two types of anchors: global anchors, that appear in all pages, and local anchors, that are specific to each page, the designer could specify that the global anchors must go in red and the others in green). So, this parameter has three possible values: portal, page, free.

These parameters determine the finale rendering of a page. But this does not mean that the designer has to set these parameters for each of the portal pages. Indeed, ensuring a common look-and-feel throughout the portal pages is one of the main concerns for the portal designer to improve usability and the user experience. To this end, *skins* have been proposed, i.e. templates defined at the portal level that set some presentation properties and are then inherited from all the portal pages. Besides ensuring presentation homogeneity, this mechanism accounts for maintainability as (some) changes in the presentation uniquely involve updating the skin rather than modifying the distinct portal pages.

However, this mechanism can be too coarse grained, i.e. presentation is defined at either the portal or the page level. There is nothing in between. For large portals where pages can be grouped into clusters based on content or functional grounds, finer grained skins could be most convenient.

The idea is to use the statechart for this purpose. Both states and transitions will have a presentation counterpart (see next subsection). Rather than attaching presentation parameters to pages, now these parameters are associated with states, and their scope includes all substates. That is, *a parameter set for state* S *affects any widget associated with any substate directly or transitively below S*. A parameter is overridden if newly defined in a substate.

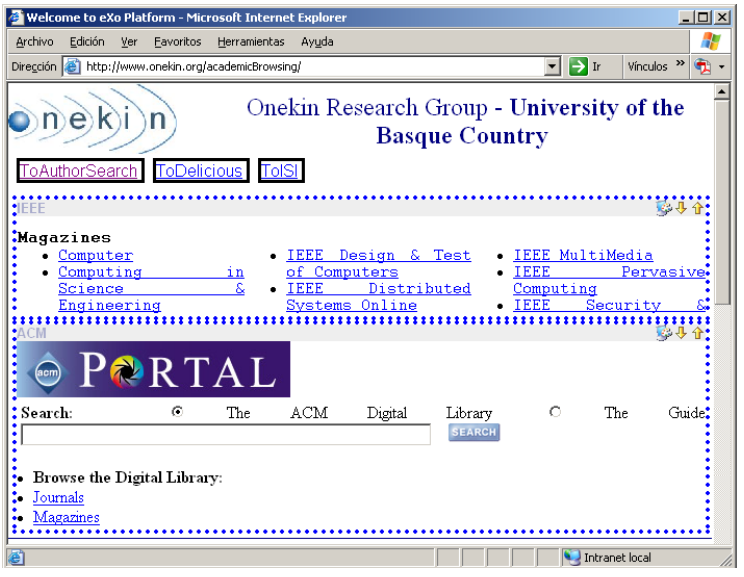


Fig. 3. "Presentation configuration" counterpart of the state configuration *{Browsing, Search, PaperSearch, IEEESearch, ACMSearch}*

This approach offers a balance between the generality required to provide a common look-and-feel along the portal while at the same time addressing specificities of certain tasks or task clusters where presentation singularity is sought. Traditional skins correspond to presentation parameters defined for the upper state (e.g. *Browsing*) whereas it is still possible to completely override the skin for a particular atomic state (e.g. *IEEESearch*). Table 1 describes the presentation model complementary to our *Browsing* statechart. In Table 1 we can see that at *Browsing* state level the designer has specified that the font type and size for displaying portlet information will be *Times* and 12, but at the *IEEESearch* state has redefined them, and she has decided that specifically the *ieee* portlet will show its information in *Courier* font and 14pt. Moreover at *Browsing* state the designer has defined that the anchors will be detached and they will be shown with a solid 5px border-line and in a italic font-style, by default. At the *PaperSearch* state, she has redefined the description and the anchors with origin in PaperSearch will be shown with a dotted 7px border-line. As we can see in figure 2 the presentation of the page follows this specification. The states that are not in the table do not have a specific presentation model, they inherit from their parente state.

Both models are independent, and using the same deskflow (statechart) it is possible to describe different presentations. In our example, for the statechart in figure 1 the designer could have decided another presentation model; for example, in *Browsing* state: *... distribution=together; alignment=column ... AnchorAppDescriptor{ ...fontStyle= normal; fontSize=12pt; borderStyle=solid; borderWidth=3px; ...} ....* In this case, the first page of the portal would have been as shown in figure 3. All anchors are together in the same row, and in normal 12pt font with a thinner border-line than before.

4.2 From State Configurations to Presentation Configurations: Generating the Portal Pages

In order to get the presentation configurations (portal pages) from the state configurations, statecharts are extended with two mapping functions, namely:

- **s2p** is a function that maps from states to their presentation counterpart. Only basic and OR states are considered. Only basic states can be mapped to portlets[2]. In our example our six basic states *(ISIWok, DeliciousStore, IEEESearch, ACMSearch, CiteSeerSearch, DBLPSearch*) are mapped to six portlets: isi, delicious, ieee, acm, citeseer and dblp. AND states are not mapped into widgets, and its unique purpose is to express concurrency in statecharts. An AND state indicates that information in widgets associated to its substates is to be shown simultaneously. This function also maps from transitions to anchors (the portal widget to perform the navigation).
- **a2e** is a function that associates anchors to statechart events that control the firing of transitions. A statechart transition must have a unique event, although an event may appear in multiple transitions. In this way, HMBS supports reuse of link components, as transitions and events may be reused to define anchors in different navigational contexts. An anchor within a page must be mapped into a transition originating from one of the states of the state configuration associated to the page. In figure 1, for example, an anchor in the widget associated to state *PaperSearch* should be mapped to the transition fired by *2AuthorSearch*. We can see that anchor inside the black dotted box in Figure 2.

The statechart and the mappings are provided by the portal designer. From then on, presentation is handled by the portal engine as follows. When the user clicks on an anchor, the system performs the following actions:

1. Applies the a2e mapping to generate the event that causes the triggering of the transition associated to that anchor.
2. Activates all states that are target sets for the transition fired, thus generating the next state configuration and disabling the previous one. To this end, the execution semantics of statecharts is used to describe the navigation of the portal. At a given moment, the system is in a certain configuration state. When an event happens, the system transmits it to the transition associated to the triggered event[3]. Then, the corresponding guard condition is evaluated, and if satisfied, the statechart sets the target state as active, which in turn, leads to a new state configuration.
3. Applies s2p to the new state configuration to obtain the next "presentation configuration", i.e. a set of artifacts that should be simultaneously rendered to the user.
4. New "presentation configuration" is delivered to the user. This ends the cycle.

Figures 2 and 4 depict the screen-shot counterparts of the 1 and 4 state configurations which correspond to our sample problem.

[2] Disclaimer: the portlets used in this paper were implemented as wrappers of third-party sites. They are used only for illustrative purposes, and not for commercial advantage.

[3] The operational semantics of statecharts relies on a discrete time model that assumes that system states have an associated duration, whereas transitions are instantaneous.

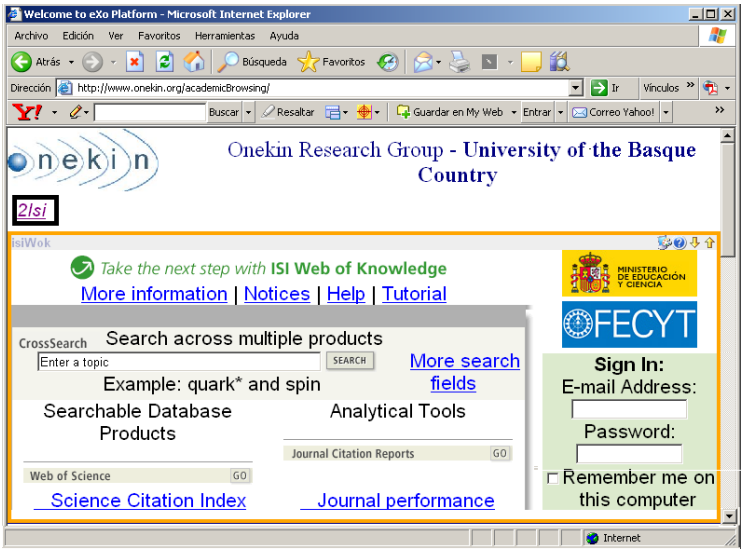


Fig. 4. "Presentation configuration" counterpart of the state configuration *{Browsing, ISIWoK}*

5 Related Works

Our work is related to WebML (www.webml.org), a Web modeling language that contemplates the interplay between Web applications and Web services. Whereas WebML considers the design of the Web application from scratch, our approach takes portlets as the starting point. Our intuition is that a component-based approach is more akin to the notion of the portal as a doorway to other applications. Moreover, we hope that the use of familiar models such as statecharts will facilitate the adoption of this approach in contrast with *ad-hoc* notations.

Pinheiro [12] explains that to develop abstract user interface descriptions the standard UML facilities are enough. They use UML class diagrams to represent presentation models and UML activity diagrams for task models, and then, they use object flows in activity diagrams to relate interaction objects (from the presentation model) to action states (from the activity diagram). The transitions in statecharts are triggered by events and not by the end of an activity as it happens in the activity diagrams. We see portals as aggregations of portlets, and after using one portlet, the user is who decides to go through another one. He clicks in the portal, i.e. triggers events. So, for this reason, we consider statechart mechanism more suitable for portal design. Moreover, besides the task and trigger design, our approach considers the look-and-feel of the portal and the state hierarchy helps the designer to define it in a modular and hierarchical manner.

Maamar [9] presents an approach for the design and development of service-driven applications: service chart diagrams are proposed as a specification technique. These diagrams are based on UML statecharts. With these diagrams a service is represented from five perspectives, one of them is the statechart. Another one is the information,

which identifies the data exchanged between services. As we see, this is the most important contribution of that work and an aspect that we should consider to add in our statechart diagrams, in order to represent the information exchange between two portlets.

In [4], we present our proposal to portlet aggregation, extending fragment markup of the portlet with semantic information. This set of data might be used by the designer of the portal to specify the mapping rules between the portlets she wants to pipe. Our work is complementary to this approach as we focus on the support of the portal as a whole. However, if we could make the portal aware of events coming from portlets then, statecharts could be expanded to contemplate also portlet events, besides those coming from the end user.

At last, Hong [8] proposes a DEVS modeling language called DEVS Specification Language (DEVSpecL) based on which discrete event systems are modeled, simulated and analyzed. The DEVS formalism specifies a model in a hierarchical, modular form: it is a state transition graph. DEVSpecL is a set of rules to define messages, input/output events, conditions, etc. With DEVSpecL the designer can specify textually the DEVS models. After that, a model specification in DEVSpecL is translated into DEVSpecL Abstract Syntax Tree (AST) and AST is used to check ill-structured coupling relations. The advantage of our approach is that using UML statecharts we do not need any new formalism to make the verification, because there are tools that make the analysis of statecharts.

6 Conclusions

The recent release of the WSRP and JSR168 standards promises to achieve portlet interoperability. This will certainly fuel the transition from content syndication to portlet syndication. In this context, the ability to aggregate portlets will become crucial to improve the user experience.

This work proposes the use of extended statecharts (i.e. HMBS statecharts) to model "portlet-intensive" portals. The approach benefits from all the advantages brought by the use of statecharts as well as improvements in the maintainability and modularity of the portal presentation.

Moreover we present a first approach for the design of this type of portals: (1) finding the tasks to show in the portal and then finding the portlets that will implement those tasks; (2) defining the s2p mapping function (each portlet is related to a basic state) and the deskflow (statechart) model (the designer decides the transitions between states, i.e. portlets); (3) describing the presentation model based on the navigation model. The designer is not concerned about the construction of portal pages, they will be generated by the portal engine, based on the deskflow and presentation models and s2p mapping function. Details about the implementation of the process of portal page generation are out of the aim of this paper.

Acknowledgments. Maider Azanza enjoys a doctoral grant for the Basque Government under the "Young Researchers Program".

References

1. D. Berardi, D. Calvanese, G. de Giacomo, M. Lenzerini, and M. Mecella. Automatic composition of e-services that export their behavior. In *Proc. of the 1st International Conference on Service Oriented Computing (ICSOC 2003)*, Lecture Notes in Computer Science, pages 43–58. Springer, 2003.
2. F. Casati and M. C. Shan. Dynamic and adaptive composition of e-services. *Information Systems*, 26(3):143–163, May 2001.
3. O. Díaz and J.J. Rodríguez. Portlets as Web Components: an Introduction. *Journal of Universal Computer Science*, 10(4):454–472, Apr 2004. http://www.jucs.org/jucs_10_4/portlets_as_web_components.
4. Oscar Díaz, Jon Iturrioz, and Arantza Irastorza. Improving portlet interoperability through deep annotation. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 372–381, New York, NY, USA, 2005. ACM Press.
5. M.C. Ferreira de Oliveira, M.A. Santos Turine, and P.C. Masiero. A Statechart-Based Model for Hypermedia Applications. *ACM Transactions on Information Systems*, 19(1):28–52, January 2001.
6. D. Harel and A. Naamad. The STATEMATE Semantics of Statecharts. *ACM Transactions on Software Engineering and Methodology*, 5(4):293–333, October 1996.
7. D. Harel, A. Pnueli, J.P. Schmidt, and R. Sherman. On the Formal Semantics of Statecharts. In *2nd IEEE Symposium on Logic in Computer Science*, pages 54–64, 1987.
8. K.J. Hong and T.G. Kim. Devspecl: Devs specification language for modeling, simulation and analysis of discrete event systems. *Information and Software Technology*, 48(4):221–234, April 2006.
9. Z. Maamar, B. Benatallah, and W. Mansoor. Service Chart Diagrams - Description and Application. In *Proc. of the 12th International World Wide Web Conference (WWW2003)*, May 2003.
10. I. Manolescu, M. Brambilla, S. Ceri, S. Comai, and P. Fraternalli. Model-driven design and deployment of service-enabled web applications. *ACM Transactions on Internet Technology (ACM TOIT)*, 5(3):439–479, August 2005.
11. M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara. Semantic Matching of Web Services Capabilities. In *1st International Semantic Web Conference*, pages 333–347. Springer-Verlag, June 2002.
12. P. Pinheiro da Silva and N.W. Paton. User interface modeling in UMLi. *IEEE Software*, 20(4):62–69, Jul/Aug 2003.
13. E. Sirin, J. Hendler, and B. Parsia. Semi-automatic Composition of Web Services using Semantic Descriptions. In *1st Workshop on Web Services: Modeling, Architecture and Infrastructure. In conjunction with ICEIS 2003*, pages 17–24. ICEIS Press, April 2003.

Calculation of Target Locations for Web Resources

Saeid Asadi[1], Jiajie Xu[1], Yuan Shi[2], Joachim Diederich[1], and Xiaofang Zhou[1]

[1] School of ITEE, the University of Queensland, Brisbane, QLD 4072, Australia
{asadi, zxf, joachimd}@itee.uq.edu.au
S4112087@student.uq.edu.au
[2] State Key Lab of Software Engineering, Wuhan University, China 430072
yuanshisklse@gmail.com

Abstract. A location-based search engine must be able to find and assign proper locations to Web resources. Host, content and metadata location information are not sufficient to describe the location of resources as they are ambiguous or unavailable for many documents. We introduce target location as the location of users of Web resources. Target location is content-independent and can be applied to all types of Web resources. A novel method is introduced which uses log files and IPs to track the visitors of websites. The experiments show that target location can be calculated for almost all documents on the Web at country level and to the majority of them in state and city levels. It can be assigned to Web resources as a new definition and dimension of location. It can be used separately or with other relevant locations to define the geography of Web resources. This compensates insufficient geographical information on Web resources and would facilitate the design and development of location-based search engines.

Keywords: location-based search, Web search, geographical search engine, target location.

1 Introduction

The World Wide Web has caused significant changes in traditional models of publishing, communicating, shopping and so on. Search engines have emerged as the main tool to capture resources on the Web and sophisticated search tools handle hundreds of millions of queries every day. Over 150 million queries are sent to Google and Yahoo each single day from the United States alone [1]. Among different services offered by search engines, location-based search has received significant attention. Location-based or geographic Web search intends to associate Web resources with real locations and offer the results based on the geographical area a query refers to. The Web has made it possible to have world wide access to information which was a desire for many centuries. However, the fast growth of Web-based services emerged in the recent years and demands for location-based information, has challenged the existing engines. Many queries (e.g. for driving, shopping or accommodation) have geospatial dimensions. Users search for services and facilities in a suburb, city or any other geographic area.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 277 – 288, 2006.

Studies such as [2,3] reveal that a significant portion of queries on existing general search engines have geospatial. So far, general search engines have not been able to answer location-based queries properly and their results are poor for this kind of queries [4]. Lack of geographic information and poor definition and allocation of locations to Web resources is a basic challenge that frustrates any attempts on developing location-based search engines.

This paper sets out to illustrate the challenges of defining the location of webpages and other resources on the Web. We think information extraction techniques can be applied to only a small portion of Web resources and complementary techniques must be employed to include more documents in the scope of location-based search engines. We introduce target location as a reliable way to calculate at least one location for every document on the Web. Target location deals with geographic areas that the users of a webpage belong to. The contribution of this paper is describing how to obtain and calculate target locations for all Web resources including texts, photos and other types of media.

2 Related Studies

Search engines have emerged since 1993 in parallel with the development of the World Wide Web. Major search engines have employed sophisticated techniques to collect as many documents as possible and to improve results effectively by using clustering or ranking models. Unlike general search, location-based search tools have not been used widely. Google Local[1] can search only for local business information in USA, UK and Canada [7]. Other location-based search engines such as SPIRITS, Northern Light and GeoSearch have remained incomplete or they have stopped their limited services. Local search homepages intend to search among webpages with limited domains or languages. For example, Yahoo7[2] is a search interface for Australia and New Zealand. These services are close to location-based tools but they are still too general and limited to particular domains and regions.

A geographical search engine must present the result in an effective way which is often different from general search. Map-based presentation, geographical ranking and location-based clustering are related techniques. Watters & Amoudi [4] report on an algorithm for re-ranking search results in a geographical approach. GeoSearcher tries to identify the geographic coordinates (latitudes and longitudes) of webpages by assistance of online gazetteers. Having the coordinates, it calculates the distance of selected webpages from a reference point. The results are then ranked in ascending order of distance. The distance-based ranking is a useful model for location-based search engines but it often results in wrongly ordered results as many geographical areas on earth do not have regular rectangular or circular shapes.

A significant challenge for location-based search engines is defining and assigning different locations to Web resources. Basically, a location-aware engine must be able to analyze webpages for geographical information. Information extraction rules and

[1] http://local.google.com/
[2] http://au.yahoo.com/

techniques have been used to extract addresses and location names from webpages [8,9,12,13]. Olga [10] describes an algorithm that assigns a geographical class to unknown names by adding patterns to a candidate name and sending the results as new queries to same search engines. Based on the number of returned documents, the algorithm determines which category is suitable for a name. Gravano et al. [11] divided Web queries into global and local based on search results. If the best results of a query refer to local webpages, the query is considered a local query; otherwise it will be global. Many webpages do not have addresses, geographic metadata or geographic features and IE techniques are unable to find related locations for these. Ding & Gravano [6] used the link structure of Web to assign locations to Web resources. The *geographic scope* of a Web resource is calculated as a city, state etc. where most of the back links refer to. This method can assign locations to webpages even if they are not mentioned in the text. It can also tag non-textual resources e.g. photos and films with location names. However, as many resources don't have enough back links, the model is not always applicable. Other studies [14,15] have focused on geographical indexing, address and location extraction, and tagging webpages with location names

3 Target Locations of Web Resources

In this section, we first describe different definitions of location on the Web and then introduce target location in detail.

3.1 Web Resource Locations

The location of a Web resource can be defined in many ways. We divide them in three groups: host, content and target locations.

Host's Location. Every website is stored on one or more servers which are located somewhere in the world. Therefore, the physical location of a resource or $L_{(w)}$ is defined as the location of the server which is hosting the resource *w*:

$L_{(w)}$: location of *w*'s hosting servers

Host's location is not the best way to define webpage location. Websites usually are hosted on servers which are not necessarily related to a topic of them.

Content Location. By content location we mean location(s) mentioned in a webpage's content including geographic features mentioned in text, footnotes, contact addresses, metatags, headings and so on. In this case, the location of a webpage is the geographical location of its objects or entities:

$L_{(w)}$: location of geographic entities in *w*

Different algorithms have been developed to extract addresses, postcodes and geographic features. Geo-tagging techniques extract all locations and disambiguate them by using gazetteers or pattern learning approaches. Web-a-Where [5] finds one or more focuses for a page. For example, if Brisbane, Gold Coast and Cairns are mentioned on a page, the focus of the page will be Queensland.

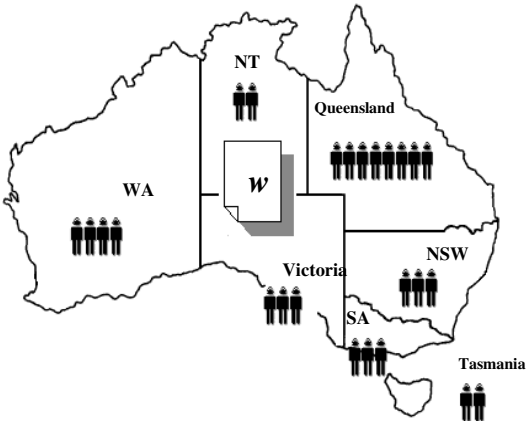


Fig. 1. An example of distribution of a Web resource users in different geographical locations (Here within the states of Australia)

Target Location. Websites are usually designed to serve a specific group of people. This population is often considered in the design and development of the page. Target location is an area a webpage is dedicated to or the location of its users. It can be said that webpages have at least one target which could be a small area like a suburb or town or a very large region. For example, the target location of a page entitled 'Pizza delivery in Toowong' can be Toowong suburb in Brisbane, Australia; while the target of Google search engine is the entire world. Target location is defined as:

$L_{(w)}$: location of people that use w

The geographical distribution of the users could be used as the target location of that page. If a website is designed for people of Toowong suburb for example, it is likely that most of the people who use this site be from Toowong or areas around it. As the target locations of many resources are not determined explicitly, a reverse approach can be used to find this information. Suppose that 1000 people have visited webpage w and most of them are from Queensland. Queensland is then a possible target location of w (Fig. 1).

3.2 Sources of Geographical Information

There are different sources for obtaining geographical information from Web resources. HTML tags are a basic source for extraction of location names and addresses. However, this information is often limited to a portion of textual resources. Link structure of the Web is another source. It can be assumed that a webpage receives more citations from its geographical target area. Ding & Gravano [6] introduced the concept of *geographic scope* which refers to one or more locations that most of the back links to a Web resource come from. The problem is that the majority of resources do not have enough back links to calculate their geographic scope.

We propose *visitor location* analysis instead of link and content analysis or as a complementary solution whenever other methods are not available. A Web resource *w* is visited or used by people in different geographic areas. Our assumption is that people in the target location of *w* visit it more than people in other areas. For example, the website *www.qldevents.com.au* that provides information about current events and programs in Queensland might be visited by people in Queensland State more than the people in Victoria. As a result, the target location of this webpage would be Queensland. Visitor information could be a useful way to find a Web resource's target location and it can be more practical than the link-based approach. Similar to geographic scope, visitor information can be applied to all resources and it is not limited to textual documents. Visitor information can be tracked and extracted by analyzing IP addresses which often refer to geographic areas. However, extracting geographic information from IPs is time consuming and requires complex information extraction and normalization steps. Surfing the Web anonymously can also affect the quality of visitor-based approach.

A typical connection to the Internet occurs through an ISP which registers a range of IPs. Geographically, an ISP is located in a city and serves that city or region. When a user visits a website, his/her information is stored in a log file. Log files are stored on hosting servers and contain valuable information about the history of visits to a website. Logs contain the user's IP, date, time and length of the visit and the list of pages and files visited by a user. There are some online databases and which provide information about the IPs. The structured data on these databases tagged as telephone, address, country name etc. provides a useful source of geographic information.

3.3 Measures for Calculation of Target Location

Different measures can be used to aim at the selection of the best locations as targets. We measure *popularity, distribution* and *stability* of candidate locations to calculate the temporary and constant target locations of a Web resource.

Popularity. If most of the users of Web resource *w* are from the location ℓ, it can be said that *w* is a popular page or document in ℓ rather than other locations. Popularity of Web resources has been measured by [6] using a *power* formula. In a particular location ℓ, *power* is the number pages with link to *w* compared to the total number of webpages existing in ℓ. The main problem with this formula is that there is no reliable way to find out how many Web resources exist in ℓ. Instead, our *power* formula compares the popularity of *w* in ℓ to the total number of visitors of *w:*

$$Power(w,\ell) = \frac{Visitors(w,\ell)}{Visitors(w)} \tag{1}$$

Distribution. Geographical areas often consist of several smaller divisions. For example, a country usually has several states or provinces. Calculations on large locations ignoring their sub-locations can lead to poor accuracy. For example, if many visitors of *w* are from Australia but all of them are from Queensland State, then it

seems that Queensland is a better target than Australia. If they are distributed smoothly over different states, then Australia can be a target location. A vector-space definition of distribution or *spread,* described in [6], computes the similarity between vectors of webpages and vectors of links. Instead of links, we use the number of visitors in a sub-location ℓ_i :

$$Spread \quad (w, \ell) = \frac{\sum_{i=1}^{i=n} \ell_i}{\sqrt{\sum_{i=1}^{i=n} \ell_i^{\,2}}} \tag{2}$$

We use spread as a threshold to select few candidates and then use power as a threshold to select temporal targets (TTLs).

Stability. A location is eligible as target if it remains in the scope of a webpage over time. To consider the freshness of webpages, the extracted target locations can receive different time-dependent weights. The more recent a location has been selected as a target, the higher weight it receives. Stability is measured as:

$$Stability \quad (\ell, w, T) = \frac{\sum_{n=t_0}^{n=t_n} \frac{1}{2^n} \ell}{\sum_{n=t_0}^{n=t_n} \frac{1}{2^n} L} \tag{3}$$

where t_i is a subdivision of time T, ℓ is a location selected as a temporal target (TTL) of w in t_i , and L refers to all TTLs selected in t_i. We use stability score as a threshold to select CTLs from TTLs.

Pruning Models. Three pruning strategies are used to choose constant targets (CTLs) from candidate TTLs after measuring their stability:

Top-k Threshold. Given an integer k, constant target locations are selected from the top of the list while candidates are ordered based on the decrease in *stability*.
Fixed Threshold. A fixed threshold of *stability* is defined and candidates above this threshold are selected.
Relative Threshold. A percentage threshold of *stability* is defined and candidates above this threshold are selected.

We use same pruning models to choose temporal targets (TTLs) from the extracted locations. In this case, power is used instead of stability as a threshold.

3.4 The Algorithm

A summary of the algorithm for the proposed visitor-based model of estimating and calculation of target locations is mentioned below. In a sub-period t_i, we calculate power and spread of the extracted locations and then select a few of them as temporal targets (TTLs) using three pruning models. In the end of T, we calculate the stability of all locations selected as TTLs in t_0, t_1,t_2,…,t_n. Finally, we select constant target locations (CTLs) from TTLs using stability as a threshold for different pruning strategies.

For t_i =a time subsequence of T:
  For $\ell \in N$ (N =extracted locations from IPs):
    A = list of candidate targets
    Calculate power and spread of ℓ
    if spread $\ell \geq$ threshold τ_s
      Add ℓ to A
    Select TTLs from A:
      Case 1. Select k $\ell \in A$ with the highest *power*
      Case 2. Select *all* $\ell \in A$ with *power* score $\geq \tau_p$
      Case 3. Relative threshold pruning:
        Select all $\ell \in A$ with $power \geq \frac{n}{m}$ top power in A

Then (in the end of *T*)
For $\ell \in B$ ($B = \sum TTLs$):
  Calculate stability of ℓ
Select CTLs from *B:*
  Case 1. Select k $\ell \in B$ with the highest *stability*
  Case 2. Select all $\ell \in B$ with *stability* > τ_b
  Case 3. Pruning with relative threshold of stability:
    Select all $\ell \in B$ with $Stability \geq n/m$ top stability in B

4 Experiments and Evaluation

4.1 Dataset and Experimental Setup

A set of 90 resources was chosen for this study and each document was tagged with at least one CITY, STATE/PROVINCE or COUNTRY name which indicates the most relevant locations to that page. Textual pages, photos and sound files were included in the dataset. We used addresses and other geographical information to judge the most relevant locations for a resource. For example, if a webpage is talking about a restaurant in Paris, it would be tagged with Paris as its location. A picture of this restaurant will be tagged with Paris too. The next step was finding the location of visitors. The log files of all selected resources were collected in 12 weeks. We pick an IP from a log and search it in online databases like Whois and RIPE[3] and automatically extract country codes, telephone codes and addresses. The output of this procedure is a list of locations accompanied with IPs. Locations were disambiguated manually as this wasn't the aim of this paper.

Privacy is a big issue in a visitor-based approach. Only authorized people have access to log files. It is notable that for search engines, the privacy problem could be solved with a substitute method. Instead of analyzing real server log files, search engines can track and analyze URLs. A URL represents a Web resource and therefore, clicking on a URL link can be interpreted as using the corresponding

[3] http://www.ripe.net/

resource. We have used server log files whenever they were available; otherwise, websites with traffic and statistic tools were selected for our dataset. Most of the resources selected for this research use either Webstats4u[4] or Statcounter[5] services with a setting that allows all people to see the history of visits. The total number of analyzed visitor entries is 5530 and for unique visitors it was 3563.

After extracting different locations from the logs, *power* and *spread* formulas were used respectively to find the weight of each location. The spread formula requires the number and weight of sub-locations. Any unrecognized sub-location was counted as a sub-location called OTHER to facilitate calculation of geographic distribution. We have limited the granularity of our model to city level as it is hard to disambiguate suburbs and streets inside a city. As a result, we have treated cities different from states and countries and the spread weight is considered 1 for all cities.

The procedure continues with using a threshold of spread τ_s to select candidates and then select temporal target locations (TTLs) with *top-k* model, fixed and relative pruning models described in 3.3. Power was used as a threshold τ_p to select TTLs in a 12-week period. At the end of this period, the *stability* of TTLs were calculated and used as a threshold τ_b to select constant target locations (CTLs) with three pruning models.

Recall, precision and f-measure have been used to measure the accuracy of visitor-based model. As we have already tagged each document in our dataset with a correspond- ing geographic name, we define recall and precision as following (here for TTLs):

$$\mathrm{Re}\,call = \frac{TTLs\_estimated}{(TTLs\_estimated) + (TTLs\_not\_estimated)}$$

$$\mathrm{Pr}\,ecision = \frac{TTLs\_estimated}{(TTLs\_estimated) + (Others\_estimated)}$$

As our approach might find more than one target location for each document, we use the average precision formula to measure how accurately it ranks the calculated locations. Average precision is the sum of the precision at each relevant hit divided by the total number of relevant documents in the collection:

$$Average\ \mathrm{Pr}\,ecision = \sum_{j=1}^{j=n} \frac{\mathrm{Pr}\,ecision(j) * \mathrm{Re}\,levance(j)}{R}$$

where j is a hit in the hit list and R is the total number of relevant documents in the entire dataset. *Relevance (j)* is 0 if it is not a relevant hit and 1 if it is relevant.

4.2 Experimental Results

We have evaluated our proposed model in different ways. Because of the limitation on the length of this paper, a selection of our experiments and results are presented here. Table 1 shows how successfully our model returns location names for an IP. The total number of IPs is 5530. The last column indicates that using online databases such as

[4] http://www.webstats4u.com/
[5] http://www.statcounter.com/

Whois, our model can return at least one country for a single IP in 98.77% of all cases. The successful tagging rate is around 78% and 74.5% for STATE/PROVINCE and CITY levels consequently. This indicates that target location can be applied to almost all Web resources at country level and to a majority of them at state and city levels.

We have used the geographic data to calculate the *target location* of documents in the dataset. As an example, we show how target locations are calculated for the webpage *http://www.boroujerd.info/english.htm* which is related to Boroujerd City in the Lorestan Province of Iran. Table 2 summarizes the results for calculation of power and spread in the 1st week as well as calculation of stability and CTLs in the end of the 12th week. The spread threshold τ_s is 0.6. In the Top-*k* model, *k*=3. The fixed threshold of τ_p=0.150 and in the relative model, τ_p=35%. At the end of the 12th week, we calculate the *stability* of each TTL and then estimate CTLs. Again for the Top-k model, k=3. For the fixed threshold pruning, τ_b=0.150. And finally, for the relative threshold model, τ_b=15% of the highest stability in the TTL set.

Table 1. Successful tagging of IPs with locations

	No. of IPs with zero match	No. of IPs with 1 match	No. of IPs with 2 or more matches	% of success in the total set
COUNTRY	68	5343	119	98.77
STATE/PROVINCE	1216	3924	390	78.01
CITY	1414	3717	399	74.43

Table 2. Calculation of power and spread in the 1st week; and calculation of stability and CTLs in the end of the 12th week for the webpage http://www.boroujerd.info/english.htm

Measure		Level	Matched	1st Top	2nd Top	3rd Top
Power in week 1		COUNTRY	35	Iran	USA	UAE
		STATE/PROVINCE	31	Lorestan (Iran)	Tehran	California
		CITY	31	Boroujerd	Tehran	Dubai (UAE)
Spread in week 1		COUNTRY	35	Iran	USA	Germany
		STATE/PROVINCE	31	California	Isfahan (Iran)	New York
		CITY	31	-	-	-
Stability		COUNTRY	85	Iran	USA	UAE
		STATE/PROVINCE	61	Tehran (Iran)	Khorasan (Iran)	Lorestan
		CITY	50	Tehran	Borujerd	Dubai
Constant Target Locations (CTL)	Top-3	COUNTRY	3	Iran	USA	Germany
		STATE/PROVINCE	3	Tehran	Lorestan	California
		CITY	3	Boroujerd	Tehran	Dubai
	Absolute T=0.15	COUNTRY	2	Iran	USA	-
		STATE/PROVINCE	2	Tehran	Lorestan	-
		CITY	4	Boroujerd	Tehran	Dubai
	Relative T=15%	COUNTRY	5	Iran	USA	UAE
		STATE/PROVINCE	2	Tehran	Lorestan	-
		CITY	2	Boroujerd	Tehran	

Table 3 shows recall and precision of our model for calculation of constant target locations (CTLs). The last row shows that the recall of our model is at least 0.80 using any pruning approach and the precision is 0.64 or more.

The basic precision formula indicates how many correct target locations our system can find. However, it cannot determine how accurately the results are ranked. Ideally, our model must be able not only to find correct locations but also to rank them based on their relevance to documents. For example, if a Web resource w has been judged to be geographically related to Australia, New Zealand or Fiji respectively, our model must be able to select these locations and rank them correctly. Average precision has been used to evaluate how well our model ranks target locations. Fig. 2 shows the average precision of different strategies used to select constant target locations. The Top-*k* pruning model shows a higher average precision than the other techniques; it can rank the most relevant target locations higher than other locations better than other pruning models. It is also indicated that visitor-based algorithm performs better for calculation of target location at country level.

Table 3. Recall and Precision of three models for calculation of constant target locations (CTLs) at different geographical levels

	Top-k		Absolute *T*		Relative *T*	
	Recall	Precision	Recall	Precision	Recall	Precision
Country	0.92	0.68	0.93	0.51	0.88	0.62
State/Province	0.88	0.75	0.80	0.69	0.81	0.67
City	0.87	0.76	0.67	0.71	0.81	0.67
All locations	0.89	0.73	0.80	0.64	0.83	0.65

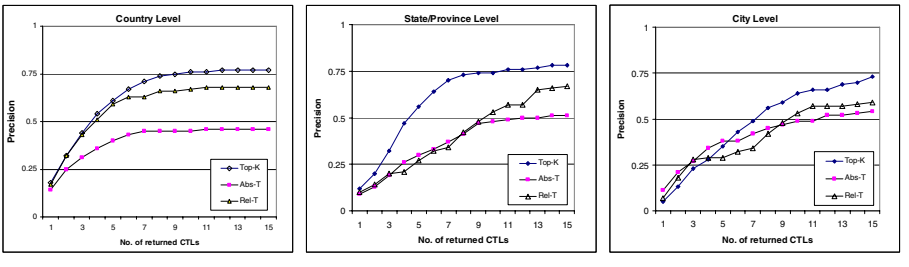


Fig. 2. Average precision of different pruning models for selection of constant target locations.

We have compared our visitor-based model with content-based and link-based approaches. Table 4 indicates the rate of successful extraction or assignment of locations to Web resources using different models. The visitor based approach receives the highest score in country and state levels. This is because of the vast information which is stored with registered IPs.

Table 5 summarizes the results of comparing f-measures of different pruning techniques in visitor, link and content-based approaches. The visitor-based approach works better at country level. The link-based approach is better at city level.

Table 4. A comparison among visitor, link and content based approaches for accuracy of extraction of correct location names

	Visitor	Link	Content
Country	98%	76%	64%
State/Province	78%	69%	58%
City	78%	76%	65%
All locations	85%	74%	62%

Table 5. A comparison of the f-measures of visitor, link and content based approaches for calculating the target location of Web Resources

	Top-k			Absolute Threshold			Average Threshold		
	Visitor	Link	Content	Visitor	Link	Content	Visitor	Link	Content
Country	**0.85**	0.77	0.65	**0.68**	0.52	0.66	**0.76**	0.74	0.68
State	**0.74**	0.72	0.58	0.58	0.5	**0.61**	0.66	**0.74**	0.66
City	**0.73**	**0.73**	0.65	0.57	0.5	**0.66**	0.65	**0.74**	0.69
Average	**0.77**	0.74	0.63	0.61	0.51	**0.64**	0.69	**0.74**	0.68

5 Conclusion

A location-based search engine must be able to relate Web resources with geographical locations. In this paper we showed that content and metadata information are not sufficient to judge the geography of a webpage. Information extraction techniques in isolation cannot address the location of non-textual resources on the Web. Target location was introduced in this paper as the locality of the majority of users or visitors of a webpage. Target location is content-independent and can be applied to different media types on the Web. We introduced a novel model for extraction and calculation of target locations. Target location can be calculated and for almost all types of media on the Web, best at country level. It can be assigned to Web resources as a new definition and dimension of location. It can also be used separately or in conjunction other relevant locations to define the geography or location of a Web resource.

References

1. Sullivan, D.: Searches per day. *Search Engine Watch* (April 20, 2006). Available at: http://searchenginewatch.com/reports/article.php/2156461 [Last visit April 22, 2006].
2. Sanderson, M., Kohler, J.: Analyzing geographic queries. Proceedings of Workshop on Geographic Information Retrieval *GIR* in *SIGIR'04,* Sheffield, UK (2004).
3. Asadi, S., Chang, C., Zhou, X Diederich, J.: Searching the World Wide Web for local services and facilities: A review on the patterns of location-based queries. *WAIM'05*, Hong Zhou, China (2005).
4. Watters, C., Amoudi, G.: GeoSearcher: Location-based ranking of search engine results. *JASIST*, 54(2) (2003) 140-151.
5. Amitay, E., Har'El, N., Sivan, R., Soffer, A.: Web-a-Where: Geotagging Web Content. *SIGIR'04*, Sheffield, UK (2004) 273-280.

6. Ding, J., Gravano, L., Shivakumar, N.: Computing geographical scopes of Web resources. Proceedings of the 26th *VLDB* Conference, Cairo, Egypt (2000).
7. Newcomb, K.: Google Gets Local in Canada. ClickZ News, (Sep. 23, 2004).
8. Available at: http://www.clickz.com/news/article.php/3411681 [Last visit M 12, 2006].
9. Buyukkokten, O., et al.: Exploiting geographical location information of Webpages. *SIGMOD WebDB'99*, Philadelphia, USA (1999).
10. Pouliquen, B., et al.: Geographical Information Recognition and Visualization in Texts Written in Various Languages. *SAC'04*, Nicosia, Cyprus (2004).
11. Olga, O.: Extracting Geographical Knowledge from the Internet. Proceedings of
12. ICDM-AM International Workshop on Active Mining, Maebashi, Japan (2002).
13. Gravano, L., et al.: Categorizing Web Queries According to Geographical Locality. *CIKM'03*, New Orleans, USA (2003).
14. Li, H., et al.: Location normalization for information extraction. Proceedings of 19th Int'l Conference on Computational Linguistics, Taipei (2002) 549-555.
15. Tu, H.: Pattern recognitions and geographical data standardization. Proceedings of Geoinformatics'99 Conference, Ann Arbor (1999) 1-7.
16. Markowetz, A. et al.: Design and implementation of a geographic search engine. *WebDB'05*, Baltimore, Maryland, USA (2005).
17. Woodruff, A. G., Plaunt, C.: Gipsy: Automated geographic indexing of text documents. *JASIST*, 45(9) (1994) 645-655.

Efficient Bid Pricing Based on Costing Methods for Internet Bid Systems

Sung Eun Park[1] and Yong Kyu Lee [2, *]

[1] Department of Computer Engineering, Dongguk University
[2] Department of Computer Engineering, Dongguk University
Pil-dong, Jung-gu, Seoul 100-715, Republic of Korea
{pse76, yklee}@dongguk.edu

Abstract. Internet bid systems are being widely used of late. In these systems, the seller sets the bid price. When the bid price is set too high compared with the normal price, chances of a successful bid may decrease. When it is set too low, however, based on inaccurate information, it can result in a successful bid yet one with no profit at all. To resolve this problem, an agent is proposed that automatically generates bid prices for sellers based on the similarity of the bidding parameters using past bidding information as well as on various costing methods such as the high-low point method, the scatter diagram method, and the learning curve method. Performance experiments have shown that the number of successful bids with appropriate profits can be increased using the bid pricing agent. Among the costing methods, the learning curve method has shown the best performance. The manner of designing and implementing the bid pricing agent is also discussed.

1 Introduction

Internet bid systems are being widely used of late. In these systems, the sellers set the bid prices when the items to be sold are registered for bidding. The bidding can be unsuccessful if the bid price is unreasonably high compared with the normal price. Moreover, a successful bid may result in a loss on the part of the seller when the latter carelessly sets the bid price too low. This could occur when sellers do not have accurate information about normal prices or when they make mistakes during the registration. Therefore, an unsuitable bid price is one of the main reasons for an unsuccessful bidding.

Recommendation systems for the activation of e-commerce systems have been used to propose new items to prospective buyers according to their interests [4][7][17][18]. The system handles only items that are likely to be purchased by people, and does not deal with pricing. Even though a program for buying can be found that automatically generates the estimated price for buyers, it is meant only for buyers and not for sellers. Therefore, it is difficult to determine which price will yield a high profit for sellers [2][9][12][16]. In addition, sellers may face certain problems. First, they may earn but a small profit if a successful bid comes with a low bid price. They may also sustain a heavy loss from the total selling price by calculating the price of one item by mistake,

* Corresponding author.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 289 – 299, 2006.

etc. Therefore, a precise cost calculation method is required that would enable sellers to set an appropriate bid price--that is, one that can decrease the probability of a successful bid at a loss and increase the probability of a successful bid at a profit.

To solve this problem, this paper describes a method of designing and implementing a bid pricing agent for Internet bid systems. The agent automatically generates bid prices for sellers based on the similarity of the bidding parameters to past bidding information as well as on costing methods such as the high-low point method, the scatter diagram method, and the learning curve method. To earn an appropriate profit from the bidding, a bid price can be set using the agent. Also by using the agent, unsuccessful bids at too high prices can be decreased, to yield reasonable profits.

Through experiments, the performance of the proposed bid pricing agent is evaluated using a number of costing methods. In these experiments, the learning curve method performed best. The experiments also showed that the rate of successful bids with appropriate profits can be increased by preventing sellers from making unreasonably low or high bid prices compared to normal prices.

2 Related Work

Agents with various functions have been developed for the activation of the e-commerce system [10][14][17][18]. That is, a commodity recommendation system has been developed by analyzing the past purchase records of purchasers or shopping agents, which helps purchasers to shop by providing them with e-commerce shopping information. In addition, studies have been conducted not only to describe these commodity recommendation functions, but also to systematically solve price and profit problems by developing an intelligent agent based on the margin push multi-agent system for Internet auctions [2][12][16]. Even though these studies aim to simply activate the electronic commerce market, however, because the margin algorithm is applied using the bidding history database or the auction method, it is difficult to understand how similar the estimated winning bid prices are to prices that can yield profits.

Furthermore, a study [6] has recently been conducted to suggest a reserve price suitable to the corresponding goods after a search for the most similar goods cases is made using past purchase records and current market prices for purchasers in the group buying system. Another study [9] has also been performed that used the simple high-low point method to estimate costs by linking the highest and lowest points among cost account theories to create bidding prices. These methods are also price suggestions according to purchasers' purchase records, however, and when the high and low points are used, the highest and lowest points occur in irregular conditions and are too weak to be used as representative values [3][11].

Accordingly, this study shall examine a costing method that yields proper profits and successful bids for sellers by comparing and analyzing more costing methods.

3 Bid Pricing Agent

3.1 Generating the Bid Price

Fig. 1 illustrates the procedure whereby an appropriate bid price for a bid item is generated. The bidding history database stores records on bid prices, successful bid prices,

and successful bid rates with detailed item information from past biddings. The company cost database contains information about item production costs such as the total cost of the items, the quantity of the items, and the selling price of the items, among others. The total cost of the items consists of the depreciation cost, wages, the material cost, and other expenses, among others. The buying history database stores past purchase records.

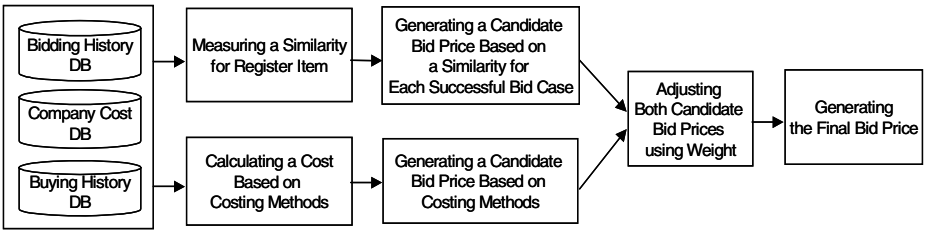


Fig. 1. Bid Pricing Procedure

First, to retrieve similar cases from the bidding history database, the similarity measure is used. Information is obtained on past bidding items similar to the item being bidded. One candidate bid price is obtained by applying the weighted average of the retrieved successful bid prices. Second, another candidate bid price is obtained using such costing methods as the high-low point method, the scatter diagram method, and the learning curve method. The final bid price is determined by calculating the weighted average of the two candidate bid prices.

The price that will yield an appropriate profit using the past winning bid price shall be the final bid price.

The aforementioned explanation is shown in Eq. 1. The details of the similarity measure and the costing methods will be discussed in the next sections. The final bid price is obtained using Eq. 1, as follows:

$$P_{bid-price} = w_1 \times P_{price-by-costing-method} + w_2 \times P_{price-by-similarity} \tag{1}$$

in which $P_{bid\text{-}price}$ is the final bid price, $P_{price\text{-}by\text{-}costing\text{-}method}$ is the candidate bid price based on the costing method, $P_{price\text{-}by\text{-}similarity}$ is the candidate bid price based on the similarity measure, and w_1 and w_2 are the weights ($w_1 + w_2 = 1$).

After the creation of the candidate bid prices, the values of w_1 and w_2 are set at 0.65 and 0.35, respectively. The value of w_1 in Fig. 6. is thus obtained.

3.2 Similarity Measure

The similarity measure is used to retrieve ranked similar cases from databases. That is, after searching the bidding history database, information on past bid items similar to the item to be bidded is obtained. The bid prices of similar items can also be searched for in the bidding history database. The obtained information is used in the next section to generate candidate bid prices. The similarity measure is as follows:

$$similarity \quad (\boldsymbol{r}_j, \boldsymbol{q}_k) = \frac{\sum_{i=1}^{n} (\boldsymbol{tr}_{ij} \times \boldsymbol{tq}_{ik})}{\sqrt{\sum_{i=1}^{n} \boldsymbol{tr}_{ij}^2 \times \sum_{i=1}^{n} \boldsymbol{tq}_{ik}^2}} \tag{2}$$

in which r_j is the record of the j^{th} case, q_k is the query of the k^{th} case, tr_{ij} is the i^{th} term in the vector for record *j*, tq_{ik} is the i^{th} term in the vector for query *k*, and *n* is the number of fields in the record. From the results of the search of the bidding history database, the candidate bid price is calculated using the following formula:

$$P_{price-by-similarity} = \left(\frac{1}{n} \sum_{i=1}^{n} w_i \cdot p_i \right) \times Q_{total} \tag{3}$$

in which *n* is the number of similar cases, w_i is the weight of the i^{th} case, p_i is the unit bid price of the i^{th} case, and Q_{total} is the total selling quantity.

3.3 Costing Methods

In this section, representative costing methods among cost account theories are explained and costs are calculated using three costing methods: the high-low point method, the scatter diagram method, and the learning curve method. Eq. 4 is equally applied to these three costing methods, as follows:

$$P_{price-by-\cos ting-method} = a + b \times Q_{total} \tag{4}$$

in which $P_{price\text{-}by\text{-}costing\text{-}methodt}$ is the total cost, *a* is the fixed unit cost, *b* is the variable unit cost, and Q_{total} is the total quantity. The manners of calculating parameters *a* and *b* differ for each costing method. Therefore, the total cost is also presumed to be different.

The High-Low Point Method. The high-low point method [1][3][11], which is part of the statistical presumption method, is a method of presuming the fixed unit cost and the variable unit cost using cost data on the highest operating rate and the lowest operating rate among past cost data. This method is an equation connecting the highest point value to the lowest point value on a straight line. It is used to get the variable unit cost *b*, as shown in Eq. 5 below:

$$b = \frac{P_{\cos t-\max} - P_{\cos t-\min}}{Q_{\max} - Q_{\min}} \tag{5}$$

in which *b* is the variable unit cost, $P_{cost\text{-}max}$ is the highest total cost, $P_{cost\text{-}min}$ is the lowest total cost, Q_{max} is the highest quantity, and Q_{min} is the lowest quantity. The fixed unit cost *a* is obtained from Eq. 6 below, using *b*, calculated from Eq. 5:

$$a = P_{month} - \left(b \times Q_{month} \right) \tag{6}$$

in which *a* is the fixed unit cost, *b* is the variable unit cost, P_{month} is the highest total cost or the lowest total cost for a month, and Q_{month} is the highest quantity or the lowest quantity for a month.

The Scatter Diagram Method. The scatter diagram method [1][3][11] is a method that intuitively decides on the straight line that best explains the relationship between the costs and the work operations, among the observed data shown on the diagram.

Therefore, no specific equation is produced. In this paper, to use the scatter diagram method objectively, the standard deviation formula [8][13] is used, as shown in Eq. 7 below:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(p_i - m)^2 \tag{7}$$

in which σ is the standard deviation, n is the number of costs, p_i is the distributed cost of the i^{th} case, and m is the average of the distributed costs.

The Learning Curve Method. The learning curve method [1][3][11] represents the learning effect in which business productivity increases when workers repeat their fixed work. To obtain costs using the learning curve method, the simple regression analysis method and the least squares method are used. In the simple regression analysis method, a tropic is derived according to the least squares method by thinking of cost changes as straight lines that are related to an independent variable. In the least squares method, an equation is derived that minimizes the total squares of cost deviations. To determine costs using the simple regression analysis method, the formulas shown in Eqs. 8 and 9 were used. The values of a and b were obtained by combining Eqs. 8 and 9, as follows:

$$\sum_{i=1}^{n} P_i = na + b\sum_{i=1}^{n} Q_i \tag{8}$$

$$\sum_{i=1}^{n} Q_i P_i = a\sum_{i=1}^{n} Q_i + b\sum_{i=1}^{n} Q_i^2 \tag{9}$$

in which P_i is the average accumulated unit cost of the i^{th} case, Q_i is the accumulated quantity of the i^{th} case, a is the fixed unit cost, b is the variable unit cost, and n is the number of costs.

4 Design and Implementation of the Bid Pricing Agent

4.1 Design of the Bid Pricing Agent

Fig. 2 shows the bid pricing agent structure. First, to retrieve similar cases from the bidding history database, the similarity measure is used. Information on past bidding items similar to the item being bidded is obtained.

One candidate bid price is obtained by applying the weighted average of the retrieved successful bid prices. Second, the cost calculation module selects the data suitable to the costing policy by applying each costing method, derives the cost equation from the data, then calculates the cost using this cost equation.

Then another candidate bid price is obtained. Third, the final bid price generation module generates the final bid price of the seller using information on the two candidate bid prices. The bid history management module updates recent bid information, and the production cost management module updates recent cost information.

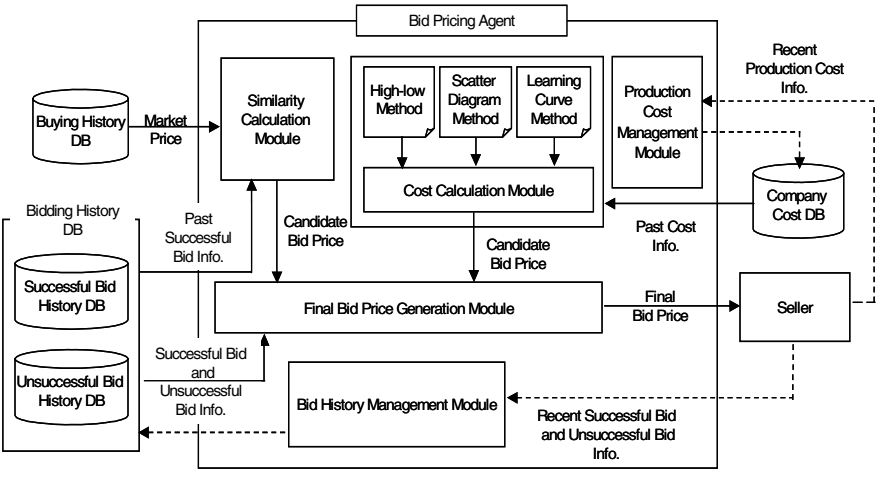


Fig. 2. Bid Pricing Agent Structure

4.2 Implementation of the Bid Pricing Agent

In this section, the bid pricing agent designed in the previous section is implemented and its main windows are explained. The bid pricing agent consists of a cost calculation module and a final bid pricing module. The cost and the bid price can be generated procedurally using the agent. The agent was developed using Visual Basic 6.0.

Fig. 3 shows the window for the cost calculation module. The window displays information on bid items and data on production costs from the databases. If a seller clicks the 'calculation' button in the window, he or she can see the cost calculated by the selected costing method.

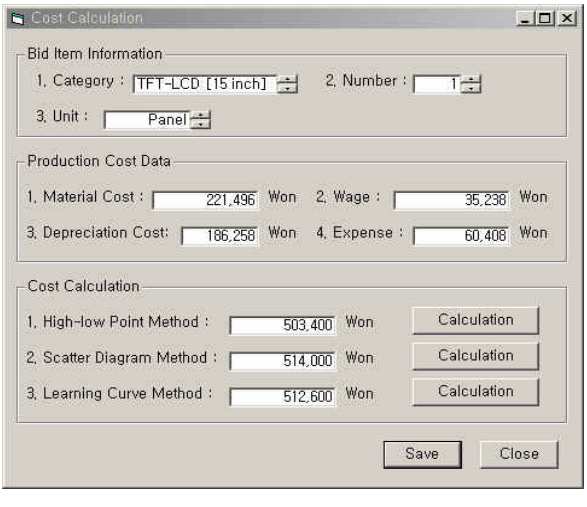


Fig. 3. Cost Calculation Window

Fig. 4 shows the window for the bid pricing module. The window displays the results of the calculation of a standard unit bid price, a standard unit profit, and a standard unit profit rate using the cost calculated in Fig. 3. It also calculates the total bid price, the total profit, and the total profit rate according to the bid quantity.

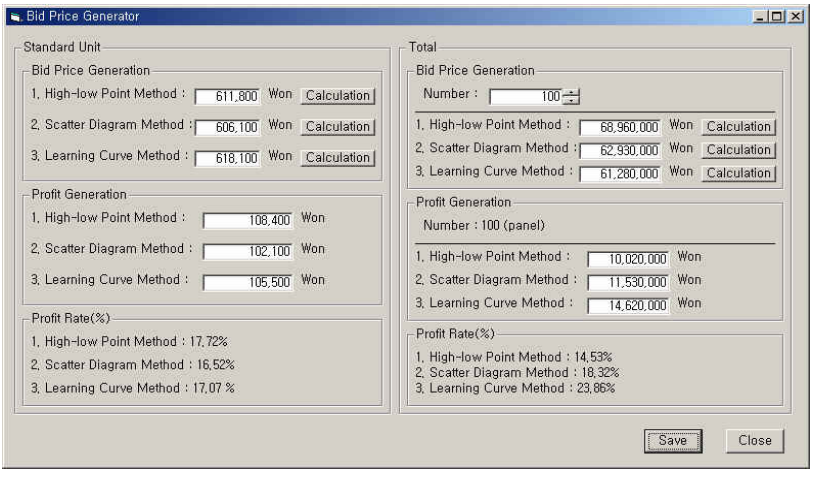


Fig. 4. Bid Pricing Window

5 Experimental Evaluation

5.1 Experimental Environment

Performance experiments were performed using data for a 15-inch TFT-LCD notebook monitor such as its total cost, its total selling quantity, its selling price, etc., which are presented in researches on the LCD industry [15][19]. For the performance experiments, 120 cases were simulated from the aforementioned data. From the performance experiments, the possibility of successful bids and the expected profit of the seller can be compared both when no agent was used and when an agent was used with different costing methods.

For this comparison, the difference between the real cost and the calculated cost based on a costing method is calculated using the adjusted AE (Absolute Error) measures, as shown in Eq. 10 below:

$$E_1 = \left| l_i - m_i \right| / l_i \tag{10}$$

in which l_i is the real cost of the i^{th} case, and m_i is the calculated cost based on the costing method of the i^{th} case.

The difference between the real bid price and the bid price generated by the agent is calculated using the adjusted MAE (Mean Absolute Error) measures, as shown in Eq. 11 below:

$$E_2 = \frac{1}{n} \sum_{i=1}^{n} \left(\left| b_i - v_i \right| / b_i \right) \tag{11}$$

in which n is the number of items, b_i is the real bid price in the i^{th} case, and v_i is the bid price generated by the agent in the i^{th} case.

5.2 Experimental Results

Table 1 shows the results of the performance experiments. The Successful Bid Rate means that the success rate to all bids without failure. In this section, both the Profit Case and the Loss Case mean that profits were made in successful bids or not.

Table 1. Experimental Results of Total Bid Cases

	No Agent	High-low Point Method	Scatter Diagram Method	Learning Curve Method
Successful Bid Rate	55%	48%	47%	44%
Loss Bid Rate	6.5%	5.8%	5.6%	5.3%

Comparison of the Error. From the previous section, an error value is represented between the real cost and the cost based on each costing method with 120 bidding data. The E_1 values of the learning curve method were lower than those of the other costing methods in the graph. This means that the learning curve method can calculate the cost more closely to the real cost than the other methods.

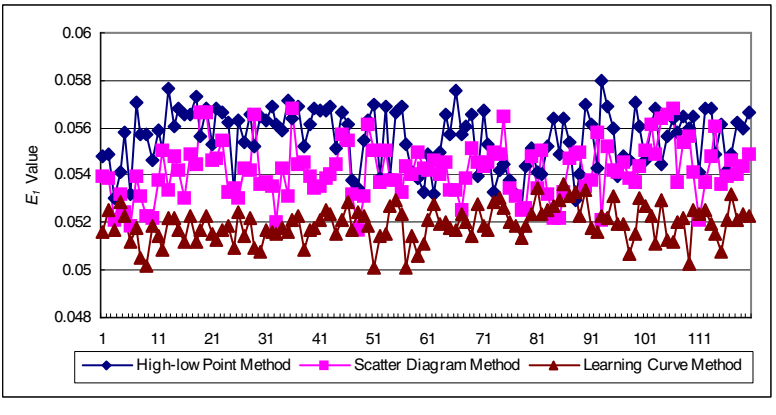


Fig. 5. Comparison of E_1 Value

The performance of the bid price of the formula in Eq. 11 was tested with varying w_1 values. Fig. 6 shows the results, in which the performance was best when w_1 was 0.65.

Comparison of the Successful Bid Rates. Fig. 7 (a) compares the successful bid rates with cases when profits were made in the number of successful bids. The

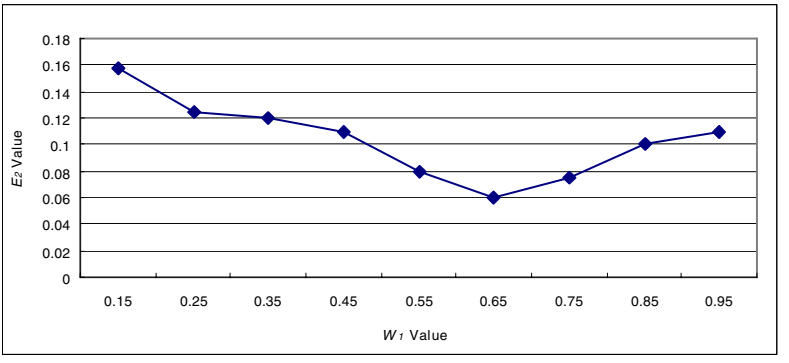
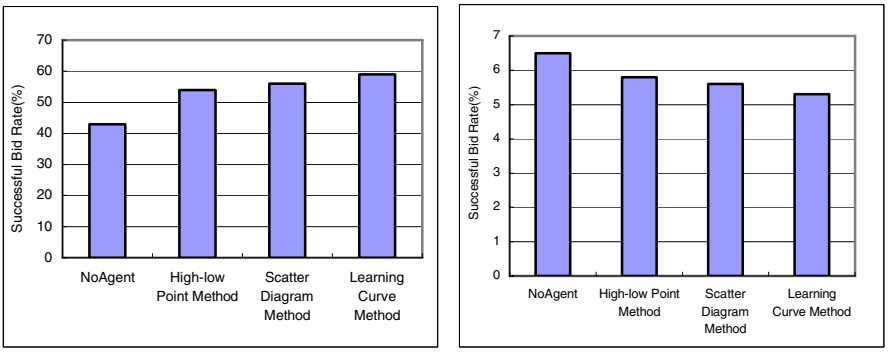


Fig. 6. Performance of w_1 Value

successful bid rate when an agent was used with the costing methods was higher than when no agent was used. It can be seen that the agent using the learning curve method is excellent.

Fig. 7 (b) compares the rate of successful bids with cases when losses were suffered. The rate of successful bids when losses were suffered is generally low. It can be seen that the rate of successful bids when losses were suffered is higher when no agent was used. This means that the rate of participation in the bidding with losses was higher when no agent was used.



(a) Profit Cases (b) Loss Cases

Fig. 7. Comparison of Successful Bid Rates

Comparison of the Average Profit per Unit. The average profit per unit according to the cumulative selling quantity of 15-inch TFT-LCD notebook monitors is represented in Fig. 8. From this graph, the profit patterns of each method can be analyzed according to the cumulative selling quantity.

When no agent was used, the spread of the average profit was significant and the average profit was relatively low. When the high-low point method was used, the spread of the average profit was more significant than when other costing methods

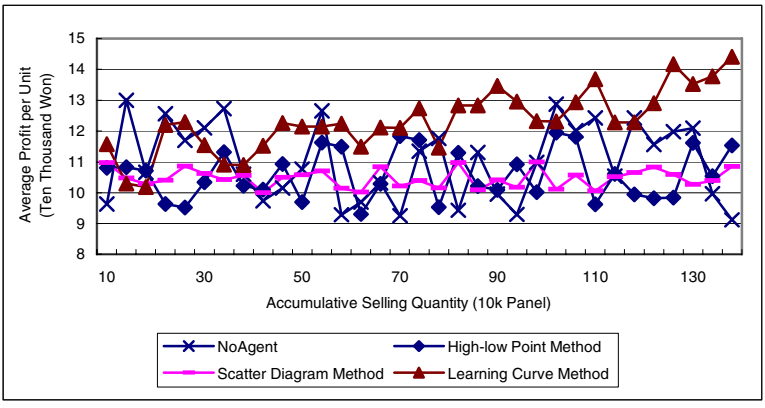


Fig. 8. Comparison Average Profit per Unit as Accumulative Selling Quantity

were used, but the average profit was higher than when no agent was used. When the scatter diagram method was used, the spread of the average profit was insignificant and the average profit was low. When the learning curve method was used, the spread of the average profit was insignificant but the average profit was relatively high. In addition, the more cumulative the selling quantity was, the higher the average profit was.

6 Conclusions

In this paper, a bid pricing agent has been proposed and implemented that automatically generates bid prices for sellers based on the similarity of the bidding parameters with previous pricing information as well as on costing methods such as the high-low point method, the scatter diagram method, and the learning curve method.

The performance experiments conducted showed that the proposed bid pricing agent can increase the probability of successful bids with appropriate profits by preventing sellers from making unreasonable bid prices. In particular, in the experiments using the test database of 120 cases obtained from researches on the LCD industry, the rate of successful bids with appropriate profits was highest and the rate of successful bids with losses was lowest when the agent was used with the learning curve method. This means that the agent, used with the learning curve method, can decrease the rate of successful bids with losses by preventing biddings from becoming successful due to too low a bid price compared to the normal price. In addition, it can decrease the probability of unsuccessful bids with prices that are too high to yield much profit.

In the future, experiments on this bid pricing agent with a large database of various bidding items are required. In addition, we make an effort to research on e-commerce solutions with the bid pricing agent.

References

1. Barfield, J. T., et. al.: Cost Accounting:Traditions and Innovations. 5th edn. South Western College Pub (2002)
2. Dasgupta, P., Das, R.: Dynamic Service Pricing for Brokers in a Multi-Agent Economy. Proc. of the 4th Int'l Conf. on Multi Agent Systems, Boston USA (2000) 375-376
3. Horngren, C. T., et. al.: Cost Accounting. 12th edn. Prentice Hall (2005)
4. Jang, E. S., Lee, Y. K.: A Method of Recommending Buying Points for Internet Shopping Malls. Proc. of the 10th Int'l Conf. on Knowledge-Based & Intelligent Information & Engineering Systems, Bournemouth UK (2006) (to apper in LNCS)
5. Jeng, J. J., Chang, H., Chung, J. Y.: A Policy Framework for Business Activity Management. Proc. of the IEEE Int'l Conf. on E-Commerce Technology, California USA (2003) 238-245
6. Ko, M. J., Kim, S. W., Park, S. E., Lee, Y. K.: A Reserve Price Generation Agent for an Internet Auction System. Journal of Korea Information Processing Society, Vol. 9-D No. 5 (2002) 955-962
7. Ko, M. J., Lee, Y. K.: Reserve Price Recommendation by Similarity-Based Time Series Analysis for Internet Auction Systems. Proc. of the 10th Int'l Conf. on Knowledge-Based & Intelligent Information & Engineering Systems, Bournemouth UK (2006) (to apper in LNCS)
8. Larson, R., Farber, E.: Elementary Statistics : Picturing the World. 3rd edn. Prentice Hall (2005)
9. Lee, Y. K., Kim, S. W., Ko, M. J., Park, S. E.: Pricing Agents for a Group Buying System. Lecture Notes in Computer Science, Vol. 2510 (2002) 693-700
10. Lu, L., Zhang, Y. Q.: Intelligent Mobile Agents for Efficient and Inexpensive e-Shopping. Proc. of the 27th Annual Int'l Computer Software and Applications Conf., Texas USA (2003) 607-609.
11. Maher, M. W., et. al.: Fundamentals of Cost Accounting. McGraw-Hill/Irwin (2004)
12. Marshall, M.: Dynamic Pricing for E-Commerce, an Integrated Solution. Proc. of the 3th Int'l Workshop on WECWIS 2001, CA USA (2001) 192-194
13. McQuarrie, D. A.: Statistical Mechanics. University Science Books (2002)
14. Mehrdad, J. S., Feza, B.: A Multimodal Shopping Assistant for Home E-Commerce. Proc. of the American Association for Artificial Intelligence, Florida USA (2001) 2-6
15. Park, S. B.: A Study on the Analysis of Firm-specific Learning Curves: Application to the TFT-LCD Industry. The Graduate School of Seoul National University (2003)
16. Raju, C. V. L., Narahari, Y., Ravikumar, K.: Reinforcement Learning Applications in Dynamic Pricing of Retail Markets. California USA (2003) 339-346
17. Raposo, J., et. al.: A Web Agent for Automating E-Commerce Operations. Proc. of the IEEE Int'l Conf. on E-Commerce Technology, California USA (2003) 16-19
18. Xiao, B., Aïmeur, E., Fernandez, J. M.: PCFinder: An Intelligent Product Recommendation Agent for E-Commerce. Proc. of the IEEE Int'l Conf. on E-Commerce Technology, California USA (2003) 181-188
19. Yoo, J. H.: A Study on The Analysis of Learning Curve in LCD Industry. The Graduate School of Seoul National University (2003)
20. Yukawa, T., et. al.: An Expert Recommendation System Using Concept-Based Relevance Discernment. Proc. of the 13th Int'l Conf. on Tools with Artificial Intelligence, Dallas USA (2001) 257-264

An Enhanced Super-Peer Model for Digital Library Construction

Hao Ding, Ingeborg Sølvberg, and Yun Lin

Dept. of Computer and Information System, Norwegian Univ. of Sci. &Tech.
Sem Sælands vei 7-9, NO-7491, Trondheim, Norway
{haowing, ingeborg, yunl} @idi.ntnu.no

Abstract. Peer-to-Peer (P2P) overlay network has emerged as a major infrastructure for constructing future digital libraries. Among various P2P infrastructures, super-peer based P2P network receives extensive attention because the super-peer paradigm allows a node to act as not just a client, but also serve for a set of clients. As different from conventional file-sharing paradigm, digital library applications have more advanced requirements on system independence/autonomy, robustness and flexible communication. This paper is devoted for constructing digital library systems built upon such super-peer based network, i.e. JXTA framework. Evaluation results are to be presented concerning network initialization, loading balancing and self-organizing.

1 Revisit Super-Peer System Model for Digital Library Construction

1.1 Requirements in Constructing Future Digital Libraries

Digital libraries, evolved from traditional libraries, share a general characteristic of making information sources available to a wider audience. However, due to the huge varieties in coverages, subjects, representations and operabilities (e.g. operation systems), a single library is absolutely not able to provide all what one wants. Therefore, many mutually interested digital libraries need to find a way to share their information. In the 1990s, client/server computing architecture became at the peak of its popularity, and many digital libraries had been built upon such architecture. However, the client/server architecture was also accompanied with some serious problems:

- Performance bottleneck: servers become bottleneck when too many requests are sent across the network.
- Dependence: all clients depend heavily on servers. That is, clients have to trade their *autonomy* to reach an agreement with each other, such as accessibility, data structure, etc.

It is thus necessary to balance the load on servers, e.g. separating requests to multiple 'servers' instead of a unique one; and support autonomy for those libraries which require independences.

In the rest of this paper, an enhanced super-peer model will be presented, in terms of alleviating potential problems in classic super-peer systems. Evaluations on this model will be presented as well.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 300–307, 2006.

1.2 Classic Super-Peer System Models

To make it consistent, we bind the understanding of super-peer network to the definition in [1]. According to Yang[1], *super-peer networks* present a cross between pure and hybrid systems. A *super peer* acts as a central server to a subset of clients. Clients submit queries to the corresponding super peer and receive results from it. Basically, connections among super peers form a *pure* P2P system, and super peers are responsible for submitting and answering queries on behalf of client peers and themselves.

A super-peer and its clients are called a *cluster*, where *cluster size* is the number of nodes in the cluster, including the super-peer itself. In an extreme condition, a pure P2P network can be regarded a regressive super-peer network where cluster size is 1 - every node is a super peer with no clients [1].

Applying super-peer networks to digital library construction, we can regard a cluster as a set of digital libraries where one of them is selected or simply agreed by other libraries as a super node. To maintain such a cluster, this super node can keep an index over its clients' data or simply provide a lookup service. In file sharing applications, the super peer may just keep inverted keyword lists over the titles of files owned by its clients. If the super peer finds any results, it will return with message including results and the address of clients. But in a more advanced situation where it is impractical or expensive to keep a central index, e.g., advanced search requirements on multiple metadata fields or SQL-alike search, the super peer evolves into a 'lookup' server providing coordinating services for its clients. When a super peer receives a query from a client, it will submit the query to its neighboring super peers (*neighbors* for brevity in the rest paper) as if it was its own query, and it will forward any returned messages back to the requesting client. That is, outside the cluster, a client's query will not be distinguishable from a super peer's query. The advantage of this mechanism is that clients are saved from processing any extra queries and network traffic, so weak nodes (e.g. nodes having limited network bandwidth) could act as clients, while strong nodes can be united into an efficient network.

However, extreme scalability and dynamism pose a problem for a generic super peer network. In an extreme unbalanced situation, there is a high possibility that super peer are overloaded. Furthermore, when a super peer comes down or simply leaves, all its client peers will be disconnected from the network until they can find a new cluster to connect to. Although reliability can be increased by introducing redundancy, e.g. k super peers instead of one super peer, into the design of the super peer [1], such solution comes at a cost. Inside a cluster, there is an extra cost of maintaining clients' information on super peers and the communication among them would be k times greater than before. Also, the communication cost with neighbors increases since all super peers in a cluster may receive messages from neighbors. Hence, in this paper, instead of applying super peer redundancy, we come up with an enhanced super-peer model to alleviate previous mentioned problems.

2 Enhanced Super-Peer Model

To cope with the practical application, we consider a federated digital library system $\mathcal{DL}$ consisting of a large collection of *nodes* which exceed the capability of

conventional client/server model. We assume that all nodes in $\mathcal{DL}$, both client peer and super peer, can be located via unique identifiers. Also, $\mathcal{DL}$ is highly dynamic that new nodes may join at any time and existing nodes may come down or simply leave. Furthermore, a super-peer's capability is limited in contrast to the large scale of $\mathcal{DL}$, so an individual super peer can only connect to a constricted number of neighbors. To extend the capabilities of the super-peer model described in [1], we make the following enhancements.

Super-Peer Network Initialization. In many research projects, often are super-peer networks generated 'artificially'. For example, in [1] a graph-based topology is set up, where each of them represents actually a single cluster. These nodes are transformed into individual super peer and a number of clients are added to each super peer. The number of clients in a cluster follows the normal distribution $N(u_c, 0.2u_c)$, where u_c is the mean cluster size. However, the actual size of a cluster is decided in a 'bootstrapping' manner since it is a progressive procedure for client peers to discover super peers. Similar to [1], we assume also that there are initially a large number of average nodes $\mathcal{N}$ where no 'role' (i.e. client or super peer) has been assigned to each node yet. But as different from the approach in [1], we allow each node to pro-actively discover existing super peer to connect to, or declare itself as a super peer if it can not find super peer but has capability to accept incoming peers. In system $\mathcal{DL}$, initially there is no super peer available, so only those who have greater capability have the chances to become a super peer.

The rationality behind this extension is that: in practical situation, different peers may have varied computing capabilities or resources, so it is necessary to take into consideration the heterogeneity of peers. That is, weak nodes can be made into clients, while the core of the system can run efficiently on a network of 'strong' super peers. It is also in much closer to practical applications where peers' computing capabilities are varied.

Load Balancing. Load balancing is especially important for the super-peer based networks where it's difficult to predict the number of new client peers that may connect to a super peer or the number of requests that will be issued to the super peer. Consider the situation of peer joining, we observe that the actual number of clients in a cluster is determined by the maximum capacity c_s a super peer s can afford. Generally, c_s is determined by peer's computing capability, storage and bandwidth, etc. Hence, some super peers may have greater capabilities to accept new client peers, while other super peers can only host limited number of incoming peers. Clearly, in the latter situation, if the load can not be alleviated from the weaker super peers, performance bottlenecks (i.e. overloads) may occur in these nodes.

To alleviate such problem, we propose a two-step load balancing mechanism which is applied to distribute loads *dynamically* between a super peer and its neighbors. In the first step, when new nodes are added to the network, we initialize them all as super peers such that gossiping protocol can be reused for the reconnections with other super peers. Similar to the initialization, peers with lower capacities can be merged to a cluster led by a super peer with greater capacity. The next step which is also the critical one, is transparent to external observers. Generally, if a super peer can not accept a new client

any more, it may *push* such client peer to a neighbor whose current load does not reach the limit. On the contrary, 'strong' super peers who can afford greater load can also proactively *pull* client peers to their clusters. Behind the *push-and-pull* mechanism, a list of unclustered peers u_c act as a *coordinator* to accept 'abandoned' but active peers. As soon as a client node is added to a proper cluster, it is removed from the list.

Self-organizing. The model in [1] does not allow one client peer to communicate with other super peers except the one in its cluster. Under this assumption, if a super peer fails or leaves, all clients connected to it will be lost. To increase the robustness, we introduce the idea of self-organizing to cope with the situation when super peers are unavailable. Basically, first we assume that all nodes (i.e. both super and client peers) in our prototype have unique identifiers, as similar to that in Internet, which are used to locate individual nodes. Second, $\mathcal{DL}$ allows client or super peers sending messages to each other to check whether they are still online. Hence, if a super peer becomes unavailable, all client peers connected will be added into a 'unclustered' list. Each peer in this list, following the strategy of *Load Balancing*, can be assigned to particular clusters; or, simply declare themselves as a *new* super peer which in turn can accept connections from other client peers and finally lead to a new cluster.

3 Evaluation

3.1 Evaluation Setting

To validate our approaches, we have conducted numerous simulation-based experiments. The reason for applying a simulation-based approach is that: P2P systems are usually extremely large scale and dynamic systems where nodes may join and leave freely. However, experiments in a practical environment, e.g. thousands of nodes, turn out to be not an easy work at all. In addition, by applying simulation based approaches, it is convenient to set up different application scenarios to study P2P systems' behaviors which are of great interests to us.

A number of experiments have been conducted in this work in order to investigate the performance of different *configuration* of systems. Default configurations defined by a set of parameters are shown in Table 1. We are to explain these parameters in detail as they appear later in the section.

The configuration parameters in Table 1 describe the desired topology of the network. Unlike [1] where client peers are specifically added to a cluster, we generate a

Table 1. Configuration parameters and default values

Name	Default	Description
Super-peer Network Initialization	Gossiping protocol	The way how super-peer network is generated (the intial status).
Network Size	10^5	The number of peers in the network
Maximum Out-degree	30	The number of neighbors
Cluster size	power law distribution	The sizes of clusters are in a power law distribution
Max Capacity c_{max}	100	The maximum capacity of a peer to host client peers
Min Capacity c_{min}	1	The minimum capacity of a super peer to host client peer

10^5 size network by gossiping protocol [2], where each node is initialized as a super peer and randomly connected with each other. This scenario is the most 'artificial' one in this experiment but it provides us a mechanism for bootstrapping the overlay network in a natural way. In a straightforward way each node in this network is assigned a maximum *out-degree* of neighbors it connects to. A super peer is selected from neighbors randomly provided that the neighbor's capacity is greater than the current local node. Note that the network topology follows power-laws [3], so without losing the generalization, we generate peers' capacities by using a power-law distribution $P(x) = \beta x^{-\alpha}$, where x is for the capacity of a peer in a range of $[c_{min}, c_{max}]$ (i.e.$[1, 100]$ in our settings), $P(x)$ is the probability of having specific capacity x, and β and α are constant parameters. For simplicity, we herein define $\beta = 1$ and $\alpha = 2$. Observe that although one node with maximum capacity may not be willing to host more clients, we thus collect qualified neighboring nodes (i.e. having greater capabilities) and randomly select one from them. From certain perspective, it is also in line with the random procedure of generating super peers.

In addition, all experiments are conducted in a *round-driven* manner in order to capture system snapshots in specific situations.

3.2 Experiment 1 - Super-Peer Network Generation Via Gossiping Protocol

Figure 1 illustrates a network of 10^5 size. Ten individual experiments are conducted with out-degrees varying from 10 to 100 but with peer's maximum capacity fixed (i.e. 100). The curves represent the change of the number of super peers in the network after specific rounds.

The value of out-degree indicates a peer's partial view on the network (i.e. its neighbors). The gossiping protocol is adapted in initializing the super-peer network. At round *zero*, super peers are initialized to connect to their neighbors via their partial views. Then, the bootstrapping step is conducted by a custom-developed super peer protocol where connections between super peers are limited by corresponding *capabilities*. Intuitively, networks should vary significantly by different out-degree parameters, but it turns out in Figure 1 that the fluctuations on the numbers of super peers are relatively small. The reason is that in the bootstrapping the decisive factor, the peer's capacity, determines how many neighbors a super peer can connect to. To justify our judgments, we then fix the out-degree parameter and assign different values for peer's maximum capacities, namely, 100, 200 and 500. The results are shown in Figure 2.

In Figure 2 we see that the variations of the number of super peers are in the same pace with peer's max capacity. The larger the capacity, the fewer super peers will be needed, and the less communication overhead among super peers will be. In addition, both Figure 1 and Figure 2 illustrate that the number of super peers decreases logarithmically before reaching a comparatively stable situation. Interestingly, the time (i.e. rounds) required to reach a stable situation, i.e. with relatively constant number of super peers, is between 10 and 20. This is due to the fact that no super peers should be removed from the network if an approximate optimized situation is reached, i.e., all client peers are connected to a super peer respectively and super peers are not overloaded.

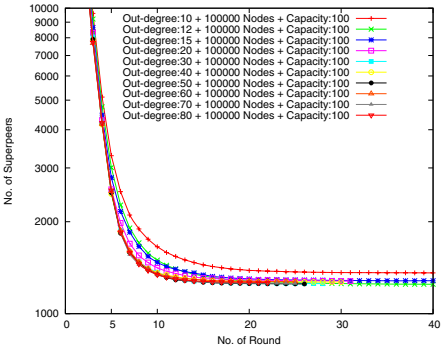


Fig. 1. Super-Peer Network Generated by Gossiping Protocol(Outdegree)

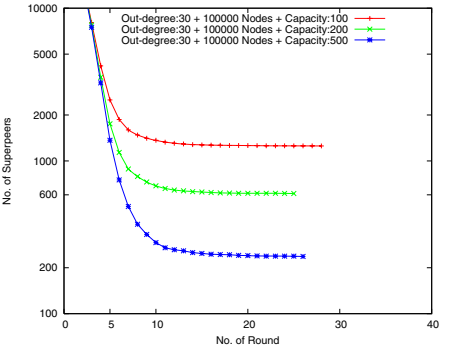


Fig. 2. Super-Peer Network Generated by Gossiping Protocol(Capacity)

3.3 Experiment 2 - Load Balancing

To evaluate the system performance under increasing load, we set up an experiment where 1000 new nodes are added to the network continuously from round 25 to round 35. The reason for choosing a round 25 is that at that moment the network is to have a relatively stable optimal status. If new nodes are added, the system will churn out.

Figure 3 illustrates that the number of super peers increases sharply as new nodes are added. It is because we initially assign the role of 'super peer' to all new nodes and adjust them in later rounds. As shown in Figure 3 as well, an obvious decrease happens after round 35, when no new nodes are added into the system. Although the system comes to a stable situation finally (c.f. after round 50), the number of super peer is slightly larger than the system which has a fixed number of nodes.

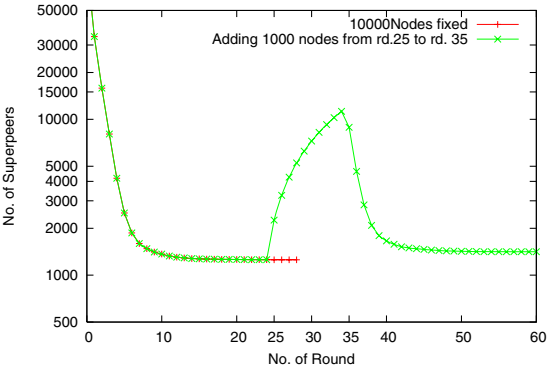


Fig. 3. Load Balancing (1000 Nodes join the network each time from round 25 to round 35)

3.4 Experiment 3 - Self Organizing

The self-organizing functionality we aim to achieve in this work is to dynamically adapt to node failure and degradation, and react to changes caused by them. The general strategy below is applied for nodes acting different roles.

- Super peer: it maintains only neighboring super peers' information without worrying about the existence of its client peers. Periodically it checks the connection status of its neighbors and remove those unreachable.
- Client peer: it periodically checks whether its super peer is online, especially before it sends requests. If the super peer is down, it searches nodes (i.e. super peer) which have more capacity and may join such a cluster led by the new super peer. Otherwise, if no super peer wants to host it, it may finally declare itself as a super peer, waiting for incoming client peers.

Two kinds of scenarios are investigated in this paper. The first scenario is that peers keep leaving while the second one is that disastrous failures happen in peers, e.g., half of the peers in the network are crashed. Results are shown in Figure 4 and Figure 5 respectively.

In the first scenario, we force 500 peers to leave the network from round 10 to 90 continuously. Figure 4 illustrates the changes in the number super peers follow a undulant manner due to the self-adjustment each time. As we can see from the enlarged figure, the tendency in the number of super peers is decreasing along with the running rounds, as is in accordance with the fact that the total number of peers is reduced.

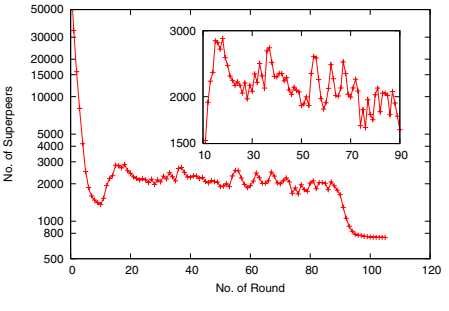


Fig. 4. Self-Organizing in Scenario of Continuous Peer Leaving

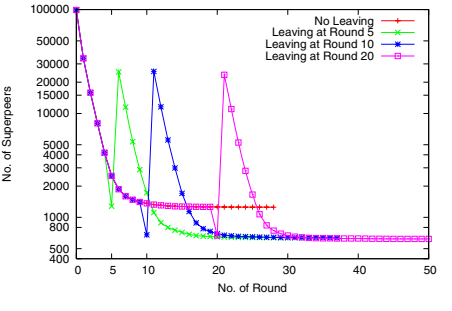


Fig. 5. Catastrophe Recovery (50000 Nodes left in round 5)

In the experiment illustrating effects brought by catastrophic failure in peers, we assume half of the peers in the network are crashed in round 5, 10 and 20 (c.f. Figure 5), which are at the beginning, middle and end of the procedure for the network reaching optimal status (c.f. the 'no leaving' curve in Figure 5).

Figure 5 shows that no matter when such a catastrophe happens, a transient phenomena that the number of super peers may surge to around 25000. The reason is that at that moment a large number of client peers which lost connection to their super peers

have the chances to declare themselves as super peers, especially when they can not find neighboring super peers who are willing to accept them. In fact the system is actually sparsely connected as in the initialization status. However, after the stumbling, the system heals itself quickly to an optimal status after 15 rounds, which justifies that merging between these newly generated super peers helps reduce the number of total super peers.

Results illustrated in Figure 4 and Figure 5 prove that the protocol helps the system adapt dynamically to continuous peer's failure (e.g. peer leaving) and huge degradation and changes in the network.

4 Conclusions and Future Work

In this paper, we have presented the urgency of exploiting advanced infrastructure for future digital library constructions and described fundamental requirements, i.e. system bottleneck and dependences, etc. It is critical for supporting advanced searching, such as metadata search (multiple fields), ontology-based search and so on, which are beyond the capability of simple file-sharing -based systems, such as Kazaa [4]. Due to that classical super-peer networks may still suffer from the problems caused by system's extreme scale and dynamism, we proposed an enhanced model concerning the initialization of super-peer networks and strategies for load-balancing and self-organizing. Evaluation results show that the enhancements are feasible and can be applied in large scale super-peer overlay networks. Further work includes supporting complex search in super-peer based digital library systems since semantic search or complicated metadata search require more advanced processing on individual information collections.

Acknowledgement

This work is partially supported by the Norwegian Research Foundation in the framework of Information and Communication Technology (IKT-2010) program.

References

1. Yang, B., Garcia-Molina, H.: Designing a super-peer network. In: IEEE International Conference on Data Engineering, 2003. (2003) http://dbpubs.stanford.edu:8090/pub/showDoc.Fulltext?lang=en&doc=2003-33&format=pdf&compression=.
2. Karl Aberer, Philippe Cudre-Mauroux, M.H.: The chatty web: Emergent semantics through gossiping. In: The International World Wide Web Conference, Hungary (2003)
3. Palmer, C.R., Steffan, J.G.: Generating network topologies that obey power laws. In: Proceedings of GLOBECOM '2000. (2000)
4. : KaZaa - a completely distributed peer-to-peer file sharing service. http://www.kazaa.com/ (Available: 2006.08)

Offline Web Client: Approach, Design and Implementation Based on Web System*

Jie Song, Ge Yu, Daling Wang, and Tiezheng Nie

School of Information Science and Engineering, Northeastern University
Shenyang 110004, P.R. China
sy_songjie@163.com, yuge@mail.neu.edu.cn

Abstract. Offline web client is a new type of Rich Internet Applications which supports a user's web operations no matter with network connection or not. It is very useful in many situations since many offline scenarios involve the users who work with explicitly disconnecting from the network. This paper proposes the approaches of designing offline web client as a guidance of developing offline web client applications, and describes the details of how to design and implement them. Finally, a prototype system is presented to verify the proposed approach. The experiments show that offline web client can increase user productivity and satisfaction, and enhance the usage of web-based systems.

1 Introduction

As the fast development of web-based system, Rich Internet Applications (RIA) takes an important part in the client of web applications. RIA is a powerful alternative to thin client applications [1]. It can provide users with a rich and responsive user interface, the ability to work offline, and a way to take advantage of local hardware and software resources. RIA can be designed to combine the traditional benefits of fat client applications with the manageability benefits of thin client applications [2, 3, 4].

Offline web client is a new type of RIA, which can work with or without network connection. Many offline scenarios involve the users explicitly disconnecting from network, who work without, intermittent or low quality of connectivity. With this condition, a web client must be designed to take maximum advantage of a connection when it is available for ensuring that both applications and data are as up to date as possible, without affecting the performance of the application.

Many studies have been made the offline model. Offline Business Objects (OBO) is a framework for implementing functionality of occasionally connected to the network [5]. Small Business Model [6] is based on traditional desktop applications as a data-oriented approach. Zahir Tari [7] proposes a caching approach that optimizes the remote invocation of clients. Smart Client Architecture for Microsoft.NET [8] is also good architecture to support offline functionality. This paper presents a general framework for the offline model which is more universal and abstract than both of them.

* This work is supported by National Natural Science Foundation of China (No. 60573090).

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 308 – 314, 2006.

2 Approaches

Three approaches of implementing offline functionality are proposed based on many predecessors' work, according to the different level of functionality dividing. Fig. 1 shows the structure of three approaches. They are data-oriented approach as a lightweight solution, the operation-oriented approach as a middleweight solution, and the service-oriented approach as a heavyweight solution.

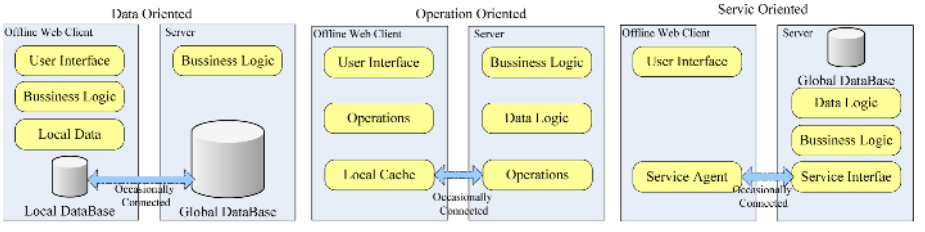


Fig. 1. Structures of Three Approaches

2.1 Data-Oriented Approach

Applications that are coupled to the data on the server employ the data-oriented approach. A local database system manages the changes to the locally cached data and propagates the changes back to the server in the manner of merge-replication. The characteristics of the data-oriented approach include:

- It provides robust data conflict detection at the fine granular level of data, for example, at the column and row level of a relational database. It also provides data validation and constraints at the fine granular level of data.
- The client is coupled to the data store tightly so that the changes to the data store schema have a direct impact on the client. The client can provide the offline support for the data stores to which it subscribes.
- Merge-replication is a two-tier architecture and therefore has the control on the manageability and maintainability.

The approach requires a local database to be installed on the client that can replicate with the server. This may not be suitable for the applications running on small devices or on the devices that require a light-touch deployment mechanism. All the change tracking code is contained inside a DBMS. There is no need to write additional change tracking or conflict detection and resolution code.

2.2 Operation-Oriented Approach

Operation-oriented approach treats the server-side business logics as the operations. An operation is combined with the necessary data and encapsulated as a single object using an object oriented programming language. Some operations have "shadow" instances at the client side. A local cache locally holds operations and then the

merge-replication mechanism is implemented by comparing object versions. The characteristics of the operation-oriented approach include:

- An operation with its data is the basic unit of conflict detection at the object level, which is close to the reality.
- Instances of one operation are on both the server side and the client side, but their versions are distributed. The client primarily chooses the server-side operation instance, and chooses the local instance in case of disconnecting from the server.
- A business logic is well encapsulated on client with its data, which makes good use of local computing resources.

The approach also requires a local data store, which can be a database, a file or a memory cache, so this approach may theoretically be suitable for the applications running on any environments such as a desk-top computer, a PDA or a smart phone. All the change tracking code and merge-replication mechanism rely on the object version comparison and hand-written programming. Although it is more complex but it is flexible than the data-oriented approach which relies on a DBMS.

2.3 Service-Oriented Approach

With this approach, an offline web client interacts with the services on the network through service requests. These services are always implemented as Web Services. The service-oriented approach is a good practice of SOA (Service Oriented Architecture).

The characteristics of the service-oriented approach include:

- Offline logic is encapsulated totally on the client.
- Client-side logic and server-side logic are loosely-coupled.
- More design and coding are required to implement.

To support web clients when working offline, asynchronous communication is primarily used when interacting with the services located on the network. This is done by using a message bus or other types of message-based transport. It expects a delay between the request and any response, or expects no response at all. An infrastructure is required that allows the details of the service requests to be stored so that they can be executed when the client reconnects to the network.

3 System Design and Implementation

In this section, a system with offline web client is designed, named as OCWeb (Occasionally Connected Web System), which consists of four important parts: operations caching, operations synchronizing, dependencies handling and subsystem's cooperation.

3.1 Operations Caching

Operations of OCWeb are all cached on the client. Stateless operation is the one that is used for reference purposes and never changed by the client. Therefore, from the client's point of view, the operation is read-only, and the client performs no update,

insert, or delete operations on it. Read-only stateless operations are readily cached on the client, thus they not only provide offline capabilities, but also improve the performance of applications.

Stateful operations can be changed on the client as well as the server. Generally, changes are as a direct or indirect result of user input and manipulation. In this case, changes that are made on either the client or the server need to be synchronized at some point. The system should keep track of any client-side changes of stateful operations, until the operation is synchronized with the server and any conflicts have been resolved. Any unconfirmed stateful operations should be handled before they are used.

Both stateful operations and stateless operations all need to be cached at the client side. OCWeb provides two types of cache:

- **Short-term cache:** Short-term cache is good for performance but it is not persistent. It is used for caching stateful operations, which can be changed on the client as well as the server.
- **Long-term cache:** Long-term cache is persistent. It is used for caching stateless operations.

3.2 Operation Synchronizing

The changes that are made in a client in offline mode must be synchronized or reconciled with the server at some point. This raises the possibility of a conflict between client-side operation and server-side operation. When conflicts do occur, OCWeb ensures that they are detected and resolved by three mechanisms: operations partition and lock, stale operations update and conflicts reconciliation.

Partitioning and Locking Operation mechanism is aimed on how to reduce the conflicts. Any system that allows multiple parties to access shared operations has the potential for producing conflicts. But OCWeb partitions operations and their each part are locked by client, this mechanism is helpful to reduce or even avoid conflicts. Most systems have the situations that different individuals controlling over separate sections of operations. For example, a team leader may have a privilege of firing a team member. In this case, only the client of leader can access the operation of deleting accounts. Partitioning the operations in this way allows users to make arbitrary changes to the operations without fear of encountering operation conflicts. Locking the operations is that the system uses mutually exclusive locks to ensure that only one party operates on system at a time. Operation partitioning and locking is often very restrictive, and so is not a good solution in many cases. However, this approach is strongly considered by OCWeb in specific situation, because it helps reduce the number of conflicts produced by target application.

Handling Stale Operation mechanism is for detecting and preventing stale operations from being used. Even if data in the operation have not changed, it may incorrect because it is no longer current. The operation with this data is known as stale operation. An operation may be current when a client first goes offline, but may become stale before a client goes online again. Stale operations should be detected and prevented from being used. OCWeb uses metadata to describe the validity of operation and shows when the operation will expire. This can prevent stale operations being passed to the client or being synchronized to the server. The metadata of operation include last-modify time and type. Not “old” operations are stale operations,

what operations are stale is business-logic related. So a good general rule is employing business logic in operation to handle whether itself is stale or fresh.

Reconciling Conflict mechanism may not have a formally pattern because all conflicts are business-logic relative. But OCWeb can reconcile the conflicts by three different rules which are adopted in different place:

- **Rule 1 (Automatical Reconciliation on Server):** OCWeb first chooses to reconcile the conflicts at the server-side by the automated processes based on different business logic without affecting the client, so as to ensure that the latest change always takes precedence, or the two dates of operation are merged, or more complex business logic is used.
- **Rule 2 (Manual Reconciliation on Client):** If the server-side programming is not able or too complex to reconciling conflicts, OCWeb chooses the second rule: sends the conflicts back to the client and reconciles them by the user himself.
- **Rule 3 (Third-party Reconciliation):** If both of the above two rules are unfit, OCWeb relays on third party, such as, an administrator or supervisor to reconcile any operation conflicts.

3.3 Dependencies Handling

If an operation is dependent on others, OCWeb handles this operation very carefully when offline. For example, suppose that accomplishing one business logic needs three operations which are requested orderly, the subsequent one may succeed only if the former one succeeds. Dependencies between individual operation requests can be both forward and reverse dependencies:

- **Forward Dependency.** Forward dependency means the when the first operation fails during operation synchronization, all the subsequent operations need to be deleted or ignored and all the following assumptions are nullification.
- **Reverse Dependency.** Reverse dependency means the subsequent operation fails when the former operations succeed during operation synchronization. It needs to undo the former operations through a compensating transaction.

Absolutely, the reverse dependency requirement can add significant complexity to the application. So the rule of business operations is trying to avoid putting the failure-prone operation forward. Or the server should join several correlative operations together and provide as a single operation. An atomic operation keeps track of the operation request dependencies, it may simplify the server-side programming but complicate the implementation of the business operations. This approach handles dependencies at the “outside” level and fits for settling reverse dependency.

Another effective approach is handling dependencies at the “inside-operation” level. Each operation defines a subsequent operation list to point out when it fails, which operations that follow it should be omitted. This is a light-weight solution but only fits for settling forward dependency.

3.4 System Structure

Fig. 2 illustrates the main modules of OCWeb and their cooperation. The following are the steps when a client requests an operation on the condition of offline and online:

- User sends the request through *client interface*. *Request dispatcher* captures this request and redirects it to the *connection detection*.
- *Connection detection* checks the network connectivity and reports the absence of connection. Then *operation manager* finds the requesting operation in *short-term cache* or *long-term cache*.
- The client-side operation is executed and an offline warning is prompted to the user through *client interface*.
- When the client is online again, this operation is sent to the server through *operation agent* and *request proxy*. *Operation Synchronizer* synchronizes client-side operation to the server-side. Some database related operations will employ *Data Accessor* to communicate with the database.

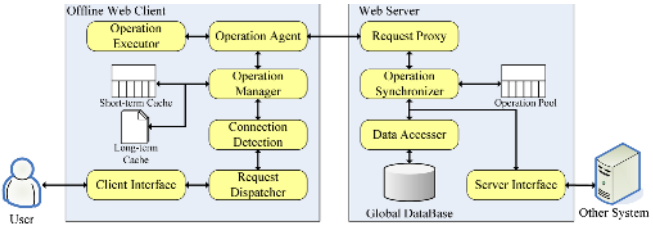


Fig. 2. The Components of OCWeb

4 Conclusions

In this paper, three approaches of offline web client are presented and one of them is designed in detail, which is primarily used in new generation web-based systems. Our primary work is in the two aspects as follows:

- Propose data-, operation- and service-oriented approaches which summarize the available approaches to develop an offline web client.
- Describe the design detail of the operation-oriented approach, which is also helpful to the other two approaches.

The offline web client has practical application in many web systems such as China Nation Marine Data and Information Service (NMDIS) [9]. Future work of this research includes the security considerations of offline web client, and using multiple threads techniques at the client to improve the system performance.

References

1. J.Preciado, M.Linaje T.Sanchez, S.Comai: Necessity of methodologies to model Rich Internet Applications. WSE 2005: 7-13
2. R.Reinhardt: Building communities with Rich Internet Applications. SIGGRAPH Web Graphics 2003

3. L.Derechin: The Benefits of Rich Internet Applications on network performance. Int. Conf. on Internet Computing 2005: 114-121
4. Y.Liu , X.Liu , L.Xiao , L.Ni, and X.Zhang , Location-Aware Topology Matching in P2P Systems, IEEE INFOCOM, 2004
5. P.Gruszczynski, S.Osinski, A.Swedrzynski: Offline Business Objects: Enabling data persistence for distributed desktop applications. OTM Conf. (2), 2005: 960-977
6. P.Nguyen, D.Sharma: Smart clients and small business model. KES 2005: 730-736
7. Z.Tari, S.Hammoudi, S.Wagner: A COBRA object-based caching with consistency. DEXA 1999: 321-331
8. Microsoft Corp.: Guide of smart client architecture for Microsoft.NET, http://msdn.microsoft.com
9. S.Feng, J.Song, X.Bai, D.Wang, G.Yu: The Web-Based Transformation System for Massive Scientific Data. Accepted by WMDP 2006

A Latent Image Semantic Indexing Scheme for Image Retrieval on the Web

Xiaoyan Li, Lidan Shou, Gang Chen, and Lujiang Ou

Zhejiang University, Hangzhou 310027, P.R. China
kricel_lee@yahoo.com.cn, should@cs.zju.edu.cn, cg@cs.zju.edu.cn,
wizardzju@gmail.com

Abstract. In this paper, we present a novel latent image semantic indexing scheme for efficient retrieval of WWW images. We present a hierarchical image semantic structure called HIST, which captures image semantics in an ontology tree and visual features in a set of specific semantic domains. The query algorithm works in two phases. First, the ontology is used for quickly locating the relevant semantic domains. Second, within each semantic domain, the visual features are extracted, and similarity techniques are exploited to break the "dimensionality curse". The target images can then be efficiently retrieved with high precision. The experimental results show that HIST achieves good query performance. Therefore, our method is promising in diverse Web image retrieval.

1 Introduction

With the advent of digital image databases and prevalent availability of World Wide Web, huge repositories of image data have ever heavier usage in WWW and become available to a large public. The amount of image data has grown rapidly and enormously which has created an urgent need of an effective image search engine or system which can retrieve relevant images quickly on demand from the widely diverse WWW image sources. Most commercial image search engines available on the Web employ keyword-based search, which mainly refers to the textual information of images, while the retrieval results are often not satisfactory [11]. With the advances in image processing techniques, a lot of efforts have been made in a new area called Content-Based Image Retrieval (CBIR), to address this issue. CBIR is a challenging topic of the current research. A collection of research prototypes and commercial systems [13] have exploited visual features of images, such as colors, textures, and shapes to represent and index image contents. Due to the "semantic gap" between the visual features and the semantic meanings, CBIR has not yet seen a successful commercial application.

Recent research efforts in CBIR focuses on bridging the so-called "semantic gap" which combines *Relevance Feedback* techniques or text and keyword predicates to get powerful retrieval methods for image collections [3,9,10]. Users intend to specify the semantic class of the scene or the objects it should contain when they search for images. That requires content-based retrieval to recognize

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 315–326, 2006.

the generic classes of objects and concepts. A few research works have been deployed in this respect[1,2], but no general methodology has been proposed yet.

Content-related keywords carry more semantics of images, so text-based techniques can capture high-level abstraction and concepts, which can fast prune topic irrelevant images. Content-based techniques can capture low-level image features but can hardly capture the high-level concepts. Though low-level features have more potential in visual comparison, the similarity measure is effective only when the semantics is well correlated. Otherwise, the "dimensionality curse" is unavoidable and the retrieval accuracy would degrade considerably.

In our proposed method, we integrate text-based and content-based techniques to take advantage of their complementing strengths. We present a novel latent image semantic indexing scheme to organize and retrieve the diverse WWW images. The goal of our research is to develop a general and scalable architecture to support fast and effective searching and querying of very large and diverse WWW image source with simple object annotation. The main contributions of this paper include the following: (1) We construct high-level image-semantic ontology and use it for quickly locating the relevant concept units. (2) Within each concept unit, the distinguishable low-level features are extracted, and similarity techniques are exploited to break the "dimensionality curse". The target images can then be efficiently retrieved with high precision. (3) We propose a query algorithm designed for users to specify their queries without detailed knowledge of the underlying metrics. It provides more user-friendly and convenient search interface.

The rest of this paper is organized as follows. In Section 2, we briefly review some relevant researches and demo systems in the literature. In section 3, we present the HIST structure in advance. We describe the query algorithm of HIST in Section 4. The construction technique of HIST is explained in Section 5. We show the performance results and the evaluations in Section 6, followed by conclusions and possible future work in Section 7.

2 Related Work

There have been many research works on the advanced CBIR techniques. Some of those implement robust content-based image retrieval systems based upon a combination of higher-level and lower-level vision principles, such as CIRES [14] and SIMPLIcity [4]. We refer to some other well-known CBIR systems as follows:

- *CBIR*: Understands the content in two-dimensional imagery, and permits user to search large image databases for particular objects or combinations of objects.
- *CBIR2*: Provides a large variety of image-distance measures that can be used singly or in combination to satisfy a wide range of user needs and provide rapid access to images, even in an extremely large database.
- *NETRA*: Uses color, texture, shape and spatial location information in segmented image regions to search and retrieve similar regions from the

database. A distinguishing aspect of this system is its incorporation of a robust automated image segmentation algorithm that allows object or region based search.
- *WebSEEK*: An image search and cataloging system on the WWW. It utilizes text and visual information synergistically to provide for cataloging and searching for the images.

Several research works have indicated that ontology can be a better candidate for modeling semantically rich and complicated image data [1,6]. In paper [6], the authors propose a method to construct domain-dependent ontology automatically in bottom-up fashion based on clustering and vector space model. Each concept in the ontology contains a label name and a feature vector. During the image clustering, the size of the vector is same as the total number of objects. Unfortunately, that seems not applicable for large diverse image collections. In [1], the authors employ an ontology to describe the relationships between words, and that can serve as an interpretation to their clustering algorithm. Motivated by these methods, we propose a keyword-based ontology structure, which can captured in HIST, as a hierarchical abstraction mechanism for capturing the category semantics associated with an input image through its salient objects and its background annotation. The objective is to provide an efficient mechanism for quickly reducing the search range to local domains, so that only minimum number of visual similarity operations are needed in local domains to obtain the final result.

In [8], LSI is reported to be able to capture the underlying semantics of corpus, therefore achieving improved retrieval performance. The authors try to overcome the problem of lexical matching by using statistically derived conceptual indices, instead of individual words or features for retrieval. In our work, we apply LSI in image data retrieval to implement the second phase (local content-based retrieval) of our novel HIST-based scheme.

3 The HIST Structure

In this section, we present the HIST (Hierarchical Image Semantics Tree) structure, which can capture both ontological knowledge and visual features for large image collection with general contents. The structure consists of two parts: A single *ontology tree* that organizes the ontological knowledge, and a set of *atomic semantic domains (ASDs)* pointed to by the leaves of the ontology tree.

3.1 The Ontology Tree

We utilize the salient objects and background description words (the annotations) of images as the core source for our image-semantic ontology analysis. The ontology structure in HIST combines the generic ontological knowledge and more domain specific semantics. This abstraction can manifest itself effectively in conceptualizing objects contained in the images.

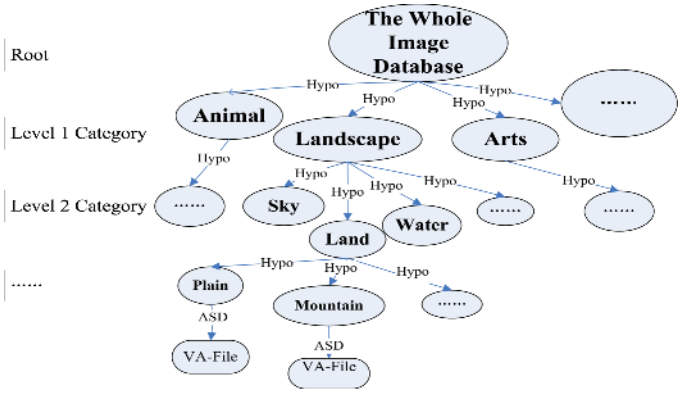


Fig. 1. An overview of HIST structure

Generally, a node of the *ontology tree* represents a category (or sub-category) of the images that the node indexes. Each node n contains the following information:

$\overrightarrow{K}$ A keyword vector which contains keywords implying the semantics of the category that the node represents.

$\overrightarrow{W}$ A weight vector where the ith element corresponds to the weight of the ith keyword in $\overrightarrow{K}$.

$\{L_i\}$ A number of links pointing to its child nodes. Each link denotes a hyponym relation to its child node.

l The level in the tree (the root node has level 0).

The leaf nodes of the ontology tree contain the same fields excluding the children information. Additionally, each leaf node contains a pointer to an *ASD* , in which further indexing data (namely visual features) are stored.

Fig 1 shows a simple four-level HIST structure. Nodes in the higher levels of the tree are usually marked by generic keywords. In contrast, the lower levels of the ontology tree are much more domain-specific. Therefore, the keyword vectors in the lower levels are more likely to include specific keywords as hyponym of their ancestors.

To traverse and query the ontology tree of HIST structure, we also maintain an additional *keyword-node map* structure. We store the quantitative representations of relations among the keywords according to linguistics knowledge. We use three relations: *synonym*, *hypernym* and *hyponym* in HIST. The synonymic keywords are stored together in this map structure. For each keyword K_i, we record the following information:

$\{Sign_i\}$ An encoded representation of the lexical knowledge of K_i, which can be used to determine if K_i has a binary relation with other keywords. That is, we can determine if K_i and K_j have one of the three relations by calculating a simple function $lex\_rel(Sign_i, Sign_j)$, which may return four possible values: NONE, SYNONYM, HYPONYM, and HYPERNYM.

$\{P_{hypo_i}\}$ The hyponym probability of K_i with respect to its hypernym, if any. In our keyword-node map, each keyword has at most one hypernym.
$\{N_i\}$ The ID set of nodes whose keyword vectors contain K_i.

If K_i has a binary relation with K_h, and K_h has the same kind of binary relation with K_j, K_i and K_j are said to have a *cascade relation* of length 2. Cascade relations can have any length l, where $l \geq 2$.

3.2 The Atomic Semantic Domains

The atomic semantic domains (ASDs) contain visual features of the images indexed by them. The visual features of an image are represented by a high-dimensional feature vector.

As wavelet transformation is a powerful tool in effectively generating compact representation of visual features of images [10], we adopt the Daubechies' wavelets to extract the wavelet coefficients using the LUV color space for its good perception correlation properties [15,7]. The feature elements chosen from the wavelet coefficients are independent of image resolution and scaling. That is an advantage of our wavelet-based approach.

Each ASD contains a feature vector matrix F_{n*m}, where n is the total number of images indexed by it, m is a constant number of visual dimensions of the visual features (64 in our case). Each row vector in the matrix represents a feature vector for one image. We employ Singular Value Decomposition (SVD), as a dimension reduction method, to select the k-most dominating coefficients (normalized) to form the local visual feature vector.

***Choosing Dimension Reduction Factor k*:**
The singular value decomposition of F_{n*m} is the product of three matrices $F_{n*m} = U\Sigma V^T$, the $\Sigma = diag(\sigma_1, \sigma_2, \cdots, \sigma_r)$ is an $r \times r$ matrix, where $\sigma_1 \geq \sigma_2, \cdots \geq \sigma_r$. U is an $n \times r$ matrix and V is an $m \times r$ matrix.

We need to choose the k-largest singular values in the above decomposition as $\Sigma_k = diag(\sigma_1, \sigma_2, \cdots, \sigma_k)$. The value of k should be large enough to adequately preserve the content of the original information, and small enough to ensure the retrieval efficiency. If we use a pre-defined threshold θ to denote the preserving ratio of information, the following equation must hold:

$$\sum_{i=1}^{k} / \sum_{i=1}^{r} \geq \theta \tag{1}$$

Therefore, the minimum value of k can be obtained by solving the above equation. By taking the first k row vectors in U and V, we obtain a matrix of rank k as the approximation of F_{n*m}: $F^k_{n*m} = U_k \Sigma_k V_k^T$.

4 The Query Algorithm

In this section, given a query image(provided by user or in the database) and his relevancy keyword annotations, we propose our two-phase query algorithm for HIST.

4.1 The First Phase: Querying the Ontology Tree

We present a non-metric matching and searching space, which provide a hierarchical browsing environment. In the following, we elaborate the search algorithm which can quickly locate the highest relevant semantic domain to dramatically reduce both memory and computation requirements by narrowing down searching range sharply.

Given the query keyword set $\{K_1, \cdots, K_m\}$, we refer to the keyword-node map (as discussed in section 3.1). Let N_{leaf} denote the final overlapped leaf node set implied by all these keywords. We perform the procedure listed in Figure 2 to find these nodes.

```
ALGORITHM: OverlappedLeaves
BEGIN: N_leaf includes all the leaves in HIST
FOR1(each node n in N_leaf) Do{
      FOR2(each K_i, i = 1, ..., m) Do{
          IF(the keyword vector K⃗ of n contains K_i or
            a hyponym, or a cascade hyponym of K_i)
            Continue FOR2;
          ELSE
            Delete node n from N_leaf, continue FOR1;
      }
}
Output the remaining nodes in N_leaf;
END
```

Fig. 2. Computing the leaf node set implied by a keyword set

We use the normalized weight W_i for keyword K_i in node n as the predictive probability: $P(n/K_i)$. Let N_{leaf_i} denote one leaf node contained in set N_{leaf}, we rank the predictive probabilities $P(N_{leaf_i}/(K_1, \cdots, K_m))$ where

$$P(N_{leaf_i}/(K_1, \cdots, K_m)) = \sum_{i=1}^{m} P(N_{leaf_i}/K_i), \tag{2}$$

- if K_i exists in the keyword vector of N_{leaf_i}: $P(N_{leaf_i}/K_i)$ is equal to W_i.
- Otherwise, one its hyponyms or cascaded hyponyms $K_{i_{hypo}}$ does exist, $P(N_{leaf_i}/K_i)$ is given by $W_{i_{hypo}}$ multiplied by the hyponym probability (or the cascaded hyponym probabilities as defined in subsection 2.2) between $K_{i_{hypo}}$ and K_i.

The possible results of the above algorithm are:

- If $\{N_{leaf_i}\}$ is NULL, we should scan the whole image source instead.
- Otherwise, choose the top-k leaves from $\{N_{leaf_i}\}$ based on the ranked probability $P(N_{leaf_i}/(K_1, \cdots, K_m))$, where k is normally set to 1.

4.2 The Second Phase: Querying Local ASDs

The output from the first phase of the query is a set of ASDs relevant to the query. In the second phase, we shall perform visual comparisons between the query image and the images in the candidate ASDs. The query image is described by a full-length feature vector (64-dimension normalized wavelet coefficients). It must be transformed into a vector in the k-dimensional space of each candidate ASD to make the similarity comparison with the images contained in that ASD. That is:

$$\bar{q} = q^T U_k \Sigma_k^{-1} \tag{3}$$

By comparing the similarity between the querying vector $\bar{q}$ and other dominant vectors of the candidate images one by one, we get the final query results. Let $\bar{D}_i$ indicates one image in this ASD, we use Manhattan Distance to compute the similarity between the feature vectors of these two images:

$$sim(\bar{q}, \bar{D}_i) = 1 - (\sum_{j=1}^{k} |v_{q_j} - v_{d_j}|)/k \tag{4}$$

where v_{q_j} and v_{d_j} are the respective normalized feature elements.

5 Generating the HIST Structure

The salient objects and the main background contained in one image are sensibly important for image semantics recognition. We start the ontology construction process by attaching these annotations.

Given the abundance of the available automatic image annotation algorithms [5], we have the freedom to choose one of them for the image annotation. However, we choose not to use automatic image annotation in this work as we would prefer to look at the effectiveness of our proposed HIST structure alone.

The annotation keyword set, denoted as K, is used as the elements for our ontology construction. Basically, we quantify the relationships among the keywords, including synonym, hypernym, overlap, covering, disjoin etc. The lexical hierarchy can be constructed respectively, such as: {hep, rose}→{flowers, bloom, blossom}→{plant, flora}(Symbol "→" indicates "is a kind of". The elements in {} are synonyms).

We build the ontology using the *keyword set*, the *common-sense knowledge*, and the *domain knowledge*. We apply a Generative Hierarchical Clustering pattern (GHC) to construct a tree-like conceptual taxonomy in a top-down fashion. We firstly initialize the general categories, and associate each category (node) with a set of relevant keywords, so that to form a coarse and simple hierarchical semantic structure. After that, a second off-line process which exploits the latent and more specific semantics for finer node-splitting will follow up.

For each node need to be split(the number of the training images fall into this node category is large enough and the semantic of this category need for further distinguished as well), we perform GHC algorithm in the following steps:

- *Trial-Query Construction*:
  For the hyper-node categories, we consult the relevant lexical hierarchy and the corresponding domain-specific knowledge as well as sets of rules defined by domain experts, and prepare a set of keyword-based trial-queries $Q = \{q_1, q_2, \ldots, q_m\}$ to induce related sub-categories. Images relevant to the keywords are then retrieved for each sub-category being trial-queried by the corresponding q_i.
- *Relevance Feedback Refinement*:
  We adopt a stochastic process, which is similar to [12], as the fuzzy clustering algorithm. The semantic relationship can be learned from the history of user's access pattern and access frequencies on the images in the training set. The irrelevant images will be dropped, while those highly-correlated ones are grouped together. For a given category to be trained, we construct $N \times m$ Matrix V which logs the access history of the images, where value N denotes the number of images in hyper-category node n, and m denotes the number of the split sub-categories. The value of entry v_{ij} indicates the frequency of i^{th} image ID accessed by query q_j, and f_j is the number of times performed by different users for query q_j. Assuming that p_{ij} is the acceptance probability of image i by q_j:
  $$p_{ij} = v_{ij}/f_j \tag{5}$$
  As long as p_{ij} exceeds a threshold λ, image i will be put into this sub-category for its high relevance. We note that one image might fall into more than one category. That is why we call it a fuzzy clustering. The clustering algorithm reveals the distinguishable traits of our image source organization, which allows an image to be associated with multiple "classes" rather than a single one.
- *Statistical Data Analysis*:
  By using the output from the relevance feedback step, we extract the representative elements (keywords) for each sub-category. Suppose there are t images falling into one sub-category n_i, and $K = \{k_1, k_2, \ldots, k_r\}$ denotes the general keyword set, we can construct a $t \times r$ Matrix U^i with each element defined as :
  $$\begin{cases} {u^i}_{jk} = p_{ji} & : \ if \ \ image \ \ j \ \ annotated \ \ with \ \ word \ \ k_k \\ {u^i}_{jk} = 0 & : \ otherwise \end{cases} \tag{6}$$
  where p_{ji} is defined as the weight we assign to the image i pertaining to sub-category n_i. We compute the normalized weight of word k pertaining to sub-category i using the following equation:
  $${w^i}_k = \sum_{j=1}^{t} {u^i}_{jk}/t \tag{7}$$
  Considering the effect of each keyword in other sub-categories, we modify the weight as:
  $${w^{i'}}_k = {w^i}_k / \sum_{i=1}^{m} {w^i}_k \tag{8}$$

If there is a hyponym relation between k_i and k_j (k_i is a kind of k_j in the lexicon hierarchy), we add the modified weight of k_i to the modified weight of k_j.

$$w^{i'}_{j} = w^{i'}_{j} + w^{i'}_{i} \tag{9}$$

We rank the keywords to form a vector for each category according to their final weights. We can employ a top-k or threshold δ to select the leading keywords, and form a keyword vector $\overrightarrow{K}$ with the corresponding weight vector $\overrightarrow{W}$ to represent each sub-category node.

6 Experimental Results and Evaluations

In order to measure the accuracy of the high-level concept structure as well as the efficiency and effectiveness of the HIST structure, we have implemented a demo system on a Pentium PC platform running Windows NT. We use a collection of 2000 images, which contain a variety of images with various contents and textures. The experimental results and evaluations are shown and discussed as following.

6.1 The Quality of High Level Ontology

We manually annotate the salient objects and main background for each image in the training set. Each image is represented by an annotation of one to five keywords, and there are 112 keywords in total in our database.

Following the GHC algorithm described in Section 5, the construction of ontology taxonomy tree is conducted utilizing the training image set. We perform the statistic based query algorithm described in the first part of Section 4 to test the quality of the high-level concept structure (We adopt threshold $\delta = 0.03$ to set the keyword vector for each ASD, and choose the rank-1 ASD). The average recall and precision of each ASD are computed and plotted in Fig 3, where recall is the ratio of relevant items retrieved to the total number in the collection, and precision is the ratio of relevant items retrieved to the total number of items retrieved. Generally, the results of the final semantic organization are promising. Whereas, the results of a few topics are still not satisfactory. We need to consult additional rules for these topics to enhance the performance. Based on the salient objects and background in the query image, we can effectively attach it to the proper sub-category, which can further accelerate the process of finding visually-matched images locally in the sub-category.

6.2 Performance Comparison with Sequential Scan

For sequential scan, we simply scan the whole image source, and compute their similarities with the query image – the Manhattan distance of the full 64-dimension wavelet coefficient vector. There we set a threshold ε, and the image whose similarity with the query image exceeds ε is retrieved. By adjusting the similarity threshold value ε as shown at X axis of Fig 4 , the average precision

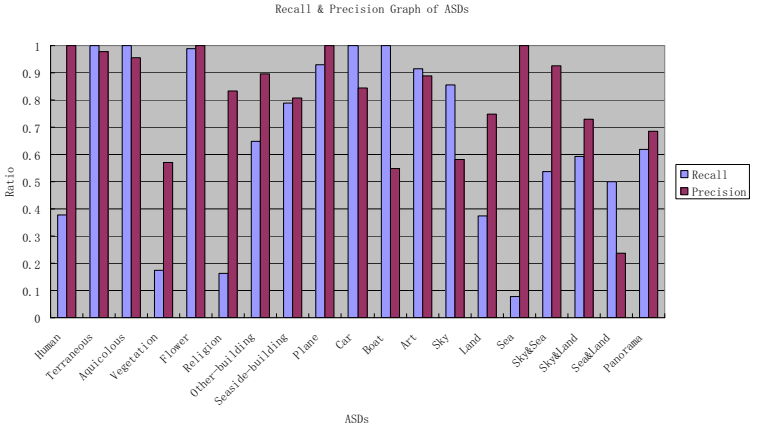


Fig. 3. Recall and Precision for Each ASD

of the retrieval results is plotted at Y axis accordingly. The average precision of each ASD is shown in Fig 3 as above, where the whole average precision of the located ASD can achieve 0.89. By comparison, our method remarkably outperforms the naïve sequential scan indexing with much higher average precision.

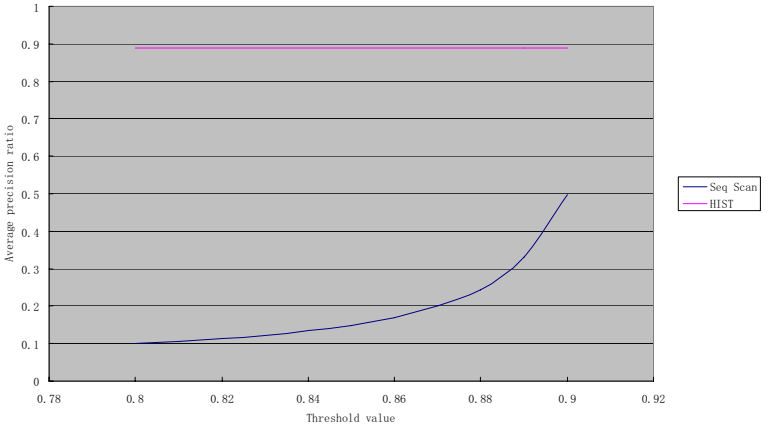


Fig. 4. Comparison of Precision: HIST Vs Naïve Sequential Scan

For the second phase of HIST query, the value of dimension reduction factor k is 10 (the threshold θ is set as 0.7 in our experiment). We first locate the highest-relevant ASD, and then compute the Manhattan distance between the feature vectors of the query image and the images of the ASD one by one. The top-8 final results are listed in Fig 5 (a) and (b) respectively in terms

of the ranked similarity values, where the query image is just the one in the top-right corner(with its relevancy keyword annotations: star, plants, ocean). Obviously, HIST obtains much better retrieval results for their high relevance. To address that, our scheme can also achieve advanced retrieval efficiency for its narrow-down searching range and lower-dimension vector which sharply reduce the processing time.

(a) top-8 results using LISI

(b) top-8 results using sequential scan

Fig. 5. Comparison of Query Output

7 Conclusions and Future Work

In this paper, we proposed a novel HIST scheme for efficient retrieval of WWW images. Query to HIST structure consists of two phases. Firstly, it locates the relevant semantic domains (ASDs) using the ontology tree. Secondly, a query makes visual similarity comparison in the reduced domains. Our experimental results indicated that our method substantially outperforms sequential scan in retrieval precision.

As the query time for these methods were trivial, we did not use it as a performance metric. However, as our method uses hierarchical pruning in the first phase, we do expect it to be very efficient. We would try our method on larger datasets, as one of the future tasks. We also plan to implement interactive query refinement techniques on HIST.

References

1. T. D. Breaux, J. W. Reed. Using Ontology in Hierarchical Information Clustering. Proceedings of the 38th Hawaii International Conference on System Sciences, 2005.
2. J. Chen, C. Bouman, and J. Dalton. Hierarchical Browsing and Search of Large Image Databases. IEEE Trans on Image Processing, Vol. 9, No. 3, March 2000, pp. 442-455.
3. Q. Iqbal, J. K. Aggarwal. Feature Integration, Multi-image Queries and Relevance Feedback in Image Retrieval. 6th International Conference on Visual Information Systems (VISUAL 2003), Miami, Florida, Sep. 24-26, 2003, pp. 467-474.
4. Q. Iqbal, J. K. Aggarwal. CIRES: A System For Content-Based Retrieval in Digital Image Libraries. Seventh International Conference on Control, Automation, Robotics And Vision (ICARCV'02), Dec 2002, Singapore.
5. J. Jeon, V. Lavrenko, R. Manmatha. Automatic Image Annotation and Retrieval using Cross-Media Relevance Models. 26th Annual Int.ACM SIGIR Conference, Toronto, Canada, 2003.
6. L. Khan, L. Wang. Automatic Ontology Derivation Using Clustering for Image Classification. Multimedia Information System 2002: 56-65.9.
7. J. Li, J. Z. Wang. Automatic Linguistic Indexing of Pictures by a Statistical Modeling Approach. IEEE Trans.on Pattern Analysis and Machine Intelligence, vol.25, no.9, pp.947-963,2003.
8. C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent Semantic Indexing: A Probabilistic Analysis. Proc. 17th ACM PODS, 1998.
9. H. T. Shen, B. C. Ooi, K. L. Tan. Giving Meanings to WWW Images. Proceedings of the 8th ACM international conference on multimedia, 30 October - 3 November 2000, Los Angeles, pp 39-48.
10. H. T. Shen, X. F. Zhou, B. Cui. Indexing Text and Visual Features for WWW Images. Lecture Notes in Computer Science, Volume 3399, Jan 2005, Pages 885 - 899.
11. M. L. Shyu, S. Chen, M. Chen, C. Zhang. A Unified Framework for Image Database Clustering and Content-based Retrieval. ACM International Workshop On Multimedia Database, 2004.
12. M. L. Shyu, S. Chen, M. Chen, C. Zhang, C. M. Shu: MMM A Stochastic Mechanism for Image Database. Proceedings of the IEEE 5th International Symposium on Multimedia Software Engineering (MSE2003), pp. 188-195, December 10-12, 2003, Taichung, Taiwan, ROC.
13. A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain. Content-Based Image Retrieval at the End of the Early years. IEEE Trans.Pattern Analysis and Machine Intelligence, vol.22, no.8, pp.1349-1380, Dec.2000.
14. J. Z. Wang, J. Li, G. Wiederhold. SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries. IEEE Transaction on Pattern Analysis and Machine Intelligence, VOL. 23, NO. 9, September 2001.
15. J. Z. Wang, G. Wiederhold, O.Firschein, Sha Xin Wei. Content-based image indexing and searching using Daubechies' wavelets. Int.J.on Digital Libraries 1(4):311-328,1998.

Hybrid Method for Automated News Content Extraction from the Web

Yu Li[1], Xiaofeng Meng[1], Qing Li[2], and Liping Wang[2]

[1] School of Information, Renmin Univ. of China, China
{liyu17, xfmeng}@ruc.edu.cn
[2] Computer Science Dept., City Univ. of Hong Kong, HKSAR, China
{itqli@, 50095373@student}.cityu.edu.hk

Abstract. Web news content extraction is vital to improve news indexing and searching in nowadays search engines, especially for the news searching service. In this paper we study the Web news content extraction problem and propose an automated extraction algorithm for it. Our method is a hybrid one taking the advantage of both sequence matching and tree matching techniques. We propose *TSReC*, a variant of tag sequence representation suitable for both sequence matching and tree matching, along with an associated algorithm for automated Web news content extraction. By implementing a prototype system for Web news content extraction, the empirical evaluation is conducted and the result shows that our method is highly effective and efficient.

1 Introduction

In recent years, besides conventional keyword based and general purpose search, most search engines have recently launched a new searching service, named "Web news search", which mainly focuses on Web pages of news. Web news search causes some nontrivial problems to traditional information retrieval techniques. One of them is how to differentiate Web news content from others in Web pages. As we know, a Web news page usually contains not only the content (the title, date and context) of news, but also other facilities such as the navigation area of the whole Web site, links to related topics, links to supplemental materials, advertisements, and even ongoing events. Separating the news content from the others, and only indexing the content, are believed to be the way to further improve the searching quality.

In this paper we focus on automatically differentiating Web news content from others. In general, identifying the content of an article in conventional techniques requires the knowledge on semantics of phases, which is a hard problem in itself. However, in Web news scenario, alternative solutions are possible, because Web pages are semi-structured documents written in markup language (mostly in HTML), which contain tags and structures dividing the context into fragments with specific semantics. By properly utilizing them, recently proposed techniques to reduce patterns or to match similar structures, reported can be applied to automatically extract specific parts of HTML documents. Therefore we are motivated to find a similar solution to automatically extract Web news content.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 327–338, 2006.

According to the Web page modeling method, existing Web extraction techniques generally fall into two main categories: tag sequence based v.s. tree based. After continuous improvement, these techniques can be easily applied, with an acceptable time complexity; but they still rely on heuristics to certain extent for solving the semantic problem. Therefore in this paper, we propose to combine the advantages of tag sequence based techniques and tree based techniques, so as to come up with a hybrid, effective and efficient solution. In particular, our contributions are as follows:

1. We propose an extended sequence representation of Web page, named as *TSReC*(Tag Sequence with Region Code), which can reserve necessary tree structure information. Building from one pass scanning of HTML and region code encoding, it is suitable for both tag sequence based and tree based extraction.
2. We propose an effective algorithm based on *TSReC* for automated Web news content extraction. It contains two procedures, namely, Sequence Matching and Tree Matching. The former one can detect and remove the identical parts of Web news pages, such as navigation bars, copyright notes. The latter one can match and remove the similar structures of Web news pages, such as advertisements and activities. Consequently, our algorithm can differentiate Web news content from others.
3. Empirical evaluation of our algorithm is performed on news pages crawled from real Web sites. The results show that our algorithm is both effective and efficient.

The rest of this paper is organized as follows. In section 2, we review some key Web extraction techniques from the literature, as well as the major work related to Web news extraction. In section 3 the problem of Web new content extraction is defined, and *TSReC* is discussed in section 4 along with an algorithm for building it. In section 5, we discuss how the Web news content extraction can be applied to a search engine, and actually provide the details of our algorithms. Finally, empirical evaluation is studied in section 6, and the conclusion is given in section 7.

2 Related Work

The great demand on Web data extraction has attracted many researchers [11] in recent years, and great efforts have been paid in finding automatic solutions to free people from manual work. Earlier research works were more on semi-automatic tools [8], whereas later ones on automated extraction techniques have become more popular. However, depending on how to model the HTML Web pages, these techniques mainly fall into two categories: tag sequence based v.s. tree based techniques.

Representative research works on tag sequence based are Stalker [14] and RoadRunner [5]. In Stalker (which also has a commercial version called FETCH), a wrapper program for extraction is learned from the tag sequence of a Web page. For example, a template based on the fragment "513 Pico, Venice, Phone 1-800-555-1515" may be derived as "513 Pico, *, Phone 1-*-555-1515", which contains two fields for location and district number. By providing enough positive and negative examples, a simple fail-and-release strategy is proposed, which starts from the rigid template, and relaxes the restrictions whenever of the accuracy is found to be not good. As an advantage, utilizing tag sequence enables the authors to apply existing

sequence matching techniques. In RoadRunner system, an algorithm is proposed to infer union-free regular expressions that represent page templates, based on tag sequence as well. Regular expression techniques are adopted thus a solid theory foundation exists. However, problems arise when iterating the sequence from one section into another (e.g., from "navigation" to "news body"), due to that there is no extra information in the tag sequence for differentiating them easily. The algorithm in RoadRunner has to try every possible template for the best result, which incurs an unacceptable huge searching space, and therefore leads to an exponential time complexity. In [1], Arasu and Hector proposed an improved version called ExAlg with a polynomial time complexity by employing several heuristics. There are other works based on tag sequence-like data structures [17, 4], but still encounter similar problems. Overall, viewing a Web page in HTML as a tag sequence (single words and tags) makes the pattern detection to be easier, but causes handling nesting structures to be harder.

More recently, tree based extraction methods are proposed, such as [15] and [18]. Both of them limit the matching process to work only under sub trees, which helps to differentiate unrelated sections of a Web page. Although sub tree structure brings out some light, pattern reduction turns out to be the dark side. Based on the results of restricted tree edit distance computation process, the work in [15] attaches wildcards to tree nodes and employs heuristics when there is a need to generalize. In [18], an improved version, based on a novel technique named partial tree alignment, is proposed. It can align corresponding data fields in data records without doing wildcards generalizations. However, its assumption on no complex nesting structures limits its use in applications [13], and in special cases adaptation or new techniques have to be made.

Additionally, there are some other related works to ours, which try to utilize the visual cues of Web page to do extraction [19, 12, 7, 2, 3]. Visual cues of a Web page can be derived after a parsing and rendering process. According to visual features, such as layout, font size and color, extraction of specified pieces of Web page content is possible [3]. Further solutions for more complex extraction tasks can also be obtained [19, 12]. However, the feature selection nature of these techniques requires many thresholds or heuristics, which should be trained first and is usually domain specific. An example work in this category is [7], which attempts to automatically extract Web page titles in this category. Comparing to it, our method does not rely on as many heuristics or thresholds.

3 Web News Content Extraction - The Problem

As an example, Fig. 1 shows a Web news page crawled from Yahoo News, in which we can see that, besides news content, there are other components like navigation areas, events, related links, and advertisements. The objective of Web news content extraction is to find out just *the content*, which is the main part of the Web page after removing the components mentioned above.

One reasonable assumption is that related Web news pages coming from the same Web site share an almost similar page layout and structure, which is usually called the *template*. Actually these web pages are generated by filling a web page template with

Fig. 1. Example Web news Page from Yahoo

Fig. 2. An fragment HTML and its *TSReC* representation

values queried from the backend database. Therefore most Web data extraction works rely on comparing or matching Web pages that follow the same template. Our method is also based on this idea.

Retrieving Web pages following the same template is possible and practical. Research on clustering or classifying Web pages [15, 6] enables us to do such retrieval in theory, and link analysis techniques in nowadays search engines make it practical. In the Web news domain, as we find out, even simply using the most similar links to the Web pages could achieve acceptable results.

Before we proceed to discussing the algorithms for identifying and extracting the content of Web news, we first give out the definition of *template*.

Definition (Template): A *template* is an incomplete Web page based on which a complete Web page can be generated by filling reserved fields with values. It usually consists of a *common part*, *regular part* and *content part*:

1. *Common part* refers to reserved texts which can not be changed.
2. *Regular part* refers to reserved rigid structure which contains unfilled fields for future values.
3. *Content part* is the reserved area which can be filled with arbitrary html fragments.

Referring back to Fig. 1, we have a feel for various parts in the template acccording to the above definition. Navigation areas (marked by number 1) are the *common parts*, since they are fixed. Events (marked by number 2), relate links (marked by number 3), and advertisements (marked by number 4) are the *regular parts*, because they adhere to the same structure across different Web news pages even though the inner fields may change. The *content parts* are the rest of that page (not marked), which can be freely filled in during generation, and tend to have no fixed patterns.

With the understanding of the *template*, we can divide the Web news content extraction problem into two sub problems, namely, matching for the *common part* and matching for the *regular part*. As we have reviewed in section 2, the former one can be easily solved by sequence matching, whereas the latter one should be better done by tree matching. In order to combine them into the same framework, we have to first design a data structure that is suitable for both techniques.

4 *TSReC*: Tag Sequence with Region Code

In this section we introduce an extended version of tag sequence, namely, Tag Sequence with Region Code (*TSReC*), which is designed for applying both tag-sequence based and tree based extraction algorithms.

As we know, a tag sequence is suitable for extraction but it does not hold any structural information. This backward prevents the utilization of sub trees' information from solving cross trees' ambiguity. For example, by matching on the sequence, we are not able to differentiate the content and advertisement in HTML; but with the help of structural information, the boundaries of them are clear because they are usually resident under different sub trees.

Therefore the basic idea is to extend a tag sequence with extra structural information. In recent database research, the region code in XML processing [9] has proven to be an ideal way to attach structural information in element based storage. With extra storage of a few numbers (region code), all structural relationships can be reserved, such as parent-child, ascent-decedent, and sibling relationships. For our work, we adopt the idea behind the region code, and define TSReC as follows.

Definition (TSReC): TSReC is a sequence of elements, each of which is defined as

$$TS =< N, RC_b, RC_e, RC_p, RC_l, C >$$

where:

1. N is the name of TS, which usually is the same as the HTML tag creating it.
2. RC_b, RC_e, RC_p, and RC_l are region codes, which correspond to begin, end, parent, and level, respectively.
3. C is the content of TS, which may contain inner HTML tags and text, or be empty.

In Fig. 2, there is an example illustration of *TSReC*. At the top part there is a fragment of HTML which shows some links of related articles in science, and there are two categories. In the bottom is its corresponding *TSReC* fragment. With respect to the definition of *TSReC*, we can see that each element represents a tag in HTML with a corresponding (identical) name. For each of them, four numbers, say RC_b, RC_e, RC_p, and RC_l , are stored in order to keep the structural information. As we will learn in the later algorithm for generating *TSReC*, RC_b and RC_e are begin-end region code identical to XML processing. We use the parent's begin code as the child's RC_p , and also calculate the child's RC_l according to the parent's level. Some TS in *TSReC* may have content consisting of simple HTML tags and texts, which is the same as the conventional tag sequence. As *TSReC* is an extended tag sequence, the sequence matching is supported naturally. On the other hand, the tree matching method can also be applied. Necessary operations, such as calculation of parent-child relationship (by utilizing the RC_p), calculation of ascent-decedent relationship (by utilizing the RC_b and RC_e), are supported. Taking the line 108, line 110 and line 117 as the example, with simple calculation, we know the line 108 and line 110 are under the same sub tree (for $RC_b(108) < RC_b(110) < RC_e(110) < RC_e(108)$), whereas the line 117 in another sub tree (for $(RC_b(117), RC_e(117))$ is not in $(RC_b(108), RC_e(108))$).

```
1  Algorithm:buildTSReC(w)
   /* w as input is a web page */
2  TSReC tsrec /* variable for TSReC holder */
3  TS temp_TS /* temp variable of TS */
4  Stack S /* a stack facility used in the algorithm */
5  int count, level, parent
6  while t = readNextTerm(w) do
7      if t is open Tag then
8          ts = getTop(S)
9          if t breaks text flow then
10             if ts is null then count = 0, level = 0, parent = 0
11             else count++, level = ts.RC_b, parent = ts.RC_l + 1
12             temp_TS = createNewTS(t, count, level, parent)
13             push(S, temp_TS)
14         else/* t does not break text flow */
15             appendToContent(ts, t)
16     else if t is close Tag then
17         if t breaks text flow then
18             ts = pop(S)
19             count++, ts.RC_e = count
20             append(tsrec, ts)
21     else if t is Text then
22         ts = getTop(S)
23         appendToContent(ts, t)
24 return tsrec
```

Fig. 3. buildTSReC - Algorithm for building *TSReC* from Web Page

TSReC can be easily built up by one-pass scanning of a Web page. Fig. 3 shows the algorithm for building it, which is modified from the conventional one for building tag trees. Basically the algorithm scans the Web page tag by tag (line 6, where text is also treat as a tag), and *TSReC* elements are created in lines 7-23 while computing the begin-end region code. According to the different types of tokens (open tag in lines 7-15, close tag in lines 16-20), different actions are taken. Note that a new TS is created only when a tag comes to break the text flow (line 9, line 17), such as "P", "DIV", "TABLE" and so on. This heuristic helps us get rid of the effect of HTML decoration tags (such as "B", "FONT", "A"), which extraction algorithms usually do not care. Finally, after one pass scanning, a *TSReC* instance is returned as the result.

5 Automated Web News Content Extraction

Having defined *TSReC*, in this section we discuss a hybrid method for automated Web news content extraction. We lable our algorithm as hybrid because it combines sequence matching techniques with tree matching techniques, in which the former is used for identifying and removing the *common part*, and the latter is for *regular part*. The overall flow chart of our method is as given in Fig. 4.

The process starts with a Web news page (n1) as its input, and returns the content as its output. Firstly, a function called `getMostRelated` is invoked to get another Web news page probably sharing the same template (cf. section 3). In our evaluation, we simply use the Web page from the related links with its URL most similar to n1's. Then we build *TSReC*s for both Web news pages (i.e., (n1 and n2). After the *TSReC* structures

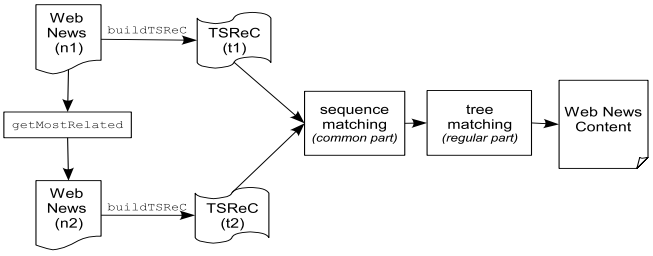


Fig. 4. The Processing Flow of Our Hybrid Method for Web News Content Extraction

are built up, sequence matching is performed which is followed by tree matching. Each of them will identify and classify specific parts by assigning class marks to elements of *TSReC*. Finally elements with no class marks are output, which are just the news content we need.

5.1 Sequence Matching for *Common Part*

Referring back to the definition in section 3, the target of sequence matching is to find out the *common part* which is supposed identical in two individual Web news pages (cf. Area 1 in Fig. 1). Intuitively, we may think that simply comparing two sequences (*TSReC*) can lead us to find out the *common part*. However, things are often a bit more complex, since there are usually more than one *common part*, between each two of which are variable *regular* and *content parts*. The matching process thus becomes not so easy as we have to skip some tags in the sequence to get the best matching result. Due to this reason, we are motivated to adopt a conventional string edit distance calculation algorithm to achieve the goal.

String edit distance calculation is to find out how similar two strings are. A solution based on conventional dynamic programming has a byproduct showing the mappings. For example, after the calculation, besides the edit distance between S1("abc") and S2("ac"), we can also know that "a in S1 maps to a in S2", "c in S1 maps to c in S2", and "b in S1 has no mapping". Taking the advantage of that, we propose an algorithm in Fig. 5, based on the dynamic programming solution of string edit distance calculation for sequence matching.

Our sequence matching algorithm takes two *TSReCs* (t_1, t_2) as the input, does matching for the common parts and marks them. Being the same as conventional calculation of string edit distance, our algorithm also uses dynamic programming techniques (lines 4-14). Different from comparing characters in string, our algorithm compares TSs in *TSReC* (line 12). If two TSs have the same tag name and content text, we regard them as they matched, in which case corresponding marking operations are performed (line 14). Otherwise, we move on to look for further matched tags. Note that, in our implementation, we always let t_1 be the shorter sequence so as to optimize the algorithm further (ie., let the shorter sequence lead the other loop).

After the sequence matching on two *TSReC*s, the *common parts* in each of them have been matched and removed. Therefore as the next step we only have to identify and remove the *regular parts*.

```
1  Algorithm:sequenceMatch(t1, t2)
2  int t1size = sizeof(t1)
3  int t2size = sizeof(t2)
4  int M[t1size + 1][t2size + 1]
5  for i = 0 to t1size + 1 do M[i][0] = i
6  for i = 0 to t2size + 1 do M[0][i] = i
7  for i = 1 to t1size do
8      for j = 1 to t2size do
9          ts1 = the ith ts of t1
10         ts2 = the jth ts of t2
11         int match = 1
12         if ts1 and ts2 have same tag name and content text then match = 0
13         M[i][j] = min(M[i-1][j-1]+match, M[i-1][j], M[i][j-1])
14         if M[i][j] == M[i-1][j-1]+match then mark matching of ts1 and ts2
```

Fig. 5. Sequence Matching Algorithm on *TSReC* for *Common Part*

5.2 Tree Matching for *Regular Part*

However, differentiating the *regular part* from the *content part* of a Web news page is not as easy as finding *common parts*. Sometimes the mission is even impossible without utilizing semantics, if the *regular parts* in the template are too flexible to fill anything. So it is reasonable to assume that the *regular part* has rigid format structure while the *content part* has not. For example, the *regular part* is usually like "<UL>(<LI><A>(text)</A>)*</UL>", whereas the *content part* can be any free HTML fragment. Based on our observation, this assumption is common in real-life Web pages for news, and therefore our algorithm takes advantage of it in differentiating the *regular part*.

In our approach, before doing tree matching, we do grouping first. After removing the TSs of *common parts*, the grouping process tries to find out which of the left TSs are under the same sub trees. It is necessary in order to do tree matching subsequently; its algorithm is given in Fig. 6.

The grouping algorithm takes a list containing different parts (ie., the rest of the *TSReC* excluding *common parts*) as its input, and returns groups as the output. Each group is a TS (called group parent) with two numbers (called group region), namely, group beginning and group ending. Any TS whose region codes (beginning and ending) are in the group region belongs to that group. The grouping method is to simply check the parent and tree level (line 5) of siblings. If siblings share the same parent and tree level, they are under the same sub tree so we add them into the same group (line 6) by simply extending the group region. Otherwise, a new group will be created (lines 8-9).

After grouping, we get sub trees which correspond possibly to the regular or *content parts*. So we have to differentiate the *regular part* from the others. As we have discussed, the determination of whether a sub tree in a Web page is a *regular part*, is measured by weather there is a sub tree in the other Web page sharing the same rigid pattern. Accordingly, we have the tree matching algorithm as shown in Fig. 7.

Basically the tree matching algorithm tries to find for each group (sub tree) a matched group in the other Web page, which is done in a nest-loop (lines 7-14). The outer loop iterates on t_1 (the Web page to be extracted), and the inner loop iterates on t_2 (the Web page as reference). The parameter *rsm* is the sequence matching result, which is necessary in extracting different parts (lines 2-3). However, we can do a simple optimization according to the following observation: Usually regular parts share the same

```
1  Algorithm:subTreeGroup(dp)
2  List groups /* list for holding groups, each group is a list of
   TS, and initial a group using first element of dp */
3  List group = createGroup(dp[0], dp[0].level, dp[0].parent)
4  for i = 1 to sizeof(dp) do
5      if dp[i].level == group.level && dp[i].parent = group.parent then
6          append(group, dp[i])
7      else
8          append(groups, group)
9          group = createGroup(dp[i], dp[i].level, dp[i].parent)
10 return groups
```

Fig. 6. Grouping Algorithm

order even in different Web pages (e.g. "event" appearing before "advertisement" in one Web page seldom appears in reverse order in another Web page). This optimization in our algorithm is as reflected in line 9 (i.e., to start matching after the previous matching position).

```
1  Algorithm:treeMatch(t1, t2, rsm)
2  List< TS > dp1 = getDifferentPart(t1, rsm)
3  List< TS > dp2 = getDifferentPart(t2, rsm)
4  List< TS > groups1 = getSubTreeGroup(dp1)
5  List< TS > groups2 = getSubTreeGroup(dp2)
6  int nextMatch = 0
7  for i = 0 to sizeof(groups1) do
8      group1 = compactGroup(getGroup(groups1, i))
9      for j = nextMatch to sizeof(groups2) do
10         group2 = compactGroup(getGroup(groups2, j))
11         if groupMatch(group1, group2) then
12             mark group1 not content
13             nextMatch = j+1
14             break
15 return groups
```

Fig. 7. Tree Matching Algorithm on *TSReC* for *Regular Part*

A notable function in the tree matching algorithm is the one named `compactGroup` (line 10). It is designed to handle repeatable fields in the *regular part*. For example, referring back to the example of in Fig. 2, where "Science News" is a regular part which usually contains several lines for titles and links of news in identical topics. However, it is a repeatable field, which can have proper instances according to the underlying database. Since we target to find rigid patterns, multiple instances of a repeatable field should be reduced, and return in the form of "<UL>(<LI><A>(text)</A>)*</UL>". In the algorithm of `compactGroup`, we thus only check whether siblings share the same sequence patterns. It is a fairly simple process, thus we leave its detail in the full version paper due to space limitation.

After the tree matching process, we have identified both the *common part* (by sequence matching) and the *regular part*. Thus the rest of the Web page is unmarked and is the content part needed. By simply returning this part, we get the Web news content as desired.

6 Empirical Evaluation

As part of the proposed approach, we have built a prototype system for Web news content extraction, upon which empirical evaluation has been conducted. In this section, we describe the implementation of the prototype system, testing data bed used, evaluation method and evaluation results, and explanations of the results.

The Prototype System: All the proposed algorithms in this paper, as well as the *TSReC* data structure of our Web news content extraction system, are implemented in Java with the help of HTML Parser [16]. The prototype system accepts two Web news content pages (one to be extracted, one as the reference) sharing the same template, and returns the HTML fragment of Web news content. As we discussed in previous section, finding two Web news pages of the same template is not hard, and in fact, we just take the one having the most similar URL to the reference page's URL. We reserve the output in HTML form, which can be directly delivered to indexing facilities for parsing and indexing.

Testing Data Bed: The testing data bed used in our evaluation is manually crawled from real-life Web news sites. We have collected news pages from up to 50 Web news sites, covering news of politics, business, sports, entertainment, and life. For each Web news site, we randomly collect one page, then running a script to get from the related links the Web news page sharing probably the same template with the source. The 50 Web news sites, which are 25 in English and 25 in Simplified Chinese, are selected from the online categories of famous search engines (google.com and baidu.com). The names of the Web sites are given in Table 1.

Evaluation Method: As an empirical evaluation, our evaluation is conducted in the following steps. We first extract Web pages from each Web site by running our crawler. Then we manually check whether the result is the content of Web news. We take the content part as the content of Web news according to the definition in section 3, while the semantics of text is not considered.

Evaluation Results and Explanation: The result of evaluation is as shown in Table 1. For each Web site, we list out the URL[1], as well as two evaluation results (R1 and R2). R1 and R2 can be of the value S or F: S means that the extraction is successful and the content is correctly extracted (as judged by manual checking); F means that the extraction fails, which may be caused by various reasons.

In particular, F_1 means that the extraction failed with no output. The reason is that the content part is incorrectly identified as the regular part, for its content may be simply a sequence of text sentence (no highlighted sub section titles, no hyperlinks for key words). So the `compactGroup` function takes a wrong action on it. This kind of situation is somehow common as there are 5 cases in our data set. However, it is easy to be handled by using a set of simple heuristic rules, such as by judging how long the text is (eg., when it exceeds specific threshold, we stop matching it). Actually, those simple heuristic rules are actually applied, as shown by the later results (R2).

F_2 means that the extraction failed because the HTML source of the Web news page contains some invisible text, thus affecting the matching process. These texts are usually

[1] Due to the space limitation, however, the URLs are only indicative in Table 1. Readers are referred to [10] for the detailed URLs.

for Dynamic HTML effects, such as popup menu. By using a simple heuristic to remove it from the original source, the problem is easily fixed.

However, there are still error types F_3 and F_4 for which currently we do not have a solution. Specifically, F_3 means that there are regular parts not in rigid forms, and F_4 means that even the regular parts are highly dynamic. These errors do not follow the assumption we made before, therefore corrupt the extraction process. Fortunately, these errors are not common, and they are left as a future work.

Table 1. The Result of Evaluation On Real-life Web News Sites

	Chinese Web News			English Web News		
	URL	R1	R2	URL	R1	R2
1	news.sina.com.cn...	S	S	news.yahoo.com...	S	S
2	beijing.qianlong...	F_3	F_3	www.cbc.ca...	S	S
3	news.qq.com...	S	S	www.cnn.com...	S	S
4	politics.people....	F_2	S	www.msnbc.msn.co...	F_3	F_3
5	news.tom.com...	S	S	today.reuters.co...	S	S
6	news.xinhuanet.c...	S	S	www.un.org...	S	S
7	www.gmw.cn...	S	S	www.cbsnews.com...	S	S
8	news.163.com...	S	S	articles.news.ao...	S	S
9	news.espnstar.co...	S	S	www.usatoday.com...	S	S
10	www.southcn.com...	F_1	S	reuters.com.uk...	F_1	S
11	news.21cn.com...	S	S	www.latimes.com...	S	S
12	heilongjiang.nor...	S	S	www.heraldsum.ne...	S	S
13	www.cnhan.com...	F_1	S	smh.com.au...	S	S
14	gb.chinabroadcas...	S	S	www.columbiaspec...	S	S
15	www.yzdsb.com.cn...	S	S	www.news.com.au...	S	S
16	sports.sohu.com...	S	S	www.thelobeandm...	S	S
17	news.sports.cn...	F_2	S	www.iht.com...	F_1	S
18	www.chinanews.co...	F_4	F_4	www.nydailynews...	S	S
19	economy.enorth.c...	S	S	www.int.iol.co.z...	S	S
20	news.17173.com...	S	S	seattletimes.nws...	S	S
21	www.daynews.com....	S	S	www.dailynews.co...	S	S
22	sports.nen.com.c...	S	S	www.washingtonpo...	S	S
23	www.lanews.com.c...	S	S	quote.bloomberg...	F_1	S
24	news.huash.com...	S	S	www.signonsandie...	S	S
25	www.jhnews.com.c...	S	S	www.marketwatch...	S	S
A		19/25	23/25		21/25	24/25

Overall, as we can see from Table 1, the accuracy of our method is generally high (19/25=76% and 21/25=84% before applying the simple heuristic rules, and after is 23/25=92% and 24/25=96%). Given that our method does not rely on any a threshold nor machine learning knowledge, it is practically feasible and suitable to be applied in real life search engines.

7 Conclusion

In this paper we have studied the Web news content extraction problem and proposed an automated extraction algorithm for it. Our algorithm exhibits a hybrid method applying both sequence matching and tree matching techniques. Built on top of *TSReC* - a variant of tag sequence previously proposed by ourselves, the hybrid method benefits from combining the advantages of both sequence matching and tree matching. Empirical evaluation conducted shows that our method is highly effective and efficient. Because of the automatic nature of our method, it should be fairly straightforward for us to integrate it into a real life search engine, such as Yahoo! or Google, as a preprocessing procedure to improve indexing quality. In addition, we plan to further improve the algorithm and enhance our prototype system to make it more robust, able to handle more types of errors as mentioned in section 6.

Acknowledgments

This research was partially supported by the grants from the Natural Science Foundation of China under grant number 60573091, 60273018; the National 973 Basic Research Program of China under Grant No.2003CB317000 and No.2003CB317006; the Key Project of Ministry of Education of China under Grant No.03044 ; Program for New Century Excellent Talents in University(NCET).

References

1. A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *Proc. of SIGMOD 2003*, pages 337–348, 2003.
2. D. Cai, S. Yu, J. R. Wen, and W. Y. Ma. Vips: a vision-based page segmentation algorithm. Technical Report MSR-TR-2003-79, Microsoft Research Asia, 2003.
3. L. Can, Z. Qian, X. F. Meng, and W. Y. Lin. Postal address detection from web documents. In *Proc. of WIRI 2005*, pages 40–45, 2005.
4. C.H. Chang and S.C. Lui. Iepad: information extraction based on pattern discovery. In *Proc. of WWW 2001*, pages 681–688, 2001.
5. V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *Proc. of VLDB 2001*, pages 109–118, 2001.
6. V. Crescenzi, G. Mecca, and P. Merialdo. Wrapping-oriented classification of web pages. In *Proc. of SAC 2002*, pages 1108–1112, 2002.
7. Y. H. Hu, G. M. Xin, R. H. Song, G. P. Hu, S. M. Shi, Y. B. Cao, and H. Li. Title extraction from bodies of html documents and its application to web page retrieval. In *Proc. of SIGIR 2005*, pages 250–257, 2005.
8. A. H. F. Laender, B. A. Ribeiro-Neto, A. S. Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Record*, 31(2):84–93, 2002.
9. Q. Z. Li and B. K. Moon. Indexing and querying xml data for regular path expressions. In *Proc. of VLDB*, pages 361–370, 2001.
10. Y. Li. Evaluation of hybrid extraction method. available at http://idke.ruc.edu.cn/hybrid.
11. B. Liu. Wise-2005 tutorial: Web content mining. In *Proc. of WISE 2005*, page 763, 2005.
12. B. Liu, R. L. Grossman, and Y. H. Zhai. Mining data records in web pages. In *Proc. of KDD 2003*, pages 601–606, 2003.
13. B. Liu and Y. Zhai. Net - a system for extracting web data from flat and nested data records. In *Proc. of WISE 2005*, pages 487–495, 2005.
14. I. Muslea, S. Minton, and C. A. Knoblock. A hierarchical approach to wrapper induction. In *Proc. of Agents 1999*, pages 190–197, 1999.
15. D.C. Reis, P.B. Golgher, A.S. Silva, and A.H.F. Laender. Automatic web news extraction using tree edit distance. In *Proc. of WWW 2004*, pages 502–511, 2004.
16. D. Udani. Html parser project. available at http://sourceforge.net/projects/htmlparser.
17. J. Wang and F. H. Lochovsky. Data extraction and label assignment for web databases. In *Proc. of WWW 2003*, pages 187–196, 2003.
18. Y. Zhai and B. Liu. Web data extraction based on partial tree alignment. In *Proc. of WWW 2005*, pages 76–85, 2005.
19. H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. T. Yu. Fully automatic wrapper generation for search engines. In *Proc. of WWW 2005*, pages 66–75, 2005.

A Hybrid Sentence Ordering Strategy in Multi-document Summarization

Yanxiang He[1], Dexi Liu[1,2], Hua Yang[1,2], Donghong Ji[2,3], Chong Teng[1,2], and Wenqing Qi[1,4]

[1] School of Computer, Wuhan University, Wuhan 430079, P.R. China
[2] Center for Study of Language and Information, Wuhan University, Wuhan 430079, P.R. China
[3] Institute for Infocomm Research, Heng Mui Keng Terrace 119613, Singapore
[4] Huangshi Institute of Technology, Huangshi 435003, P.R. China
yxhe@whu.edu.cn, dexiliu@gmail.com

Abstract. In extractive summarization, a proper arrangement of extracted sentences must be found if we want to generate a logical, coherent and readable summary. This issue is special in multi-document summarization. In this paper, several existing methods each of which generate a reference relation are combined through linear combination of the resulting relations. We use 4 types of relationships between sentences (chronological relation, positional relation, topical relation and dependent relation) to build a graph model where the vertices are sentences and edges are weighed relationships of the 4 types. And then apply a variation of page rank to get the ordering of sentences for multi-document summaries. We tested our hybrid model with two automatic methods: distance to manual ordering and ROUGE score. Evaluation results show a significant improvement of the ordering over strategies losing some relations. The results also indicate that this hybrid model is robust for articles with different genre which were used on DUC2004 and DUC2005.

1 Introduction

With rapid growth of online information, it becomes more and more important to find and describe textual information effectively. Although it is convenient for users to obtain a great deal of documents with a search engine, users have to take the tedious burden of reading all those text documents. Automatic text summarization [1] that provides users with a condensed version of the original text, tries to release our reading burden.

Despite the beginnings of research on alternatives to extraction, most work today still relies on extraction of sentences from the original document to form a summary [2]. In extractive summarization, a proper arrangement of these extracted sentences must be found if we want to generate a logical, coherent and readable summary. This issue is special in multi-document summarization. Sentence position in the original document, which yields a good clue to sentence arrangement for single-document summarization, is not enough for multi-document summarization because we must consider inter-document ordering at the same time [3].

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 339 – 349, 2006.

Barzilay et al. [4] showed the impact of sentence ordering on readability of a summary and explored some strategies for sentence ordering in the context of multi-document summarization. According to Barzilay's experiments, naive ordering algorithms such as majority ordering (selects most frequent orders across input documents) produces a good ordering only when the input articles' orderings have high agreement; chronological ordering (orders facts according to publication date) do not always yield coherent summaries when the information is not event-based. Based on the human study that topically related sentences always appear together, Barzilay et al. proposed an algorithm that first identifies topically related groups of sentences and then orders them according to chronological information. Okazaki et al. [3] introduced an approach to coherent text structuring for summarizing newspaper articles. They improved chronological ordering by resolving antecedent sentences of arranged sentences and combining the refinement algorithm with topical segmentation and chronological ordering. Lack of presupposition obscures what a sentence is saying and confuses the readers. So Okazaki et al. employed the refinement algorithm to refine the chronological ordering by selecting some proper sentences and setting them before the sentence losing presupposition. Lapata [5] proposed another method to sentence ordering based on an unsupervised probabilistic model for text structuring that learns ordering constraints from a large corpus. This model assumes that the probability of any given sentence is determined by its immediately antecedent. Lapata estimated transitional probability between sentences by some features such as verbs (precedence relationships of verbs in the corpus), nouns (entity-based coherence by keeping track of the nouns) and dependencies (structure of sentences).

Researchers have explored some favorable strategies for ordering sentences, but the limitation of above-mentioned strategies is still obvious. Barzilay's strategy paid attention to chronological and topical relation whereas ignored sentence original ordering in the articles where summary comes from. Hence, if the articles are not event-based, the quality of summary will decrease because the temporal cue is invalid. Although Okazaki's strategy makes full use of the original articles to fill the presupposition of an arranged sentence, it is not a robust method because the detection of presupposition depends on only one original article. Moreover, the proper threshold of presupposition detection is difficult to decide. As for Lapata's strategy, the probabilistic model of text structure is trained on a large corpus, so it performs badly when genre of the corpus and the article collection are mismatched.

To overcome the limitation mentioned above, we propose a hybrid model for sentence ordering in extractive multi-document summarization that combines four relations between sentences - chronological relation, positional relation, topical relation and dependent relation. Our model regards sentence as vertex and combined relation as edge of a directed graph on which the approximately optimal ordering can be generated with PageRank analysis. Evaluation of our hybrid model shows a significant improvement of the ordering over strategies losing some relations and the results also indicate that this hybrid model is robust for articles with different genre.

The rest of this paper is organized as follows. We first present the hybrid model and the strategy to generate an approximately optimal ordering. The subsequent section (Section 3) addresses evaluation method and experiment results. In Section 4 we conclude this paper and discuss future work.

2 Model

In this section, we introduce the hybrid sentence ordering model that combines four relations between two sentences extracted from the article collection (documents to be summarized), and then explore a PageRank analysis method to search an approximately optimal ordering. Four relations are positional relation, chronological relation, topical relation, and dependent relation, which individually correspond to four strategies of master ordering, chorological ordering, topical relation, and text structure.

2.1 Four Relations

Positional Relation. Most of the sentence ordering strategies in extractive single-document summarization arrange sentences according to their original positions in the article. So we employ positional relation to match the original ordering arranged by author. Suppose s_i and s_j are two sentences in article collection, quantitatively positional relations between s_i and s_j are defined as follow:

$$P_{i,j} = \begin{cases} 1, \text{ if } s_i, s_j \text{ are extrated from the same article and } s_i \text{ precedes } s_j \\ 0 \quad , \text{else} \end{cases}. \tag{1}$$

Topical Relation. Investigations into the interpretation of narrative discourse [6] have shown that specific lexical information (e.g., verbs) plays an important role in determining the discourse relations between propositions. Centering Theory (CT) [7] is an entity-based theory of local coherence, which claims that certain entities mentioned in an utterance are more central than others and that this property constrains a speaker's use of certain referring expressions (in our model, noun is considered as entity). Accordingly, stemmed verbs and nouns are selected while calculating the sentence topical relation and other words are ignored. Topical relation can be operationalized in terms of word overlap:

$$T_{i,j} = \frac{2\left|words(s_i) \cap words(s_j)\right|}{\left|words(s_i)\right| + \left|words(s_j)\right|}. \tag{2}$$

Where $words(s_i)$ is the set of nouns and stemmed verbs in sentence s_i, $\left|words(s_i)\right|$ is the number of words in $words(s_i)$.

Dependent Relation. Following the same methodology used by Lapata [5], we use $P(s_i \left| s_{i-1}\right.)$ to describe the dependent relation of sentence s_i on its antecedent s_{i-1}. Of course estimating would be impossible if s_i and s_{i-1} are actual sentences because it is unlikely to find the exact same sentence repeated several times in a corpus. What we can find and count is the number of occurrences a given word appears in the corpus. We will therefore estimate $P(s_i \left| s_{i-1}\right.)$ from features that express its content.

$$P(s_i \left| s_{i-1}\right.) = P(\left\langle a_{i,1}, a_{i,2}, ..., a_{i,n} \right\rangle \left| \left\langle a_{i-1,1}, a_{i-1,2}, ..., a_{i-1,m} \right\rangle\right.). \tag{3}$$

Where $\langle a_{i,1}, a_{i,2}, ..., a_{i,n} \rangle$ are features of sentence s_i and $\langle a_{i-1,1}, a_{i-1,2}, ..., a_{i-1,m} \rangle$ are of sentence s_{i-1}.

In our model, noun and stemmed verb are employed as the features of a sentence. We assume that these features are independent and that $P(s_i | s_{i-1})$ can be estimated from the pairs in the Cartesian product defined over the features expressing sentences s_i and s_{i-1}: $\langle a_{i,r}, a_{i-1,k} \rangle \in s_i \times s_{i-1}$. Under these assumptions $P(s_i | s_{i-1})$ can be written as :

$$\begin{aligned} P(s_i | s_{i-1}) &= P(a_{i,1} | a_{i-1,1}) P(a_{i,1} | a_{i-1,2}) ... P(a_{i,r} | a_{i-1,k}) ... P(a_{i,n} | a_{i-1,m}) \\ &= \prod_{(a_{i,r}, a_{i-1,k} \in s_i \times s_{i-1})} P(a_{i,r} | a_{i-1,k}) \end{aligned} . \tag{4}$$

The probability $P(a_{i,r} | a_{i-1,k})$ is estimated as:

$$P(a_{i,r} | a_{i-1,k}) = \frac{f(a_{i,r}, a_{i-1,k})}{\sum_{a_{i,r}} f(a_{i,r}, a_{i-1,k})} . \tag{5}$$

where $f(a_{i,r}, a_{i-1,k})$ is the number of occurrences feature $a_{i,r}$ is preceded by feature $a_{i-1,k}$ in the corpus. The denominator expresses the number of occurrences $a_{i-1,k}$ is attested in the corpus (preceded by any feature).

Instead of a large corpus, the article collection is used as training data because we try to make full use of text structure of the original articles. Furthermore, mismatch between the article collection and training corpus is resolved. In additional, we do not smooth the probability because all sentences in summary are extracted from the corpus (the article collection), and consequently, it is impossible that the occurrence of $a_{i-1,k}$ is equal to zero.

Then, we define the dependent relation between two sentences. For two sentence s_i and s_j, suppose s_i is immediately preceded by s_j, the dependent relation of s_i on s_j is defined according to formula (6).

$$D_{i,j} = P(s_i | s_j) = \prod_{(a_{i,r}, a_{j,k} \in s_i \times s_j)} P(a_{i,r} | a_{j,k}) . \tag{6}$$

Chronological Relation. Multi-document summarization of news typically deals with articles published on different dates, and articles themselves cover events occurring over a wide range of time. Using chronological relation in the summary to describe the main events helps the user understand what has happened. In our model, we approximate the sentence time by its first publication date. It is an acceptable approximation for news events because the first publication date of an event usually corresponds to its occurrence in real life [4].

Suppose sentence s_i and s_j come from the x^{th} and the y^{th} article individually. The chronological relation between sentence s_i and s_j is defined as:

$$C_{i,j} = \begin{cases} 1 & \text{, if the } x^{\text{th}} \text{ article has earlier publication date than the } y^{\text{th}} \\ 0 & \text{, else} \end{cases} . \tag{7}$$

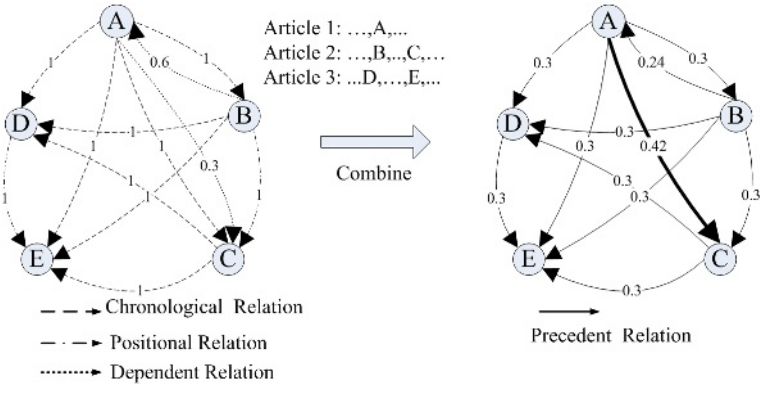


Fig. 1. An example for the hybrid model

2.2 Hybrid Model

For the group of sentences extracted from the article collection, we construct a directed graph (we call it Precedence Graph). Let $G = (V, E)$ be a directed graph with the set of vertices V and set of directed edges E, where E is a subset of $V \times V$. For a given vertex V_i, let In(V_i) be the set of vertices that point to it (predecessors), and let Out(V_i) be the set of vertices that vertex V_i points. In our hybrid model, the Precedence Graph is constructed by adding a vertex for each sentence in the summary, and edges between vertices are established using sentence relations. Three of the predefined quantitative relations are integrated to precedent relation using linear model as follows.

$$R_{i,j} = \lambda_P P_{i,j} + \lambda_D D_{i,j} + \lambda_C C_{i,j} . \tag{8}$$

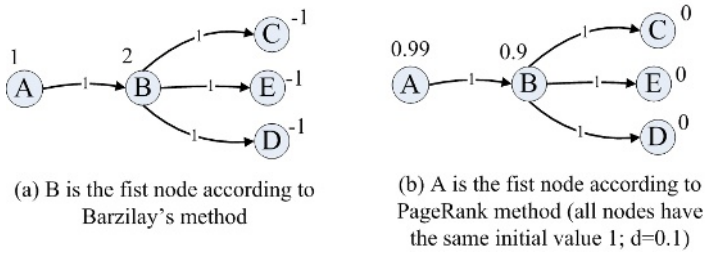
where λ_P , λ_D , λ_C are the weight of positional relation, dependent relation and chronological relation individually. If $R_{i,j}$>0, there exist a directed edge with value $R_{i,j}$ from V_i to V_j (see figure 1).

In our case, we set λ_P =0.3, λ_D =0.4, λ_C =0.3 manually. By the way, figure 1 does not contain topical relation because it will be used only after the first vertex has been selected.

2.3 Determining an Order

Unfortunately, the next critical task of finding an optimal road according to the Precedence Graph is a NP-complete problem. Cohen et al. [8] has proved it by reducing from CYCLIC-ORDERING [9]. However, using a modified version of topological sort provides us with an approximate solution. For each node, Barzilay et al. [4] assign a weight equal to the sum of the weights of its outgoing edges minus the sum of the weights of its incoming edges. In his method, the node with maximum weight first picked up and arranged as the first sentence in summary. Then the node just selected and its outgoing edges are deleted from the Precedence Graph, and the weights of the remaining nodes in the graph are updated properly. This operator iterates through all

nodes until the graph is empty. However, the method only considering the local relationship between two sentences is not precise enough. For example in figure 2, vertex "A" precedes vertex "B" and should be arranged before "B", but we get an ordering "B, A" according to Barzilay's method.



(a) B is the fist node according to Barzilay's method

(b) A is the fist node according to PageRank method (all nodes have the same initial value 1; d=0.1)

Fig. 2. An example for PageRank method and Barzilay's method

Our model employs PageRank method [10], which is perhaps one of the most popular ranking algorithms, and was designed as a method for Web link analysis. PageRank method is a graph-based ranking algorithm to decide the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. The basic idea implemented by the ranking model is that of "voting" or "recommendation". When one vertex links to another one, it is basically casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex. Score of vertex V_i is defined as:

$$S(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{S(V_j)}{\left|Out(V_j)\right|}. \tag{9}$$

where d is a parameter set between 0 and 1 (we let d=0.1 in our experiments).

In the context of Web link analysis, it is unusual for a vertex to include partial links to another vertex, and hence the original definition for graph-based ranking algorithms is assuming unweighted graphs. However, the Precedence Graph in our model includes partial links between the vertices. For example in figure 1, the precedent relation (or link) between two vertices has "weight" in [0, 1]. In formula (9), where vertex with higher "in degree" has higher score, but it is contrary in our model that vertex with higher "out degree" should has higher score. The ranking model therefore is amended as:

$$S(V_i) = (1-d) + d * \sum_{V_j \in Out(V_i)} R_{i,j} \frac{S(V_j)}{\sum_{V_k \in In(V_j)} R_{k,j}}. \tag{10}$$

While the final vertex scores (and therefore rankings) for weighted graphs differ significantly with their unweighted alternatives, the number of iterations to convergence and the shape of the convergence curves is almost identical for weighted and unweighted graphs [11].

(A) (APW19981027.0491, 2) The UN found evidence of rights violations by Hun Sen prompting the US House to call for an investigation. (B) (APW19981113.0251, 1) The three-month governmental deadlock ended with Han Sen and his chief rival, Prince Norodom Ranariddh sharing power. (C) (APW19981016.0240, 1) Han Sen refused. (D) (APW19981016.0240, 1) Opposition leaders fearing arrest, or worse, fled and asked for talks outside the country. (E) (APW19981022.0269, 7) Cambodian elections, fraudulent according to opposition parties, gave the CPP of Hun Sen a scant majority but not enough to form its own government. (F) (APW19981118.0276, 1) Han Sen guaranteed safe return to Cambodia for all opponents but his strongest critic, Sam Rainsy, remained wary. (G) (APW19981124.0267, 1) Chief of State King Norodom Sihanouk praised the agreement.

Fig. 3. An example for the result of our hybrid model (The text is selected from DUC2004 dataset; publication date and position are in the bracket before sentence.)

The next step following PageRank algorithm is the selection of vertex. We first select the vertex with highest score as the first sentence in summary. For the following vertices, not only the score itself but also the topical relation with immediately previous vertex should be taken into consideration. So the succeed vertex of V_k should satisfy:

$$V_i = \arg\max_{V_i}((1-\lambda_T)S(V_i) + \lambda_T T_{k,i}) . \tag{11}$$

where λ_T is the weight of topical relation (we let λ_T =0.4 in our experiments) . When the successor vertex of V_k is picked up, vertex V_k and all edges linked with it are deleted from the Precedence Graph. This operator iterate until the graph is empty, and then an ordered summary is produced. In context of text in figure 3, our model produces an ordering "E,D,C,A,B,F,G".

3 Experiments

We conducted an experiment of sentence ordering in multi-document summarization to test the effectiveness of our hybrid model.

3.1 Dataset

The Document Understanding Conferences (DUC) is run by the National Institute of Standards and Technology (NIST) to further promote progress in summarization and enable researchers to participate in large-scale experiments. There are 50 sets of English TDT documents. Each set contains 10 documents. Task 2 of DUC 2004 requires participants produce a short summary no more than 665 bytes for each document cluster. Four human model summaries are provided for each cluster for evaluation. We collected three groups of summaries generated by three different participants of task 2 in DUC2004 [12] where each participant was required to generate 50 summaries no longer than 665 bytes (). Three selected groups have high (id=65), fair (id=138), and low (id=111) ROUGE score [13] individually so that we can observe the degree of influence that our model affects on summarizer with different

performance. For each group, we randomly selected 10 from 50 summaries produced by one participant. Five postgraduates were employed and everyone built one ordering manually for each summary. To make full use of the positional relation and chorological relation, we tagged the position and publication date for each sentence according to its original article (each article for task 2 in DUC2004 has a publication date) (see figure 3). For the sentence not generated by extractive method, we can't find the corresponding sentence in the original article, hence, the position and publication date of the most similar sentence (decided manually) were tagged as substitute. To test the adaptive performance of our hybrid model, we also randomly selected the fourth group of testing data from DUC2005 [14], which has different genre with DUC2004.

3.2 Evaluation

We tested our hybrid model with two automatic methods. In this section, we present these methods and the evaluation results.

Distance to Manual Ordering

A number of metrics can be used to measure the distance between two rankings such as Spearman's correlation coefficient for ranked data, Cayley distance, or Kendall's [15]. We use Kendall's which based on the number of inversions in the rankings.

$$\tau = 1 - \frac{2 \times \mathrm{Inv}(O_i, O_j)}{N(N-1)/2} \tag{12}$$

where N is the number of sentences being ranked and $\mathrm{Inv}(O_i, O_j)$ is the number of interchanges of consecutive elements necessary to arrange them in their natural ordering (manual ordering in this case). If we think in terms of permutations, then τ can be interpreted as the minimum number of adjacent transpositions needed to bring one ordering to the other. We use the minimal one from 5 distance values corresponding to the 5 "ideal" orderings produced manually. Figure 4 shows the evaluation results.

As Figure 4 indicates, our hybrid ordering (HO) model yields more satisfactory result (as a whole) than any other model taking part in the comparison, including the strategy employed by the summarizer that produces the original ordering (OO). Although partially hybrid strategies such as HO-C (hybrid ordering model without chronological relation), HO-P (hybrid ordering model without positional relation), HO-T (hybrid ordering model without topical relation, and HO-D (hybrid ordering model without dependent relation) are all worse than HO, we found that different relation has significantly different influence on the ordering strategy performance. In context of DUC2004 dataset (group 1, 2, 3), HO-C performs worst and HO-T is slightly better than HO-C, whereas there aren't significant difference between HO-P, HO-D, and HO. In other word, the ordering strategy depend more on chronology and topical relation than on positional and dependent relation for DUC2004 dataset. After investigating the failure ordering of HO-C, we found that most of the articles have been used for task 2 in DUC2004 are event-based, so chronological cue is very important for ordering. Moreover, for the limitation of summary length, number of sentences in most summaries is less than 10 – the number of articles from which summary extracted, so the probability of two sentences coming from the same article is very low. Hence the importance of positional relation is not distinct.

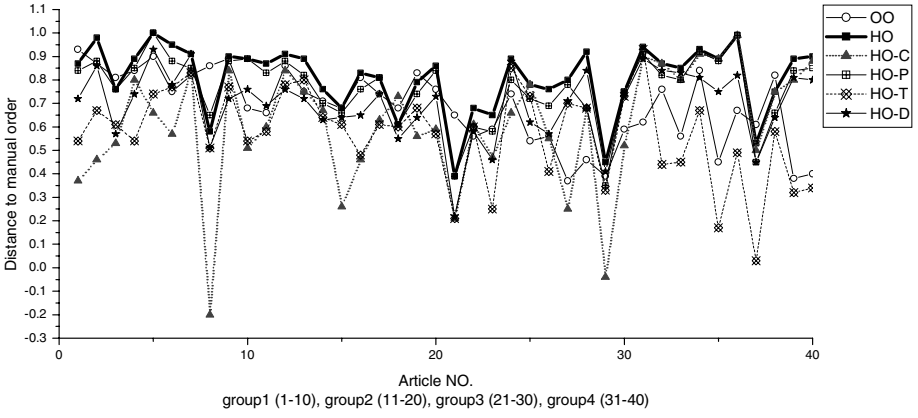


Fig. 4. Results and comparison of different models

On the contrary, HO-T and HO-P perform much worse than HO on DUC2005 dataset (group 4) while HO-D and HO-C perform closely to HO. Reason for this change is that most articles for DUC2005 are biography, where chronological cue is not so important for ordering. Although different relation plays different role in ordering summary with different genre, our hybrid model combining these relations together has robust performance. Figure 4 indicates clearly that the results of HO have no distinct change when article genre change from event-based to biography-based.

Furthermore, after reordering, the improvement for "poor" summaries (group 3) is better than for "good" summaries (group 1) because the summaries generated by a good summarizer are more readable than by bad one.

ROUGE Score

To observe whether our model is useful to improve ROUGE score of summary, we compared the ROUGE scores of the original summaries (OO) with the reordered ones (HO). Table 1 lists the results.

Table 1. ROUGE score comparison between the original and the reordered summaries

		ROUGE-1	ROUGE-2	ROUGE-3	ROUGE-4	ROUGE-L	ROUGE-W1.2
Group 1 (id=65)	OO	0.38210	0.09213	0.03352	0.01548	0.38680	0.13336
	HO	0.38210	0.09213	0.03355	0.01550	0.38702	0.13348
Group 2 (id=138)	OO	0.34298	0.07248	0.02421	0.01029	0.31179	0.10803
	HO	0.34298	0.07248	0.02430	0.01054	0.31326	0.10889
Group 3 (id=111)	OO	0.24196	0.01859	0.00279	0.00078	0.27636	0.09325
	HO	0.24196	0.01864	0.00295	0.00113	0.27951	0.09499

Table 1 shows that ROUGE-1 score have almost no change after reordering and that ROUGE2, 3, 4 scores are changed slightly. Because the content has no change after reordering, as the result, the recall-based ROUGE score has no significant

change. Although ROUGE-L score is improved by HO, but the improvement is not clear for good summaries. Detailed observation shows that the improvement of ROUGE score increases with the granularity of comparison unit (n-gram) (see figure 5). Additionally, figure 5 and table 1 show the same conclusion that sentence ordering strategy affects more significantly on bad summaries than on good ones.

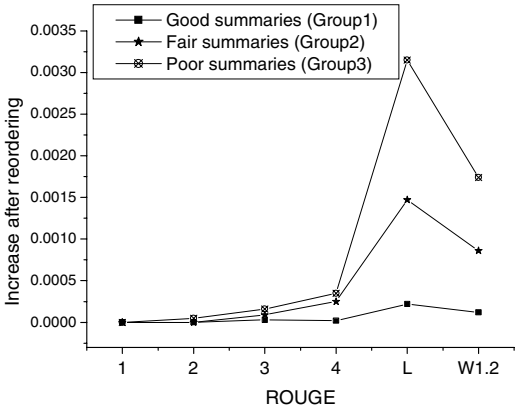


Fig. 5. Increase of ROUGE score after reordering

4 Conclusions

In this paper, we propose a hybrid model for sentence ordering in extractive multi-document summarization. We combine the four relations between sentences using a linear model and we call it precedent relation. To find a proper ordering for the group of sentences extracted from an article collection, our hybrid model regards sentence as vertex and precedent relation as edge of a directed graph, and employs PageRank analysis method to generate an approximately optimal ordering. We evaluate the automatically generated orderings against manual orderings on the testing dataset extended from DUC2004 and DUC2005. Experiment results show that the hybrid model has a significant improvement compared with other partially hybrid model. Moreover, experiment results on DUC2004 dataset and DUC2005 dataset indicates that our hybrid model is robust for articles with different genre. We also compare ROUGE score of reordered summaries with original ones. Results show ROUGE-1 score has no increase while ROUGE-L increases more higher on poor original summaries than on good ones.

For our future work, we plan to apply our hybrid model to Chinese summaries, since Chinese document is different with English in many aspects and sentence ordering on it has received little attention. If this model does not yield good orderings, we will investigate through corpus analysis Chinese summary structure and other relations between sentences. Furthermore, the readability of summary not only depends on the ordering of inter-sentence, but on the ordering of intra-sentence, our future research work will focus on intra-sentence ordering as well as inter-sentence. When combining the different relations with linear model, we decide the weights arbitrarily. To improve the performance, we will train these weights by machine learning method.

References

1. Mani, I.: Automatic Summarization. John Benjamins (2001)
2. Radev, Dragomir R., Hovy, Eduard H., McKeown, K.: Introduction to the Special Issue on Summarization. Computational Linguistics 28(4) (2002) 399-408
3. Okazaki, N., Matsuo, Y., Ishizuka M.: Coherent Arrangement of Sentences Extracted from Multiple Newspaper Articles. PRICAI (2004) 882-891
4. Barzilay, R., Elhadad, E., McKeown, K.: Inferring strategies for sentence ordering in multidocument summarization. Journal of Artificial Intelligence Research (JAIR), 17 (2002) 35–55
5. Lapata, M.: Probabilistic text structuring: experiments with sentence ordering. In Proceedings of the 41st Meeting of the Association of Computational Linguistics (2003) 545–552
6. Asher, N., Lascarides, A.: Logics of Conversation. Cambridge University Press (2003)
7. Grosz, B., Joshi, A., Weinstein, Scott.: Centering: A framework for modeling the local coherence of discourse. Computational Linguistics 21(2) (1995) 203–225
8. Cohen, W., Schapire, R., Singer, Y.: Learning to order things. Journal of Artificial Intelligence, 10 (1999) 243-270
9. Galil, Z., Megido, N.: Cyclic ordering is np-complete. Theoretical Computer Science, 5 (1977) 179-182
10. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 30 (1998) 1–7
11. Mihalcea, R., Tarau, P.: A Language Independent Algorithm for Single and Multiple Document Summarization. http://acl.ldc.upenn.edu/I/I05/ (2005)
12. Paul, O., James, Y.: An Introduction to DUC-2004. In Proceedings of the 4th Document Understanding Conference (DUC 2004). (2004)
13. Lin, C.Y., Hovy, E.: Automatic Evaluation of Summaries Using N-gram Co-Occurrence Statistics. In Proceedings of the Human Technology Conference (HLTNAACL-2003), Edmonton, Canada (2003)
14. Dang, H. T.: Overview of DUC 2005 (DRAFT). In Proceedings of the 5th Document Understanding Conference (DUC 2005). (2005)
15. Lebanon, G., Lafferty, J.: Combining rankings using conditional probability models on permutations. In Proceedings of the 19th International Conference on Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA. (2002) 363-370

Document Fragmentation for XML Streams Based on Query Statistics

Huan Huo, Guoren Wang, Xiaoyun Hui,
Chuan Xiao, and Rui Zhou

Institute of Computer System, Northeastern University, Shenyang, China
wanggr@mail.neu.edu.cn

Abstract. Recently XML fragments processing prevails over the Web due to its flexibility and manageability. In this paper, we propose two techniques for document fragmentation considering the query statistics over XML data: path frequency tree (PFT) and Markov tables. Both techniques work by merging the nodes of low inquiring frequency to enhance fragment utilization, or by merging the nodes of high inquiring frequency to enhance fragment cohesion. Performance study shows that our algorithms perform well on query cost and other metrics.

1 Introduction

XML [6] is emerging as a *de facto* standard for information representation and data exchange over the web [5]. However, as semi-structural data, XML processing poses an overwhelming overhead on runtime factors. Many research work, such as XFrag [1], Xstream [7], XStreamCast [2] and etc, support processing data fragments rather than documents.

In order to correlate each XML fragments, *Hole-Filler* model is proposed in [3]. In the model, a hole represents a placeholder into which another rooted subtree (a fragment), called a filler, could be positioned to complete the tree. In this way, queries on parts of XML data require less memory and processing time. Furthermore, changes to XML data may pose less overhead by sending only fragments corresponding to the changes, instead of sending the entire document.

Unfortunately, existing methods for XML fragmentation are limited to simply slicing XML data into fragments in terms of the contents and the structure of the document itself, without taking query information into account. In [4], two document fragmentation polices and the algorithms are proposed, which fragment an XML document according to the element's fan-out (i.e. the number of the element's children). In Xstream [7], it partitions XML documents into autonomous units as *XDU* (*Xstream Data Unit*) in a breadth-first order, which includes both the child nodes and their parent nodes.

This paper presents two techniques for fragmenting XML data based on the statistics of path expressions. The first technique is to construct a *path frequency tree* representing the query frequency of each nodes, denoted as PFT. We then determine the fragment policy by merging low-frequency sibling nodes to enhance the fragment utilization and by merging the high-frequency parent-child

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 350–356, 2006.

nodes to retain the structural relationship between them. The second technique is to store all paths in the XML data up to a certain length and their inquiry frequency in a table of paths that we call the Markov table. We group the low-frequency paths by suffix merging to determine the fragmentation policy.

The rest of this paper is organized as follows. Section 2 introduces *Hole-Filler* model. Section 3 proposes our document fragmentation policies. Section 4 shows experimental results. Our conclusions are contained in Section 5.

2 Model for Fragmented XML Data

In our approach, we adopt hole-filler model [3] to describe XML fragments. In order to summarize the structure of XML fragments, *tag structure* [3] is exploited to provide structural and fragmentation information. A tag structure is itself structurally a valid XML fragment that conforms to XML Schema or DTD, where a tag corresponds to an XML element and is qualified by a unique id, a name (the element tagname), and a type. Given an XML document tree $T_d = (V_d, E_d, \Sigma_d, root_d, Did)$, a filler $T_f = (V_f, E_f, \Sigma_f, root_f, fid, tsid)$ is a subtree of XML document associating an fid and a $tsid$, where V_f, E_f, Σ_f is the subset of node set V_d, edge set E_d and element type set Σ_d respectively, and $root_f$ $(\in V_f)$ is the root element of the subtree; a hole H is an empty node $v(\in V_d)$ assigned with a unique hid and a $tsid$, into which a filler with the same fid could be positioned to complete the tree. We can reconstruct the original XML document by substituting holes with the corresponding fillers at the destination as it was in the source. Figure 1 gives an example of XML fragments.

```
Fragment 1:
<commodities filler id="0" tsid="1">
  <vendor>
    <name>Wal-Mart</name>
      <items>
        <stream: hole id="10" />
        <stream: hole id="20"  />
        ...
  </vendor>
  ....
</commodities>
```

```
Fragment 2:
<stream: filler id="10" tsid="5">
  <item>
    <name>PDA</name>
    <make>HP</make>
    <model>PalmPilot</model>
    <price currency="USD">315.25</price>
  </item>
</stream: filler>
```

```
Tag Structure:
<stream: structure>
  <tag name="commodities" id="1" Filler="true">
    <tag name="vendor" id="2"  Filler="true">
      <tag name="name" id="3" />
      <tag name="items" id="4">
        <tag name="item" id="5"  Filler="true">
          <tag name="name" id="6" />
          <tag name="make" id="7" />
          <tag name="model" id="8" />
          <tag name="price" id="9" />
        </tag> ...   </tag>
 </stream: structure>
```

Fig. 1. XML Fragments based on Hole-Filler Model

3 Fragmenting XML with Query Statistics

3.1 Path Frequency Tree

Definition 1. *Let E_i and C_i represent steps and path connectors in absolute path expression $E_1C_1E_2C_2 \ldots C_{n-1}E_n$ for $i(1 \leq i \leq n)$, respectively. The Mark Nodes in path expression are defined as follows:*

- *Filter Node E_j, if E_j has predicates or is part of predicates;*
- *Key Ancestor E_k, if C_{k+1} is "//";*
- *Result Node E_n, the output of the query.*

Definition 2. *Let $P = \{p_1, p_2, \ldots, p_n\}$ be the set of path expressions on an XML document D, and $V_i = \{v_1, v_2, \ldots, v_k\}$ be the set of Mark Nodes. Frequency tree is a path tree $PFT = (V_d, E_d, \delta_d, \Sigma_d, ID, root_d, \Sigma_f)$, where V_d is the set of element type nodes for each node in D; E_d is a set of edges; δ_d is the mapping function from nodes to nodes indicating the containment relationship; Σ_d is the set of tag names; ID is a set of tsid identifying the tag nodes; Σ_f is the set of occurrence number of v_i in P, which is defined as the* frequency *of v_i.*

Figure 2 presents a path frequency tree (PFT), which contains the information for query statistics. For the node with higher frequency, it is inquired by a larger number of query clients, which means the majority of the clients will access the node. Once the fragment where the high-frequency node resides fails to be transmitted to the clients or needs to be updated, the server has to repeat or replace the fragment as soon as possible to cater the inquiry demands. Hence the utilization of such fragment is higher due to frequent transmission and massive evaluation. On the other hand, for the node with lower frequency, only small number of clients will access it while the majority will treat it as redundant information and filter it out. Our goal is to enhance the utilization of fragment by merging the low-frequency sibling nodes and enhance the query performance by merging the high-frequency parent-child nodes.

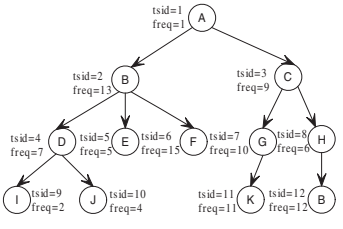


Fig. 2. A Path Frequency Tree

Sibling Merging. In the first phase for merging the nodes in PFT, which we call sibling merging, we repeatedly choose the node with the lowest frequency and mark it for deletion with "*". When we mark a node, A, for deletion, we check its siblings to see if they contain a node, B, that is either a merged node or a regular node that has been marked for deletion. If the frequency of such a node is below the threshold, nodes A and B are coalesced into one merged node.

Each merged node represents multiple sibling nodes deleted from the path tree. The parent of a merged node is the parent of the deleted nodes it represents, and the children of these deleted nodes become children of the merged node. When a node A, is coalesced with a merged node, the children of A become

children of the merged node. Coalescing the children of coalesced nodes is repeated recursively if needed. We use the average frequency of the nodes merged together as the frequency of a merged node and keep the number of nodes in the original unmerged tree that the merged node represents for estimation.

We coalesce the nodes in PFT until the minimum frequency of the nodes reaches the threshold excluding the nodes without siblings. The merged nodes in the PFT after coalescing represent the nodes should be grouped in a fragment. Figure 3 presents a merged PFT with threshold 8.

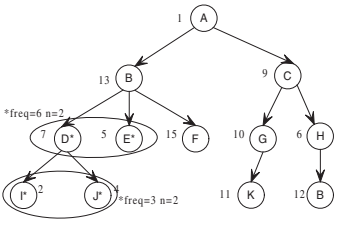


Fig. 3. The Sibling Merging on Path Frequency Tree

Parent-Child Merging. In the second phase for merging the nodes in PFT, which we call parent-child merging, we repeatedly choose the path frequency tree node with the highest frequency to merge. Figure 4 presents a parent-child merged path frequency tree after sibling merging. We set the threshold with 2. The nodes of this tree are marked for deletion in the order F, B(2), B(12), K, G, C. And the merged node KG can be further coalesced with node C.

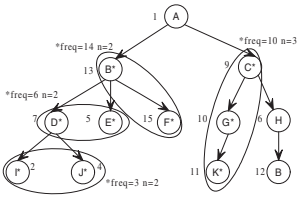


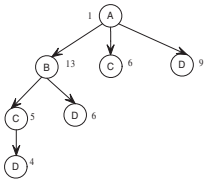
Fig. 4. The Parent-Child Merging on Path Frequency Tree

3.2 Markov Tables

We construct a table of all the distinct paths in XML data of length up to m and their inquiry frequency, where m is a parameter ≥ 2. The table provides frequency for all path expressions of length up to m. To estimate the frequency of longer path expressions, we combine several paths of length m using the formula

$$f(t_1/t_2/\ldots/t_n) = f(t_1/t_2/\ldots/t_m) \times \prod_{i=1}^{n-m} \frac{f(t_{1+i}/t_{2+i}/\ldots/t_{m+i})}{f(t_{1+i}/t_{2+i}/\ldots/t_{m+i-1})} \qquad (1)$$

where $f(t_1/t_2/\ldots/t_n)$ is the frequency of the path $t_1/t_2/\ldots/t_n$. $f(t_1/\ldots/t_k)$ for any k $\leq m$ is obtained by a lookup in the table of paths. Figure 5 presents a path tree and its corresponding Markov table for $m = 2$.



Path	Freq	Path	Freq
A	1	AC	6
B	13	AD	9
C	11	BC	5
D	19	BD	6
AB	13	CD	4

Fig. 5. A Path Tree and the corresponding Markov Table

Suffix Merging. The method for merging paths in Markov tables, which we call suffix merging, has a special *-paths, representing the merged paths of length 1, and a path, */*, representing the merged paths of length 2. When merging a low-frequency path of length 1, it is similar to sibling merging process and the merged node is added to the path *. When merging a low-frequency path of length 2, we do not add it to the path */* right away.

We keep a set of merged paths of length 2, S_M. When we merge a path of length 2, A/B, we look for any path in the set S_M that starts with the same tag as the path being merged, in this case A. If no such path is found, we remove the path being merged, A/B, from the Markov table and add it to S_M. If we find such a path in S_M, A/C, we remove A/B from the Markov table, remove A/C from S_M, and add a new path A/* that represents these two paths to the Markov table. And we get the merge list {A,B,C}.

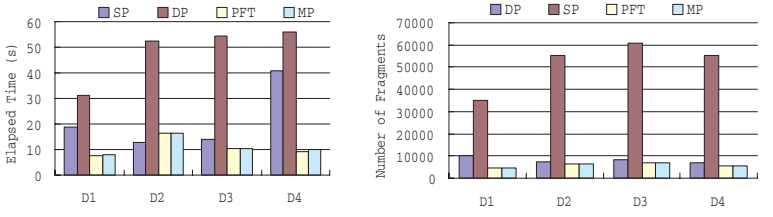
A/* represents all merged paths that start with the tag A. We call the path A/* a "suffix-*" path. When we merge a path of length 2, before we check the set S_M, we check the Markov table to see if there is a suffix-* path that has the same starting tag as the path being merged. If we find such a path, the path being merged is combined with it. In our example, if we merge A/D, we would combine it with A/*, and the merge list becomes {A,B,C,D}.

At the end of suffix merging, paths still remaining in S_M are added to the path */*, and the average frequencies of all *-paths are computed. The frequency for *-path is the average frequency of merged paths represented by this *-path.

4 Experimental Evaluation

We have implemented the following fragmentation algorithms: DOM-based algorithm(DP) [4], Schema-based algorithm(SP) [4], PFT-based algorithm(PFT) and Markov-based algorithm(MP) on the same platform. In the case that the fragmentation results for PFT and MP are the same, we denote these two algorithms as query-based algorithm (FP). All experiments are run on a PC computer with 2.6GHz CPU, 512M of memory and 80G hard disk. The operating system is WindowsXP. The programs are written with JBuilder 9.0.

Performance vs Different Document Types. We design four kinds of testing XML documents($D1, D2, D3, D4$). With different input parameters, we generate $10M$, $20M$ and $40M$ documents for each case. $D1$ represents deep tree document structure, the depth of the tree is $20 - 23$, the breadth of the tree is $8 - 19$; $D2$ represents wide tree structure, with the maximal depth 8 and the maximal breadth $20 - 22$; the element nodes in $D3$ have attributes, whereas in $D4$ no element nodes have attribute nodes.



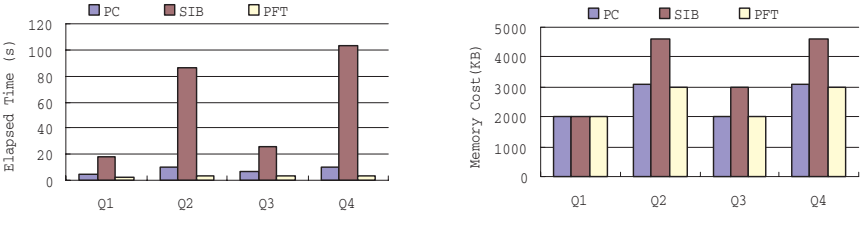
(a) The Elapsed Time of Fragmentation (b) The number of Fragments

Fig. 6. Fragmentation Performance with Different Document Types

Figure 6 shows the scenarios of the four algorithms conducted on different kinds of documents. In (a), FP and SP greatly outperform the DP algorithm, because they avoid the comparisons between the fan-out of element nodes and the threshold. In (b), FP generates a less fragments according to the query statistics using PFT or Markov table, which keeps fragment number in an average level for both wide and deep documents.

Performance vs Different Queries. We take PFT-based algorithm for example in the following and compare the algorithms on different scenarios: the sibling merging algorithm(SIB), the parent-child merging algorithm(PC) and the PFT-based algorithm(PFT), which combines the two algorithms together.

Figure 7 demonstrates the elapsed time and space cost on different fragmentation solutions. PC algorithm achieves better performance than SIB since the frequent inquired parent-child nodes are grouped in the same filler. PFT adopts both SIB and PC algorithms and achieves the best performance.



(a) The Elapsed Time of Q1~Q4 (b) The Memory Cost of Q1~Q4

Fig. 7. Query Performance with Different Fragmentation Solutions

5 Conclusions

This paper presents two techniques for fragmenting XML data based on the statistics of path expressions: *path frequency tree*(PFT) and *Markov tables*. The former determines fragmentation by merging the sibling nodes of low inquiring frequency to enhance fragment utilization and by merging the parent-child nodes of high inquiry frequency to enhance fragment cohesion. The latter achieves this by suffix merging. The performance evaluation shows these fragmentation algorithms perform well on execution time and space cost of queries.

Acknowledgement. This work is partially supported by National Natural Science Foundation of China under grant No. 60573089 and 60273074 and supported by Specialized Research Fund for the Doctoral Program of Higher Education under grant SRFDP 20040145016.

References

1. S. Bose and L. Fegaras. XFrag: A query processing framework for fragmented XML data. In *WebDB 2005*, Baltimore, Maryland, June 16–17, 2005.
2. S. Bose, L. Fegaras, D. Levine, and V. Chaluvadi. A query algebra for fragmented XML stream data. In *DBPL2003*, Potsdan, Germany, September 6–8, 2003.
3. L. Fegaras, D. Levine, S. Bose, and V. Chaluvadi. Query processing of streamed XML data. In *CIKM 2002*, McLean, Virginia, USA, November 4–9, 2002.
4. H. Huo, X. Hui, and G. Wang. Document fragmentation for XML streams based on hole-filler model. In *2005 China National Computer Conference*, page 250253, Wu Han, China, October 13–15, 2005.
5. S. Li, M. Liu, and Z. Peng. Wrapping HTML tables into XML. In *The Fifth International Conference on Web Information Systems Engineering*, pages 147–152, Brisbane, Australia, November 22–24, 2004.
6. W3C Recommendation. *Extensible Markup Language (XML) 1.0 (Second Edition)*, October 6 2000. http://www.w3.org/TR/REC-xml.
7. E. Wang and et al. Efficient management of XML contents over wireless environment by Xstream. In *ACM-SAC 2004*, pages 1122–1127, March, 2004.

A Heuristic Approach for Topical Information Extraction from News Pages

Yan Liu, Qiang Wang, and QingXian Wang

Information Engineering Institute, Information Engineering University
Zhengzhou, 450002, P.R. China
liu_yan_hello@yahoo.com.cn

Abstract. Topical information extraction from news pages could facilitate news searching and retrieval etc. A web page could be partitioned into multiple blocks. The importance of different blocks varies from each other. The estimation of the block importance could be defined as a classification problem. First, an adaptive vision-based page segmentation algorithm is used to partition a web page into semantic blocks. Then spatial features and content features are used to represent each block. Shannon's information entropy is adopted to represent each feature's ability for differentiating. A weighted Naïve Bayes classifier is used to estimate whether the block is important or not. Finally, a variation of TF-IDF is utilized to represent weight of each keyword. As a result, the similar blocks are united as topical region. The approach is tested with several important English and Chinese news sites. Both recall and precision rates are greater than 96%.

Keywords: Information Retrieval, Entropy, Naive Bayes classifier.

1 Introduction

The innovation of the Web creates numerous information sources published as HTML pages on the Internet. Obviously, different information inside a web page has different importance weight according to its location, occupied area, content, etc. Hence, it is of great advantage to have a technique which could automatically analyze the web page and find the topical information to facilitate news reading and other application.

To distinguish different information in a web page, we first need to segment a web page into a set of blocks. The number of news content blocks in a news page may be more than one. For instance, one block maintains the principal part of news and another block lists the related story. Sometimes the segmentation algorithm only achieves a news block adulterated with some advertisements. In order to eliminate the unusual parts, the partition requires smaller granularity. However, topical information tends to be separated. Thus, it is necessary to integrate the related content blocks and achieve the whole topical information.

The rest of the paper is organized as follows. In Section 2, related work is described. In section 3, the application of our work is introduced. A weighted

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 357–362, 2006.

Naive Bayes classifier is used for judging block importance and a variation of TF-IDF is utilized to unite the similar content blocks to topical information of the page. Then experimental evaluation is presented in Section 4. Finally, we conclude our work.

2 Related Works

Related work could be categorized into two classes. One class of techniques aims to detect the patterns among a number of web pages from the same web site. The common idea of these approaches is that in a given web site, noisy blocks usually share some common contents and presentation styles[1][2]. However, if the amount of web pages is huge, the computation for similarity will cost expensively. The other class of techniques tries to detect important regions in a single web page [3][4][5]. However, they did not consider the variation of features' influence among different pages.

There are some differences between the study of block importance and topical information extraction. The former achieves importance score of all blocks in a page, while the latter only focuses on some most important blocks.

3 Discover Topical Informative Contents

Our system, *NewsDigger*, is a system that personalizes news for users by retrieving, ranking, indexing, classifying, clustering and delivering personalized news information extracted from news sites. In this system, the function of *TopicalExtractor* is to achieve the topical information content in each news page. Fig.1 shows the process of *TopicalExtractor*. We aim at the adaptive adjusting

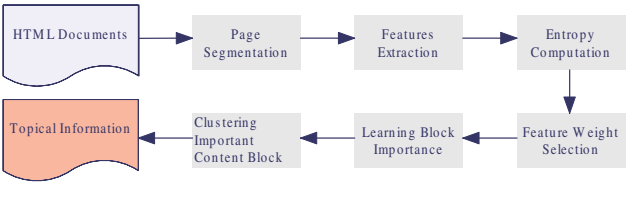


Fig. 1. Process of *TopicalExtractor*

of features' weight in the process of classification and improving the efficiency in integrating process.

Firstly, a Web page is partitioned into blocks and the block features are extracted to represent the blocks. Then each feature is studied to show how much it influences the block importance judgment in a given page. The weighted Naive Bayes classifier is then used to estimate whether the block is important or not. Finally, a variation of TF-IDF is utilized to represent weight of each keyword in the given page. Thus, the similar blocks could be united as topical region.

3.1 Page Segmentation

Several methods have been explored to segment a Web page into regions or blocks. Compared with the DOM-based segmentation approach , vision-based page segmentation (VIPS) [6] excels in both appropriate partition granularity and coherent semantic aggregation. VIPS makes full use of page layout features such as font, color and size. It firstly extracts all the suitable nodes from the HTML DOM tree, and then finds the separators between these nodes. Here, separators denote the horizontal or vertical lines in a Web page that visually do not cross any node. Based on these separators, the semantic tree of the Web page is constructed. Consequently, VIPS can efficiently keep related content together while separating semantically different blocks from each other.

3.2 Block Features

Typically, topical information in a news page usually contains lots of texts with little links. Therefore, the contents in a block are useful to differentiate block importance. On the other hand, Web designers would put most important information in the center and put the navigation bar on the header or the left side and the copyright on the footer .Thus, the importance of a block could also be reflected by spatial features like position, size, etc.

Content Features. The content features in a block could reflect the importance of a block. Examples of features are the number of links, the length of anchor text, the length of non-anchor text, the number of images etc. So the following features are used to represent the content of a block:

{*LinkNum, LinkTextLength, TextLength, ImgNum, FormNum, InteractionNum*}

All of these features are related to the importance. Of course the degrees of relation are varied from each other. It depends on the layout and content of the page.

Spatial Features. The result of VIPS helps further describe the spatial features which are made up of five items:

{*Relative BlockCenterX, Relative BlockCenterY, Relative BlockWidth, Relative BlockHeight, Width-Height Ratio*}

BlockCenterX and *BlockCenterY* are the coordinates of the block's center point and BlockWidth, *BlockHeight* are the width and height of the block. However, considering the difference of width and height among various pages, it is hard to compare the spatial features from different Web pages reasonably. So it is necessary to normalize the absolute spatial feature to relative spatial feature.

3.3 Entropy of Features

As discussed before, content features give many clues in learning block importance. Each feature corresponds to a certain aspect of the measurement. The motivation of this approach is, the more uniformly a feature distributes the less

information it carries to users. In contrast, those appears evidently in fewer blocks in a page carry more information of interest. Hence, entropy, which is determined by the probability distribution of a feature around the whole page, can represent its information strength (rich or poor). Shannon's information entropy [7] is applied on the feature-block matrix which is generated from statistical step to calculate the frequency of each feature. By definition, the entropy E can be expressed as $-\sum_{i=1}^{n} p_i \log p_i$, where p_i is the probability of $event_i$ and n is the number of events. In order to normalizing the weight of a term to be [0,1] and express the feature normally, the entropy of feature F_i is:

$$E(F_i) = -\sum_{j=1}^{n} w_{ij} \log_n w_{ij}. \tag{1}$$

In which n=|B| , B is the set of blocks in a page, w_{ij} is a entity in the feature-block matrix to represent the frequency of a feature, i.e., $w_{ij} = {tf_{ij}}/{\sum_{k=1}^{n} tf_{ik}}$, where tf_{ij} is the frequency of the $feature_i$ in a $block_j$.

3.4 Redundant Block Eliminating

Practically, people have consistent opinions about the importance of the same block in a page [4]. The redundant block eliminating could be viewed as a classification problem. Specially, some blocks are pre-labeled and thus each labeled block can be represented as (x, y) where x is the feature representation of the block and y is a BOOL value. TRUE represent the block is valuable and otherwise is FALSE.

Naïve Bayes. Naïve Bayes is often used in text classification applications and experiments for its simplicity and effectiveness. Redundant block eliminating could adopt the method. There are only two classes (V_+:valuable/V_-:redundant). The score of an input block B is calculated as follow:

$$score(B) = \log P(V_+) + \sum_{k \in |F|} \log P(f_k|V_+) - (\log P(V_-) + \sum_{k \in |F|} \log P(f_k|V_-)). \tag{2}$$

where f_k represent the k-th feature of block and |F| is the number of features. If $score(B) > 0$, the block will be assigned to V_+, and V_- otherwise.

Weighted Naïve Bayes Classifier. Naïve Bayes classifiers rely on Bayes' decision rule with the assumption that the probability of each attribute value is independent from the values of other attributes. This assumption is not valid in general. Actually, the appearance of features has some coherence based on roles of blocks in a given page. In this section, an approach is proposed to modify the Naïve Bayes by simply introducing weight which can help to improve the accuracy of a Naïve Bayes block classifier dramatically.

The basement of our scheme is that some features may have stronger ability for distinguish than other features. The natural extend to the Naïve Bayes is to introduce weight to every content feature, then the *formula(2)* can rewrite as:

$$score(B) = \log P(V_+) + \sum_{k \in |F|} (\log P(f_k|V_+) + g(E_k)) \\ -(\log P(V_-) + \sum_{k \in |F|} (\log P(f_k|V_-) + (1 - g(E_k))) \quad . \tag{3}$$

where $g(E_k)$ is the weight appended to each feature. The selection of weight is dynamic, which depends upon the entropy achieved in section 3.3. The function $g(E_k)$ is inversely proportional to the entropy of f_k and in direct proportion to the support degree of f_k, which means the more outstanding f_k appears in a block, the more important the block is.

3.5 News Content Blocks Uniting

A content block can be viewed as a type of document and it can also be expressed by a group of keywords. Thus, block could be represented as a vector of keywords and the similarity could be calculated for uniting all the similar blocks as topical region. Given two blocks, the similarity measure, *sim*, returns the cosine between their block information vectors. We make use of a variation of TF*IDF[5] to represent the information of block:

$$w_i = \frac{\sum_{j=1}^{BN} BWeight_j * BTf_{ij}}{\sqrt{\sum_{i=1}^{n} (\sum_{j=1}^{BN} BWeight_j * BTf_{ij})^2}}. \tag{4}$$

where BN is the amount of content blocks; n is the amount of keywords; $BWeight_j$ is the score of block j achieved by section 3.4; BTf_{ij} is the frequency of keyword i appeared in block j.

4 Experiment

In our experiments, the dataset contains eight English and two Chinese news Web sites. News in these Web sites covers several domains, including politics, finance, sports, life, international issues, entertainment, health, cultures, etc.

We use our weighted Naïve Bayes classifier to implements redundant blocks eliminating. The result shows that our classifier can reasonably recognize the ability of each feature for distinguishing importance of blocks and adaptively adjust the weights in the computation.

In the process of *TopicalExtractor*, it selects the topical block from every Web page and compares the result with predefined block. Ratio of Noise is achieved in the computing progress, means how many noise texts are filtered in this topical extraction work. Our approach is able to extract correctly an average of 96.5% of the news, while 3.5% are erroneously extracted and 3.01% are not extracted. These experiments prove that the entropy-based bayes classifier achieves high recall and precision.

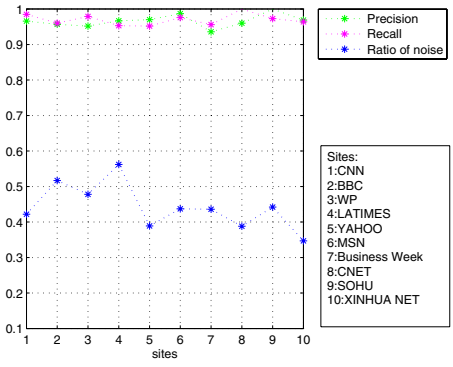


Fig. 2. Block level Precision, Recall values and Ratio of Noise

5 Conclusion

We devised a simple but powerful approach to identify primary portions within Web pages. The VIPS algorithm is adopted to partition a Web page into multiple semantic blocks. Shannon's information entropy is adopted to represent each feature's ability for differentiating. And the weighted Naïve Bayes classifier is presented to eliminate the redundant blocks.

The approach is tested with several important English and Chinese news sites and achieved precise results. Evidently, it can be applied to general Web IR systems to reduce the size of index and increase the precision of retrieval.

References

1. Lin, S.-H. and Ho, J.-M.: Discovering Informative Content Blocks from Web Documents, in the proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (SIGKDD'02).(2002)
2. Sandip Debnath, Prasenjit Mitra, C. Lee Giles.: Automatic Extraction of Informative Blocks from Webpages.SAC'05 Santa Fe, New Mexico, USA. March 1317 (2005)
3. Gupta,S., Kaiser,G., Neistadt,D. and Grimm,P.: DOM based Content Extraction of HTML Documents, in the proceedings of the 12th World Wide Web conference (WWW 2003), Budapest, Hungary. May (2003).
4. Ruihua Song, Haifeng Liu, Ji-Rong Wen, Wei-Ying Ma.: Learning Block Importance Models for Web Pages. WWW 2004, New York, USA. May 17-22 (2004)
5. Zhang Zhigang,Chen Jing,Li Xiaoming.: An Approach to Reduce Noise in HTML Pages. JOURNAL OF THE CHINA SOCIETY FOR SCIENTIFIC AND TECHNICAL INFORMATION . April 23, (2004)
6. Cai, D., Yu, S., Wen, J.-R. and Ma, W.-Y., VIPS: a vision-based page segmentation algorithm, Microsoft Technical Report. MSR-TR-2003-79, (2003)
7. C. E. Shannon. A mathematical theory of communication.Bell System Technical Journal, 27:398-403.(1948)

Defining a Data Quality Model for Web Portals

Angélica Caro[1], Coral Calero[2], Ismael Caballero[2], and Mario Piattini[2]

[1] Department of Auditoria e Informática, University of Bio Bio
La Castilla s/n, Chillán, Chile
ancaro@inf-cr.uclm.es
[2] Alarcos Research Group. Information Systems and Technologies Department
UCLM-SOLUZIONA Research and Development Institute.
University of Castilla-La Mancha
{Coral.Calero, Ismael.Caballero, Mario.Piattini}@uclm.es

Abstract. Advances in technology and the use of the Internet have favoured the appearance of a great variety of Web applications, among them Web Portals. These applications are important information sources and/or means of accessing information. Many people need to obtain information by means of these applications and they need to ensure that this information is suitable for the use they want to give it. In other words, they need to assess the quality of the data.

In recent years, several research projects were conducted on topic of Web Data Quality. However, there is still a lack of specific proposals for the data quality of portals. In this paper we introduce a model for the data quality in Web portals (PDQM). PDQM has been built upon the foundation of three key aspects: a set of Web data quality attributes identified in the literature in this area, data quality expectations of data consumers on the Internet, and the functionalities that a Web portal may offer to its users.

Keywords: Data Quality, Information Quality, Web Portal, Quality Model.

1 Introduction

Over the past decade the number of organizations which owns Web portals grows dramatically. Their have established portals to complement, substitute or widen existing services to their clients. In general, portals provide users with access to different data sources (providers) [19], as well as to on-line information and information-related services [31]. Also they create a working environment that users can easily navigate in order to find the information they specifically need to quickly perform their operational or strategic functions and make decisions [7]. It will be up to the owners of a Web portal to obtain and maintain a high level in the quality of information [17].

In the literature, the concept of Information or Data Quality is often defined as "fitness for use", i.e., the ability of a data collection to meet user requirements [5],[27]. Besides, the terms "data" and "information" are often used as synonyms [28]. In this work we will use them as synonymous.

Research on data quality (DQ) began in the context of information systems [18],[27] and it has been extended to contexts such as cooperative systems [9],[20],[30], data warehouses [2],[32] or e-commerce [1],[13], amongst others. Due

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 363 – 374, 2006.

to the particular characteristics of Web applications and their differences from the traditional information systems [31], the research community started to deal with the subject of DQ on the Web [11].

However there are no works on DQ that address the particular context of Web portals [6], in spite of the fact that some works highlight the DQ as one of the relevant factors in the quality of a portal [22],[25]. Likewise, except for few works in the DQ area, like [4],[5],[29], most of them have looked at quality from the data producers or data custodians perspective and not from the data consumers perspective [4]. The last perspective differs from the two others in two important aspects [4]: (1) data consumer has no control over the quality of available data and (2) the aim of consumers is to find data that match their personal needs, rather than provide data that meet the needs of others.

In this paper, we present a portal data quality model (PDQM), focused on the data consumer's perspective. As key pieces in the construction of our model we have taken (1) a set of Web DQ attributes identified in the literature, (2) the DQ expectations of data consumers on the Internet, described by Redman in [26], and (3) the functionalities that a portal Web may offer its users [7].

To produce the PDQM model, we defined the process shown in Fig.1. During the first phase, we have recompiled Web DQ attributes from the literature, which we believe should therefore be applicable to Web portals. In the second phase we have built a matrix for the classification of the attributes obtained in previous phase. This matrix reflects two basic aspects considered in our model: the data consumer perspective and the basic functionalities which a data consumer uses to interact with a Web portal.

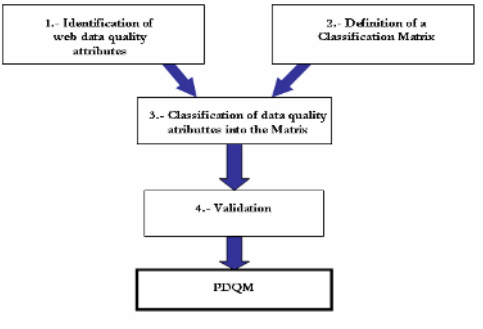


Fig. 1. Phases in the development of the PDQM

In the third phase we used the obtained matrix, to analyse the applicability of each Web DQ attribute in a Web portal. Finally, in the fourth phase, we must validate PDQM. The remainder of this paper is organized as follows. In section 2, we describe the phase of identification of attributes of Web DQ and show a summary of the results obtained (phase 1 of Fig.1). Section 3 is dedicated to the description of the phase of classification matrix construction (phase 2 of Fig.1). We set out the result of classification of attributes on the matrix in section 4 (phase 3 of Fig.1). Section 5 shows how we aim to validate our model (phase 4 of Fig.1). In section 6 we present the conclusions and the future work.

2 Identification of Attributes of Web Data Quality

The first stage in the development of our model consisted of a systematic review of the relevant literature [15]. From this task we aimed to identify data quality attributes which have been proposed for different domains in the Web context (Web sites [8], 14],[23], data integration [3],[24], e-commerce [13], Web information portals [25], cooperative e-services [9], decision making [11], organizational networks [21] and DQ on the Web [10]). The idea was to take advantage of work already carried out in the Web context and apply it to Web portals.

In this review we studied 55 works, published between 1995 and 2004. From the studied work we selected the projects in which DQ attributes applicable to the Web context were proposed. We thus obtained a total of 100 Web DQ attributes. We wanted to reduce this number, having also detected certain synonymous amongst the attributes identified. Those attributes were combined and also the ones which had similar name and meaning, obtaining a final set of 41 attributes. Table 1 shows these attributes, pointing out for each of these the work where they were put forward, as well as the total number of pieces of work where they can be found referred to. In addition, the symbols × and ⊗ were used to represent how they were combined (× indicates the same name and meaning and ⊗ marks the fact that only the meaning is the same).

Table 1. Web Data Quality Attributes 1-41

Author	Year	Accessibility	Accuracy	Amount of data	Applicability	Attractiveness	Availability	Believability	Completeness	Concise Representation	Consistent Representation	Cost Effectiveness	Customer Support	Currency	Documentation	Duplicates	Ease of operation	Expiration	Flexibility	Granularity	Interactive	Internal Consistency	Interpretability	Latency	Maintainable	Novelty	Objectivity	Ontology	Organization	Price	Relevancy	Reliability	Reputation	Response time	Security	Specialization	Source's Information	Timeliness	Traceability	Understandability	Validity	Value-added	Number of Attributes
Naumann and Rolker	2000		×	×			×	×	×	×	×		×		×								×	×			×			×	×	×	×	×			×	⊗	×		×	22
Katerattanakul and Siau	1999, 2001	×	⊗			×																						×							×			⊗			6
Eppler	2001	×	×		×				×	⊗	⊗			×			⊗				⊗			×								⊗	×		×	×	⊗				16
Fugini et al	2002		⊗		×			⊗	×													×								⊗			⊗		×					8
Pernici and Scannapieco	2002		×						×									×													⊗										4
Graefe	2003	⊗					⊗	×															×			×				⊗							⊗	⊗		8
Bouzeghoub and Peralta	2004													×																						×					2
Gertz	2004								×					×		×				×								×													5
Melkas	2004	×	×	⊗					×	×	×	×					×		×				×				×				×		×	⊗	×		×	⊗		×	20
Moustakis	2004				⊗			⊗																						⊗				×						4
Yang et al.	2004		×					⊗	×					×																⊗										5
Number of references		4	7	2	3	1	1	6	7	3	3	1	1	4	1	1	2	1	1	1	1	2	3	1	1	1	2	1	1	6	2	2	3	4	1	1	5	3	4	1	3	

3 Matrix Classification

In the second step we defined a matrix which would allow us to perform a preliminary analysis of how applicable these attributes are to the domain of Web portals. The matrix was defined based on the relationship that exists between: (1) The functionalities of a Web portal, identified in [7]: Data Points and Integration, Taxonomy, Search Capabilities, Help Features, Content Management, Processes and Actions, Communication and Collaboration, Personalization, Presentation, Administration and Security; and (2) The data quality expectations of Internet consumers as stated in [26]: Privacy, Content, Quality of Values, Presentation, Improvement and Commitment

On this matrix we carried out an analysis of what expectations were applicable to each of the different functionalities that a portal offers to a data consumer represented in Fig.2 with a "√" mark.

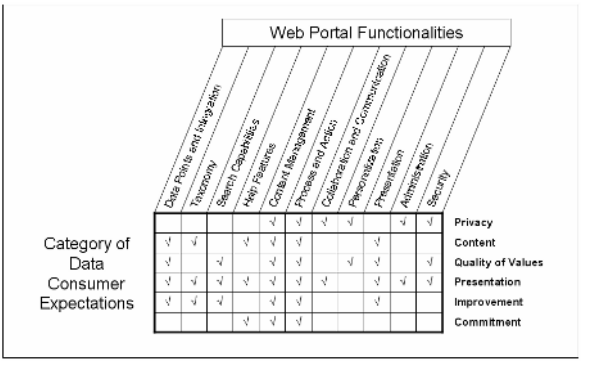


Fig. 2. Matrix for the classification of attributes of Web data quality

In the following paragraphs we explain each relationship (functionality, expectation):

- *Data Points and Integration*. They provide the ability to access information from a wide range of internal and external information sources and display the resulting information at the single point-of-access desktop. The expectations applied to this functionality are: *Content* (consumers need a description of portal areas covered, use of published data, etc.), *Quality of value* (data consumer should expect the result of searches to be correct, up-to-date and complete), *Presentation* (formats, language, and other aspects are very important for easy interpretation) and *Improvement* (users want to participate with their opinions in the portal improvements and to know what the results of applying them are).

- *Taxonomy.* It provides information context (including the organization-specific categories that reflect and support the organization's business). The expectations applied to this functionality are: *Content* (consumers need a description of which data are published and how they should be used, as well as easy-to-understand definitions of every important term, etc.), *Presentation* (formats and language in the taxonomy are very important for easy interpretation; users should expect to find instructions when reading the data), and *Improvement* (the user should expect to convey his/her comments on data in the taxonomy and know what the result of improvements are).

- *Search Capabilities*. This provides several services for Web portal users and needs to support searches across the company, the World Wide Web, and in search engine catalogs and indexes. The expectations applied to this functionality are: *Quality of values* (the data consumer should expect the result of searches to be correct, up-to-date and complete), *Presentation* (formats and language are important for consumers, both for searching and for easy interpretation of results) and *Improvement* (the consumer should expect to convey his/her comments on data in the taxonomy and be aware of the result of improvements).

- *Help Features*. These provide help when using the Web portal. The expectations applied to this functionality are: *Presentation* (formats, language, and other aspects are very important for easy interpretation of help texts) and *Commitment* (the consumer should be able to ask and obtain answers to any question regarding the proper use or meaning of data, update schedules, etc, easily.).

- *Content Management*. This function supports content creation, authorization, and inclusion in (or exclusion from) Web portal collections. The expectations applied to this functionality are: *Privacy* (a privacy policy for all consumers to manage, to access sources and to guarantee Web portals data should exist), *Content* (consumers need a description of data collections), *Quality of values* (a consumer should expect all data values to be correct, up-to-date and complete), *Presentation* (formats and language should be appropriate for easy interpretation), *Improvement* (the consumer should expect to convey his/her comments on contents and their management and be aware of the result of the improvements) and *Commitment* (the consumer should be able to ask any question regarding the proper use or meaning of data, etc.,).

- *Process and Action*. This function enables the Web portal user to initiate and participate in a business process of a portal owner. The expectations for this functionality are: *Privacy* (the data consumer should expect there to be a privacy policy to manage the data about the business on the portal), *Content* (consumers should expect to find descriptions about the data published for the processes and actions, their uses, etc.), *Quality of values* (that all data associated to this function are correct, up-to-date and complete), *Presentation* (formats and other aspects are very important in order to interpret data), *Improvement* (the consumer should expect to convey their comments on contents and their management and to know the improvements) and *Commitment* (the consumer should be able to ask and obtain an answer to any question).

- *Collaboration and Communication*. This function facilitates discussion, locating innovative ideas, and recognizing resourceful solutions. The expectations for this functionality are: *Privacy* (the consumer should expect a privacy policy for all consumers that participate in activities of this function), and *Commitment* (a consumer should be able to ask and have any question answered regarding the proper use or meaning of data for the collaboration and/or communication, etc.).

- *Personalization*. This is a critical component in creating a working environment that is organized and configured specifically for each user. The expectations applied to this functionality are: *Privacy* (the consumer should expect privacy and security as regards their personalized data, profile, etc.), and *Quality of values* (data about the user profile should be correct and up-to-date).

- *Presentation*. It provides both the knowledge desktop and the visual experience to the Web portal user that encapsulates all of the portal's functionality. The expectations for this functionality are: *Content* (the presentation of a Web portal should include data about areas covered, appropriate and inappropriate uses, definitions, etc.), *Quality of values* (the data of this function should be correct, up-to-date and complete.), *Presentation* (formats, language, and other aspects are very important for the easy interpretation and appropriate use of data.) and *Improvement* (the consumer should expect to convey their comments on contents and their management).

- *Administration*. This function provides a service for deploying maintenance activities or tasks associated with the Web portal system. The expectations for this functionality are: *Privacy* (Data consumers need security for data about the portal administration) and *Quality of values* (Data about tasks or activities of administration should be correct and complete).

- *Security.* This provides a description of the levels of access that each user (groups of users) are allowed for each portal application and software function included in the Web portal. The expectations for this functionality are: *Privacy* (the consumer needs a privacy policy regarding the data of the levels of access.), *Quality of values* (data about the levels of access should be correct and up-to-date) and *Presentation* (data about security should be in a format and language for easy interpretation).

4 Classification

The third phase of the development process in the PDQM model (see Figure 1), consisted in classifying the Web data quality attributes (shown in section 2) in each of the relationships (functionality, expectation) established on the classification matrix created in stage 2 (and presented in section 3). In this paper we do not show the attributes applicable to each relationship (functionality, expectation), we just set out a summary of the attributes applicable to each portal functionality (Table 2).

Table 2. Data quality attributes for functionality

Functionalities	Accessibility	Accuracy	Amount of data	Applicability	Attractiveness	Availability	Believability	Completeness	Concise Representation	Consistent Representation	Cost effectiveness	Customer support	Currency	Documentation	Duplicates	Ease of operation	Expiration	Flexibility	Granularity	Interactive	Internal consistency	Interpretability	Latency	Maintainable	Novelty	Objectivity	Ontology	Organization	Price	Relevancy	Reliability	Reputation	Response time	Security	Specialization	Source's information	Timeliness	Traceability	Understandability	Validity	Value-added	Total of Attributes
Data Points and Integration	✓	✓	✓				✓		✓	✓		✓	✓			✓	✓								✓				✓	✓							✓	✓	✓		15
Taxonomy	✓		✓				✓		✓			✓				✓													✓	✓							✓	✓	✓		11
Search Capabilities	✓		✓				✓	✓	✓			✓	✓			✓									✓				✓	✓							✓	✓			13
Help Features	✓		✓						✓			✓				✓													✓	✓							✓	✓			8
Content Management	✓		✓	✓		✓	✓	✓	✓			✓	✓	✓	✓	✓									✓	✓		✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	24
Process and Action	✓	✓	✓	✓		✓	✓	✓	✓			✓				✓	✓					✓				✓			✓	✓	✓		✓	✓	✓		✓	✓	✓		21
Collaboration and Communication						✓						✓										✓							✓				✓				✓				6
Personalization		✓						✓					✓				✓																			✓	✓	✓		7
Presentation			✓		✓		✓		✓			✓	✓			✓	✓	✓				✓				✓			✓	✓					✓		✓	✓	✓		15
Administration			✓						✓	✓						✓																✓				✓				6
Security	✓	✓	✓					✓	✓							✓						✓											✓				✓	✓		10
Number of References	7	4	9	2	1	3	6	5	9	1	0	8	5	1	1	8	4	1	0	0	0	5	0	0	3	2	0	1	0	7	7	2	0	5	3	1	0	7	11	8	1	

As can be seen in Table 2 there are some quality attributes which were not classified in the matrix. This is basically due to the fact that these are not able to be assessed by the data consumers, for example- Ontology and Latency.

5 Validation

Until this point, in the production of the PDQM model we have established relationships between the functionalities of a web portal and the expectations of the

Internet users. On the basis of these relationships we have identified, intuitively, Web DQ attributes which could be applicable in a portal obtaining the first version of the PDQM model. The next phase consists in the validation of our model (see Fig.1).

In this process we decided to carry out a study by means of a survey. In the survey, consumers of web portal data would be asked to give their opinion about what aspects they think are important when assessing the quality of the data they get from a portal.

We decided to draw up independent questionnaires for each one of the functionalities because we thought that to use only a questionnaire for the whole model would be tiring for the subjects. In this paper we will describe the validation tasks carried out on a single portal functionality.

The first functionality we chose was Data Points and Integration. This decision was made on the basis that this functionality is the first one that the consumer comes across on entering a Web portal and also because this aspect will help the portal user to assess whether the data which the portal offers match their expectations or not.

5.1 Definition of the Objectives of the Survey

The first activity carried out corresponds to the definition of the objectives of the survey, since these are directly related to the information that will be gathered [16]. For the chosen functionality these are:

1. To obtain the opinion of the consumers of a Web portal with respect to how they characterizes the quality of the data obtained in a portal through the functionality of Data Points and Integration.
2. To ascertain the importance given by the data consumers to each one of the DQ attributes identified in each relationship (Data Points and Integration, expectation).
3. To identify other aspects, in particular, Data Points and Integration which are important for consumers but which have not been considered in our model.

5.2 Choosing of the Subjects

With the above objectives in mind, the target population of our survey was defined as the whole set of Web portal data consumers. As to work with the whole population was not feasible, we used just a representative sample of it. That is, people who usually use portals to get data. To obtain the sample we used the snowball sampling method.

5.3 Preparation of the Questionnaire

In this section, we explain the stages used for preparing the questionnaire (the instruments is in http://FreeOnlineSurveys.com/rendersurvey.asp?id=140254).

1. **Search in Literature.** In the pertinent literature we looked for studies in which surveys had taken place in the validation of data quality attributes. Amongst those found we could mention [8],[23],[29].
2. **Construction of the instrument (or use of an existing one).** As there are advantages in using an already-proven instrument [16], we have built a new questionnaire based on the one that has been created in [29]. The questionnaire was constructed with an automatic tool and was posted on the URL given by the tool's provider.

3. **Choice of questions.** Basically, to produce the questions, we have considered the purpose and goals of the survey [16]. We therefore chose 2 questions for the first objective, one (a compound one) for the second and another for the third. All this was done taking into account factors such as the level of understanding of those being interviewed, an appropriate number of questions and the standardization of reply formats. A set of demographic questions were included too. Apart from all this, we included a question where we asked the subjects their opinion about the survey itself. With this we pretended to pick up on any defect or problem as far as the questionnaire is concerned.
4. **Making the questions.** We tried to create questions which have evident sense and which are specific [16]. The language used is conventional, expressing simple ideas. Negative questions are not included.
5. **Type of questions.** Our instrument mixes open and closed questions. In the first question, an open question, we asked the respondents about the attributes they considered important for a data point (the functionality under study). In the second question, open too, we asked about the attributes which were important for the data obtained through a data point. In the third question, another open question, we showed the 15 attributes classified for the functionality and we asked the respondents about other attributes that they consider need to be included. In the fourth question, a closed question, the subjects had to give a value (among 1 and 7) to the importance given for each one of the 15 attributes classified for the functionality. After this, we asked, with 6 closed questions, demographic data. Finally, in the last question, an open question, we asked the subject's opinion about the survey.
6. **Format of the questionnaire.** Our questionnaire was self-administered (it will be put on the Web and the user would be able to access to it when and where he/she decides). In order to check the format and instructions of the questionnaire we have used a check list proposed, to this purpose, in [16].
7. **Evaluation of the survey instrument.** In spite of the fact that we use a previously -proven model for our survey, we have carried out a pilot study whose aim was to ensure that the survey was a reliable instrument. For the pilot study we asked a group of data consumers, via e-mail, to take part. Twenty participants were notified in this way, of whom 15 replied. We thus reached a response rate of 75%. Among those who replied, and after analysing the observations given, a new version of the questionnaire was produced. The changes consisted, basically, in adjusting the presentation format, so that in each one, key aspects would be clearly highlighted.

5.4 Application and Results of the survey

The sample on which we applied the survey was obtained by means of personal contacts. The questionnaire was posted on the Web and each person included in the sample (150 approximately) was sent an e-mail. The e-mail contained an invitation to take part in the survey (at this point we explained its purpose, as well as the importance of the cooperation of each person). We also asked them to send the e-mail to their contacts in order to obtain more answers to our survey. The Web address of the questionnaire was provided at the same time.

A total of 91 responses were obtained within two weeks. After data screening, we eliminated 32 incomplete and repeated questionnaires. As a result, the total effective sample was 69 subjects (46% of the subjects contacted).

With the demographic question we obtained the respondents' profile. 43% of the respondents were female and 57% were male; 58% were between 36 and 45 years old; 97% used the Web more than once a day; 94% used Web portals; and 97% had more than 3 years of experience using the Web.

For the Data Points and Integration functionality, in our classification we considered 15 data quality attributes (see Table 2). The results of the first open question showed that the most mentioned attributes were: Accessibility, Understandability, Currency and Consistent representation (all considered in our model for the functionality under study), Organization, Source's Information and Response time (all considered in our model but not for this functionality). All these attributes were mentioned by more than 10% of the participants. The results of the second open question showed the following attributes as the most mentioned: Accuracy, Relevance (considered in our model for this functionality), Organization, Source's Information and Response time (considered in our model but not for this functionality). All these attributes were mentioned by more than 10% of the participants.

In the third open question we showed all the attributes considered in our model and we asked the data consumer for other attributes that they consider necessary. As a result, the most-proposed attribute was Attractiveness with 22%, Security with 12% and Source's Information, Response time and Ease of Operation with 10%. All were considered in our model but not classified within this functionality.

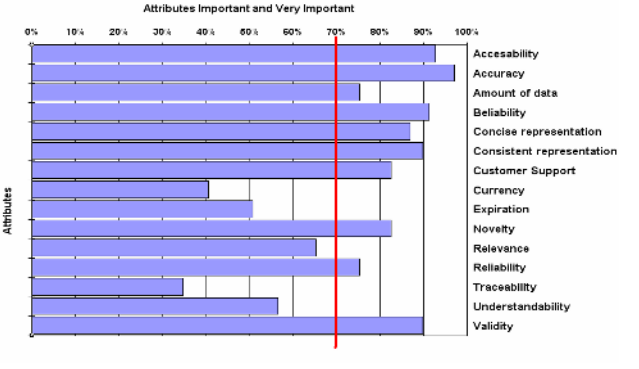


Fig. 3. Importance assigned by respondents to the data quality attributes proposed

Lastly, in the fourth question (the closed one), the participants had to assign a degree of importance between levels 1 and 7 (1 not important and 7 very important) to each attribute. The attributes that had at least 70% of preferences (adding the percentages of level 6 and 7) will be considered as important for the data consumer. Among the fifteen attributes, Accessibility, Currency, Amount of data, Reliability, Believability, Understandability, Accuracy, Relevance, Consistent representation and Validity appeared to be relevant (see Fig.3). This result coincided with our initial

classification of DQ attributes for "Data point and Integration" functionality, in at least 66 %.

With the results of this validation we are able to fine-tune our model. This means, for the functionality being studied, looking at only the ten attributes that have been judged most important by the respondents (adding some particular attribute proposed by them). We do believe, however, that it would be better to await the result of the survey on all the functionalities, and then fully adjust our model to obtain a definitive PDQM model.

Finally, although we have results just for one portal functionality considered in our model, we believe that allows us to draw some conclusions. One of them is that our model, in regards to the functionality Data points and Integration, is very close to the data consumer perspective. We affirm this because our assignation of DQ attributes to evaluate this functionality coincides to a very large degree with their responses. Furthermore, we consider that the results of this survey demonstrated that using the surveys to validate our model is appropriate. Thus we can also best-fit our model to the data consumer perspective.

6 Conclusions and the Future Work

The Web portals are applications that have been positioned like information sources and/or as means of accessing information over the last decade. The other side to this is that those who need information by means of these portals need to be sure, somehow, that this information is indeed suitable for the use they wish. In other words, they really need to assess the level of the quality of the data obtained.

In the literature we have studied, there are no specific proposals for data quality models for Web portals. In this article we have presented a preliminary version of a data quality model for Web portals (PDQM) that consider the consumers point of view. This has been built on three fundamental aspects: a set of Web data quality attributes set out in the relevant literature, data quality expectations of data consumers on the Internet, and the functionalities which a Web portal may offer its users.

As a first step in the validation of the model, we carried out a survey to validate the "Data point and Integration" functionality. We started with this functionality because we believe it is the most important one from the data consumer perspective. In order to achieve a complete validation of our PDQM model, we will apply the questionnaire in full, i.e. with the 11 questionnaires created (one per functionality). The population sample chosen for this phase corresponds to the users of the portal of Castilla-La Mancha (www.castillalamancha.es).

We are very aware that giving 11 questionnaires to a person might turn out to be really tiring and none- too- motivating, so just three of the 11 questionnaires will be given to each person undergoing this survey. The said three will be chosen at random. Besides that, a Web application will be installed in the portal we have selected and this will manage the free distribution of the survey to the users who log on to the portal. The application will be made up of three modules: an administrative one (by which we can administer the survey and the application), an analyzing module (which will show the results: statistics, diagrams, response rate, etc) and a data collection

module (which creates the questionnaires for users from a random choice of 3 questionnaires from the possible 11 and which will be in charge of storing the answers. At the end of the time allowed for the application, when an acceptable level of response has been reached, we will analyse the results obtaining the final version of PDQM.

As future work, once we obtain a final version of PDQM, we will use it as basis to de-fine a framework for evaluating and improving the DQ for Web portals. Our aim is to offer the different consumers of Web portal data the possibility of evaluating, accor-ding to their needs and criteria, the DQ they receive from the Web portal.

Acknowledgments. This research is part of the following projects: CALIPO (TIC2003-07804-C05-03), CALIPSO (TIN20005-24055-E) supported by the Ministerio de Educación y Ciencia (Spain) and COMPETISOFT (506PI0287) financed by CYTED.

References

1. Aboelmeged, M., A Soft System Perspective on Information Quality in Electronic Commerce, In Proceedings of the Fifth Conference on Information Quality, 2000.
2. Bouzeghoub, M. and Kedad, Z., Quality in Data Warehousing, in Information and Database Quality, Piattini, M., Calero, C., and Genero, M., Eds.: Kluwer Academic Publishers, 2001.
3. Bouzeghoub, M. and Peralta, V., A Framework for Analysis of data Freshness, In International Workshop on Information Quality in Information Systems, (IQIS2004), Paris, France, 2004.
4. Burgess, M., Fiddian, N., and Gray, W., Quality Measures and The Information Consumer, In Proceedings of the 9th International Conference on Information Quality, 2004.
5. Cappiello, C., Francalanci, C., and Pernici, B., Data quality assessment from the user´s perspective, In International Workshop on Information Quality in Information Systems, (IQIS2004), Paris, Francia, 2004.
6. Caro , A., Calero, C., Caballero, I., and Piattini, M., Data quality in web applications: A state of the art, In IADIS International Conference WWW/Internet 2005, Lisboa-Portugal, 2005.
7. Collins, H., Corporate Portal Definition and Features: AMACOM, 2001.
8. Eppler, M., Algesheimer, R., and Dimpfel, M., Quality Criteria of Content-Driven Websites and Their Influence on Customer Satisfaction and Loyalty: An Empirical Test of an Information Quality Framework, In Proceeding of the Eighth International Conference on Information Quality, 2003.
9. Fugini, M., Mecella, M., Plebani, P., Pernici, B., and Scannapieco, M., Data Quality in Cooperative Web Information Systems, 2002.
10. Gertz, M., Ozsu, T., Saake, G., and Sattler, K.-U., Report on the Dagstuhl Seminar "Data Quality on the Web", SIGMOD Record, vol. 33, Nº 1, pp. 127-132, 2004.
11. Graefe, G., Incredible Information on the Internet: Biased Information Provision and a Lack of Credibility as a Cause of Insufficient Information Quality, In Proceeding of the Eighth International Conference on Information Quality, 2003.
12. Huang, K.-T., Lee, Y., and Wang, R., Quality information and knowledge: Prentice Hall PTR, 1999.

13. Katerattanakul, P. and Siau, K., Information quality in internet commerce desing, in Information and Database Quality, Piattini, M., Calero, C., and Genero, M., Eds.: Kluwer Academic Publishers, 2001.
14. Katerattanakul, P. and Siau, K., Measuring Information Quality of Web Sites: Development of an Instrument, In Proceeding of the 20th International Conference on Information System, 1999.
15. Kitchenham, B., Procedures for Performing Systematic Reviews, 0400011T.1, 2004.
16. Kitchenham, B. and Pfleeger , S. L., Principles of survey research: part 3: constructing a survey instrument, SIGSOFT Softw. Eng. Notes, ACM Press, 27, pp. 20-24, 2002.
17. Kopcso, D., Pipino, L., and Rybolt, W., The Assesment of Web Site Quality, In Proceeding of the Fifth International Conference on Information Quality, 2000.
18. Lee, Y., AIMQ: a methodology for information quality assessment, Information and Management. Elsevier Science, pp. 133-146, 2002.
19. Mahdavi, M., Shepherd, J., and Benatallah, B., A Collaborative Approach for Caching Dynamic Data in Portal Applications, In Proceedings of the fifteenth conference on Australian database, 2004.
20. Marchetti, C., Mecella, M., Scannapieco, M., and Virgillito, A., Enabling Data Quality Notification in Cooperative Information Systems through a Web-service based Architecture, In Proceeding of the Fourth International Conference on Web Information Systems Engineering, 2003.
21. Melkas, H., Analyzing Information Quality in Virtual service Networks with Qualitative Interview Data, In Proceeding of the Ninth International Conference on Information Quality, 2004.
22. Moraga, M. Á., Calero, C., and Piattini, M., Comparing different quality models for portals. (2006). To appear on Online Information Review.
23. Moustakis, V., Litos, C., Dalivigas, A., and Tsironis, L., Website Quality Assesment Criteria, In Proceeding of the Ninth International Conference on Information Quality, 2004.
24. Naumann, F. and Rolker, C., Assesment Methods for Information Quality Criteria, In Proceeding of the Fifth International Conference on Information Quality, 2000.
25. Pressman, R., Software Engineering: a Practitioner's Approach. Fifth ed: McGraw-Hill, 2001.
26. Redman, T., Data Quality: The field guide. Boston: Digital Press, 2000.
27. Strong, D., Lee, Y., and Wang, R., Data Quality in Context, Communications of the ACM, Vol. 40, Nº 5, pp. 103 -110, 1997.
28. Wang, R., A Product Perspective on Total data Quality Management, Communications of the ACM, Vol. 41, Nº 2, pp. 54-65, 1998.
29. Wang, R. and Strong, D., Beyond accuracy: What data quality means to data consumers, Journal of Management Information Systems; Armonk; Spring 1996, 12, pp. 5-33, 1996.
30. Winkler, W., Methods for evaluating and creating data quality, Information Systems, Nº 29, pp. 531-550, 2004.
31. Yang, Z. a. C., S. and Zhou, Z. and Zhou, N., Development and validation of an instrument to measure user perceived service quality of information presenting Web portals, Information and Management. Elsevier Science, 42, pp. 575-589, 2004.
32. Zhu, Y. and Buchmann, A., Evaluating and Selecting Web Sources as external Information Resources of a Data Warehouse, In Proceeding of the 3rd International Conference on Web Information Systems Engineering, 2002.

Finding Reliable Recommendations for Trust Model

Weiwei Yuan, Donghai Guan, Sungyoung Lee, Youngkoo Lee*, and Andrey Gavrilov

Department of Computer Engineering, Kyung Hee University, Korea
{weiwei, donghai, sylee, avg}@oslab.khu.ac.kr, yklee@khu.ac.kr

Abstract. This paper presents a novel context-based approach to find reliable recommendations for trust model in ubiquitous environments. Context is used in our approach to analyze the user's activity, state and intention. Incremental learning based neural network is used to dispose the context in order to detect doubtful recommendations. This approach has distinct advantages when dealing with randomly given irresponsible recommendations, individual unfair recommendations as well as unfair recommendations flooding regardless of from recommenders who always give malicious recommendations or "inside job" (recommenders who acted honest previous suddenly give unfair recommendations), which is lack of consideration in the previous works. The incremental learning based neural network used in our approach also enables to filter out the unfair recommendations with limited information about the recommenders. Our simulation results show that our approach can effectively find reliable recommendations in different scenarios and a comparison is also given between previous works and our method.

1 Introduction

Computational models of trust have been proposed for ubiquitous environments because they are capable of deciding on the runtime whether to provide services to requesters who are either unfamiliar with service providers or do not have enough access rights to certain services. The basis for the trust model to make decision for unfamiliar service requesters are the recommendations given by recommenders who have past interaction history with the requesters. However, in the large-scale, open, dynamic and distributed ubiquitous environments, there may possibly exist numerous self-interested recommenders who give unfair recommendations to maximize their own gains (perhaps at the cost of others). Since recommendations given by recommenders are the key point for the trust model to make decision, finding ways to avoid or reduce the influence of unfair positive or negative recommendations from self-interested recommenders is a fundamental problem for trust model in ubiquitous environments. At the same time, because of the highly dynamic nature of ubiquitous environments, it is not always easy to get enough information about the recommenders. Hence the trust model is required to find the reliable recommendations with limited information about the recommenders.

* Corresponding author.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 375 – 386, 2006.

The objective of this paper is to contribute to an approach which can find the reliable recommendations for the trust model in ubiquitous environments. This paper sets the stage by identifying a novel context-based approach using incremental learning algorithm. Context is used in our approach to analyze the user's activity, state and intention. The learning of context is incrementally increased by a Cascade-Correlation architecture neural network. By analyzing the context under which the recommendation was given, our method is able to filter out unfair recommendations in different scenarios. The advantages of our proposed approach are: (1) it can filter out randomly given irresponsible recommendations, individual unfair recommendations as well as unfair recommendations flooding no matter the recommendations are from recidivist (recommenders who always give malicious recommendations) or inside job (recommenders who acted honest suddenly give unfair recommendation on the benefit of themselves), (2) it can differentiate doubtful recommendations due to different reasons: the changing behaviors of service requesters in front of different recommenders, the incorrect observations by recommenders, as well as malicious intention of recommenders, (3) it is able to detect the malicious recommendations when limited information is available for the recommenders which is usually the case in a real scenario.

The rest of the paper is organized as follows. Section 2 gives the recommendation scenarios in ubiquitous environments. Section 3 gives our proposed approach in details. Section 4 gives the simulation results. Section 5 shows the comparison between our approach and previous works. The last section concludes the paper and points out the future work.

2 Recommendation Scenarios in Ubiquitous Environments

For the trust model in ubiquitous environments, the possible scenarios for the recommendations given by recommenders are as follows:

(1) Normal Recommendations.

a. Honest recommenders give accurate recommendations.

(2) Abnormal Recommendations.

b. Honest recommenders give inaccurate recommendations due to their incorrect observation.

c. Due to the changing behavior of service requester in front of different recommenders, honest recommenders give exceptional recommendations compared with recommendations given by other recommenders.

d. Recommenders give random recommendations at ease due to the lack of responsibility.

(3) Malicious Recommendations.

e. Recommenders who acted honest give unfair high or low recommendations individually. The past behaviors of these recommenders were always honest. However, they suddenly give unfair recommendations due to the relationship with the service requester or their own benefits. (Called Inside Job)

f. Recommenders who acted malicious give unfair high or low recommendations individually. Different from the recommenders in scenario e, these recommenders always give malicious recommendations in the past.

g. A number of recommenders who acted honest collude to give unfair recommendations (more than 50% of total recommendations), which causes the flooding of unfair recommendations. (Called Inside Job Flooding)

h. Unfair recommendations flooding similar as scenario g, but caused by recommenders whose past behaviors were always malicious.

A reliable trust model in ubiquitous environments should have the ability to filter out the recommendations in scenario b, d, e, f, g and scenario h, distinguish recommendations in scenario b and scenario d from recommendations in scenarios a, and differentiate scenario c apart from scenario b, d, e and scenario f.

3 The Proposed Approach

Trust is subjective since it is based on each user's own understanding of information. Hence it is relatively easy for the malicious recommender to pretend honest and for the honest recommender to be misunderstood as malicious, which makes it difficult to differentiate between the unfair and fair recommendations. Our key idea for the solution is that: different recommenders may give different recommendations due to their different understandings on the same information, however, from the view of psychology, one recommender will follow his own rule on recommendation giving, i.e., one recommender usually gives similar recommendations in similar context. In case one recommender gives exceptional recommendations compared with his previous ones in similar context, the reason lies in two aspects. One reason is that this exceptional recommendation is a malicious recommendation and should be filtered out. The other reason is that the recommender's rule on recommendation giving has changed, e.g. due the environments' effect on him, one recommender now only gives positive recommendation to the service requester whose past interaction with him are more than 80% successful in stead of 60%. In this case, our architecture will use incremental learning algorithm to catch up the recommender's new rule.

Table 1. Context used in our approach

ID	AB.	Context	Example
1	NA	User Name	Weiwei
2	RE	Relationship with other agents	Senior member
3	TD	Time/date of request	Weekday/daytime
4	CS	Current state	Busy working
5	PI	Past interaction history with service requester	57% successful
6	TL	Time of last communication	Within 3 days
7	CF	Confidence for the service requester in given time window	Number of total Communication is 27

To learn each recommender's rule on recommendation giving, we use incremental learning based neural network to learn the recommendations as well as the context under which the recommendations were given by the recommenders. The reason we use incremental learning is that the acquisition of a representative training data for the rule is time consuming and the rule is also possible to dynamically change from time to time. Consequently, it is not uncommon for such data to become available in small

batches over a period of time. In such setting, it is necessary to update an existing classifier in an incremental fashion to accommodate new data without compromising classification performance on old data [1]. The context which may relate to the learning of the rule is shown in Table1. Attributes 1, 3 and 4 of the context are specifically bounded to the recommender's activity or state, and the other attributes are supposed to hold regardless of the recommender's state since it depends on relationship of the external world where the recommender is currently situated.

We use the following architecture (shown in Fig.1.) to filter out the unfair recommendations.

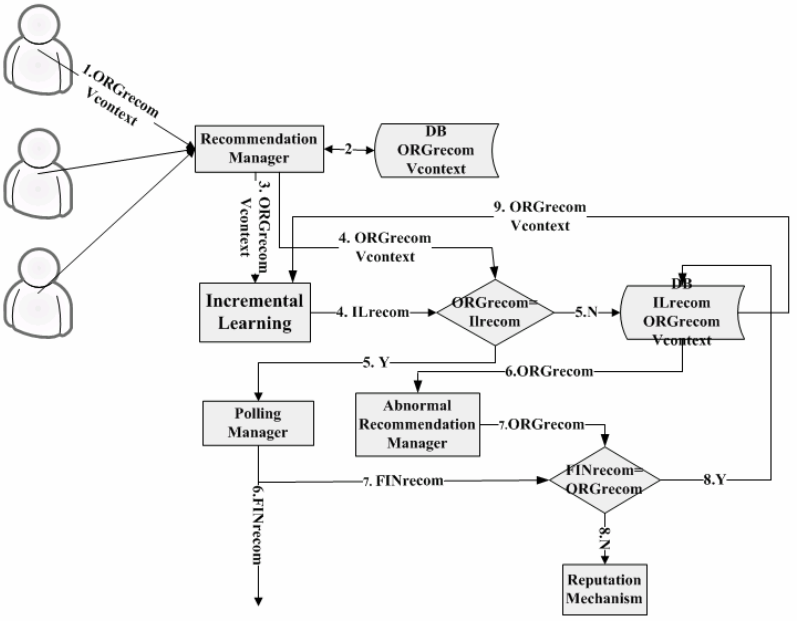


Fig. 1. Architecture Used for Filtering out Unfair Recommendations

Step 1: Recommendation Manager collects recommendations (REC_{org}) from all the recommenders, along with the context value $V_{context}[NA, RE, TD, CS, PI, TL, CF]$ under which recommendations were given by each recommender, where NA, RE, TD, CS, PI, TL, CF represent the context attributes mentioned in Table 1 from (1) to (7) respectively.

$$REC_{org} = \begin{cases} 1 & trusted \\ 0 & untrusted \end{cases}$$

Step 2: We use this step to find the doubtful recommendations from those gotten by step 1. To achieve this, we use incremental learning neural network, in particular, the Cascade-Correlation architecture.

Cascade-Correlation is useful for incremental learning, in which new information is added to an already-trained network. It is an architecture of neural network which begins with minimal network, then automatically trains and adds new hidden units one by one, creating a multi-layer structure. Once a new hidden unit has been added to the network, its input-side weights are frozen. This unit then becomes a permanent feature-detector in the network, available for producing outputs or for creating other, more complex feature detector [2]. Fig.2 gives the process of training Cascade-Correlation. In 1, we train the weights from input to output. In 2, we add a candidate unit and train its weights to maximize the correlation with the error. In 3, we retrain the output layer. We train the input weights for another hidden unit in 4. Output layer is retrained in 5, etc. The usage of Cascade-Correlation architecture has several advantages over others: it learns very quickly; the network determines its own size and topology; it retains the structures it has built even if the training set changes; and it requires no back-propagation of error signals through the connections of the network.

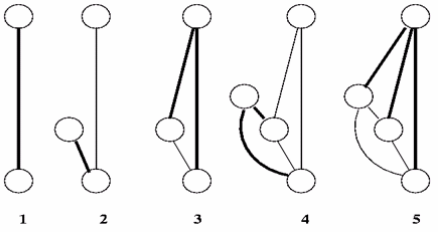


Fig. 2. Training Procedure of Cascade-Correlation Architecture

For each recommender who gives recommendation, the context $V_{context}[NA, RE, TD, CS, PI, TL, CF]$ under which he gave recommendation REC_{org} is used as the input of the Cascade-Correlation architecture. Using the Cascade-Correlation architecture trained by the previous context and corresponding recommendations, we will get the output REC_{IL}. REC_{IL} is the recommendation that the recommender will give due to his past behavior when given the input context $V_{context}[NA, RE, TD, CS, PI, TL, CF]$.

Compare REC_{org} and REC_{IL}:

$$REC_{com} = \begin{cases} REC_{IL} & REC_{IL}=REC_{org} \\ -1 & else \end{cases} \quad (1)$$

If $REC_{com} = REC_{IL}$, it means that the recommender gives the same recommendation as previous behavior in similar context, the possible scenarios are scenario a and scenario c. Otherwise if $REC_{com} = -1$, REc_{org} is regarded as a doubtful recommendation, which implies that the recommender's current behavior on

recommendation giving is different from previous. We will use following steps to judge whether this doubtful recommendation is malicious.

Step 3: We use this step to get the final recommendation. Since by using step 2 we have already found the doubtful recommendations, we use basic voting mechanism to calculate the final recommendation REC_{fin} . The voting is among the recommendations which are not doubtful ($REC_{com} = REC_{IL}$ in (1)).

$$REC_{fin} = \begin{cases} 1 & NUM\,[REC_{comi}=1 \,|\, REC_{comi} \neq -1] \geq \dfrac{NUM\,[REC_{comi} \neq -1]}{2} \\ 0 & else \end{cases}, \qquad (2)$$

where REC_{comi} is the REC_{com} of recommender i , $i \in N$. $NUM[REC_{comi} \neq -1]$ means the number of all the recommendations which are not doubtful. $NUM[REC_{comi} = 1 \,|\, REC_{comi} \neq -1]$ is the number of undoubtful recommendations which equal to 1.

Step 4: This step is used to find the malicious recommendations (scenario e, f, g and scenario h) and incorrect recommendations (scenario b and scenario d) from the doubtful recommendations gotten in step2.

In (1), if $REC_{com} = -1$, the possible situations are: A. the recommendation is malicious or incorrect, the possible scenarios are scenario b, d, e, f, g and scenario h, B. (1) the recommender's rule on recommendation giving has changed, i.e. the recommender gives different recommendation compared with previous one in similar context, however, this recommendation is also right. This kind of situation is reasonable since all the things in the world are always in movement, (2) the currently neural network is not enough to reflect the recommender's rule on recommendation giving since the Cascade-Correlation architecture begins with a minimal network and the knowledge on the recommender's rule is incrementally increased. We use the following method to differentiate between situation A and B.

$$result = \begin{cases} situationA & REC_{org} \neq REC_{fin} \,|\, REC_{com} = -1 \\ situationB & REC_{org} = REC_{fin} \,|\, REC_{com} = -1 \end{cases}, \qquad (3)$$

where $REC_{org} \neq REC_{fin} \,|\, REC_{com} = -1$ and $REC_{org} = REC_{fin} \,|\, REC_{com} = -1$ means $REC_{org} \neq REC_{fin}$ and $REC_{org} = REC_{fin}$ when given $REC_{com} = -1$ respectively.

If the result equals to situation B, $V_{context}[NA, RE, TD, CS, PI, TL, CF]$ as well as REC_{org} will be given back as retrain data to retrain the Cascade-Correlation architecture (as showed in Fig.1) to catch up the recommender's current rule on recommendation giving. Otherwise if the result is situation A, the record of this recommender will be given to a separated disposal unit and mark it as doubtful recommender. If one user appears several times as a doubtful recommender, it will be considered either as a malicious recommender or a recommender who is unable to

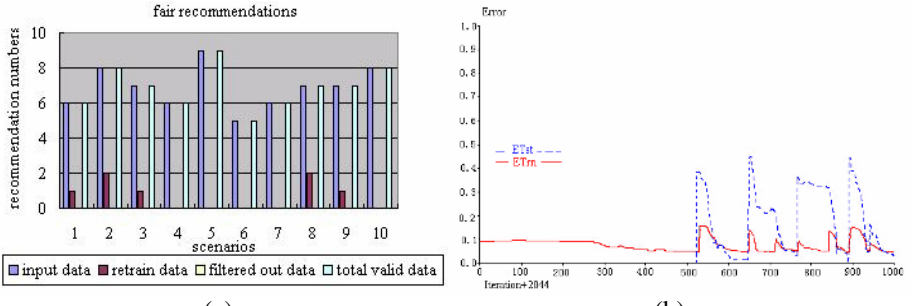
give correct recommendations. The recommendations given by this recommender later will be directly filtered out in step1.

4 Simulation Results

Our simulation is based on a ubiquitous computing supported smart office [3]. The possible recommenders in this smart office have different positions (in other words, they have different trust levels), such as professor, senior member. For the training of the original Cascade-Correlation architecture, we randomly choose 5 services requesters, the recommendations given by different recommenders as well as the context act as the training data. Each context is a vector including 7 attributes as shown in Table 1. The learning rate of the Cascade-Correlation neural network is set to 0.01 and the error tolerance is 0.05. The iterations for training the original neural network is 2252. As time goes by the Cascade-Correlation architecture will incrementally learn the new information (retrain data) which will make it closer to each recommender's current rule on recommendation giving. The following subsections give our simulation results in different scenarios mentioned in section 2.

4.1 Fair Recommendations

Fig.3 shows the simulation results when there is no unfair recommendation among all the recommendations (scenario a and scenario c). Since our approach compares the recommendations with the recommender's own previous behaviors, so there is no difference when dealing with scenario a and c in our architecture. The test data are the recommendations given by recommenders for different service requesters. The retrain data in Fig.3 (a) is the recommendations and the corresponding context which got $result = situationB$ in formula (3).



(a) (b)

Fig. 3. There is no unfair recommendation in total recommendations

As shown in Fig.3 (a), the total valid data is the same as input data, which means that when there is no unfair recommendation, even there are some recommendations have the result $REC_{com} = -1$ in formula (1), no data will be filtered out, in other

words, our approach does not have negative bias when the recommendations are all fair. Fig.3 (b) shows the simulation result of the Cascade-Correlation architecture using the data shown in Fig.3 (a). ETrn is the error of training data and ETst is the error of test data. When the error of training data is less than 0.05, the iteration is stopped. The sudden raise of ETrn is because of the adding of retrain data which were found as shown in Fig.3 (a). Due to the retrain of the Cascade-Correlation architecture, ETst also has the sudden raise along with the raise of ETrn. When ETrn is reduced to a reasonable level(less than 0.05 in our model), after a number of iterations, ETst is also reduced to a low level(less than 0.05), which means that the new neural network can fit the recommender's current rule.

4.2 Individual Unfair Recommendations

Using the input data as shown in Fig.3 (a), we randomly choose two recommenders and set their recommendations to be unfair. These data act as input data in this subsection. Since the recommenders are randomly chosen, they may have different trust levels. The possible scenarios are scenario b, e and scenario f.

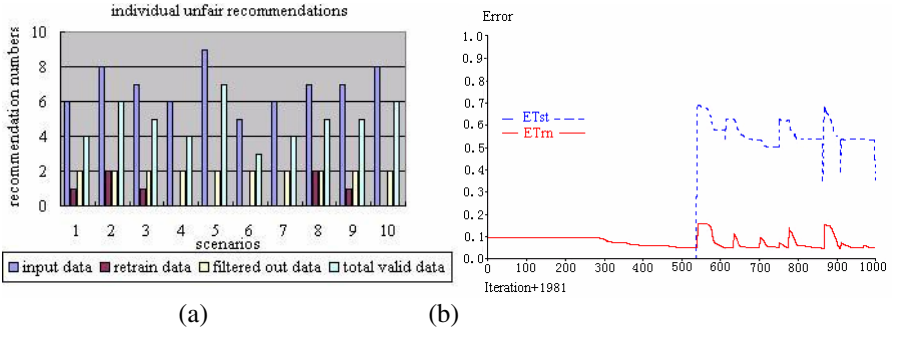


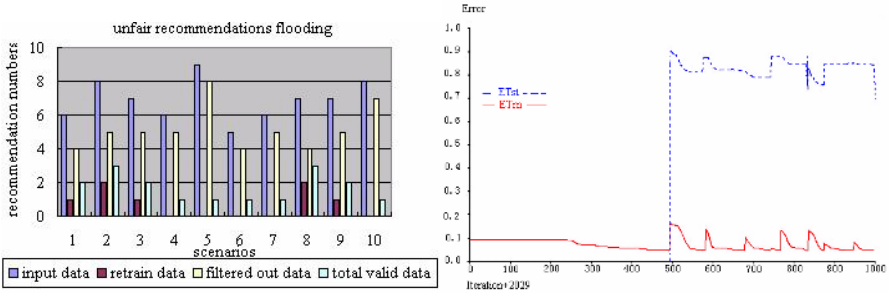
Fig. 4. There are individual unfair recommendations in total recommendations

The filtered out data as shown in Fig.4 (a) are those got *result* = *situationA* in formula (3), while if *result* = *situationB* in formula (3), these data will act as retrain data. Though $REC_{com} = -1$ in both case, only the data those act as retrain data can be valid data. Fig.4 (a) shows that our approach can accurately find the unfair recommendations (two for each input data as we set previous).

Using formula (2) and the input data showed in Fig.4 (a), we got the same final recommendation as section 4.1. Fig.4 (b) shows the simulation results for Cascade-Correlation architecture using the data in Fig.4 (a). The difference between Fig.4 (b) and Fig.3 (b) is that in Fig.4 (b), ETst is still big (more than 0.3) when ETrn is reduced to a reasonable level using the retrain data. The reason is that there are unfair recommendations among the test data (the input data in Fig.4 (a)). Since our approach can differentiate unfair recommendations from fair recommendations, ETst can not be reduced to a very low level along with ETrn.

4.3 Unfair Recommendations Flooding

Using the input data as shown in Fig.3 (a), we randomly choose the majority of recommenders and set their recommendations to be malicious. These data act as input data in this subsection. Since the recommenders are randomly chosen, they may have different trust levels. Therefore the possible scenarios are scenario g and scenario h. The result of the final recommendation using formula (2) in this case proves the correctness of our approach when dealing with unfair recommendation flooding since it is the same as the final decision in section 4.1 and 4.2.



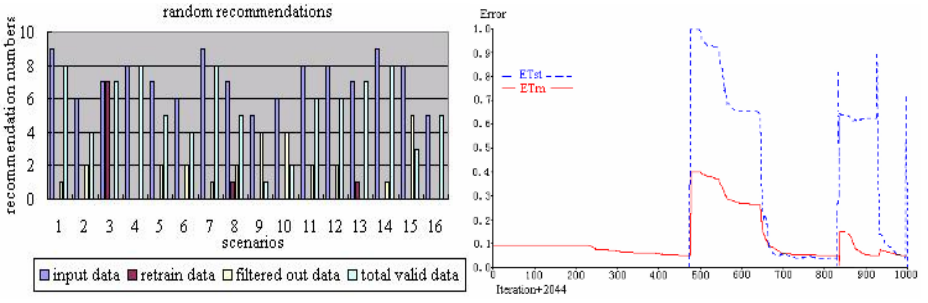
(a) (b)

Fig. 5. There are unfair recommendations flooding in total recommendations

If the unfair recommendations flooding can successfully influence the final recommendation, the fair recommendations will be regard as unfair. In this case since the unfair recommendations are majority, ETst can be reduced to a relatively low level (less than 0.5, the curve will be similar as Fig.4 (b)) along with ETrn. However, our simulation result shows that for ETst, when ETrn is reduce to a reasonable level, ETst in Fig.5 (b) is much higher than in Fig.4 (b) (above 0.7), which means that our approach can defend the unfair recommendations flooding.

4.4 Randomly Given Recommendations

The possible scenario in this case is scenario d. Since the recommendations are randomly given, there will be random number of filtered out data (as shown in Fig.6 (a)) as well as retrain data. Fig.6 (b) gives the error of Cascade-Correlation architecture given the data in Fig.6 (a). The curve of ETst does not have regularity. It can be up to a very high level (near 1.0) since most of recommendations for certain service requester are unfair and it can also be a very level (lower than 0.05) since the recommendations for certain service requester are fair. The reason our approach can filter out the randomly given recommendations is that if the recommendations are given randomly, it must be different with its own past behavior in similar context. Therefore, we will get *result* = *situationA* in formula (3).



(a) (b)

Fig. 6. The recommendations are randomly given

5 Comparison with Related Works

There are some researches that gave helpful attempts on how to get reliable recommendations, especially for scenario f and scenario h. One method is to use polling method, e.g. in [4], the authors used basic polling as well as enhanced polling. The enhanced polling differs from the basic polling by requesting voters to provide their servent_id to prevent a single, possible malicious user to create multiple recommendations at a time. Another very popular method is to give weighted value to each recommender (also called a reputation-based method) to choose reliable recommendations. The reputation-based method had been used in a number of works, e.g. weighted majority algorithm is used in [5], and a Rating Reputation Feedback mechanism is used in [6] to train the weighted values. In [7] [8] [9] [10], the authors measure the reputation for each recommender and filter out unfair recommendations based on the usage of the reputation. Using the combination of different filters is also a reasonable method to filter out the unfair recommendations, as mentioned in [11] [12] [13] [14]. Their simulation results suggest that cluster filtering is suitable to reduce the effect of unfairly high recommendations and positive discriminations and frequency filtering can guarantee the calculation of trust not be influenced by the unfair raters flooding.

Table.2 gives the comparison between different methods when dealing with different scenarios as mentioned in section 2. The reason previous methods can not deal with some of the scenarios is that they took at least one of the following assumptions: (1) most recommendations are close in the range of the real quality of the product, (2) recommendations provided by different recommenders on a service requester will follow more or less the same probability distribution, (3) top ranked recommenders are the expert recommenders in the trust category, i.e., the higher rank recommender has, the more authority his recommendation will have. For example, as the most popular method, reputation-based method takes assumption (3), hence it is impossible for this method to filter out any of the inside job (scenario e and scenario g). What's more, if scenario e and scenario g happens, the higher the recommender's rank is, the more serious aftereffect there will be. Our approach is effective in different scenarios because the comparison used to filter out the unfair recommendations is not between different recommenders but between each recommender's own current behavior and previous behavior. Hence we do not take any of these assumptions.

Table 2. Comparison between different methods

Scenario	Polling	Weight-based (Reputation-based)	Combination of Filters	Our Approach
a	Y	Y	Negative Reputation Bias	Y
b	Y	High Rank User N Low Rank User Y	Y	Y
c	N	High Rank User Y Low Rank User N	N	Y
d	N	High Rank User N Low Rank User Y	Effective when variance is large	Y
e	Y	N	Y	Y
f	Y	Y	Y	Y
g	N	N	Y	Y
h	N	Y	Y	Y

However, the cost of our method is that it takes longer time for our approach to find the reliable recommendations. The reason is that to judge the validity of recommendations, each recommender's current behavior should be compared with his past behaviors. However, since these calculations take place in the service agent which has enough computing ability, we believe that it does not distinctly affect the efficiency of the whole trust model.

6 Conclusions

In this paper we propose a robust trust model for ubiquitous environments, in which a context-based approach is used to filter out the unfair recommendations. The learning of the context is based on incremental learning neural network. The filtered out recommendations may be the intended unfair recommendations as well as the mis-observation by the recommenders. We also focus on the flooding of unfair recommendations in this paper. Since our approach concentrates on the doubtful behavior of each entity, it has special advantages when dealing with inside job, which is lack of considerations in the current trust models.

In the future work, we plan to focus on the scalability of trust model since our trust model is merely used in a smart office now. And we also want to find the relationship between different attributes in the context shown in Table.1, and try to find the most important ones. We also plan to add risk analysis in our context-based trust model. Based on our comparison between our context-based approach and other methods, we believe that the usage of context-based trust model and incremental learning neural network within ubiquitous environments application presents a promising path for the future research.

Acknowledgment. This research was supported by the MIC (Ministry of Information and Communications), Korea under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment) in collaboration with SunMoon University.

References

1. Robi Polikar, Lalita Udpa, Satish S. Udpa and Vasant Honavar, "learn++: An Incremental Learning Algorithm for Supervised Neural Networks", IEEE transactions on systems, man, and cybernetics-Part C: Applications and Reviews, Vol. 31, NO.4, Nov.2001
2. Scott E. Fahlman, Christian Lebiere, "The Cascade-Correlation Learning Architecture". Technical Report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University
3. Hung Q. Ngo, Anjum Shehzad, Saad Liaquat Kiani, Maria Riaz, Kim Anh Ngoc, Sungyong Lee.: Developing Context-aware Ubiquitous Computing Systems with a Unified Middleware Frame Work. The 2004 International Conference on Embedded & Ubiquitous Computing (EUC2004)
4. Damiani, Vimercati, Paraboschi, Samarati, and Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks", 9th ACM CCS 2002
5. Bin Yu, Munindar P. Singh, and Katia Sycara, "Developing trust large-scale peer-to-peer systems", First IEEE Symposium on Multiagent Security and Survivability, 2004
6. Ping Xu, Ji Gao, Hang Guo, "Rating Reputation: a necessary consideration in reputation mechanism", Proceedings of 2005 International Conference on Machine Learning and Cybernetics
7. Whitby, A., Josang, A. and Indulska, J. "Filtering out unfair ratings in Bayesian reputation systems", AAMAS 2004, New York, USA
8. Weihua Song, Vir V. P hoha, and Xin Xu, "An adaptive recommendation trust model in multiagent system", IEEE/WIC/ACM IAT'04
9. Weihua Song, Vir V. Phoha, "Neural network-based reputation model in a distributed system", pp. 321-324, 2004 IEEE International Conference on E-Commerce Technology (CEC'04), 2004
10. Huang Baohua; Hu Heping; Lu Zhengding, "Identifying local trust value with neural network in p2p environment", The First IEEE and IFIP International Conference in Central Asia on Internet, 2005
11. C. Dellarocas, "The design of reliable trust management systems for electronic trading communities", MIT Working Paper
12. C. Dellarocas , "Building trust online: the design of robust reputation reporting mechanisms for online trading communities" A combined perspective on the digital era, Doukidis, G., Mylonopoulos, N. and Pouloudi, N. (Eds.), Idea Book Publishing (2004)
13. C. Dellarocas. "Immunizing online Reputation Reporting systems against unfair ratings and discriminatory behavior", In Proceedings of the ACM Conference on Electronic Commerce, pages 150--157, Minneapolis, Minnesota, USA, 2000
14. Chrysanthos Dellarocas , "Mechanisms for coping with unfair ratings and discriminatory behavior in online reputation reporting systems", In ICIS, pages 520--525, 2000

Self-Updating Hash Chains and Their Implementations*

Haojun Zhang[1,2] and Yuefei Zhu[1]

[1] Network Engineering Department, Information Engineering University, Zhengzhou, Henan 450002, P.R. China
[2] Computer Engineering Department, Henan University of Technology, No.140, SongShan South Road, ZhengZhou, Henan 450052, P.R. China
zhj@haut.edu.cn

Abstract. Hash Chains are widely used in various cryptography applications such as one-time passwords, server-supported signatures and micropayments etc. However, the finite length ('limited-link') of hash chains limits their applications. Some methods of re-initializing hash chains or infinite hash chains introduced in literatures are inefficient and un-smooth. In this paper, a novel scheme (a new kind of hash chain) is proposed, which re-initializes or updates by itself, named *Self-Updating Hash Chain* – SUHC. Highlights of SUHC are *self-updating*, *fine-authentication* and *proactive updating*. The updating process of SUHC is smooth, secure and efficient and does not need additional protocols or an independent re-initialization process, and can be continued indefinitely to give rise to an infinite length hash chain. An improved *Server-Supported Signature* with SUHC is also presented to show the application of SUHC.

Keywords: hash chain, updating, authentication, Server-Supported Signature.

1 Introduction

Hash functions have been widely used in cryptography techniques such as signature, authentication etc. Cryptographic secure hash functions must have two important properties: one-way and collision-resistance.

A hash chain is generated by multiple iterations of a secure hash function upon a secret seed and a unique sequence of proof materials (links) is generated. Then the tip T of the chain is securely distributed and then the elements of the hash chain are spent one by one by starting from T and continuing until the value of s is reached. This technique was first suggested by Leslie Lamport [1].

Owning features of public-key cryptography, hash chains are widely used for data-origin or entity authentication. Examples of system built on the conception of hash chains include password systems based on authentication – one-time password (OTP) [1], [2], server-supported signatures [3], certificate revocation [4], efficient key distribution [5], micropayments and digital cash, efficient multicasting and so on.

* Research supported by the NNSF of China (No. 90204015, 60473021); the Key Technologies R&D Program of Henan Province of China (No. 0524220044, 062426001); the S&R Found of HAUT of China (No. 0401009, 050211,050215,050216).

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 387 – 397, 2006.

However, in practical applications, a hash chain has a finite length. A hash chain would be exhausted quickly if its length were short. On the contrary, if its length were long, overhead of computation (for computing links of a chain) would go up and security would go down. When a hash chain has been exhausted, re-initializing a new chain is needed. We have not found an "elegant" method in literatures for re-initializing a hash chain smoothly and efficiently. In this paper, we propose a scheme that achieves infinite length hash chains and eliminates inconvenience of re-initialization in conventional hash chains (CHC) applications.

Rest of the paper is organized as follows – Section 2 introduces the related works. Section 3 states some definitions. Section 4 presents a formalized description for our scheme – self-updating hash chain. Section 5 discusses security and efficiency of the proposed scheme. Section 6 gives an application example of our scheme. And section 7 concludes the paper.

2 Relative Works

As an authentication root, the tip of a hash chain is usually signed along with the identity of its owner and published by a Certification Authority (CA), or delivered /distributed directly and securely to a certifier. Once a chain is exhausted, a new chain needs to be re-created or re-initialized, and then the new tip needs to be re-signed by the CA and be redelivered securely to the certifier. It is obviously that above process needs complex management infrastructure.

Paper [6] defined a concept called Infinite Length Hash chains (ILHC) and gave an application for OTP. The basic idea of ILHS is:

Let algorithm A be a public-key algorithm (e.g. RSA) where d is the private key and e is the public key. Let s and c constitute a pair such that $A(s,d)=c$ and $A(c,e)=s$.

$A^N(s,d)$ denotes that applying algorithm A recursively N times to the initial input (seed) s using the private key d. The result is an infinite length hash chain –ILHC:

$$s, A(s,d), A^2(s,d), A^3(s,d), ..., A^N(s,d), ...$$

Public key e and seed s are transmitted to a server securely in advance. The user owning a ILHC produces a first one time password $P^0=A(s,d)$. Upon receiving P^0, the server can verify it by applying $s=A(P^0,e)$. The general formula for the i-th password is $P^i=A^i(s,d)$. Correspondingly the server verifies if $P^{i-1}= A(P^i,e)$.

However, in our view, using public-key cryptography defeats the basic purpose of hash chains as it compromises on efficiency. Note that in most cases, hash chains are mainly used to remove/replace/complement the usage of public key cryptography for gaining efficiency.

Vipul Goyal [7] introduced a method for re-initializing hash chains (RHC). He gave a notion of 'tying' multiple finite length hash chains by using one time signature. However, the re-initialization process of RHC is independent relatively and needs lager storage for one time signature (OTS) key components even though these components are revealed as the same manner of links of hash chains.

Referring to OTS conception [8], but completely differing from RHC, we propose a novel scheme (an improved kind of hash chain) that achieves a hash chain

re-initialization automatically and smoothly during being consumed. We call it self-updating hash chain (SUHC).

3 Definitions

Firstly, we give some basic concepts used by our scheme.

Definition 1. Cryptographic Secure Hash Function (CSHF)

A function h is a CSHF if it satisfies following properties:

- Define k is a security parameter of h.
- Function h maps bit strings, either of an arbitrary length or a predetermined length, to strings of a fixed length k. Denote h: $\{0,1\}^* \mapsto \{0,1\}^k$.
- Given x, it is easy to compute $h(x)$.
- Given $h(x)$, it is computationally infeasible to compute x. That means inverting h is hard.
- For any given x, it is computationally infeasible to find $y \neq x$ such that $h(x)=h(y)$.
- It is computationally infeasible to find any pair (x,y) such that $h(x)=h(y)$.

Definition 2. Hard Core Predicate

A hard core predicate of a function f: $\{0,1\}^* \mapsto \{0,1\}^*$ is a Boolean predicate $B:\{0,1\}^* \mapsto \{0,1\}$ such that for every probabilistic polynomial-time algorithm A', every polynomial $p(.)$, and all sufficiently large n there is:

$$\Pr[A'(f(U_n)) = B(U_n)] \leq \frac{1}{2} + \frac{1}{p(n)},$$

where U_n denotes a random variable uniformly distributed over $\{0,1\}^n$.

Definition 3. One-Way Hash Chain

Select a CSHF h defined by definition 1 with security parameter k, h: $\{0,1\}^* \mapsto \{0,1\}^k$. Pick a seed s randomly and apply h recursively N times to a initial seed s to generate a hash chain. The tip ω of the chain equals $h^N(s)$.

$$\omega = h^N(s) = h(h^{N-1}(s)) = \underbrace{h(h(h(...h}_{N}(s)))) \ .$$

Note that h may be selected from collections of one-way functions, but we focus on a single h in order to avoid the notation gets mess.

Definition 4. Basic One-time Signature (OTS)

OTS signature scheme is a collection (Gen, Sig, Ver). Where

- Key generation Function - *Gen*(): Chose a hash function h defined by Definition 1. Generate two k-bits private keys x^0, $x^1 \in_R \{0, 1\}^k$ ($\in_R$ denotes uniformly and randomly pick). And compute two corresponding public keys $y^0=h(x^0)$ and $y^1=h(x^1)$.

- Signing Function - *Sig*(): To sign a single-bit message m, compute signature σ: $\sigma = x^0$ if m=0, and $\sigma = x^1$ if m=1.
- Verifying function - *Ver*(): To verify (m, σ), check if $f(\sigma) = y^m$.

The scheme OTS was first proposed by Lamport [8]. Merkle, Winternitz etc. improved and developed it [9].

Theorem 5. The scheme OTS defined by definition 4 is secure.

The proof sketch refers to appendix A.

4 The Proposed Scheme

In our scheme, a certifier is an entity which generates a hash chain and uses links of the hash chain as his *authentication proof materials* or *tokens*, and a verifier is an entity which authenticates the identity of certifier by verifying the validity of these *proof materials*. A verifier is usually an authentication server in practical applications.

4.1 Initialization

Firstly, the certifier and the verifier negotiate a secure hash function h: $\{0,1\}^* \mapsto \{0,1\}^k$ with a security parameter k. The output of h is a k-bits string. Chose a predicate B:$\{0,1\}^* \mapsto \{0,1\}$ which is a hard core of h. Next the certifier does as follows:

- Initialize a hash chain. Pick a secure seed s from space $\{0,1\}^k$ randomly (suppose the length of s is k-bits but not must be) and generate a hash chain according to definition 2. Tip of the chain is $\omega = h^k(s)$.
- Generate next hash chain. Pick a random number $s' \in_R \{0,1\}^k$, construct another hash chain whose $\omega' = h^k(s')$, save s' and ω' securely at local.
- Initialize i=1.
- Generate the first pair of private/public key for signing bit. Select a random number denoted SK_1 such that $B(SK_1) = \omega'[i]$, and compute $PK_1 = h(SK_1)$. Save SK_1 securely.
- Compute $g_1 = h^{k-1}(s)$ and $\zeta_0 = h(g_1, PK_1)$, then save g_1.
- Distribute (ω, ζ_0).

Remark. (ω, ζ_0) is made public or securely distributed/delivered to the appropriate verifier. Signing a bit with a pair private/public key is conceptually and functionally identical to OTS, so we adapt the same concept in this work (or call it semi-OTS).

4.2 Certification

In certification processes, the certifier produces *proof materials* for authentication and reveals them. For the i-th verification the certifier does:

- Compute hash values $g_{i+1} = h^{k-i-1}(s)$ and compute / or retrieval $g_i = h^{k-i}(s) = h(g_{i+1})$.
- Prepare next bit commitment. Pick a random number SK_{i+1} such that $B(SK_{i+1}) = \omega'[i+1]$, and compute $PK_{i+1} = h(SK_{i+1})$ and save SK_{i+1} securely.
- Compute message authentication code:

$$\zeta_i = h(g_{i+1}, PK_{i+1}) \ .$$

– Construct an authentication frame (auth-frame) as following, and reveal it.

$$(g_i, PK_i, \zeta_i, SK_i) \ .$$

Remark. The value i is from 1 to k, so reveling links of a hash chain k times will carry k bits of the tip of the next/a new chain to the verifier. When $i=k$, the secret seed s of current hash chain is revealed and the current hash chain has been exhausted. At the same time we notice wondrously that the tip of a new hash chain has been conveyed to the verifier securely and the new chain has been prepared for used.

4.3 Verification

Upon receiving an auth-frame (g_i,PK_i,ζ_i,SK_i) from a certifier, the verifier does:

– Compute and check if $h(g_i) = g_{i-1}$, where g_{i-1} is sent in the last session and recorded.
– Verify integrity of public key of one time signature. Compute $h(g_i,PK_i)$ and check if it equals ζ_{i-1}.
– Get the signed bit. If $PK_i=h(SK_i)$, which means the signature is valid, then compute and record bit value $B(SK_i)$, otherwise report error.
– If all above validity verifications are passed, the verifier has verified successfully the certifier.

4.4 Self-Updating

After k times successful certification-verification, the verifier exactly obtains a k-bits value which is the tip ω' of the next/a new hash chain. At this time, the certifier replaces ω with ω' and s with s'. The new chain begins work and another new chain is generated including a seed s' and the corresponding tip ω' (actually, which are generated in (k-1)-th certification of previous chain). Iteration of above processes has a result that hash chains work continuously and infinitely.

Intuitively, every link's reveal conveys a bit of the tip of next hash chain "on its shoulder". In other words, the updating process is carried out during the chain is used. So, we call the proposed kind of hash chain or scheme self-updating hash chain (SUHC).

5 Security and Performance Analysis

5.1 Correctness

A SUHC works in the same manner as a normal/conventional hash chain (CHC), which has the whole properties of a CHC. It can be used for entity authentication or data-origin authentication, and others see no difference as the functionality it provides with respect to CHC. A certifier reveals links (or called proof materials) of a hash chain conversely to proof his identity.

The difference between a SUHC and a CHC is that the former does not only accomplish basic functions of a hash chain but also update by itself. In every authentication message one bit of next chain's tip is signed and conveyed. A new chain is born as soon as the current chain is exhausted.

We call this process self-updating or automatic re-initialization of hash chains. This is the most significant feature of the new kind of hash chains we proposed.

5.2 Security Analysis

Intuitively, the security of SUHC depends on the security of hash function which is employed as a core and sole algorithm for achieving hash chains and one time signature. The unforgery and integrity of massages are ensured by the properties of one-way and collision-resistance of hash functions.

While a hash chain is generated securely, adversaries cannot forge legitimate and available links from having been revealed links because inverting hash functions is computationally infeasible.

Bits of the tip of a new chain are conveyed one by one, and they are authenticated by bit signature – OTS. Further more, the integrity of the public key of a signature is protected by message authentication codes MAC ζ_i (ζ_i authenticates $(i+1)$-th public key).

Theorem 6. The scheme SUHC is secure if selected hash functions are cryptographic secure.

Proof. We say SUHC is secure means that SUHC achieves certification securely and bits conveyed in message are secure. In other words, adversaries could not construct legitimate links of a hash chain without knowing secret seed s, and could not control or select bits conveyed.

First of all, we assume that a hash function is selected satisfying definition 1. It means that finding converse function or finding collision in polynomial time is computationally infeasible. Next, we discuss security of SUHC in several cases as follows.

Case 1. An adversary successfully forge a link g_i in current session so that $h^m(g_i)=\omega$ with (t,e), where $1\leq m\leq p(k)$, and t denotes time consumption that the adversary spends, e denotes the adversary obtains success probability. In this case, we say that the adversary has the ability in (t,e)-inversing the hash function or finding collisions in (t,e). If t is polynomial and e is non-negligible, the adversary will succeed in inversing hash function h or finding a collision in polynomial time t with non-negligible probability e. It means that the adversary can successfully break SUHC in $(p(t),e)$. This contradicts assumption.

Case 2. An adversary could control or tamper bits conveyed, which is parts of the tip of a new chain. We can further separate this case into two sub-situations.

Case 2-1. The adversary breaks bit signature successfully with (t,e) through constructing the private key SK of a signature so that $h(SK)$ equals verification public key PK which has been computed in previous authentication and can be certified by MAC ζ. That means the adversary (t,e)-inverses the hash function or finds a collision in (t,e) because the signature algorithm is hash function too. This causes the same effect as case 1.

Case 2-2. The adversary could select pairwise key *SK*/*PK* of a signature so that $B(SK)$ is a desired value. If he succeeds, that means $h(g,PK)$, where g is current link of the chain and PK is a value the adversary picks choicely, equals the message authentication code ζ constructed and conveyed in previous authentication session (the trust can be traced back to root ζ_0 which is securely distributed). We notice that this case is as same as above cases, it needs the adversary succeed in constructing pre-image under only knowing image of h – inversing h, or finding collisions. It means that the adversary succeeds in computing g and PK when he knows $h(g,PK)$, or finding SK' and a collision (g',PK') so that $h(g,PK)= h(g',PK')$, $PK'=h(SK')$ and $B(SK')$ is a desired value. This causes the same contradiction with assumption too.

Remark. If an adversary want to control the next chain tip, he must control every SK so that $B(SK)$ equals exactly correspond bit in the tip which is selected/generated by himself. So, one time occasional success, i.e. he find a collision occasionally, may forge a validated bit signature or a link, but he cannot success always unless he finds an efficient inversion function or an efficient collision algorithm.

All of above, we notice, in SUHC, that the core and sole algorithm employed in our scheme is hash function. When hash function is constructed or deployed as a strong one-way function, finding a conversion function is computationally infeasible, so SUHC is secure. □

We also notice that SUHC is usually used for active certification. It means that certification activity is initiated by the certifier who owns hash chains and controls the whole authentication data, and an adversary can only eavesdrop communication passively unless he corrupts the host of the certifier. In other words, there are no opportunities for adversaries to request actively a certifier to sign any data constructed by them.

In addition to working in the same manner of CHC, our scheme has several special properties which enhance the security of CHC.

Property 1. *Fine-Authentication*

In our scheme, the granularity of authentication is finer: Intuitively, SUHC is a combined scheme involving hash chain and OTS so one of them occasionally failure may not mean the whole scheme is fail. For example, suppose that an adversary finds (fortunately) a number r (collision) such that $h^m(r)=h^n(s)$ where $0\le i\le m\le n\le k$, and i is the position of link being published. For CHC, an adversary can masquerade as a legitimate entity $(m\text{-}n)$ times. But for SUHC, the adversary cannot masquerade successfully because they cannot forge OTS keys, i.e. PK_i, which are authenticated by message authentication code (MAC) ζ_{i-1} in previous auth-frame. These OTS keys are “fresh” in every frame.

So, in CHC all links are authenticated by one value - "authentication root" - ω. In SUHC, in addition to ω, every link is also authenticated directionally by MAC in the previous auth-frame. Every MAC is "fresh" and independent since every OTS public/private key pair is different and randomly drawn every time. We call authentication in our scheme *fine- authentication*.

Property 2. *Proactive Recovery/Updating*

We call life of a hash chain is a vulnerable window. Said otherwise, a vulnerable window is a period in which a hash chain works in the manner of revealing links one by

one until all of links are spent. In a vulnerable window adversaries attempt to find secret seed of the hash chain or forge links. We notice a fact that a hash chain needs to be re-initialized frequently if its length is shorter; by the contrary, the security goes down if its length is longer – the vulnerable window is bigger.

In our scheme, the length of a SUHC equals the security parameter k of the corresponding hash chain. It means that the size of vulnerable window of SUHC is linear with k, which is relatively small.

For example, in our scheme, the length of a SUHC is 128 when MD5 be used and is 160 when SHA-1 be used. The security of SUHC is linear with its length. For example, using one time password [2] a user logs a network 4 ~ 5 times per day averagely for obtaining services or accessing resources (e.g. E-mail, FTP service). A hash chain will be exhausted in one month approximately. In this case, vulnerable window of SUHC is a month. So, a hash chain must be updated or re-initialized monthly. This kind of periodical updating is also called *proactive recovery/updating*. So, SUHC has *proactive security* property and can be deployed in distributed multi-party computation environment.

Property 3. *Updating Smoothly*

The re-initialization/updating process of hash chains in SUHC is smooth since it is embedded in authentication process of hash chains rather than an isolated operation, and there are no relations between any two secret seeds of hash chains in different vulnerable windows. Adversaries could not break updating phase separately because the whole value of the tip of a new chain is not exposed until the final link (seed) of current chain is exhausted, in other words, adversaries could not obtain the verification value of next session prematurely.

5.3 Availability

In practice, MD5, SHA-1[10] and SHA-256/384/512[11] are selective instances of hash functions. They are satisfied the one-way and collision-resistance in polynomial bound, so they can be deployed for SUHC. We say that they are applicationally or practically secure.

The core predicate is used for generating expected bit securely. It exits if strong one-way function exits [14]. Here we do not discuss how to construct it. In practical application it can be replaced with a simple function such as all or part bits of pre-image of hash function XOR, another alternative way is getting the least bit value of SK (which equals the desired value) drawn randomly.

5.4 Efficiency and Performance

The purpose of using OTS is that it is efficient since it is based on hash functions too. In SUHC, only a sort of cryptography algorithm – hash function is involved.

The length of an auth-frame in SUHC is more $3k$-bits (k is the output length of a hash function) than in CHC. In SUHC, an auth-frame (i.e. the i-th auth-frame) includes one core link $h^{k-i}(s)$ of a hash chain, one public key PK_i, one MAC ζ_i, and one private key SK_i. If MD5 is employed, the length of auth-frame is 4×16bytes=64bytes.

To construct an auth-frame, compared with CHC, SUHC needs one additional random number generation for generating a private key of OTS, and two hash computations: one for computing OTS public key and the other one for computing ζ_i.

SUHC needs very little additional memory to save OTS's private/public keys. For example, if MD5 is employed, SUHC needs 2×16bytes=32bytes to save a private key for next authentication and ω'.

Table 1 gives a comparison of SUHC, CHC, and RHC[7].

Table 1. SUHC v.s. CHC and RHC

	Initialization		Certification-verification			Updating/Re-init.	
	Comp.	Memo.	Comp.	Comm.	Memo.	Feature	Comm.
CHC	1kH	1M	1C	1M	1M	No	
RHC	5kH	(1+4k)M	1C	1M	(1+4k)M	Independent	4kM
SUHC	1kH	3M	1C+2H	4M	3M	Smooth	0

Remark. In table 1, H denotes a computation of a hash function. k denotes security parameter of hash functions. Assume that the bit-length of seed equals the length of output of hash function. M denotes store space of k bits (e.g. M=16bytes for MD5). C denotes average consumption of computing a link of a hash chain. We notice that there is C>>H. '*Comp.*' denotes computation. '*Comm.*' denotes communication.

It is obvious that our scheme does not increase more requirements of computation and storage than CHC and the total consumption is still small. So SUHC is suitable for employment in devices such as IC, mobile phones, PAD and so on nowadays.

6 Application - Server Supported Signature with SUHC

Asokan in paper [12] presented a delegated signing scheme – Server-Supported Signature S^3. It is based on one-way hash functions and traditional digital signature, and all signatures of users are done by third parties, called *signature servers*.

With SUHC, we proposed an improved S^3 scheme that does not achieve the same function of original S^3 but also has the properties of SUHC. Unlike the original S^3 needing two links of a hash chain per-authentication, our system split one auth-frame into two parts. The concrete protocol is depicted in Fig. 1.

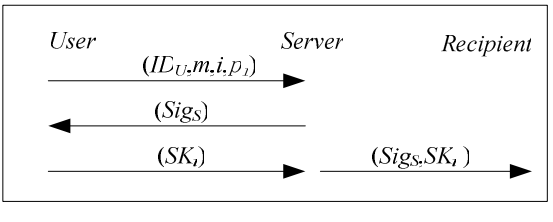


Fig. 1. Server-Supported Signature with SUHC

In Fig. 1, $p_i=(g_i,PK_i,\zeta_i)$ is user's i-th non-repudiation token (authorization certification), and Sig_S denotes signature on message (ID_U,m,i,p_i) with server's private key upon traditional signature algorithm.

In above protocol, *User* sends message (ID_U,m,i) to its signature *Server* along with current token p_i in the first protocol flow. *Server* verifies the *User*'s token (g_i,PK_i,ζ_i) then creates only one NRO (Non-repudiation of Origin) Sig_S for given (ID_U,m,i,p_i). *User* verifies received signature and then produces token which actually authenticates m, by revealing SK_i. Those tokens can be verified by *recipient*. In our scheme, *User* confirms *server*'s signature by revealing private key SK_i of OTS instead of consuming another link of the hash chain (, which is not forbidden). This method expands the property *fine-authentication* of SUHC. The third protocol flow SK_i is authenticated by PK_i in p_i. So the whole protocol S^3 with SUHC looks more smooth and fluent. SUHC enhances the security of S^3 since several important security properties *fine-authentication* and *proactive updating* are transfused into the original system.

7 Conclusions

The need for elegant methods for the secure re-initialization of hash chains was clear due to the 'limited-link' limitations most systems employing hash chains suffer from. In response, we have presented a novel scheme or a new kind of hash chain named SUHC. The most significant feature of SUHC is that it achieves a hash chain self-updating or re-initialization smoothly (by itself), securely (combined OTS for conveying bits) and efficiently (only several hash computations are involved). The updating process in SUHC does not need additional protocols or independent re-initialization processes and can be continued indefinitely to give rise to an infinite length hash chain or more precisely an infinite number of finite length hash chains tied together. So it improves the flexibility of hash chains applications.

SUHC offers several enhanced security properties: *fine-authentication*, *proactive updating* and *updating smoothly*, as well.

Compared with CHC, SUHC does not consume more additional computation and memory. These additional requirements in computation and memory are so low that devices like mobile phones and ICs can easily deploy SUHC.

In this paper, we give an application instance of SUHC in Server-Support Signature system - S^3 with SUHC, which looks more smooth and fluent. SUHC enhances the security properties of S^3 since some novel security properties of SUHC are transfused into the original system.

References

1. Lamport L.: Password Authentication with Insecure Communication. Communications of the ACM 24 (11), (1981) 770-772
2. Haller N., Metz C., Nesser P., Straw M.: A One-Time Password System. RFC 2289. (1998) Available from http://www.ietf.org
3. Asokan N., Tsudik G., Waidners M.: Server-supported signatures. Journal of Computer Security. (1997)

4. Micali S.: Efficient Certificate Revocation. Proceedings of RSA '97. U.S. Patent No. 5,666,416.
5. Ramkumar M., Memon N.: An Efficient Key Pre-distribution Scheme for MANET Security. IEEE Journal on Selected Areas of Communication. (2005)
6. Bicakci K., Baykal N.: Infinite Length Hash Chains and Their Applications. Proceedings of IEEE 11th International Workshops on Enabling Technologies (WETICE2002). (2002)
7. Goyal V.: How To Re-initialize a Hash Chain. http://eprint.iacr.org/2004/097.pdf. (2004)
8. Lamport L.: Constructing digital signatures from a one way function. Technical Report CSL-98. SRI International. (1979)
9. Merkle R.C.: A certified digital signature. In: Brassard G. (ed): Proceedings of the CRYPTO 89. Lecture Notes in Computer Science, Vol. 435. Springer-Verlag, Berlin (1990) 218–238
10. National Institute of Standards and Technology (NIST). Announcing the Secure Hash Standard, FIPS 180-1, U.S.Department of Commerce. (1995)
11. National Institute of Standards and Technology (NIST). Announcing the Secure Hash Standard, FIPS 180-2, U.S.Department of Commerce. (2002)
12. Asokan N., Tsudik G., Waidners M.: Server-supported signatures. Journal of Computer Security. (1997)
13. Ding X., Mazzocchi D., Tsudik G.: Experimenting with Server-Aided Signatures. Network and Distributed Systems Security Symposium (NDSS '02). (2002)
14. Oded G.: Foundations of Cryptography: Basic Tools. Cambridge University Press. (2001) 64-74

Appendix A

Theorem 5. The scheme OTS defined by definition 4 is secure.

Proof sketch. Suppose it's not. Then exist an adversary E which breaks OTS successfully, we will build an inverter A for the one-way function f upon output of E. On input y, A flips a random coin to get b. Suppose $b = 0$. Then A picks $x0$ at random, lets $y0 = f(x0)$, then runs E on the public key $(y0, y)$.

At some point, E may ask for a single signing query on some message m, for $m = 0$ or $m = 1$. If $m = 0$, A returns $x0$ in response to the query; else A aborts.

If A did not abort and E outputs a forgery, which must be for $m = 1$ (because E is required to forge on a new message), so A will learn σ such that $f(\sigma) = y$ and thus will invert f. Similar reasoning works for $b = 1$.

Note that if E succeeds with probability ϵ then A will succeed with probability $\epsilon/2$, because the view of E does not depend on A's choice of b – hence, the choice will be "lucky" (i.e., will match the query asked by E) half the time.

Modeling Web-Based Applications Quality: A Probabilistic Approach

Ghazwa Malak[1], Houari Sahraoui[1], Linda Badri[2], and Mourad Badri[2]

[1] DIRO, University of Montreal, Montreal, Qc, Canada
{rifighaz, sahraouh}@iro.umontreal.ca
[2] DMI, UQTR, Trois-Rivières, Qc, Canada
{linda.badri, mourad.badri}@uqtr.ca

Abstract. Quality assurance of Web-based applications is considered as a main concern. Many factors can affect their quality. Modeling and measuring these factors are by nature uncertain and subjective tasks. In addition, representing relationships between these factors is a complex task. In this paper, we propose an approach for modeling and supporting the assessment of Web-based applications quality. Our proposal is based on Bayesian Networks.

1 Introduction

Web-based applications are complex software systems that allow the user to create, publish, handle, and store data. Developing such applications in an ad hoc way may compromise their success and their viability. On the other hand, assuring their quality requires the use of sophisticated models. Nevertheless, in most cases, Web-based systems development lacks systematic approach and quality control [9].

Many authors have proposed guidelines [7], metrics and tools [6], methodologies and models [12] to assess the quality of Web sites or pages. Most of these proposals focused almost exclusively on Web applications usability aspects. In addition, several works proposed hierarchical quality models that are considered as oversimplified. Moreover, as Web-based applications are evolving software systems, they often yield uncertain and incomplete measurements [1], which major existing studies do not specifically address.

For Web applications, the quality is a multidimensional notion that involves a set of interdependent factors [3]. The majority of the studies recognize the importance of these factors, but some divergences exist for their definition, classification, and measurement [8]. Indeed, several quality factors are subjective [3]. In addition, modeling relationships that may exist between some factors is complex. From the measurement point of view, defining metrics for Web applications is difficult and often confusing [4]. These particular problems influence the objectivity of the assessment methodologies when evaluating the quality of Web applications.

To be efficient, a Web quality model must consider the inherent subjectivity, uncertainty and complexity of the quality factors, their measures and their relationships. Our objective is to propose a framework that considers specifically these properties. This can be done through probabilistic approaches, particularly using

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 398 – 404, 2006.

Bayesian Networks (BNs) [11]. Indeed, according to Baldi [1], the dynamic evolution of the Web is probabilistic in nature and probability methods apply to diverse areas of Web modeling. In addition to the handling of uncertainty, BNs offer a good support for causality relationships between factors.

The rest of the paper is organized as follows: Section 2 discusses why and how to define a Bayesian-based quality model for Web applications. Section 3 illustrates the application of the approach to the evaluation of the Navigability Design criterion. Section 4 concludes the paper and gives some future work directions.

2 A Probabilistic Approach to Model Web Applications Quality

In this work, we are interested in the nonfunctional aspects of quality as defined by ISO/IEC 9126 standard [5]. Considering the criteria for Web applications quality, when looking into the related work, many limitations are reported. First, some criteria are subjective [3, 12], and optimal values are often contradictory for many of them [6]. This makes it hard to define realistic threshold values. Second, balancing criteria is important w.r.t. the variety of application domains [8]. However, although necessary, sub-criteria weight assignment adds a new subjective dimension. Finally, the same criterion can affect simultaneously several criteria [8]. These interdependences are difficult to represent in a hierarchical way. Thus, selecting a particular grouping (hierarchy) means that some relationships has to be ignored.

Consequently, we propose a framework that addresses specifically the subjectivity in criteria evaluation, the uncertainty in the determination of the threshold values, the difficulty in balancing criteria, and the representation of interdependences between criteria. Both theory and experience have shown that probabilities are powerful tools for modeling uncertainty [1]. In the context of quality models, reasoning with probabilities allows to weight criteria and to handle uncertainty issues. Moreover, using graphical representation provides a naturally interesting interface by which we can model interacting sets of criteria [10]. This convey to the adoption of the probabilistic approach of Bayesian Networks (BNs) to build a quality model for Web applications. A BN model can be used to evaluate, predict and possibly optimize decisions when assessing Web applications.

Building a BN for a quality model can be done in two stages. The first one consists in constructing the graph structure. Criteria are considered as random variables and represent the nodes of the BN. Criteria affecting the same criterion should be independent variables. The second step deal with the definition of node probability tables (NPTs) for each node in the graph. A conditional probability function models the uncertain relationship between each node and its parents [11]. As usual, probability tables are built using a mixture of empirical data and expert judgments.

In our previous works [8], we attempted to collect Web applications quality criteria proposed by several authors. We extended the obtained list, refined it by applying GQM paradigm [2] and classified hierarchically retained criteria on the basis of the characteristics and sub-characteristics definitions in ISO 9126 standard [5]. Then, to apply the probabilistic approach, we represented the criteria gathered hierarchically in the form of a Bayesian Network. Considering the great number of criteria and sub criteria gathered, the resulting BN model is large and complex.

However, as explained in [11], a BN can be built starting from semantically meaningful units called network "fragments". A fragment is a set of related random variables that could be constructed and reasoned about separately from other fragments. They should formally respect the syntax and semantics of BNs. However, their use decreases the complexity when dealing with large models. Although our ultimate objective is to propose a comprehensive BN model for Web applications quality, we concentrate in this paper on the definition of navigability design fragment (at the page level). This will illustrate our approach.

3 Evaluation of the "Navigability Design" Criterion

Several works recognize the navigability design as an important quality criterion for Web applications [7, 12]. For some authors, the navigability design is a criterion of functionality [12], for others it characterizes usability [6, 7]. Authors propose many design elements, control points, directives, and guidelines to ensure the quality of navigability design. According to many definitions [7, 15, 16], navigability design in a Web application can be determined by: "the facility, for a given user, to recognize his position in the application and to locate and link, within a suitable time, required information. This can be done via the effective use of hyper links towards the destination pages". This criterion can be also assessed at the page level and be determined by the presence of some design elements and mechanisms that improve the navigability design.

3.1 Network Structure Construction

For the rest of this section, let's consider the *NavigabilityDesignP* as the variable representing the quality of the navigability design criterion at a Web page level, *Locate* as the variable representing the facility, for a given user, to know exactly in which page of the application he is and localize required information within the page, *Access* as the variable representing the facility, for a given user, to access to the required information in the destination page from the selected page, and *Revisit* as the variable representing the possibility, for a given user, to return to the selected page within a suitable time.

NavigabilityDesignP, Locate, Access, and *Revisit* are represented by four nodes (Fig. 1). The node *NavigabilityDesignP* is defined in terms of the three others. The direction of the edges indicates the direction of the definition/influence between criteria [11]. As the definitional relation is by nature uncertain, this structure will be completed later by probabilistic functions to state the degree to which parent nodes define the child node [11].

The obtained first structure is recursively refined. The same process is followed to construct the sub networks for *Locate, Access*, and *Revisit* nodes. Let's now discuss the *Locate* fragment. To ensure a good identification of "where I am", the presence of many design elements (as shown in fig. 2) can help user to recognize his position [7, 12]. Also, a good indication about the destination will help the user to determine his goal with less error [7]. Moreover, the presence of many navigational mechanisms supports the user in his information retrieval [7, 15]. Subsequently, the relationship

that exists between *Locate* and its parents is causal and not definitional as we have seen in figure 1. However, with 9 parents, a reorganization of this network is essential to avoid the combinatory explosion during the preparation of the probability tables. For that, we propose to group some nodes together when it is possible. The introduction of new nodes (meaningful or synthetic) gathering some parents' nodes and decreasing their number helps in the definition of probability tables. According to many definitions [7, 15] we can add the synthetic nodes by grouping some other nodes when the presence of each one of these nodes is independent from the other (fig. 3).

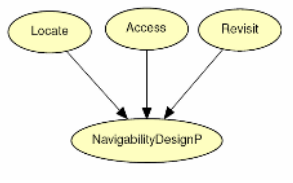


Fig. 1. BN sub network of NavigabilityDesignP criterion

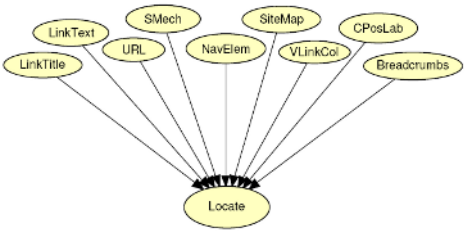


Fig. 2. BN sub network of Locate sub criterion

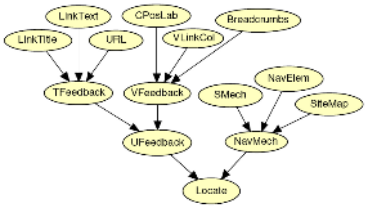


Fig. 3. Final 'Locate' BN

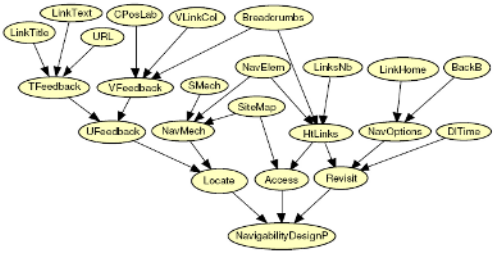


Fig. 4. Final Navigability design BN at Web page level

After constructing the sub networks for "Access" and "Revisit" nodes, all the fragments are put it together to obtain the BN of the Navigability design at a page level (Fig. 4).

3.2 Defining the Probability Tables for Navigability Design Nodes

Now, to define the probability tables for navigability design nodes, we consider two types of nodes in the BN of figure 4: intermediate and input nodes. Intermediate nodes of the BN are defined/influenced by their parents such as nodes *Locate* or *NavigabilityDesignP*. These nodes are not directly measurable and their probability distribution is determined by expert judgments. Accordingly, for each node C_c that has possible values $\{V_{c1}, \ldots V_{ck}, \ldots V_{cn}\}$ and has parents $\{C_{p1}, \ldots C_{pi}, \ldots C_{pm}\}$ *with*

possible values $\{V_{ci1}, \ldots V_{cij}, \ldots V_{cil}\}$, we need to define a table that gives the probability for all the possible value combinations

$$P(V_{ck} \left| V_{p1j}, \ldots, V_{pmj} \right.)$$

In an initial phase, the probability tables can be defined using expert judgments. They can be adjusted using automatic learning from data samples or from processed cases.

Input nodes of the BN are criteria considered as measurable variables that do not have parents. For the majority, these criteria take binary values (present or not). According to various studies [7, 15], their presence is recommended and contributes to improve the quality of the navigability design. The other input variables have measurable numerical values. As the number of possible values can be infinite, we have to transform them into discrete variables with a limited number of values. This is done to ease the definition of probabilities. According to [13], this transformation can be achieved using fuzzy logic. Indeed, the fuzzification process takes the different criterion values and replaces them with a set of functions that represent the degree of membership of each value to different fuzzy labels (usually , "High", "Medium", and "Low").

More concretely, the process of transforming crisp values into probabilities of criterion labels is defined as follows. First, we measure the criterion value for a large number of Web pages. We apply then a fuzzy clustering algorithm on them specifying the number of classes (2 or 3 classes). Figures 5 and 6 are examples of obtained clusters for the criteria *DlTime* and *LinksNb* with respectively 2 and 3 classes. The third step consists in defining cluster boundaries using an approximation method (drawing intersecting lines segments tangent to the curves of clusters as shown in figures 5 and 6). Finally, when using the BN for assessing the quality of a particular page, the measure of the input criteria is transformed into a set of probabilities corresponding each to a label/class. Note that as explained by Thomas in [14], the membership function degrees can be used as probabilities with the condition that both the fuzzy clustering algorithm and the approximation method preserve the condition that the sum of the membership degrees is always equal to 1.

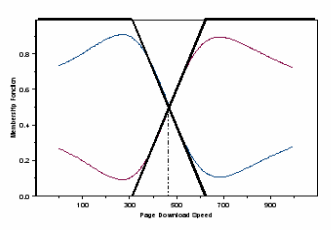


Fig. 5. Fuzzy clusters of "DlTime"

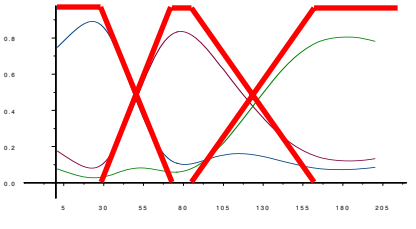


Fig. 6. Fuzzy clusters of "LinksNb"

3.3 Application Example and Preliminary Evaluation

The following example shows the feasibility of our approach using the partial BN of NavigabilityDesignP of Fig. 4. Values for input nodes are directly measured for a chosen page (in our example http://www.pbs.org/). In the present application context,

probability value calculated for a measured criterion is known as "evidence". This evidence propagates through the BN via the definitional/causal links, resulting in updated probabilities for other criteria. Although the quality of NavigabilityDesignP is not known with certainty, it is evaluated in our case as good with a probability of 76.70 % good (fig.7).

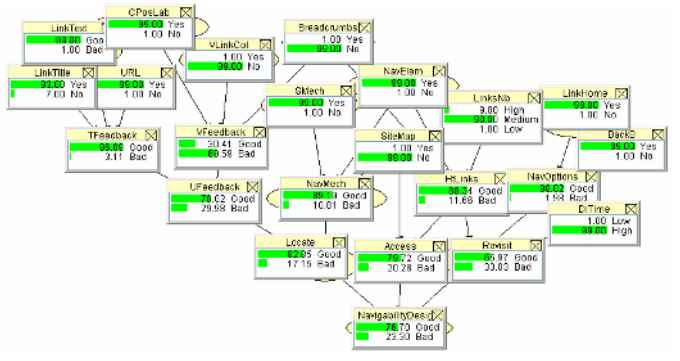


Fig. 7. State of BN probabilities showing the Navigability design quality for the PBS page

In a first experimentation to evaluate our approach, we selected two groups of web pages. The first group is composed of web pages recognized for their good quality (from *Top 100 of Webby Awards*) while the second one contains pages with poor quality (from *Worst of the Web or Webby Worthy*). We applied our BN on each page of each group and collected the probability of "good" navigability design. The idea of the evaluation is that our assessment must not contradict the status of each group.

As shown in table 1, we obtained a good score for the pages of the first group (*Webby Awards*) and low scores for the ones of the second group (*Worst of the Web or Webby worthy*). This score give a first indication that the selected and evaluated criteria are relevant.

During this experience, we used the ability of BNs to generate "What if" scenarios, i.e. giving a fixed value for one or more variables. Then, when considering high speed connection, the quality of the Navigability Design has significantly increased for many pages.

Table 1. Partial results for a rapid validation

Web applications	**Navigability design quality at page level**	**Navigability design quality at page level if High Speed Connection**	**Web applications**	**Navigability design quality at page level**	**Navigability design quality at page level if High Speed Connection**
Winner of *Webby Awards*	85.44 %	85.44 %	The *Worst of the Web or The Webby Worthy*	59.24 %	67.84 %
	85.09 %	90.68 %		51.22 %	58.26 %
	80.96 %	89.40 %		74.33 %	74.33 %
	84.83 %	92.81 %		57.88 %	57.88 %

4 Conclusions and Future Work

This paper proposes a general framework that uses Bayesian networks to support the quantitative assessment of Web applications quality. We illustrate our methodology for a BN "fragment", corresponding to the Navigability Design criterion, to demonstrate the feasibility of our model. A BN graph was constructed for the considered criterion. Many experiences, involving different Web pages, were conducted using this BN. A rapid validation of the proposed approach demonstrated its relevancy. The proposed framework is extensible and adaptable. It can be re-used for specific cases to assess a particular criterion, a super criterion, a sub characteristic, a characteristic or the whole quality of a Web application. This can be done for one page, for specific pages or for all the Web application. We believe that our proposal is for web quality evaluation a good alternative to the classical hierarchical models. As future work, we plan to extend the defined network to cover more quality characteristics, to complete a global evaluation for Web applications usability, and to validate empirically the proposed model using a large scale controlled experience.

References

1. Baldi, P., Frasconi, P., Smyth, P.: Modeling the Internet and the Web; Probabilistic Methods and Algorithms. Wiley (2003)
2. Basili, V.R., Caldiera, G., Rombach, H.D. : The Goal Question Metric Approach. (1996)
3. Brajnik, G.: Towards Valid Quality Models for Websites. Proceedings of the 7th Conference on Human Factors and the Web (2001)
4. Calero, C.; Ruiz, J.; Piattini, M.: A Web Metrics Survey Using WQM. Proceedings of the International Conference on Web Engineering (2004)
5. ISO/IEC (2001) ISO/IEC 9126: Quality characteristics and Guidelines for their use (2001)
6. Ivory, M.: An empirical foundation for automated Web interface evaluation. Thesis (2001)
7. Koyani, S. J., Bailey, R. W., and Nall, J. R.: Research-Based Web Design & Usability Guidelines. National Institutes of Health (2003)
8. Malak G., Badri L., Badri M., Sahraoui H.: Towards a Multidimensional Model for Web-Based Applications Quality Assessment. Proc. of the fifth I. C. E-Commerce and Web Technologies (EC-Web'04), Spain, LNCS Vol. 3182. Springer-Verlag, (2004) 316-327
9. Muruguesan, S., Deshpande, Y., Hansen, S., Ginige, A.: Web Engineering: A New Discipline for Development of Web-based Systems
10. Naïm, P., Wuillemin, P.H., Leray, P., Pourret, O., Becker, A.: Réseaux Bayésiens. (2004)
11. Neil, M., Fenton, N.E., Nielsen, L.: Building large-scale Bayesian Networks. The Knowledge Engineering Review, 15(3). (2000) 257-284
12. Olsina, L. Rossi, G.: Measuring Web Application Quality with WebQEM. IEEE MultiMedia, Vol. 9, No. 4 (2002)
13. Sahraoui, H., Boukadoum, M., Chawiche, H. M., Mai, G. Serhani, M. A, A fuzzy logic framework to improve the performance and interpretation of rule-based quality prediction models for object-oriented software. The 26th (COMPSAC'02). Oxford (2002)
14. Thomas, S.F.: Possibilistic uncertainty and statistical inference. ORSA/TIMS Meeting. Houston, Texas (1981)
15. W3C Recommendation 5-May-1999, Web Content Accessibility Guidelines 1.0.
16. Zhang, Y., Zhu, H., Greenwood, S.: Website Complexity Metrics for Measuring Navigability. The Fourth International Conference on Quality Software (QSIC'04) (2004)

Monitoring Interactivity in a Web-Based Community

Chima Adiele and Wesley Penner

University of Lethbridge
Lethbridge, Alberta, Canada
{chima.adiele, wesley.penner}@uleth.ca

Abstract. Over the years, Web-based communities (WBC) have evolved from social phenomena that have no business dimension to a business enabler in the virtual marketplace. WBC becomes economically interesting with the increasing size of the community where members are active participants in sharing ideas and knowledge. There are, however, inherent problems associated with the increasing size of a WBC, especially determining the contributions of members in sustaining the community's interactivity. Interactivity relates to the level of participation of a member in a given community, and the usefulness of such contributions to the needs of the community. In this paper, we present an interactivity model that captures the contributions of members of the community and also determines the community's interactivity level. We use simulation results to validate the correctness of our model.

Keywords. Web-based community, interactivity model, e-commerce and simulation.

1 Introduction

A Web-based community (WBC) is a Web-enabled communication and social interaction between a group of people that have common interests. Web-based community is also referred to as e-community, virtual community, or online community in the literature. In one of the earliest work in WBCs, Rheingold [1] envisions a WBC as social phenomena that has no business dimension. Advances in information and communication technologies, however, have given impetus to WBCs as a business enabler in the digital marketplace. For example, Hagel and Armstrong [2] observe that successful organizations take advantage of virtual communities to create interactions among consumers or suppliers of a company's product. Such a community enables the sharing of experiences, problems and solutions through established discussions, and sense of community among members [3]. Blanchard and Markus [4] argue that "the success of community support platforms depends on the active participation of a significant percentage of the community members". Community participation is necessary for sustained interactivity.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 405–410, 2006.

Participation can be in the form of collecting and indirectly sharing information, and by directly communicating with one another. Interactivity is the level of participation of a member in a given community, and the usefulness of such contributions to the needs of the community. Measuring the level of participation of members of a given WBC and indeed the interactivity level of the community is necessary to determine members' contributions and how to sustain the community. In this paper, we design an interactivity model to measure the contributions of members of a given WBC, and hence determine the community interactivity level. We provide simulation results to validate the correctness of our model.

The remaining part of this paper is organized as follows. In Section 2, we present the interactivity model and explain the interactivity life cycle. We present simulation set up, results and discussions in Section 3. Finally, Section 4 concludes the paper and provides insight into future work.

2 Modelling Interactivity

This definition refers to the degree of responsiveness of messages and how they relate to previous messages. Fiore *et al.* [5] identified a broad activity set for measuring interactivity. The set of activities include, authors, repliers, initiators, returning authors, posts, replies, thread starts (initial turns which received replies), barren posts (initial turns which received no replies), cross-posts, and cross-post targets (distinct groups with which this one shared messages). In some other studies [5], [6], a subset of the activities identified by Fiore *et al.* were used to discuss interactivity. The model we present provides a standard to measure interactivity in different communities regardless of the category of the WBC.

To measure interactivity in a given WBC, we have to capture the set of activities that are used to generate messages and their levels of importance to the given community.

Let a_i be the ith activity in a set of activities A and w_i be the corresponding weight of a_i, Then, we define the interactivity of a community over a given time window, $\overline{W_i}$ as (nA is the number of activities):

$$\frac{\sum_{i=1}^{\overline{W_i}} \sum_{j=1}^{n} a_j \cdot w_j}{nA} \quad . \tag{1}$$

To simplify our exposition, we restrict the number of activities to include:

1. Number of reads (R) (the number of messages, $message_k$, read by a member, m_j, during a given time window, $\overline{W_i}$)
2. Number of start posts (sP) (the number of messages, $message_k$, a member, m_j, initiates during a given time window, $\overline{W_i}$),
3. Number of reply posts (rP) (the number of replies to messages, $message_k$, a member, m_j, generates during a given time window, $\overline{W_i}$),
4. Number of replies (Res) a sP generates during a given time window, $\overline{W_i}$.

We assign weights to activities to reflect their importance to the community. For example, a member can choose to read (R) all the messages posted in the community without responding to none. We, therefore, assign the weight γ to R to reflect its relative importance in contributing to the community's interactivity. Similarly, start post (sP) is a way of posting messages in the community. However, a sP is relevant to the extent that it relates to existing messages in the community. We assign the weight α to sP. The number of responses (Res) a sP generates contributes to the value of the sP. Therefore, reply to a message counts in two ways: first, we assign the weight β to the member who generates the reply (rP); second, we assign the weight α' to the member who initiates the sP that is being replied to (Res). Observe that the effective weight of $sP = \alpha + \alpha'$.

Let tM be the total number of messages generated in a WBC for a given time window W_i;

$$tM = sP + rP \ . \tag{2}$$

To reduce redundancy in the system and enhance efficiency, we assume that a message cannot be read more than once by any member. The implication of this assumption is that regardless of the number of times Therefore, we have that an individual member's read is constrained to $R \leq tM$.

Let tAW be the total activity weights. Then,

$$tAW = \sum_{j=1}^{n}(w_j) = 1 \ . \tag{3}$$

We compute individual daily interactivity, I_{DI}, for an initial $\overline{W_i^\circ}$ as:

$$I_{DI} = \frac{((\alpha \cdot sP + \alpha'(Res \times sP)) + \beta \cdot rP + \gamma \cdot R)}{nA} \ . \tag{4}$$

Similarly, we compute the community daily interactivity. Let CS be the size of the community with members m_j (where $1 < j \leq CS$) then, community daily interactivity C_{DI} is given by:

$$C_{DI} = \frac{\sum_{j=1}^{CS}((\alpha \cdot sP_j + \alpha'(Res_j \times sP_j)) + \beta \cdot rP_j + \gamma \cdot R_j)}{nA} \ . \tag{5}$$

To measure individual interactivity of a member, m_j for a time window $\overline{W_i}$ (where $\overline{W_i} > \overline{W_i^\circ}$), we compute the individual daily interactivity over the width of $\overline{W_i}$. Accordingly, individual interactivity for $\overline{W_i}$, $I_{\overline{W_i}}$ is given by:

$$I_{\overline{W_i}} = \frac{\sum_{i=1}^{\overline{W_i}}((\alpha \cdot sP + \alpha'(Res \times sP)) + \beta \cdot rP + \gamma \cdot R)}{nA \cdot \overline{W_i}} \ . \tag{6}$$

Similarly, for a WBC with members m_j and a common time window $\overline{W_i}$, the community interactivity for $\overline{W_i}$ is given by:

$$C_{\overline{W_i}} = \frac{\sum_{i=1}^{\overline{W_i}}\sum_{j=1}^{CS}((\alpha \cdot sP_j + \alpha'(Res_j \times sP_j)) + \beta \cdot rP_j + \gamma \cdot R_j)}{nA \cdot \overline{W_i}} \ . \tag{7}$$

In section 3, we present simulation results to validate the correctness of our model.

3 Experimentation and Discussion

We simulated the interactivity model using a discrete event simulator with members' participation and behaviours modelled using a Poisson random process. Initially, the size of each community was set to 200 members.

The community was divided into three groups:

1. Userbase × 15% = leaders (eg: 200 × .15 = 30)
2. UserBase × 50% = active members (eg: 200 × .50 = 100)
3. UserBase × 35% = non-active members (eg: 200 × .35 = 70)

We initialize the weights w_i of activities a_i used in our model ($\gamma = 0$, $\alpha = 0.2$, $\alpha' = 0.25$, and $\beta = 0.45$). Notice that this initialization satisfies Equation (3). For simplicity, we assume that the weights of these activities do not change for the duration of the simulation time.

Let tM for a time window of $\overline{W_i^\circ} = 500$. We assume that non-active members only read, and don't post anything, leaders post 75%, and active members the remaining 25% [7], [8]. The total number of posts per member per day was then computed by generating a random integer in the range [0, Max] (where Max is the group's daily maximum). The number of start posts was derived using a random generator in the range of [0, MTDP] (where MTDP is a member's total daily posts).

The number of reads was further constrained to 30% for leaders and non-active members, and 35% for average members. Non-active members read more sporadically than they write because they are not committed to the community. The number of replies was randomly generated in the range of [0, rMax] (where Rmax is the maximum number of replies). If the system determined that the user did not create a start post, replies was set to zero.

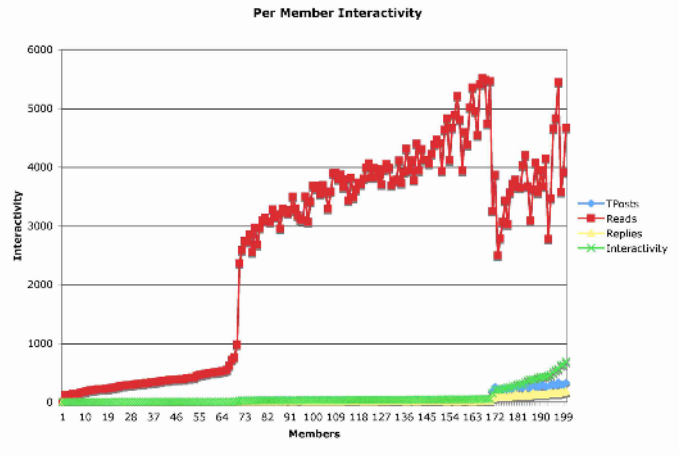


Fig. 1. Member interactivity for a standard community

Results. Figure 1 shows the interactivity of members of a WBC. The community has 200 members, and the graph is arranged in ascending order of individual member's interactivity score. There is little or no interactivity for non-active members, while active members have relatively more interactivity. On the contrary, leaders that constitute the final 15% of membership have significantly high interactivity. The graph rises quite sharply indicating the substantial contribution leaders make. Observe that the value of reads for most members were high, yet it had only slight effect on interactivity. Read on its own does not constitute interactivity.

4 Conclusions

In this paper, we designed an interactivity model to measure the interactivity of members of a WBC. We have shown through simulations that a member's contribution, and indeed a community's performance in a WBC can be measured. There are many benefits in computing members' interactivity levels, including identifying specific leaders and loafers in a given community. The concept of social loafing is crucial to the survival of a WBC.

Our model can also monitor the dynamics of a WBC and show how community dynamics affects the interactivity level of the community. We can take advantage of this metric to design promotional techniques that are tailored towards individual member's needs. In the future, we plan to measure the effects of different promotional activities to determine their efficacy. Another interesting area of extension is to determine the effects of measuring the quality of a post and its rating on interactivity.

Acknowledgments. Funding for this research is provided in part by the University of Lethbridge and SCP research grants.

References

1. Rheingold, H.: The Virtual Community: Homesteading on the Electronic Frontier. Revised edition edn. MIT Press (2000)
2. Hagel, J., Armstrong, A.: Net Gain: Expanding Markets Through Virtual Communities. Harvard Business School Press (1997)
3. Bakos, Y.: The emerging role of electronic marketplaces on the internet. Communications of the ACM **41** (1998) 35–42
4. Blanchard, A.L., Markus, M.L.: Sense of virtual community: Maintaining the experience of belonging. In: Proceedings of the Hawaii 35th International Conference on System Sciences (HICSS-3502). (2002)
5. Fiore, A.T., Tiernan, S.L., Smith, M.A.: Observed behavior and perceived value of authors in usenet newsgroups: bridging the gap. In: CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM Press (2002) 323–330
6. Teeni, D., Schwarz, A.: Communication in the is community: A call for research and design. Communications of the Association for Information Systems **13** (2004) 521–543

7. Takahashi, M., Fujimoto, M., Yamasaki, N.: The active lurker: influence of an in-house online community on its outside environment. In: GROUP '03: Proceedings of the 2003 international ACM SIGGROUP conference on Supporting group work, New York, NY, USA, ACM Press (2003) 1–10
8. Brush, A.B., Wang, X., Turner, T.C., Smith, M.A.: Assessing differential usage of usenet social accounting meta-data. In: CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM Press (2005) 889–898

A Metamodel-Based Approach for Extracting Ontological Semantics from UML Models

Hong-Seok Na[1], O-Hoon Choi[2], and Jung-Eun Lim[2]

[1] Dept. of Information and Computer Science, Korea Digital University
[2] Dept. of Computer Science and Engineering, Korea University
hsna99@kdu.edu, {pens, jelim}@software.korea.ac.kr

Abstract. UML has been a standard language for domain modeling and application system design for about a decade. Since UML models include domain knowledge in themselves, which was verified by domain experts, it is helpful to use these models as when we start to construct domain ontology. In this paper, we propose a method for extracting ontological concepts, properties, restrictions and instances from UML models. We compare the UML metamodel with the OWL metamodel, and define transformation rules for constructing OWL-encoded ontology. We expect that the generated ontology plays a role of an early stage model for ontology development.

1 Introduction

Ontology defines a common set of concepts and relations that are used to describe and represent domain knowledge [1]. It includes machine-interpretable definitions of basic concepts in the domain and relations among them. We can use ontologies to share common understanding of the structure of information among people or software agents, to enable reuse of domain knowledge and make domain assumptions explicit.

However, creating ontologies is a difficult and time-consuming process that involves specialists from several fields [3]. Especially, one of the major impediments is the difficulty of defining base concepts in the domain [5].

UML has been a standard modeling language for domain modeling and application system design since it was announced as a OMG specification. Most legacy UML models include core domain knowledge in themselves, which was verified by domain experts already, thus it is possible and meaningful to use these models as a base for domain ontology building. This paper proposes a method for domain ontology building by extracting ontological concepts, properties, instances and restrictions from UML models (especially UML structure diagrams) designed previously.

In order to construct the ontological knowledge from legacy UML models, we proposed a method to extract ontological semantics from UML structure diagrams. We classified UML model elements (defined in UML metamodel) with respect to ontology metamodel and defined transformation rules between UML and OWL model elements.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 411 – 422, 2006.

In section 2, we discuss related approaches on extracting ontologies from legacy models. The UML metamodel and OWL metamodel are explained in section 3. We define transformation rules and ontology construction processes in section 4. The transformation process and the results are shown in section 5. Finally, in section 6, we conclude with a summary of this paper and the direction for further research.

2 Related Works

In most system development projects, analysis and modeling tasks on the problem domain are essential for efficient and complete system implementation. UML is the representative language for system modeling about the basic concepts, structures and behaviors of the system. It is possible and meaningful to use UML models as an ontological basis. There are several approaches to convert legacy models to ontologies with such languages as OWL, DL and DAML+OIL.

Robert M. Comb [8] proposed a DL centered method on conceptual mapping among UML, ER and OWL. He compared mapping elements of each languages as table 1, and defined function prototypes for model converting using QVT (Query, View, Transformation) model. However, all models should be converted to DL at the first step, it is inevitable to loss some semantic information. For example, the association and the attribute of UML are semantically different elements, but both elements are mapped to the role of DL, and then transformed to the property of OWL.

Table 1. Mapping Elements of Each Models

Model Element	UML	ER	OWL	DL
Object	Class	Entity	Class	Concept
Property	Association	Relationship	Property (Object, Datatype)	Role
	Attribute	Attribute		

Java2OWL [9] suggests a transforming direction from UML to OWL. This program transforms java classes and instances to OWL using XML schema data types and cardinality restrictions. Falkovych [10] proposes a transformation method from UML class models to DAML+OIL. He classifies UML association to detailed relationships to represent the semantics of the association more accurately.

Dragan [11] used UML notations for ontology definition. He proposed the Ontology UML Profile and a transformation method from OUP-based ontology to OWL ontology. OUP defines it's own terminologies and stereotypes for ontology representation, so we can not apply the method to pure UML models.

As shown above, in order to extract domain knowledge in legacy (UML) models m ore completely without semantic loss, we have to analyze it at the metamodel level, e xtract the ontological elements in UML models and reconstruct them according to the ontology metamodel.

3 Metamodel of UML and OWL

3.1 UML Metamodel

UML is a language with a very broad scope that covers a large and diverse set of application domains. A UML model consists of elements such as packages, classes, and associations [2].

There are two major diagram types in UML – structure diagrams and behavior diagrams. Structure diagrams show the static structure of the objects in a system. Elements in a structure diagram represent the meaningful concepts of an application, and may include abstract, real-world and implementation concepts. Behavior diagrams show the dynamic behavior of the objects in a system, including their methods, collaborations, activities, and state histories.

As we focus on the static and structural aspect of UML models, we consider the UML structure diagrams as main resources of extracting ontological elements. There are six diagrams in UML structure diagrams – class, component, object, composite structure, deployment, and package diagram. Fig. 1 shows a class diagram and component diagram on university domain.

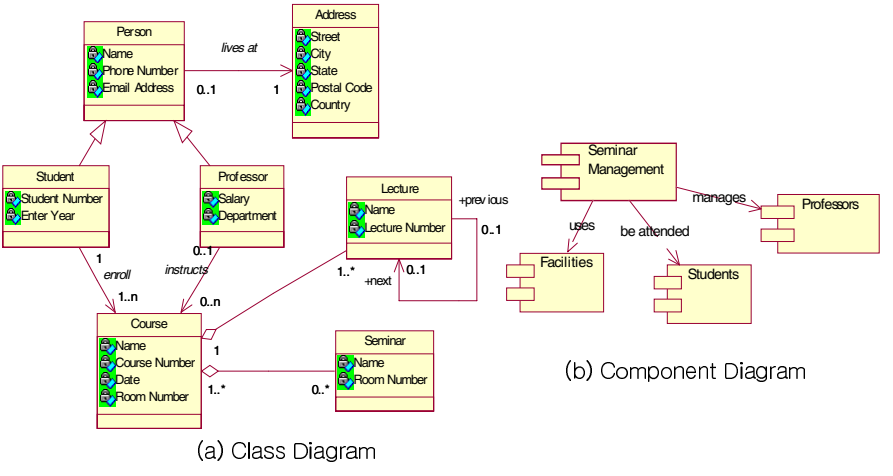


Fig. 1. Examples of UML Structure Diagram

Each diagram type has at least one metamodel which contains necessary elements and their relationships for the diagram. Fig. 2 shows the metamodel of UML class diagram. UML Class is a kind of classifier whose features are attributes and operations. Attributes of a class are represented by instances of property that are owned by the class. Some of these attributes may represent the navigable ends of binary associations.

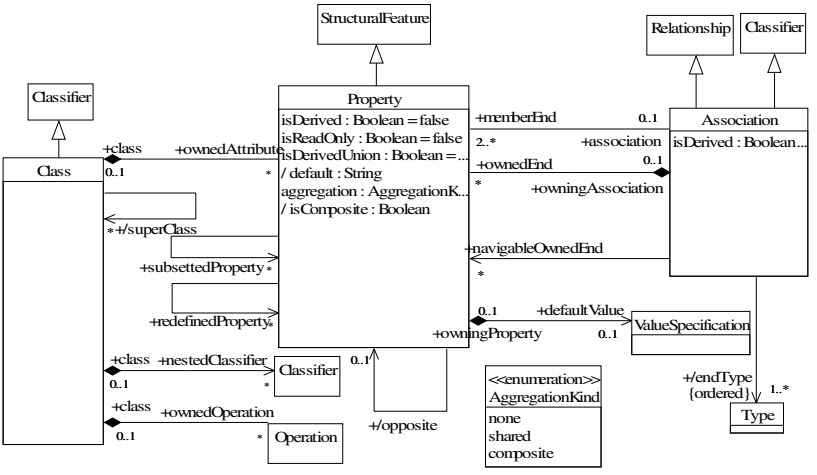


Fig. 2. Classes diagram of the Kernel package in UML2

We can derive UML model elements from the metamodel, and list them on the table 2. These model elements can be used for different use according to the contexts, so the corresponding elements of OWL should be determined based on them.

Table 2. Model Elements of UML Structure Diagrams

Diagrams	Model Elements
Class	Class, Association, Generalization, Aggregation, Composition, Dependency, OwnedAttribute, Type, Enumeration, Navigable, Non-navigable, Disjoint, Cover, Multiplicity, Realization
Composite Structure	Part, Port, Collaboration, CollaborationUse, Connector, Role binding
Component	Component, Interface, Realization, Interface Realization, Usage Dependencies, Class, Artifact, Port
Deployment	Artifact, Node, Deployment specification, Association, Dependency, Generalization, Deployment, Manifestation
Object	InstanceSpecification, Link(i.e. Assocation)
Package	Package, PackageExtension, PackageImport, Dependency

3.2 Ontology Metamodel

The term ontology has been used in several disciplines, from philosophy to knowledge engineering, where ontology is considered as a computational entity, containing concepts, relationships between concepts, instances of the concepts and

constraints [4]. We define the ontology in the following way to make a reference model for comparing the features in UML and OWL.

Definition 1. An ontology is a tuple O := (C, P, I, R), where :

1. C = {c_1, c_2, …, c_n }is a set of concepts(classes in OWL), where each concept c_i refers to a set of real world objects(instrances).
2. P = {p_1, p_2, …, p_n} is a set of properties, which can be divided into attribute and relations. They are usually first-class objects. That is, a property can exist without being attached to a particular class.
3. I is a set of instances. Instances are individual members of concepts (classes).
4. R is a set of restrictions. Each property has a set of restriction on its values, such as cardinality and range.

OWL [6] is a semantic markup language for publishing and sharing ontologies on the web, and it is a language extension of RDF Schema. OWL provides three increasingly expressive sublanguages designed for use by specific communities of users and implementers.

ODM (Ontology Definition Metamodel) of OMG defines OWL metamodel as a MOF2 compliant metamodel that allows a user to define ontology models using the same terminology and concepts as those defined in OWL [7]. The OWL metamodel includes seven diagrams for classes, restrictions, properties, individuals, datatypes, utilities and ontology. Among the seven diagrams in OWL metamodel, we focus on classes, properties, individuals and restrictions diagrams.

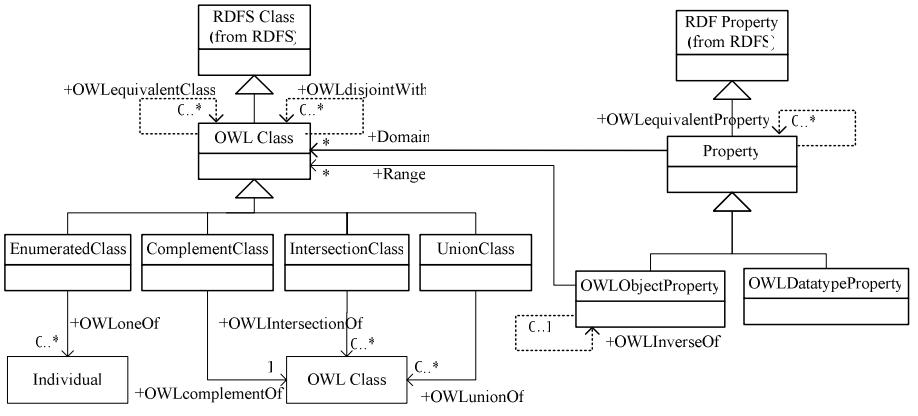


Fig. 3. The Classes and Properties Diagram of the OWL Metamodel

Fig. 3 shows parts of the classes diagram and the property diagram of the OWL Metamodel. In contrast to UML, OWL DL does not only allow to define simple named classes. Instead, classes can be formed using a number of classes, class restrictions, and enumerated classes. The EnumeratedClass is defined through a direct enumeration of named individuals. Also, OWL class can be defined through OWLcomplementOf, OWLintersectionOf, and OWLUnionOf.

Properties represent named binary associations in the modeled knowledge domain. OWL distinguishes two kinds of properties – object properties and datatype

properties. A common generalization of them is given by the abstract metaclass property. Properties can be functional and their domain is always a class. Object properties may additionally be inverse functional, transitive, symmetric, or inverse to another property. Their range is a class, while the range of datatype properties is datatypes.

An Individual is an instantiation of a Class and is the subject of a PropertyValue, which instantiates a Property. Naturally, an ObjectPropertyValue relates its subject with another Individual whilst a DatatypePropertyValue relates its subject with a DataValue, which is an instance of a Datatype [1].

In this paper, we will compare UML and OWL metamodel in the respect of this ontology model, and propose a method to extracting ontological elements from UML models.

4 OWL Ontology Building from UML Models

4.1 Extracting Ontological Semantics

Both UML and OWL are a kind of language for organizing knowledge, and there are some semantic similarities between them. For example, UML classes, components, and artifacts are the important candidates for ontology concepts (classes) in OWL. Associations and generalizations are the representative candidates for ontology properties which imply binary relations between classes. Including these elements, many of UML elements can be transformed to OWL directly. In table 3, we classified UML model elements into 4 categories according to definition 1- concept, property, restriction and instance.

Table 3. The Correspondence between Ontology and UML

Ontology	UML Model Elements
Concept	Class, Enumeration, Component, Part
Property	Association, Generalization, Aggregation, Composition, Dependency, OwnedAttribute, Link
Restriction	Navigable, Non-navigable, Disjoint, Cover, Mutilplicity
Instance	InstanceSpectification

In order to build ontologies of high completeness, it is necessary to define the individuals associated to the OWL concepts (classes and properties). However, except for object diagrams, the instances of the class in UML exist out of the models, thus additional researches on extract individuals from operational data are also needed.

There is no one correct way to model a domain. However, there are generally accepted guidelines for ontology development. Natalya et al.[13] provided a practical guideline to develop a domain ontology – defining classes in the ontology, arranging the classes in a taxonomic hierarchy, defining slots, and filling in the values for slots

for instances. In the following sections and subsections, we will propose some transformation rules and explain them with the example diagrams in Fig. 1.

4.2 Constructing OWL Classes

In UML metamodel, the most highest class is (model) element. If the type of a model element is class or component or part, then the model element corresponds to owl:Class in OWL elements

Rule 1. (Class) An OWL Class C can be created on a UML model element ME, if ME is class, component or part.

For example, we can find 7 classes and 4 components in Fig.1, and create 8 OWL clas ses named 'Person', 'Student', 'Professor', 'Address', 'Course', 'Lecture', 'Seminar' and 'Facilities', removing duplicated classes. We can translate UML class 'Person' into OWL expression <owl:Class rdf:ID="Person"/>, UML component 'Facilities' int o <owl:Class rdf:ID="Facilities"/> and so on.

4.3 Relating Classes with OWL Properties

OWL distinguishes two kinds of properties – object properties and datatype properties. Their domain is always a class. The range of a object property is a class, while the range of datatype properties is a datatype.

Rule 2. An OWL datatype property $P_{datatype}$ can be created on the UML model element ME, if ME is ownedAttribute and its type is primitive datatype.

Rule 3. An OWL object property P_{object} can be created on the UML model element ME, if ME is ownedAttribute and its type is class datatype.

The ownedAttribute with primitive type is mapped to the owl:DatatypeProperty with domain and range properties of RDF schema. For example, the attribute "Name" of Person class in Fig. 1 is represented as follows.

```
<owl:DatatypeProperty rdf:ID="Person Name">
  <rdfs:domain rdf:resource="#Person"/>
  <rdfs:range  rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
```

The attribute previous in Lecture class has class datatype. So, this attribute corresponds to object properties.

```
<owl:ObjectProperty rdf:ID="Lecture Previous">
    <rdfs:domain rdf:resource="#Lecture"/>
    <rdfs:range  rdf:resource="#Lecture"/>
</owl:ObjectProperty>
```

Rule 4. An OWL object property P_{object} can be created on the UML model element ME, if ME is association.

UML association represents relationship between classes. If a property is part of the memberEnds of an Association and the type of the property is class, then the property is mapped to owl:ObjectProperty. In this case, rdfs:domain and rdfs:range indicate the classes related to the association.

```
<owl:ObjectProperty rdf:ID="Lives at">
    <rdfs:domain rdf:resource="#Person"/>
    <rdfs:range  rdf:resource="#Address"/>
</owl:ObjectProperty>
```

Aggregation and composition represent the part-whole relationships in UML. Since OWL doesn't support this type of property, we need use additional constraints for the se type of relation. Further details are in section 4.4.

Rule 5. An OWL subclass relation $P_{subclass}$ can be created on the UML model element ME, if ME is generalization, and its memberEnds are classes.

Rule 6. An OWL subproperty relation $P_{subproperty}$ can be created on the UML model element ME, if ME is generalization, and its memberEnds are association class.

A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. UML and OWL both support generalization. The objects of inheritance in UML are only classes and associations. On the other hand, OWL suppo rts class inheritance and property inheritance separately. Thus, class inheritance in UML is mapped to the rdfs:subClassOf element of OWL, and association inheritance i s mapped to the rdfs:subPropertyOf element as follows.

```
<owl:Class rdf:ID="Student">
  <rdfs:subClassOf rdf:resource="#Person" />
  ...
</owl:Class>
<owl:Class rdf:ID="Professor">
  <rdfs:subClassOf rdf:resource="#Person" />
  ...
</owl:Class>
```

4.4 Adding OWL Restrictions

The UML constraints such as cardinalities and conditions are represented using owl:Restriction in the part of OWL class definitions. If the constraints are related to a property, the property should be shown in the OWL restriction. The following example shows the OWL definition of the class person with a cardinality constraint (owl:minCardinality and owl:maxCardinality).

```
<owl:Class rdf:ID="Person">
   <rdfs:subClassOf>
   <owl:Restriction>
     <owl:onProperty rdf:resource="#Lives at "/>
```

```
    <owl:minCardinality rdf:datatype="&xsd; Integer " > 0</owl:minCardinality>
    <owl:maxCardinality rdf:datatype="&xsd; Integer " > 1</owl:maxCardinality>
   </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Rule 7. An OWL functional property $P_{functional}$ can be created on the UML model element ME, if ME is association, and the multiplicity is less than 1.

An OWL property can be globally declared as functional or inverse functional. A functional property has a maximum cardinality of 1 on its range, while an inverse functional property has a maximum cardinality of 1 on its domain [7]. Thus, if an attribute plays a role of identifier, it will be represented as a functional and inverse functional datatype property with cardinality restriction set to one. Imposing functional and inverse functional property characteristics on a property in OWL makes sure that the domain and range values share a one-to-one relationship [6].

4.5 Populating OWL Individuals

The instances of UML class are real objects in system, and correspond to individuals of OWL. In OWL, all classes should inherit the top class named “Thing”, thus an instance of class "Course" can be represented as follows. In this example, we suppose the identifier of the instance is “CS0001”.

```
<owl:Thing rdf:ID="CS0001" />
  <owl:Thing rdf:about="#CS0001">
  <rdf:type rdf:resource="#Course"/>
</owl:Thing>
```

Where, the rdf:type, which is a RDF property, represents that the OWL individual is a member of the class of the rdf:resource property.

4.6 Other Considerations

Beside the generalization, UML supports the specialized part-of relationship between classes. The representatives are aggregation and composition. Also, the composite structure diagrams uses the part elements which specify runtime instances of classes collaborating via connectors. They are used to hierarchically decompose a class into its internal structure which allows a complex object to be broken down into parts.

Since OWL doesnot support this type of relations, to represent the part-whole type association, we need additional properties for the association. For example, we can define OWL properties from the course-lecture association by rule 3.

```
<owl:ObjectProperty rdf:ID="is a part of">
  <rdfs:domain rdf:resource="#Lecture"/>
  <rdfs:range rdf:resource="#Course"/>
</owl:ObjectProperty>
```

Like the example, on the model elements which UML supports but OWL does not, we use the name of model elements to represent the semantics of the elements. But, it is needed to define additional rules for dealing with these type of semantics. We put these problems in the future works.

5 Implementation of the Transformation Rule

OMG recommends XMI (XML Metadata Interchange) as a XML style standard for sharing and exchanging of UML models. XMI is a widely used interchange format for sharing object using XML [12]. Many UML tools including Rational Rose and Poseidon support a transforming function from UML models to XMI formatted documents, thus we use a XMI document as an input of our converting environment.

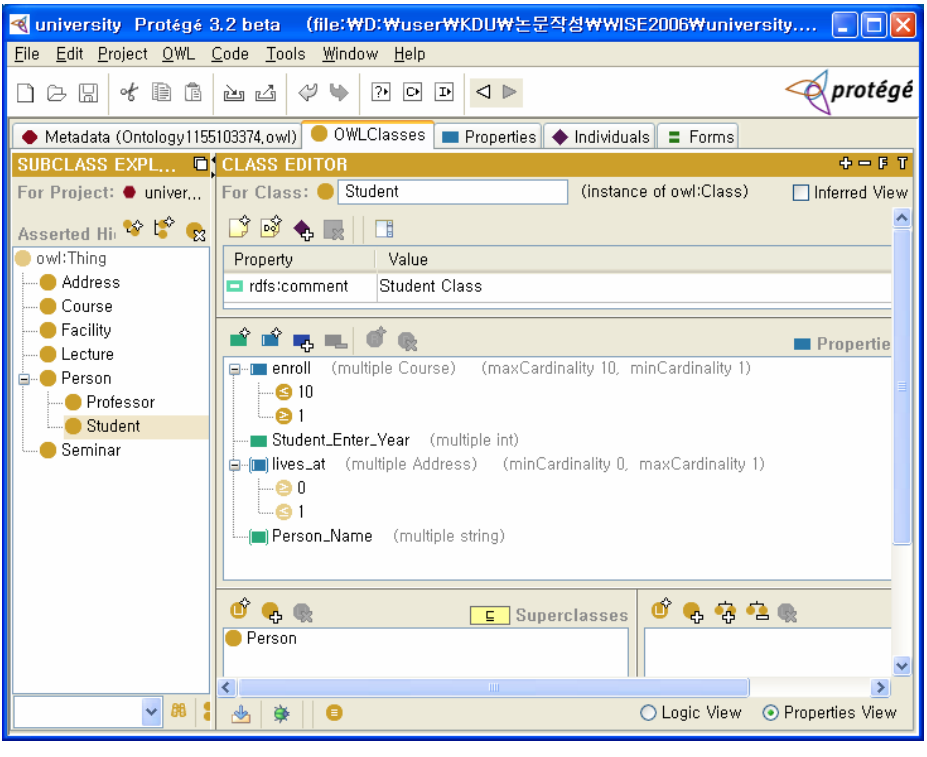


Fig. 4. The Transformed OWL ontology

Because XMI and OWL use XML syntax, we can convert the structure and contents of a XMI document into OWL style documents through XSL (eXtensible Stylesheet Language). To do this, XSL uses the idea of template. If it finds a predefined pattern in the XML documents, it replaces the pattern with another one according to the rules. As the result, the OWL ontology is automatically generated from the XMI document after XSLT transformation.

We constructed an OWL-encoded ontology from the UML diagrams in Fig. 1 applying the transformation rules defined in section 4. For graphical representation, we used protégé ontology editor to read and browse the created OWL ontology. Fig 4 shows the constructed university ontology on the UML diagrams in Fig.1. We can see that there are eight owl classes on the left-side window which shows classes and class hierarchies. The right-side window shows properties. The object properties enroll and lives_at have multiplicity restrictions.

Like this, the legacy UML diagrams can be converted into OWL-encoded ontologies, then we can edit and expand them using ontology editing tools like Protégé and Poseidon.

6 Conclusions

The objective of UML modeling is to depict functional, structural and behavioral aspects for a problem domain using several diagrams. We can find out some ontological domain knowledge, such as concepts and relations, from UML diagrams.

In this paper, we proposed a method for creating an early stage of domain ontology from legacy UML models – especially focusing on UML structure diagrams. We classified UML model elements into 4 corresponding ontological components (concepts, properties, instances and restrictions), and derive some similar features with OWL based on the metamodel of both languages. As the results, we defined the transformation rules to OWL representation for each UML model elements,

This paper shows most elements of UML with ontological semantics can be mapped to the corresponding OWL elements – especially concepts and properties. We expect that the generated OWL ontologies, even if they are not complete, play an important role as a base model for constructing domain ontologies.

The presented method is now in an early stage of development which leaves plenty of room for future work. In UML, there is a strict separation between class and its instances. This means that we have to consider the operational data in run time environment to provide sufficient ontology instances. Also, we have a plan to expend the scope to UML behavior models which include operations, activities and responsibilities.

References

1. Saartje Brokmans, Peter Haase, Pascal Hitzler and Rudi Studer, "A Metamodel and UML Profile for Rule-extended OWL DL Ontologies – A Complete Reference", Lecture Note on Computer Science, Vol. 4011, pp.303-316, 2006.
2. OMG, "Unified Modeling Language: Superstructure", version 2.0, August 2005.
3. Roberto Navigli, Paola Velardi, "Learning Domain Ontologies for Document Warehouses and Dedicated Web Sites", Computational Linguistics, Vol.30, No.2,pp.152-179, 2004
4. Bernd Amann and Irini Fundulaki, "Integrating Ontologies and Thesauri to Build RDF Schemas", In Proceedings of the Third European Conference for Digital Libraries, ECDL'99, Paris, France, 1999.

5. Marta Sabou, Chris Wroe, Carole Goble, Gilad Mishne, "Learning Domain Ontologies for Web Service Descriptions: an Experiment in Bioinfomatics", WWW2005, pp.190-198, May 2005, Japan.
6. Bechhofer, S., Harmelen, F.v., et al., "OWL Web Ontology Language Reference", W3C Recommendation, W3C, 2004.
7. OMG, "Usage Scenarios and Goals for Ontology Definition Metamodel", version 2.7, 2004.
8. Robert M. Colomb, Anna Gerber, Michael Lawley, "Issues in Mapping Metamodels in the Ontology Development Metamodel Using QVT", in The 1st International Workshop on the Model-Driven Semantic Web, 2004.
9. Dean, M. Java2OWL, DAML.org, 2003.
10. K. Falkovych, M. Sabou and H. Stuckenschmidt, "UML for the Semantic Web:Transformation-Based Approaches", In Knowledge Transformation for the Semantic Web, B. Omelayenko and M. Klein editors. IOS Press, 2003.
11. Dragan Gasevic, Dragan Djuric, Vladan Devedzic, Violeta Damjanovi, "Converting UML to OWL ontologies", Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters , Pages: 488 - 489 , USA , 2004.
12. OMG, "XML Metadata Interchange Specification", http://www.omg.org, May 2003.
13. Natalya F. Nay and Deborah L. McGuiness, "Ontology Development 101: A Guide to Creating Your First Ontology", Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.

Deeper Semantics Goes a Long Way: Fuzzified Representation and Matching of Color Descriptions for Online Clothing Search

Haiping Zhu, Huajie Zhang, and Yong Yu

Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, 200240, P.R. China
{zhu, zhjay, yyu}@apex.sjtu.edu.cn

Abstract. Indexing and retrieval by color descriptions are very important to finding certain web resources, which is typical in the example of online clothing search. However, both keyword matching and semantic mediation by the current ontologies may suffer from the semantic gap between the similarity evaluation and the human perception, which requests the exploitation of "deeper" semantics of color descriptions to reduce this gap. Considering the inherent variability and imprecision characterizing color naming, this paper proposes a novel approach to define (1) the fuzzy semantics of color names on the HSL color space together with their knowledge representation in fuzzy conceptual graphs, and (2) the associated measures to evaluate the similarity between fuzzified color descriptions. The experimental results rendered by the prototype clothing search system have preliminarily shown the strength of the deeper semantics surpassing the ability of both keywords and a concept hierarchy, in handling the matching problem of color descriptions in the targeted web resource search.

Keywords: Semantics of Color Descriptions, HSL Color Space, Fuzzy Colors, Fuzzy Conceptual Graphs, Fuzzy Color Similarity.

1 Introduction

The web has grown into the largest information base for people to acquire online resources as well as to share documents. In finding certain web resources, e.g. clothing commodities that are concerned in this research, we find that *color* plays a very important role. It is because color is not only one of the features that the human brain remembers the most [1], but also a reflection of people's likes and dislikes to a large extent. Therefore, it is suggested that special importance be attached to the color feature in a successful search system for such web resources.

However, several famous online shopping portals, e.g. *Yahoo! Shopping*[1], *Amazon.com*[2], etc., do not provide color indexing, and color names are treated as common keywords. As a result, a query of a "*crimson* T-shirt" may retrieve "Alabama *Crimson*

[1] http://shopping.yahoo.com
[2] http://www.amazon.com

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 423 – 435, 2006.

Tide" (a football club) T-shirts while filtering out those "*ruby*" or "*deep red*" ones, which are not expected by the users. Querying by color indices, seen on *eBay*[3], *ClothesAgency.com*[4], etc., overcomes the first defect; and subsequently, the emergence of semantic mediation techniques, such as the adoption of an ontology, helps people to unify the semantics of color terms by concepts (e.g., "*crimson*", "*ruby*" and "*deep red*" are possibly mapped to the same class) and organize them into a concept hierarchy (e.g. the concept "*crimson*, *ruby*, *deep red*" is subsumed by the concept "*red*"). Nevertheless, it has already been recognized that the concept hierarchy alone is not sufficient to determine the semantic closeness of concepts [2]. For example, assuming all kinds of *reds* are immediate subconcepts of "*red*", it is difficult to reveal that "*deep red*" is more similar to "*Turkey red*" than to "*orange red*". If the search system is anticipated to correctly rank "*Turkey red*" T-shirts higher than "*orange red*" ones to a "*deep red* T-shirt" query, we believe it is necessary to match color descriptions by "deeper" semantics, the semantics defined on certain color space models.

On color spaces, *color difference formulae* are most often used to define the similarity between two colors (a rather comprehensive review can be found in [3]). However, a multiplicity of uncertainties are presented in color descriptions mainly because of the subjective and context-sensitive nature of color [4]. Hence, it is quite unlikely that different people would map the same color name to a single color point in different applications. To achieve shared understanding for semantic integration, it asks for the incorporation of fuzzy semantics of colors in similarity evaluation.

Fuzzy colors, i.e. colors that are defined on fuzzy sets, have been well researched in the academia of CBIR (Content Based Image Retrieval) [5, 6]. However, in CBIR, it is the similarity between two color points (e.g. colors of pixels from two images) that is primarily investigated. Differently, when we match color descriptions (other than pixels in images), two groups of color points, denoted by two color descriptions, respectively, are actually being compared. This requests a new set of similarity measures.

Targeting the above design goals, this paper proposes a novel approach for matching color descriptions, featuring:

1. **The representation of a fuzzy color**, either *achromatic* (i.e. "*black*", "*white*" and all shades of "*gray*") or *chromatic* (the opposite to *achromatic*), as the composite of all the three component membership functions on the HSL color space (see Section 2 for more details on this color space), in the form of *fuzzy conceptual graphs* [7];
2. **The evaluation of similarity between fuzzified color descriptions**, according to the membership functions defined, for different combinations of the queried color and the color in the resource repository (hereinafter "the resource color") belonging to the two types of colors, i.e. the chromatic and the achromatic.

The prototype system for online clothing search has demonstrated the strength of the deeper semantics surpassing the ability of both keywords and a concept hierarchy, in dealing with the matching problem of color descriptions. Besides, we believe that the methodology can be applied to other domains to which matching color descriptions is also very important, e.g. color images [6, 8], flowering plants [9, 10], etc.

The rest of this paper is organized as follows. Section 2 briefly introduces the color spaces that are concerned. The fuzzified representation of color semantics is defined

[3] http://www.ebay.com
[4] http://www.clothesagency.com

in Section 3, while the similarity between two fuzzy colors defined in Section 4. Section 5 presents an introduction of the prototype system and an analysis of the preliminary experimental results. Certain related work is discussed in Section 6. Section 7 concludes the whole paper and gives several interesting topics for further research.

2 Color Spaces

The most commonly used color space to represent colors on web pages or printing materials may be RGB. In this space, each color is represented by three independent dimensions, namely *Red* (abbr. *R*), *Green* (abbr. *G*) and *Blue* (abbr. *B*). Usually, the value on each dimension falls into the range between 0 and 255. For example, "*black*" is assigned (0, 0, 0), "*white*" is assigned (255, 255, 255), and all shades of "*gray*" is located on the diagonal from "*black*" to "*white*", characterized as $R = G = B$. The other points in the cube depict chromatic colors (Figure 1(a)).

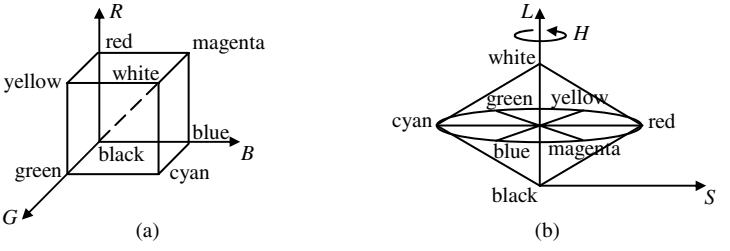


Fig. 1. The two color space models concerned in this paper: (a) RGB; (b) HSL

Despite its popular application in human life, RGB is also widely accepted as a color space that is not perceptually uniform to human vision [3, 5, 6, 10]. This paper focuses on color representation defined on the HSL space [11] which is believed to be close to the physiological perception of human eyes, and furthermore possesses easy-and-clear transformation from the RGB space (refer to [5]) as well as a conveniently realizable color difference formula (see equation (6) in Section 4.1). However, we also believe that our design rationale can be transferred to other perceptual or perceptually uniform color spaces by careful definition.

The HSL space tries to decompose colors according to physiological criteria as *Hue*, *Saturation* and *Luminance*. Hue (abbr. *H*) refers to the pure spectrum colors and corresponds to dominant colors as perceived by human beings. It is an angle that takes a value between 0 and 360. Saturation (abbr. *S*) corresponds to the relative purity or the quantity of white light that is mixed with hue, while luminance (abbr. *L*) refers to the amount of light in a color. Both of them are in the form of ratio and thus belong to the interval of [0, 1]. Figure 1(b) depicts the HSL color model. The points on the *L*-axis with *H* undefined and $S = 0$ denote achromatic colors, while the remaining the chromatic.

3 Representation of Fuzzy Colors

3.1 The Definition

In the literatures concerning fuzzy colors [5, 6], trapezoidal membership function is usually employed to model each separate dimension of a fuzzy color (Figure 2):

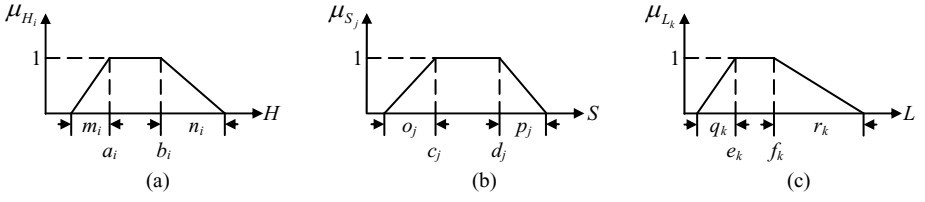


Fig. 2. The membership degree of the fuzzy (a) *H*; (b) *S*; (c) *L*

$$\mu_{H_i}(h)=\begin{cases}1, & a_i \le h \le b_i \\ 1-\dfrac{a_i-h}{m_i}, & a_i - m_i < h < a_i \\ 1-\dfrac{h-b_i}{n_i}, & b_i < h < b_i + n_i \\ 0, & \text{otherwise}\end{cases} \quad \mu_{S_j}(s)=\begin{cases}1, & c_j \le s \le d_j \\ 1-\dfrac{c_j-s}{o_j}, & c_j - o_j < s < c_j \\ 1-\dfrac{s-d_j}{p_j}, & d_j < s < d_j + p_j \\ 0, & \text{otherwise}\end{cases} \quad \mu_{L_k}(l)=\begin{cases}1, & e_k \le l \le f_k \\ 1-\dfrac{e_k-l}{q_k}, & e_k - q_k < l < e_k \\ 1-\dfrac{l-f_k}{r_k}, & f_k < l < f_k + r_k \\ 0, & \text{otherwise}\end{cases} \tag{1}$$

Investigating the set of transformation formulae from RGB to HSL [5], we can observe that the definition of *S* is dependent on *L*, which forms a triangular area (Figure 3(a)). In other words, the composite result of *S* and *L* is situated inside this triangular area, while the outside never appears in this color system. Thus, to compose the fuzzy set of *tone* (denoted as *T*), i.e. the *S*-*L* plane, we use the operation of *algebraic product* (enlightened by [5]) associated with a two-valued function f_T:

$$\mu_{T_{jk}}(s,l)=\mu_{S_j}(s)\cdot\mu_{L_k}(l)\cdot f_T(s,l)$$

$$f_T(s,l)=\begin{cases}1, & \text{if } s \le \dfrac{l}{0.5} \text{ when } l \le 0.5 \text{ or } s \le \dfrac{1-l}{0.5} \text{ when } l > 0.5 \\ 0, & \text{otherwise}\end{cases} \tag{2}$$

We further define the membership function for the fuzzy set of a chromatic color as the algebraic product of the membership grade of *H* and that of *T*.

$$\mu_{CC_x}(h,s,l)=\mu_{H_i}(h)\cdot\mu_{T_{jk}}(s,l) \tag{3}$$

Now, each chromatic color CC_x is mapped to a cake-like closed region on the HSL space with μ_{CC_x} as the membership function defining the fuzzy set on the "cake" (Figure 3(b)).

As to an achromatic color, since it satisfies the condition that *H* is undefined and *S* = 0, its membership function is measured only by the membership grade of *L*.

$$\mu_{AC_y}(l)=\mu_{L_k}(l) \tag{4}$$

Each achromatic color AC_y is mapped to a line segment located on the *L*-axis of the HSL space with μ_{AC_y} as its membership function (Figure 3(b)).

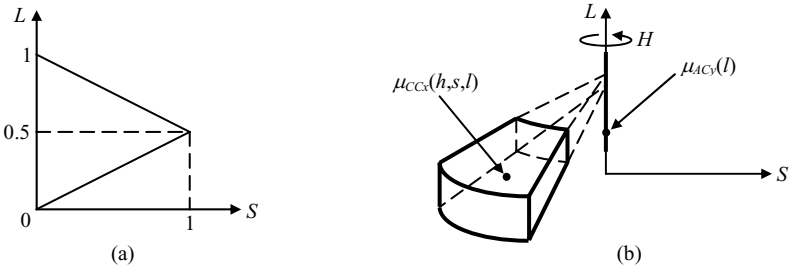


Fig. 3. The definition of fuzzy colors: (a) the tone plane; (b) a chromatic color – a "cake", and an achromatic color – a line segment

3.2 The Knowledge Representation

We employ the notion of *fuzzy conceptual graph* [7] to represent the fuzzy semantics of color descriptions. Fuzzy conceptual graph is an extension of Sowa's *conceptual graph* (abbr. *CG*) model [12], a graphic representation for logic with the full expressive power of first-order logic, to incorporate imprecise data in knowledge modeling. Therefore, this component is easy to integrate into an existing CG matching system, e.g. [13, 14]. For other search models that are highly related to *Description Logics* [15] in the community of the Semantic Web, *fuzzy description logics* [16] can also be an alternative.

A conceptual graph consists of nodes called *concepts* and *conceptual relations*, interlinked by arcs. Each concept node has a *type*, together with an *individual marker* referring to a particular instance or the *generic marker* "*" for an unspecified individual, whereas a relation node has a *type* only.

We use *fuzzy markers* introduced in [7] to extend the individual markers of the concepts "*HueValue*", "*SaturationValue*" and "*LuminanceValue*" linked to each chromatic color by the relations "*hasHue*", "*hasSaturation*" and "*hasLuminance*", respectively (Figure 4).

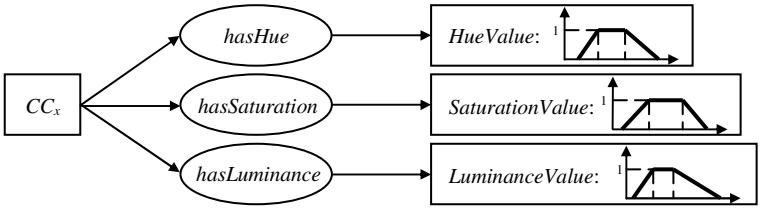


Fig. 4. The fuzzy conceptual graph for a chromatic color

Since an achromatic color features that H is undefined and $S = 0$, the fuzzy marker of its *HueValue* remains to be the generic marker, while that of its *SaturationValue* is a particular one that associates the value 1 with $S = 0$ and the value 0 with the rest (Figure 5).

When the CG matching algorithm in [13, 14] iterates to the concepts denoting fuzzy colors, a particular matching function is invoked to calculate the similarity between two colors that is to be elaborated on in Section 4.

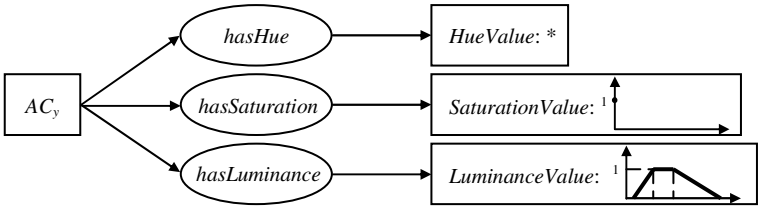


Fig. 5. The fuzzy conceptual graph for an achromatic color

4 Similarity Between Color Descriptions

As mentioned in Section 1, in our proposed approach, the comparison is not between two color points, but between two color descriptions, namely two groups of color points. In this sense, the similarity should be recognized as the conjunction (overlap) of the two colors on the color space. It should also be noted that in fact we define the similarity measure as asymmetric: we are measuring how much the resource color matches (namely, is subsumed by) the queried one, but not the reverse (take the intuitive example that a resource "*crimson*" matches a queried "*red*", but the reverse may not stand because a "*red*" can also be an "*orange red*" that does not match a queried "*crimson*"). Therefore, the similarity measure is given the form as the ratio of the overlap to the resource color.

In the following text, we divide the discussion of our fuzzy color similarity into three categories: chromatic color vs. chromatic color, achromatic color vs. achromatic color and chromatic color vs. achromatic color, according to the different characteristics between the two types of colors shown in Section 3.

4.1 Chromatic Color vs. Chromatic Color

If two chromatic colors overlap on the color space, the degree to which they overlap each other can be used to measure their similarity. Otherwise, we turn to investigate the distance between them: taking the closest two points selected from the two fuzzy colors respectively, we reckon the distance between them as the minimum cost of rotation and/or move to make the two fuzzy colors overlap.

1° Derived from the definition in Section 3, only when two chromatic colors have overlap on each dimension will they overlap on the color space (Figure 6). Special attention should be paid to the periodicity of H because it is an angle.

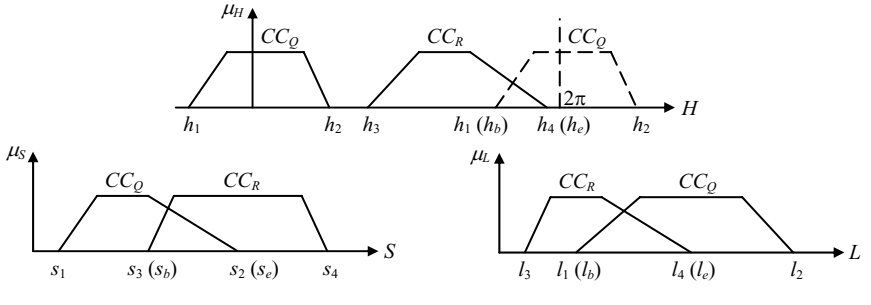


Fig. 6. Two chromatic colors have overlap on each dimension

In this case, we define the *overlap* between two colors as the ratio of two integrals in the cylindrical coordinate system with s as the radius, h the angle and l the height:

$$overlap(CC_Q, CC_R) = \frac{\int_b^{l_e} dl \int_{h_b}^{h_e} dh \int_{s_b}^{s_e} \min\{\mu_{CC_Q}(h,s,l), \mu_{CC_R}(h,s,l)\} \cdot sds}{\int_{l_3}^{l_4} dl \int_{h_3}^{h_4} dh \int_{s_3}^{s_4} \mu_{CC_R}(h,s,l) \cdot sds} \tag{5}$$

where CC_Q and CC_R are the queried chromatic color and the resource chromatic color, with μ_{CC_Q} and μ_{CC_R} as their membership functions, respectively. The integral intervals $[h_b, h_e]$, $[s_b, s_e]$, $[l_b, l_e]$ in the numerator are the intervals where the two colors overlap on the H, S, L dimensions, respectively. The integral intervals $[h_3, h_4]$, $[s_3, s_4]$, $[l_3, l_4]$ in the denominator are determined by the boundary of the resource color. The value of *overlap* ranges between 0 (any of $[h_b, h_e]$, $[s_b, s_e]$ and $[l_b, l_e]$ is reduced to a point) and 1 (CC_R is just the same as CC_Q).

2° If two chromatic colors have no overlap on each dimension (Figure 7):

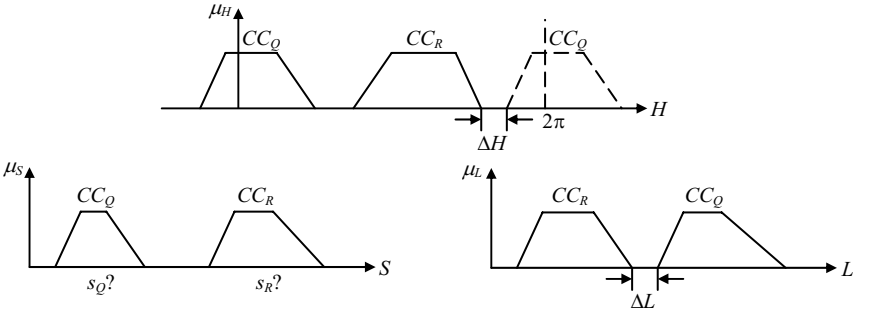


Fig. 7. Two chromatic colors have no overlap on each dimension

we turn to use the color difference formula on the HSL space to define the *distance* between the two colors:

$$distance(CC_Q, CC_R) = \sqrt{\Delta L^2 + {s_Q}^2 + {s_R}^2 - 2 s_Q s_R \cos(\Delta H)} \tag{6}$$

where ΔH and ΔL are defined as the distance on H and L, respectively, between the closest two points. Since it is not ΔS but the values of s_Q and s_R that affect the calculation of *distance*, we let $(s_Q, s_R) = \underset{\text{all } s \text{ in } CC_Q, CC_R}{\arg\min} \{{s_Q}^2 + {s_R}^2 - 2 s_Q s_R \cos(\Delta H)\} \cdot$

It is not difficult to conclude that the minimum of *distance* is 0, while the maximum is 2, reached when two saturated and complementary colors ($\Delta L = 0$, $s_Q = s_R = 1$, $\Delta H = 180$) are being compared. Moreover, the *distance* measure is symmetric.

3° If two chromatic colors have overlap on some, but not all, of the dimensions (i.e. the combination of separate cases on H, S, L in 1° and 2°), we calculate the *distance* according to equation (6) by designating $\Delta L = 0$ for overlap on L, $\Delta H = 0$ for overlap on H, and the valuation of (s_Q, s_R) in the same way as given in 2°.

Because the longer distance two colors hold the less similarity they have, we use the value of the opposite number of *distance* to measure their similarity. Thus, the similarity between two chromatic colors takes a value between -2 and 1 (from *distance* to *overlap*).

4.2 Achromatic Color vs. Achromatic Color

Achromatic colors have normal membership functions on the L dimension only. Therefore, determining the similarity between two achromatic colors is reduced to the measurement of their *overlap* or *distance* on L:

$$overlap(AC_Q, AC_R) = \frac{\int_b^e \min\{\mu_{AC_Q}(l), \mu_{AC_R}(l)\}dl}{\int_3^4 \mu_{AC_R}(l)dl} \tag{7}$$

$$distance(AC_Q, AC_R) = |\Delta L| \tag{8}$$

where AC_Q and AC_R are the queried achromatic color and the resource achromatic color, with μ_{AC_Q} and μ_{AC_R} as their membership functions, respectively. The definition of other parameters is the same as in Section 4.1.

The maximum of *overlap* is 1 when the two achromatic colors are totally the same, while the maximum of *distance* is also 1 (at the moment, the similarity is -1) when the two achromatic colors are just "*black*" and "*white*". When the two achromatic colors share only one color point, both *overlap* and *distance* reach their minimum 0. Therefore, the similarity between two achromatic colors ranges between -1 and 1.

4.3 Chromatic Color vs. Achromatic Color

Since each achromatic color is characterized as $S = 0$, the integral interval $[s_b, s_e]$ defined in Section 4.1 is either the point $S = 0$ or $\varnothing$ (i.e. no overlapping). Therefore, calculating the *overlap* between a chromatic color and an achromatic color always returns the result 0. Hence, we only observe the distance between such two colors. Because the *distance* measure is symmetric, we do not distinguish whether the achromatic color is the queried one or the resource one.

Because the H of each achromatic color is undefined, we may take $\Delta H \equiv 0$ when it is compared with a chromatic color (let the H of the achromatic color equal to that of the chromatic color). Thus, the *distance* is based on the tone plane only (Figure 8):

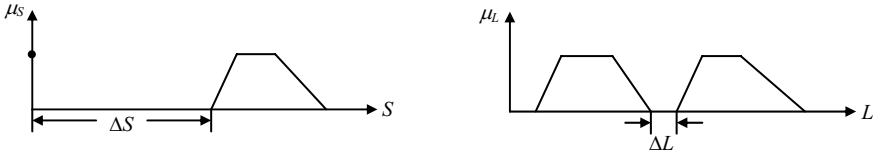


Fig. 8. The distance between a chromatic color and an achromatic color

$$distance(CC, AC) = \sqrt{\Delta L^2 + \Delta S^2} \tag{9}$$

It takes a value between 0 and $\sqrt{5}/2$ (when the chromatic color is a saturated one and the achromatic color is "*black*" or "*white*"), so the similarity, as its opposite, ranges between $-\sqrt{5}/2$ and 0.

5 Evaluation

We integrated the fuzzy color matching function into ALPHA, an existing CG matching system [13, 14] for assessment purpose. The architecture of the prototype system is shown in Figure 9. *Color indices*, according to the *fuzzy color database*, are built on both the queried fuzzy conceptual graphs and the resource ones. The *fuzzy color matching* subsystem evaluates the similarity between fuzzy colors. The whole system was implemented in C# + ASP.net and run on a P4 2.4GHz / 1GB Mem. workstation.

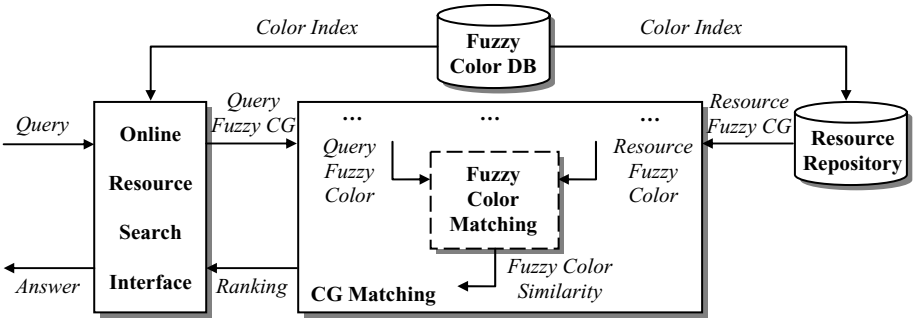


Fig. 9. The architecture of the prototype system for online resource search

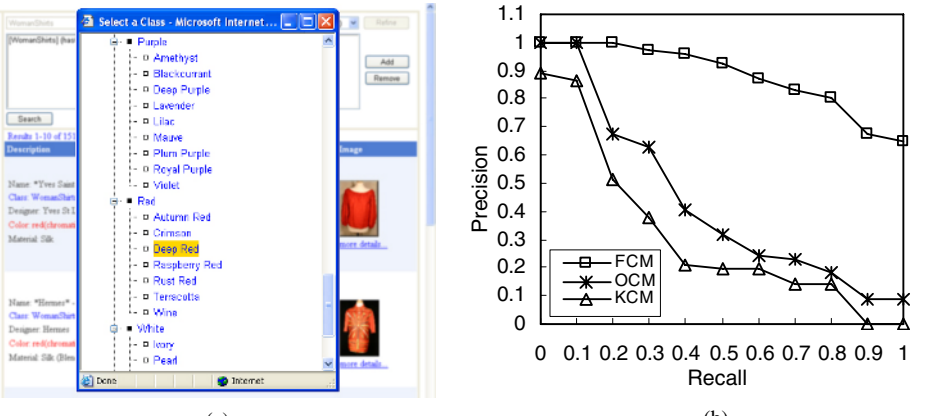
The fuzzy color database was constructed by collecting color names together with the RGB values of the sampling points for each color from "The Mother of All HTML Colors"[5], which can be taken as an open and collaborative color database. There were more than 11,650 entries of (*Color Name*(*s*), *RGB*) pairs gathered. After all the RGB values were transformed into HSL values, multiple colors belonging to the same entry were split. In respect that people seldom write query as "*gray* 1" or "57% *vivid cold blue*" (57% refers to its luminance), ordinal numbers and percentages in color names were removed. Finally, entries with identical names but different HSL values were merged, by figuring out the minimum cuboid region that covers these values, to construct the kernel of the membership function (i.e. the region in which the membership degree is 1) for that color. To determine the interval between the kernel and the border where the membership degree falls to 0, we use the heuristic to let it equal to a fixed proportion to the length of the kernel (e.g. 10% in our current implementation). If the kernel is reduced to a single point, we specify the interval as the minimum of those for non-single-point kernels. We believe in this way it to some

[5] http://tx4.us/moacolor.htm

extent models the subsumption relationship between colors, because more general color terms tend to possess longer kernels and hence larger regions on the color space. All the above steps eventually led to about 9,940 different fuzzy colors after their membership functions were defined, which we believe has an extensive coverage on most commonly-used color descriptions.

To set up the *resource repository*, we crawled the descriptions of over 3,000 clothing commodities from *ClothesAgency.com*. By filtering out those without simple color indices (e.g. multicolored categories, such as *check*, *stripe*, *print*, etc.), there remain 2,675 individuals, indexed by 100 color descriptions. Calculating the similarity between each pair of these colors is the only computation-intensive task during CG matching, because we have to discretely approximate the triple integrals. However, we can pre-compute the similarity values and store the results in a lookup table embedded in the system, so that the query results are still worked out within seconds. It should also be noted that for the moment, we use a simple normalization method to always map the ranges of values for *overlap* to [0, 0.5], and those for *distance* to [-0.5, 0], so that comparison can be made among the three color-matching categories introduced in Section 4.1–3.

Finally, we collected 30 clothing queries concerning color matching (e.g. a *deep red* shirt), and randomly chose a group of students in our lab to give their relevance judgments on the correspondingly fetched clothing descriptions from the resource repository (e.g. a *crimson* shirt can be a match to the aforementioned example query, but a *rust red* one not). The performance of the prototype system was compared with (1) that of a keyword matching system with thesaurus-based *query expansion*, implemented using Lucene[6], and (2) that of an ontology-based semantic search system (the original ALPHA system). It is understandable that the color ontology is rather difficult to obtain online, so currently we employed one (Figure 10(a)) that was derived from the structure of the color indices on *ClothesAgency.com* and refined with reference to the WordNet ontology [17]. We computed precision and recall for the



(a) (b)

Fig. 10. The evaluation. (a) The color ontology; (b) the average precision-recall curves.

[6] http://jakarta.apache.org/lucene/

different systems, defined as Precision = # relevant / # retrieved, Recall = # relevant / total # relevant. The average precision-recall curves for the 30 queries are shown in Figure 10(b), where "FCM" stands for *fuzzy color matching*, "OCM" for *ontology-based color matching*, and "KCM" for *keyword-based color matching*.

At the initial fragments of the result datasets, OCM and KCM are almost comparable to the FCM model. However, they tend to quickly introduce those irrelevant results that are either hierarchically or literally close to the query, which makes their precision curves drop markedly. The fact that the precisions at the end of their curves usually descend to 0 indicates that the two methods may fail to find some relevant results that are neither hierarchically nor literally similar to the query. We believe the above observations have proven that incorporating deeper semantics could better complement the current search models, esp. in a color-concerned domain.

6 Related Work

Wang *et al* [10] represent the semantics of each color term by a cuboid space, defined by a range triplet on the three component measures of the HSL color model. However, they only use a color reasoner to carry out subsumption checking, rather than similarity ranking. Liu and her colleagues [8] define a color naming model that also uses the combinations of value ranges on the three dimensions of the HSV color space (one that is very similar to HSL) to derive 93 different color names, but they simply perform exact string matching on color names. Besides, we think such range triplets are too rough-grained to precisely capture the fuzzy semantics of a color description.

[6] indexes dominant color(s) of an image by a tree of 96 classes consisting of *fundamental colors/non-colors* (*H*) and their *colorimetric qualifiers* (*S* and *L*) on the HSL space. Though fuzzy representation is concerned, they execute subsumption checking other than similarity measuring, so, like [10], the ranking of retrieved instances is not supported. In [5], membership functions of a set of fuzzy colors based on the HSL color space are first defined to represent the membership degrees of each pixel in an image to these fuzzy colors. Then, a fuzzy similarity measure is developed for evaluating the similarity of fuzzy colors between two pixels. However, their work does not address achromatic colors. Moreover, our approach focuses on matching color names, which can be taken as collections of color points, other than matching pixels.

7 Conclusion

Indexing and retrieval by color descriptions are very important to finding certain web resources, which is typical in the example of online clothing search. However, both keyword matching and semantic mediation by the current ontologies may suffer from the semantic gap between the similarity evaluation and the human perception. Though color similarity measures are well researched in the realm of CBIR, little work is done on how to match two color descriptions based on their fuzzy semantics. In this paper, we propose a novel approach to define a unified membership function on the three dimensions of the HSL color space for both chromatic and achromatic colors, and use fuzzy conceptual graphs to represent a fuzzified color. A set of similarity measures to

evaluate how well two color descriptions match each other are introduced as well. The experimental results rendered by the prototype clothing search system have preliminarily shown the strength of our approach in matching color descriptions by exploiting their "deeper" semantics.

There is still a lot of future work to do. For multicolored objects, different indexing and matching schemes should be investigated. Modeling and matching colors in a personalized way are also of great interest and necessity. Last but not least, the topic of this paper has shown that numerical processing can be helpful to certain semantic harmonization tasks that are beyond the capability of keywords and the current ontologies. We are expecting the experiments on other clothing features, such as size/length, material ingredient, etc., to further justify this thought.

References

1. Shoemaker, S.: Colors, Subjective Relations, and Qualia. Philos. Issues, 7 (1996) 55–66
2. Kedad, Z., Métais, E.: Ontology-Based Data Cleaning. In: Andersson, B., Bergholtz, M., Johannesson, P. (eds.): Proc. of the 6th Int'l. Conf. on Applications of Natural Language to Information Systems (NLDB 2002), LNCS 2553. Springer-Verlag (2002) 137–149
3. Sarifuddin, M., Missaoui, R.: A New Perceptually Uniform Color Space with Associated Color Similarity Measure for Content-Based Image and Video Retrieval. In: Proc. of ACM SIGIR 2005 Workshop on Multimedia Information Retrieval (MMIR 2005) (2005) 1–8
4. Binaghi, E., Ventura, A.D., Rampini, A., Schettini, R.: A Knowledge-Based Environment for Assessment of Color Similarity. In: Proc. of the 2nd IEEE Int'l. Conf. on Tools for Artificial Intelligence (1990) 768–775
5. Chien, B.-C., Cheng, M.-C.: A Color Image Segmentation Approach Based on Fuzzy Similarity Measure. In: Proc. of the 2002 IEEE Int'l. Conf. on Fuzzy Systems (FUZZ-IEEE'02) (2002) 449–454
6. Younes, A.A., Truck, I., Akdag, H.: Color Image Profiling Using Fuzzy Sets. Turk. J. Elec. Engin., 13(3) (2005) 343–359
7. Thomopoulos, R., Buche, P., Haemmerlé, O.: Different Kinds of Comparisons Between Fuzzy Conceptual Graphs. In: de Moor, A., Lex, W., Ganter, B. (eds.): Proc. of the 11th Int'l. Conf. on Conceptual Structures (ICCS 2003), LNCS 2746. Springer-Verlag (2003) 54–68
8. Liu, Y., Zhang, D., Lu, G., Ma, W.-Y.: Region-Based Image Retrieval with High-Level Semantic Color Names. In Chen, Y.-P.P. (ed.): Proc. of the 11th Int'l. Conf. on Multi Media Modeling (MMM 2005) (2005) 180–187
9. Das, M., Manmatha, R., Riseman, E.M.: Indexing Flowers by Color Names using Domain Knowledge-driven Segmentation. In: Proc. of the 4th IEEE Workshop on Applications of Computer Vision (WACV'98) (1998) 94–99
10. Wang, S., Pan, J.Z.: Ontology-Based Representation and Query of Colour Descriptions from Botanical Documents. In: Meersman, R., Tari, Z. (eds.): Proc. of the 4th Int'l. Conf. on Ontologies, DataBases, and Applications of Semantics (ODBASE 2005), LNCS 3761. Springer-Verlag (2005) 1279–1295
11. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. 2nd edn. Prentice Hall (2002)
12. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley (1984)
13. Zhong, J., Zhu, H., Li, J., Yu, Y.: Conceptual Graph Matching for Semantic Search. In: Priss, U., Corbett, D., Angelova, G. (eds.): Proc. of the 10th Int'l. Conf. on Conceptual Structures (ICCS 2002), LNCS 2393. Springer-Verlag (2002) 92–106

14. Tu, K., Lu, J., Zhu, H., Liu, G., Yu, Y.: A Semantic Search Approach by Graph Matching with Negations and Inferences. In: de Moor, A., Lex, W., Ganter, B. (eds.): Proc. of the 11th Int'l. Conf. on Conceptual Structures (ICCS 2003), LNCS 2746. Springer-Verlag (2003) 378–391
15. The Official Description Logics WWW Home Page: http://dl.kr.org/
16. Straccia, U.: Reasoning within Fuzzy Description Logics. J. Artif. Intell. Res., 14 (2001) 137–166
17. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge MA (1998)

Semantically Integrating Portlets in Portals Through Annotation

Iñaki Paz, Oscar Díaz, Robert Baumgartner*, and Sergio F. Anzuola

ONEKIN Research Group, University of the Basque Country
*DBAI, Institute of Information Systems, Vienna University of Technology
{inaki.paz, oscar.diaz}@ehu.es, baumgart@dbai.tuwien.ac.at,
jibfeans@ehu.es

Abstract. Portlets are currently supported by most portal frameworks. However, there is not yet a definitive answer to portlet interoperation whereby data flows smoothly from one portlet to a neighboring one. One of the approaches is to use deep annotation. By providing additional markup about the background services, deep annotation strives to interact with these underlying services rather than with the HTML surface that conveys the markup. In this way, the *portlet* can extend portlet markup with meta-data about the processes this markup conveys. Then, the *portlet consumer* (e.g. a portal) can use this meta-data to guide mapping from available data found in markup of portlet A to required data in markup of portlet B. This mapping is visualised as portlet B having its input form (or other "input" widget) filled up. However, annotating is a cumbersome process that forces to keep in synchrony the meta-data and the resources being annotated (i.e. the markup). This paper presents an automatic process whereby annotations are generated from portlet markups without user intervention. We detail our prototype using Lixto Visual Wrappers to extract semantic data from the markup.

1 Introduction

Portals are the ultimate integration platform that act as a doorway to distinct services. The heterogeneity of these services, and the diversity of the portal platforms make main vendors to set a common mechanism for front-end service integration: the portlets.

Broadly speaking, a portlet is a full-fledge application syndicated within a Web application (most frequently, a portal). Full-fledge application means that it not only provides the business logic (as Web Services do) but the user interface as well. Hence, a portlet request returns not just dull data (e.g. a data-centric XML document) but markup (e.g. XHTML) ready to be rendered by the portal. Moreover, a portlet service does not end with a single interaction. Portlet enactment normally involves multiple interactions with the end user.

A portal normally hosts distinct portlets that are simultaneously rendered. From this perspective, they look like Windows applications in a user desktop in the sense that a portlet renders markup *fragments*[1] that are surrounded by a decoration containing

[1] A fragment is defined as the output markup of a portlet.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 436–447, 2006.

controls. The portal page then contains a number of portlets whose fragments can be arranged into columns and rows, and minimized, maximized, or arranged to suit the user needs. Portlet aggregation is then defined as the combination of a set of portlets to achieve a common goal.

So far, portlets are aggregated "side-by-side", i.e. fragments from distinct portlets are arranged in the same portal page. This permits content from distinct services being visually aggregated. However, tighter forms of aggregation can be envisaged whereby data from one portlet can flow to another portlet. So far, this scenario is not yet possible, forcing individual user to manually copy and key in data from source to target portlets which leads to frustration, lost productivity, and inevitable mistakes.

To overcome this situation, two approaches are possible. The producer-based approach relies on the portlet producer to facilitate the mechanism to achieve a tighter integration among portlets (e.g. sharing portlet state or using the same database). By contrast, the consumer-based approach puts the burden into the portlet consumer, i.e. the portal. In this approach, portlets are kept untouched, and it is through the portal resources that portlet integration is achieved. This facilitates *existing* portlets to benefit from tighter forms of integration.

To this end, we propose the use of an annotator commodity. The annotator becomes the consumer of the portlet. It takes portlet markups as inputs, and delivers annotated markups along the portal ontology. Once all markups are annotated along the same ontology, semantic integration can be achieved by mapping data available in markup A with data required in markup B. This paper focuses on the annotator commodity. An annotation *"is a set of instantiations related to an ontology and referring to an HTML document"* [8]. In our case, the resources to be annotated are the portlet markups.

The rest of this paper is organized as follows. First, section 2 outlines a sample scenario. Section 3, 4 and 5 detail our proposed solution using an information extraction engine whose wrappers are generated through a simple Visual tool, namely, the Lixto Visual Wrapper[2]. Next, the annotator implementation is focused on section 6. Finally, related work and conclusion are presented.

2 A Sample Scenario

Before going into further details, let us present a sample case through two portlets: *dblpPortlet* and *acmPortlet*. The former mimics Ley's Web site[2] which supports a sequence of steps that lead from an academic to her research publications. The *acmPortlet* mirrors the searching facility provided by the ACM Digital Library[3]. Users can perform a search through the form, or can navigate the digital library through the taxonomy which ends up on a concrete paper. Figure 1 shows a screenshot of a portal page with both of them.

Current portal frameworks permit these two portlets to be rendered in the same portal page, i.e. "side-by-side" integration. However, user experiences can be leveraged if data could automatically flow from one portlet to a neighboring one. For instance, if

[2] *http://www.informatik.uni-trier.de/~ley/db/indices/a-tree/index.html*
[3] *http://portal.acm.org/dl.cfm*

Fig. 1. Two portlets side-by-side: *dblpPortlet* up, *acmPortlet* bottom

acmPortlet is in the state of displaying an specific paper, and *dblpPortlet* is in the state of prompting for an author then, the *dblp* author can be obtained from the authors of the paper at *acmPortlet*.

3 The Architecture

Portlets can be either local to the portal or provided through third parties as remote portlets. The annotator is itself supported as a portlet, but a local portlet. Since portlet API has been standardised through the JSR168 Java Specification [10], this option permits the annotator portlet to be vendor-independent: the commodity can be deployed in Oracle Portal, IBM WebSphere, etc.

Figure 2 depicts this scenario. Existing portlets are deployed remotely. When the portlet markup is to be annotated, the annotator portlet acts as a proxy portlet using the WSRP protocol [13]. A clone of the annotator portlet (hereafter, just annotator) is deployed for each remote portlet to be annotated. Hence, the annotator behaves as a container of a single portlet. The annotator takes the markup as input, extends this markup with the meta-data, and delivers the extended markup to the portal. The portal collects the markups coming from different annotators, and arranges them in a portal page. This page is finally rendered at the browser agent. Any interaction with a portlet is directly routed from the portal to the annotator, and from here to the portlet producer. Notice that this architecture imposes an indirection. The portal no longer addresses the portlet producer directly but through an annotator.

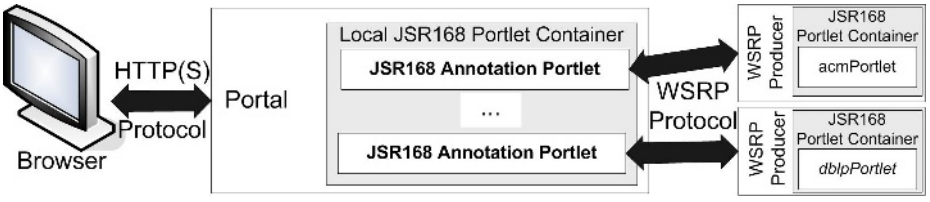


Fig. 2. Architecture

The extended markup the annotator builds is then used by the portal to realise data-flow among portlets. For the data-flow to be automatic, the portal should be able to understand what the fragments are about (i.e. the extended meta-data) and perform the mapping of that data among portlets. How this mapping is done has been previously presented at [6]. By contrast, this paper focuses on how the portlets' fragments are annotated.

Nowadays, annotators neither have access to the portal state (as portlets) nor to the portlet state (as portlet wrappers), thus the only pieces of information portlets have are the markups. This poses two questions:

First, *what characterises the markup of a portlet?* Having interoperability in mind, the markup of a portlet is characterised by its inputs and outputs.

Second, *how is markup "automatically" characterised?* Wrapping techniques are used to extract the semantic annotations from the fragments. Namely, Lixto Wrappers extract the information which will be attached to the fragment. Next sections provides the details for these two questions.

4 What Characterises the Markup of a Portlet?

For the purpose of this paper, markup can fulfill three distinct roles: (1) structure/context information markup which renders a meaningful set of data, (2) markup which prompts for some data (e.g. an input form) and (3), rendering-oriented markup whose purpose is to be interpreted by the browser (e.g. images, styles). We are only interested in the first two: what the portlet provides and what the portlet requests.

The idea is to conceive these markup chunks as the rendering counterpart of input and output operations. These operations are formalised as *OWL-S Atomic Processes* [4], which describe how web services work.

Therefore, a portlet ontology includes a task ontology, along the lines of OWL-S, and a domain ontology to describe the parameters of the task ontology. As an example, consider the *acm* portlet. The *acm* lifecycle whose reflection are the successive markups returned by the portlet, is captured as a set of *OWL-S Atomic Processes: searchQuery_IS*, *paper_OS* and the like. Figure 3 shows an excerpt of this ontology where the suffix *OS* (output service) and *IS* (input service) denote output and input Atomic Processes, respectively. The portlet ontology to be used is provided by the portal administrator when creating the annotator for the given portlet.

Fig. 3. Characterising portlet through an OWL-S ontology: an example for *acmPortlet* where (a) presents the domain part and (b) the task part

5 How Is Markup *Automatically* characterised?

Previous section describes how a markup is characterised as the rendering counterpart of *OWL-S Atomic Processes*. The issue now is to automatically extract the instantiations related to this ontology and referring to a portlet markup. That is, annotating the markup along the portlet ontology.

An example for the markup in figure 1 can be found in figure 4. Now the XHTML document returned by the portlet (i.e. the fragment) is annotated with tags coming from the ontology of the portlet (provided by the annotator designer). The *acm* namespace is introduced with this purpose. The snippet includes the following annotations:

- for each "output" markup chunk (i.e. the one that renders a meaningful set of data), an annotation is inlaid where the outcome is conceived as the result of a parameterless function. Our sample fragment conveys only one output Atomic Process (i.e. *paper_OS*). Each Atomic Process comprises its actual parameters, which correspond to instantiations of the domain ontology of the portlet, i.e. *ACM-DLOnto*. See figure 4 (a).
- for each "input" widget (e.g. an entry form), an annotation is introduced where the widgets are conceived as the realization of an input-only atomic process of the portlet's ontology. Our sample fragment includes one input Atomic Process (i.e. *searchQuery_IS*). See figure 4 (b).

It is worth noticing that input processes impose an important issue on naming conventions of parameters or data range restrictions. For example consider a select or radio input. They do limit the number of values the input process may have. Those values have to be also included into the process instance with its corresponding physical values and semantic meaning (e.g. in a country list, "Spain" has an option value of "es").

For annotation to be run automatically, Information Extraction wrappers are used. Wrappers are programs that extract data from the Web and turn the data into a more

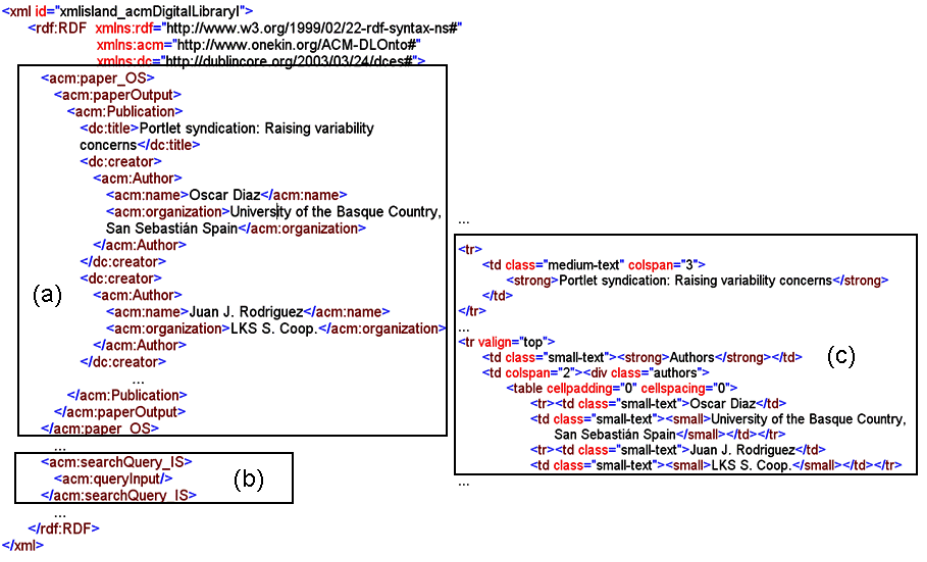


Fig. 4. Markup + annotations: an example for the *acmPortlet* fragment at fig. 1

meaningful structure such as XML, relational databases, etc. Wrappers are quite tedious to program, and visual tools are available to facilitate this process. Specifically, Lixto Visual Wrapper [2] is used in this work.

Notice that annotation is the process to be automated. To this end, wrappers are built, but not automatically. The portal administrator uses Lixto, which has been enabled to access portlet markups in order to build the wrappers[4] that, in runtime, automatically extract the meta-data to extend portlet fragments. Lixto employs a fully visual wrapper specification process. Neither manual fine-tuning and only little knowledge of HTML is necessary. Figure 5 shows an screenshot on the wrapper construction for an article fragment.

The basic building block of a Lixto wrapper program is a so-called pattern, a container for pieces of information with the same meaning. Patterns are structured in a hierarchical fashion. Inside a pattern, the wrapper designer visually defines filters, a crucial part of the pattern which defines how to extract relevant information from its parent pattern instances. Defining a filter expects the designer to select an example publication with two mouse clicks on a sample fragment. A filter definition continues with optional fine-tuning of properties for the generated generalization of the chosen example. It is possible to test filters against new fragments and based on the results, the designer decides whether to extend (i.e. add another filter) or shrink (i.e. add a condition to an existing filter) the set of matched instances. Very expressive visual wrapper generation is possible allowing the extraction of target patterns based on surrounding landmarks, on the content itself, on HTML attributes, on the order of appearance, and on semantic and syntactic concepts.

[4] Distinct wrappers will be needed for each type of fragment rendered by the portlet.

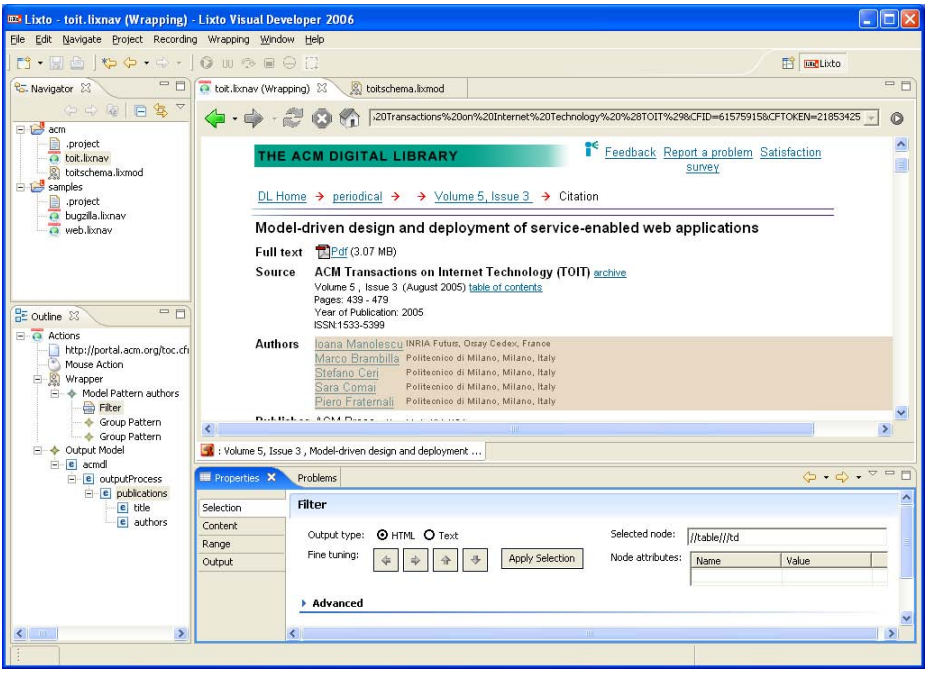


Fig. 5. Lixto Visual Wrapper specification in Visual Developer 2006

Unlike other application scenarios [1,3], the relevant meta information available on the fragments is extracted to help define a semantic context of portlets, instead of directly populating an ontology on the backend where distinct controlled sources may be integrated. Now the ontology is remote and populated through the portal. The portal then integrates the distinct and unrelated information sources that now are the portlets.

The output of a Lixto wrapper is usually XML:

```
<document><acmDL>
  <outputProcess><Publication>
    <title>Portlet syndication: Raising variability concerns</title>
    <author>
      <name>Oscar Diaz</name><organization>ONEKIN </organization>
    </author>
    <author><name>Juan J. Rodriguez</name>
      <organization>LKS, S. Coop.</organization>
    </author>
    <abstract>A Portlet is a multistep, ... </abstract>
    ....
  </Publication></outputProcess>
  <inputProcess> .... </inputProcess>
  ....
</acmDL></document>
```

In this scenario, however, the extracted meta information should be leveraged to a semantic presentation based on OWL-S. Now a mapping, by means of an XSLT or XQuery, from the XML output documents to ontology instances can be easily and automatically obtained. We do not go deeper into this process because there has been a lot of research already in this context (e.g. Harmonise [5]) that can be exploited in practice.

6 The Implementation

Portlet standards define a component model that enables portlets to be easily plugged into standards-compliant portals. Thus portlets are currently a main building block for portal construction. One of the main lifecycle differences with the traditional *http* protocol is that *http* readily provides an HTML page as a result of an *http* request. By contrast, portlet standards impose a two-phase protocol where the request is decoupled from the rendering of the next markup. In this way, two operations are intertwined[5]: (1) *processAction()* which conducts the interaction of the user and updates the portlet's state accordingly, and (2) *render()* which is a request for the next markup to be rendered.

Basically, the portlet cycle is as follows. A user is presented with a portal page and clicks on any of the fragment's anchors, which makes the portal invoke *processAction()* into the affected portlet. As a result, the portlet changes its state[6], and becomes ready to produce a new fragment. The portal now collects all the fragments resulting from the distinct *render()* enactments to all the portlets, and arranges them together into a new portal page that begins the cycle again.

To build the annotator we have used WSRP4J. WSRP4J[7] is an *Apache* project that integrates the WSRP communication protocol in a Java context, allowing a JSR168 Portlet be remotely deployed. They also provide a generic JSR168 Proxy Portlet that acts as a proxy to remotely deployed portlets. Thus the annotator extends this generic portlet by enabling the annotation and acting as a wrapper of a source portlet. That is, its markups are obtained after the markups are returned from the wrapped portlet. Figure 6 shows the interaction diagrams with the main actors.

When a user interacts with a fragment, the portal forwards the interaction to the corresponding annotator (by means of *processAction()*, *acm* one in this case*).* The annotator in turn delegates this request to the provider of the wrapped portlet (by means of *perfomBlockingInteraction*(), a WSRP operation) which serves it to the *acmPortlet.* Finally, the *acmPortlet* changes its state accordingly.

Next, and conforming to the two-phase protocol, the portal retrieves the next markup to be rendered by issuing *render()* to all the (annotator) portlets. The annotator in turn, issues a *render()* on the wrapped portlet, collects the markup, annotates the markup by invoking the right Lixto wrapper, and returns the annotated markup to the portal, where output services are matched to input services.

If some matching happens, the portal issues a second *render()* with the input services and its actual values as parameters. On receiving this second *render()*, annotator portlets

[5] JSR168 terminology has been used.

[6] In fact, distinct portlets can change their state through a producer-based portlet integration. This is the rationale behind this two-phase protocol.

[7] http://portals.apache.org/wsrp4j/

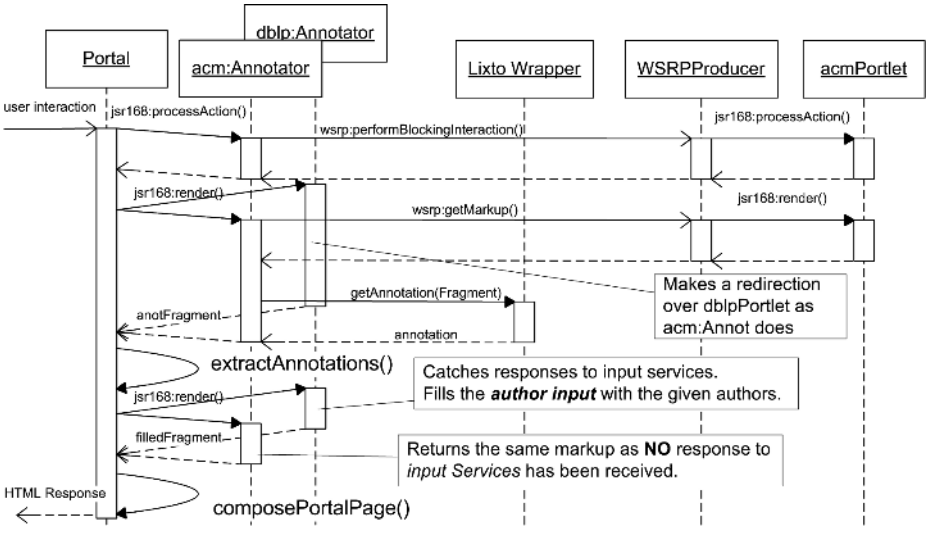


Fig. 6. Interaction diagrams for *processAction()* and *render()* of the annotator portlet

check if the *inputServices* parameter correspond to any of the input services of its current markup. If not, the same markup is returned. Otherwise, the annotator updates the markup, providing the actual values of the *inputServices* as suggestions to the user in the forms.

From the user viewpoint, all these enactments are the result of a single user interaction. Figure 7 shows the source state where the *acmPortlet* is prompting from an article (right), and the *dblpPortlet* is prompting from an author. The user provides the article by clicking on the corresponding anchor of the *acmPortlet* and this initiates all the process. From the user perspective, this transition delivers the target state. Here, not only the *acmPortlet* has changed but so does *dblpPortlet* whose input is now filled up with the names of authors. More sophisticated mechanisms can be envisaged for parameter matching as those suggested for Web Services [14][16]. However, this is outside the scope of this paper.

Processing Inputs. How the portal receives annotations is clear, but how are *inputServices* data passed to the portlet? As URI request parameter. Notice that all the possible values of inputs elements on a HTML are just strings, and that the bigger problem is addressing multi-evaluation, like in the given author example. Request parameters can be multi-evaluated. Sometimes it may be needed to transform the type of the input (e.g. from text to select [8]).

A normal request to a *render()* contains several parameters for portal use (e.g. *portal:componentId = dblpAnnotator & portal:type = render*[9]) , and several others for the portlet use. Now those for portlet use are extended with *inputServices* data (e.g. *dblp:author = Oscar%20Diaz & dblp:author = Juan%20J.%20Rodriguez*).

[8] User experience can be improved using complex scripting techniques instead.

[9] These are the parameters used in eXo Platform portal: *www.exoplatform.org.*

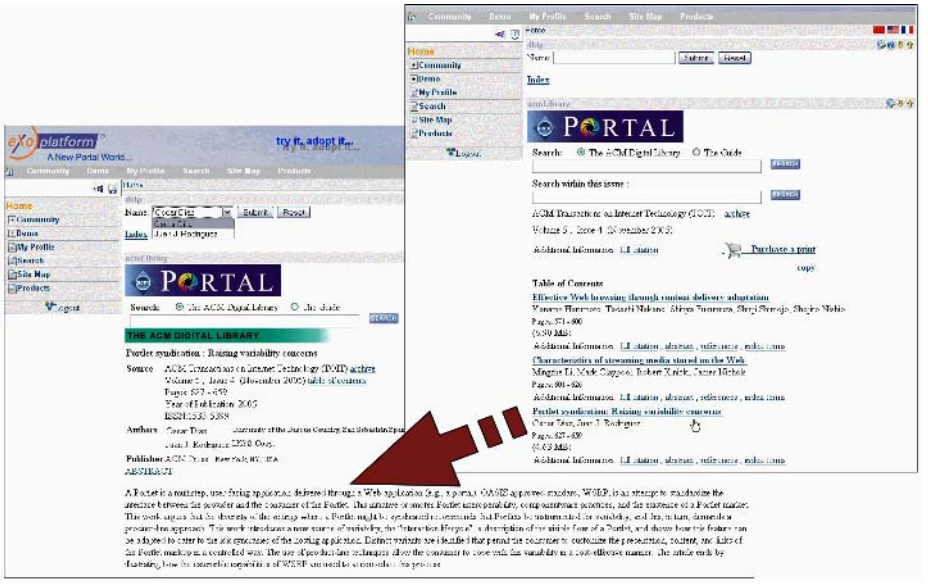


Fig. 7. *acmPortlet* integration with *dblpPortlet* on viewing a paper

To this end, locating the input element on the markup to be modified is easily done by searching the right input element with an XPath (e.g. *//input[@name='author']*). In this way, the actual value of the *inputService* becomes the default value of a parameter of the input form. Finally the resulting markup is returned back to the portal, and rendered to the user.

7 Related Work

Offering an integrated view on the information found in several web sites is not new. Piggy Bank [9], a revolutionary extension to a popular browser (limiting its functionality to a personal use), allows the easy extraction and composition of the semantic information located on the Web pages, directly by browsing them. Then, it allows to nicely view the information out of its original context, and perform complex queries not available before. However, this system does not allow you to integrate the distinct processes available in the Web applications, and of course, removes any trace of presentation found. Instead a portal is a company unified point of access where portlets are enabled to interact with each other. Web applications can be converted into portlets using [15].

The beta drafts from standard committees (WSRP V2.0 & JSR286 Portlet Specification) are also paying attention to integration capabilities of portlets by establishing new means of communicating events. Our framework actually includes the events into portlet's presentation, but moving it to a new communication standard is not a mayor problem.

There are many works that extract information from the Web, from them, Ding et al. [7] use ontologies to semantically annotate Web pages. The classes, operations and relations on the ontologies have an extra facet, data-frames, which describe how instances appear in texts (i.e. through text patterns). This enables to locate possible instances. By the use of this ontologies the annotation could be fully **automatically** done by solely generating an ontology per domain of application. However, this approach has several counterparts: It is a hard work to build a new ontology for a new domain; and is mainly oriented to extract information directly from texts. Being greatly robust to extract information, it does not consider the processes existing on a Web page (e.g. input forms). In this context, Kushmerick [11] tries to use the ontologies to capture the semantics of Web forms in order to simulate annotation of Web services, based on the labels of the inputs. However, locating the labels is a hard problem as forms' structure is not directly related with DOM position of elements [12].

In this sense, as Piggy Bank does, we have resorted to traditional information extraction technologies based on page structure. Lixto provides a fast generation of robust enough wrappers which then are integrated into the portlet, where Piggy Bank ones are JavaScript-based and manually-generated. This makes them really hard to be developed compared to Lixto.

8 Conclusion

Enhancing the user experience is one of the hallmarks of portals. This implies for the user to perceive the distinct offerings of a portal as an integrated workplace where data flows smoothly among the distinct portlets being framed by the portal. The controlled and cooperative environment that characterizes the portal facilitates the use of deep annotation to portlet interoperation.

This paper has described such approach by using annotator portlets, which wrap remote portlets. The sample portlets have been implemented and can be found at *http://www.onekin.org/wise-annotator* portal.

However, the portal implementation needs to be adapted. In our case the *eXo Platform* portal has been adapted to support portlet integration. Integrating portlets without having to touch the portal where they are aggregated represents another exciting battleground.

References

1. R. Baumgartner, S. Eichholz, S. Flesca, G. Gottlob, and M. Herzog. *Annotation for the Semantic Web*, volume 96 of *Frontiers in Artificial Intelligence and Applications*, chapter Semantic Markup of News Items with Lixto. IOSPress, 2003.
2. R. Baumgartner, S. Flesca, and G. Gottlob. Visual web information extraction with lixto. In *27th I.C. on Very Large Data Bases (VLDB 2001)*, pages 119–128. Morgan Kaufmann, 2001.
3. R. Baumgartner, N. Henze, and M. Herzog. The Personal Publication Reader: Illustrating Web Data Extraction, Personalization and Reasoning for the Semantic Web. In *European Semantic Web Conference ESWC 2005*, volume 3532 of *LNCS*, pages 515–530. Springer-Verlag Heidelberg, 2005.

4. W3C Consortium. OWL-S: Semantic Markup for Web Services, 2004. at http://www.w3.org/Submission/OWL-S/.
5. M. Dell'Erba, O. Fodor, F. Ricci, and H. Werthner. Harmonise: A solution for data interoperability. In *Towards The Knowledge Society: eCommerce, eBusiness, and eGovernment, IFIP I3E 2002*, volume 233 of *IFIP Conference Proceedings*, pages 433–445. Kluwer, 2002.
6. O. Díaz, J. Iturrioz, and A. Irastorza. Improving portlet interoperability through deep annotation. In *WWW '05: 14th international conference on World Wide Web*, pages 372–381. ACM Press, 2005.
7. Yihong Ding, David W. Embley, and Stephen W. Liddle. Semantic annotation based on extraction ontologies. Submitted manuscript, 2005. at http://www.deg.byu.edu/papers/index.html.
8. S. Handschuh, R. Volz, and S. Staab. Annotation for the Deep Web. *IEEE Intelligent Systems*, 18(5):42–48, September/October 2003.
9. D. Huynh, S. Mazzocchi, and D. Karger. Piggy bank: Experience the semantic web inside your web browser. In *4th International Semantic Web Conference ISWC 2005*, volume 3729 of *LNCS*, pages 413–430. Springer-Verlag Heidelberg, 2005.
10. JCP. JSR 168 Portlet Specification Version 1.0, September 2003. at http://www.jcp.org/en/jsr/detail?id=168.
11. Nicholas Kushmerick. Learning to invoke web forms. In *On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, volume 2888 of *LNCS*, pages 997–1013. Springer-Verlag Heidelberg, 2003.
12. Bing Liu, Robert Grossman, and Yanhong Zhai. Mining data records in web pages. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–606. ACM Press, 2003.
13. OASIS. Web Service for Remote Portlets Specification Version 1.0, 2003. http://www.oasis-open.org/commitees/tc_home.php?wg_abbrev=wsrp.
14. M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara. Semantic Matching of Web Services Capabilities. In *1st International Semantic Web Conference*, pages 333–347. Springer-Verlag, June 2002.
15. Iñaki Paz and Oscar Díaz. On portletizing web applications. *2nd revision in ACM Transactions on Internet Technology, ACM TOIT*, 2006.
16. E. Sirin, J. Hendler, and B. Parsia. Semi-automatic Composition of Web Services using Semantic Descriptions. In *1st Workshop on Web Services: Modeling, Architecture and Infrastructure*, pages 17–24. ICEIS Press, April 2003.

A Self-organized Semantic Clustering Approach for Super-Peer Networks

Baiyou Qiao, Guoren Wang, and Kexin Xie

School of Information Science and Engineering, Northeastern University, Shenyang, 110004, China
{qiaobaiyou, wangguoren}@ise.neu.edu.cn

Abstract. Partitioning a P2P network into distinct semantic clusters can efficiently increase the efficiency of searching and enhance scalability of the network. In this paper, two semantic-based self-organized algorithms aimed at taxonomy hierarchy semantic space are proposed, which can dynamically partition the network into distinct semantic clusters according to network load, with semantic relationship among data within a cluster and load balance among clusters all well maintained. The experiment indicates good performance and scalability of these two clustering algorithms.

1 Introduction

In recent years much research has been focused on improving search efficiency and scalability in P2P systems [2,7,8,9,10], and one important approach is to construct super-peer networks. Super-peer networks such as Edutella[1], KaZaA[3] combine elements of both pure and hybrid systems, it has the potential to combine the efficiency of a centralized search with autonomy, load balancing, robustness to attacks and at least semantic interoperability provided by distributed search. Therefore, super-peer networks are the research focus in this paper.

At present there are many distributed data sources which always using taxonomy hierarchy to organize and classify some kind of data, such as files in PCs, web directories and so on. And how to efficiently exchange and share information among these large-scale distributed data sources? One effective solution is to construct a taxonomy-based super-peer semantic overlay network. In this network, peers with semantically similar content are clustered together with their super-peer form a cluster, and Query can be forwarded to the clusters that probably contain results, thus reducing the communication cost and message numbers of the queries, which improves the search efficiency and network scalability. And an efficient clustering approach suitable for taxonomy hierarchy semantic space is a basis of constructing such a network. One static clustering method was proposed by Li. M. [4], in which peers are clustered into many semantic clusters by the concepts in taxonomy hierarchy, and each semantic cluster is associated with a concept. For super-peer networks, this method can lead to load imbalance of super-peer; another approach uses hash functions to map concepts in taxonomy hierarchy into many clusters [5], it will result in loose semantics of data within a cluster, cause communication increase

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 448 – 453, 2006.

among clusters while searching. Paper [6] proposes a routing and clustering strategy based on routing indices in super-peer networks, in which concepts and schemas are used to cluster peers without considering load balance. So we proposed two self-organizing semantic clustering algorithms aimed at taxonomy hierarchy semantic space, i.e. Semantics First Clustering Algorithm (SFCA) and Load Balance First Clustering Algorithm (LBCFA).

The two algorithms are based on the semantic space partitioning strategy, and a preset maximal cluster size M is used to determine the cluster range. When the cluster size exceeds M, the cluster will be split into two sub-clusters dynamically according to the semantics and load. Therefore, it is a self-organized clustering method with good adaptability for changes of cluster's members. SFCA algorithm firstly guarantees tight semantics of data within clusters and then finds a partition of semantic space, which most satisfies the load balance demand; LBFCA algorithm takes load balance and semantics into consideration synthetically and achieves a good tradeoff between them. The remainder of the paper is organized as follows: related definition and representation is provided in section 2, Section 3 describes the two algorithms in detail, and a preliminary performance evaluation is presented in Section 4. Finally we conclude this paper.

2 Related Definitions and Representation

In a taxonomy-based super-peer semantic overlay network, a set of peers together with their super-peer forms a semantic cluster; super-peers are also connected to each other and form an overlay structure. Data is stored on peers with index and routing information stored on the super-peer. Intra cluster data communication takes place via direct peer-to-peer links; inter clusters communication takes places via links between super-peers. Each super-peer manages a semantic subspace, and data on peers connected with the super-peer fall in the semantic subspace, and the semantic subspace is called cluster space, which is comprised of a set of concepts in taxonomy hierarchy. Related definitions are presented as follow.

Definition 1. A taxonomy hierarchy is a directed tree $T = <V, E>$, where V is the set of nodes, each node represents a concept, E is the set of directed edges, and an edge represents ISA relationship.

Definition 2. A cluster space $CS_i = \{T_{i1}, T_{i2}, \ldots, T_{im}\}$, where T_{ij} $(1 \le j \le m)$ is a taxonomy hierarchy sub-tree of T.

Definition 3. A semantic cluster $SC_i = (CS_i, Peerset_i)$, where CS_i is a cluster space, $Peerset_i$ is a set of peers whose data fall in the cluster space CS_i.

Form above definition, a semantic cluster can be logically represented by all taxonomy hierarchy sub-trees in its cluster space. A node in the tree is described by a quadruple (ID, weight, subtree_weight, peerlist), where ID is the identification of the node, weight is the weight of the node, i.e. load of the node, representing the number of peers classified onto the concept of the taxonomy, subtree_weight is the number of peers of the sub-tree whose root is the node, as the load of the sub-tree, equaling to the sum of weights of all nodes in the sub-tree, and peerlist is the set of peers classified onto the node.

3 Self-organizing Clustering Algorithms Based on Semantics

3.1 Clustering Strategy

Our clustering approach is based on semantic space partitioning strategy, which dynamically partitions semantic space by load and semantics, the chief idea is: suppose that there is one semantic cluster in the network initially, whose cluster space is the whole taxonomy hierarchy tree; as the load of the cluster increases with peers adding, the maximal cluster size M is reached and the cluster will be split into two sub-clusters dynamically according to the semantics and load. Every semantic cluster is responsible for answering queries falling in its cluster space, and thus an adaptive super-peer semantic network comes into being. The advantage of the method is that the number of peers within a cluster is less than the maximal cluster size M, and hence search efficiency of the system is guaranteed. Therefore, the method is adaptive to difference of density of peers in the taxonomy hierarchy semantic space, and makes system more stable and well adaptive to changes of members.

3.2 Clustering Algorithms

According to the features of semantic cluster, two clustering algorithms are proposed herein, i.e. Load Balance First Clustering Algorithm and Semantics First Clustering Algorithm. Description of the algorithms is presented as follow.

(1) Load Balance First Clustering Algorithm

Load Balance First Clustering Algorithm (LBFCA) is a top-down clustering algorithm according to the load of the semantic cluster, which considers both load balance among clusters and semantics within cluster, makes a good tradeoff. According to the algorithm, the clustering result may be one of the three:

1) If the cluster is composed of a node in taxonomy hierarchy, after partition, two new cluster spaces is same as old one, but load is equally partitioned.
2) If the cluster is composed of multi taxonomy hierarchy sub-trees, the sub-trees are partitioned into two groups, each of which forms a new cluster, and either the load of two new clusters satisfying the load interval demand or the difference of loads between two new clusters is minimal.
3) If the cluster is composed of a taxonomy hierarchy sub-tree, the two new clusters generated by the algorithm have the form: one is composed of a sub-tree, another is compose of one or more sub-trees of the same node, and either the load of two new clusters satisfying the load interval demand or the difference of loads between two new clusters is minimal.

Detailed description of LBFCA is shown in Algorithm 1, where M is the maximal cluster size, the function split-forest(*in-set*) is used to partition a cluster composed of several taxonomy hierarchy sub-trees, and *in-set* is the set of roots of taxonomy hierarchy sub-trees. For simplification, detailed description of the function will not be represented herein.

(2) Semantics First Clustering Algorithm

Semantics First Clustering Algorithm (SFCA), adopts a simple strategy that firstly ensures semantics within clusters, in which a sub-tree best satisfying the load demand is separated. The two clusters generated by the algorithm have parent-child relationship logically. The algorithm is detailed in Algorithm 2.

Algorithm 1. Load Balance First Clustering Algorithm LBFCA(node *in-set[]*, int *m*, int *x*)

Input: the set of roots of taxonomy hierarchy sub-trees *in-set*, the number of sub-trees *m*, load interval mediating parameter *x*

Output: the set of partitioning nodes *out-set*

```
(01) If (m>1) then
(02)   Split-forest(in-set);
(03) else
(04)   R=in-set[0];
(05)   if ( R.child = = NULL ) then
(06)     Out-set ←R;
(07)     return;
(08)   end if
(09)   if ( R.weight>M/2 - x) then
(10)     Out-set←{all children of R}
(11)     return;
(12)   end if
(13)   for all children C of R do
(14)     If (C.subtree_weigh≥M/2 - x and
           C.subtree_weight ≤M/2 + x ) then
(15)       Out-set←C;
(16)       return;
(17)     end if
(18)     If (C.subtree_weigh t>M/2 + x) then
(19)   temp =M - C.subtree_weight + C.weight;
(20)     If (temp > C.subtree_weight) then
(21)       out-set←C;
(22)       return;
(23)     end if
(24)     Temp= C.weight;
(25)     Temp1=C.subtree_weight;
(26)     C.weight =M - C.subtree_weight
                    + C.weight;
(27)     C. subtree_weight=M;
(28)     LBFCA(C,1,x) ;
(29)     C.weight=temp;
(30)     C.subtree_weight =Temp1;
(31)     return;
(32)   end if
(33)   end for
(34)   Temp=R.subtree_weight;
(35)   R. subtree_weight = R.weight;
(36)   In-set←R;
(37)   for all children C of R do
(38)     In-set← In-set ∪ C;
(39)   end for
(40)   Split_forest(In-set);
(41)   R.subtree_weight =Temp;
(42) end if
```

Algorithm 2. Semantics First Clustering Algorithm SFCA(node *R*)

Input: the root *R* of the tree, load interval mediating parameter *x*

Output: the root *R′* of the sub-tree to be separated

```
(01)  max_weight=0;  R′=R;
(02)  while (R′.subtree_weight > M/2 + x) do
(03)    if ( R′.child = = NULL ) then
(04)      Return;
(05)    end if
(06)    for all children C of R′ do
(07)      if ( C.subtree_weight >M/2 + x) then
(08)        R′=C;
(09)        Max_weight=M - C.subtree_weight;
(10)        exit for ;
(11)      end if
(12)      if ( C.subtree_weight≥M/2 - x) then
(13)        R′=C;
(14)        Return;
(15)      end if
(16)    if ( C.subtree_weight>max_weight) then
(17)        R′=C;
(18)        Max_weight = C.subtree_weight;
(19)      end if
(20)    end for
(21)  end do
```

4 Performance Evaluations

We evaluate the two algorithms by simulations, and construct several taxonomy hierarchy trees, each of which has 2-10 layers and 200-10K concepts. Each node has 0-30 fan-outs. Peers′ joining and leaving are simulated by modifying weights of the nodes in the taxonomy tree, and numbers of peers added to the tree obey Zipf-like distribution (α=0.8) and random distribution respectively. The simulation results are following:

First, we examine the scalability of the algorithms. A four-layered taxonomy hierarchy semantic space composed of 1706 concepts is partitioned, and results are presented in Fig. 1 and Fig. 2 when M is 100 and x is 10. The clustering result indicates fluctuating ascending trend of the average loads of clusters generated by the

two algorithms, which reflects good scalability of the two algorithms. The curves in the two figures are similar, implying that the algorithms are not sensitive to distribution of the load. It shows that more clusters are produced by SFCA than LBFCA, and therefore, LBFCA is better herein.

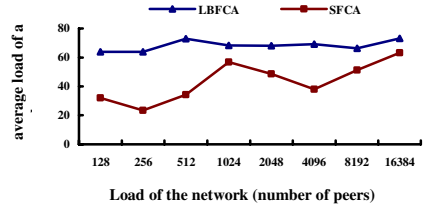


Fig. 1. Results of two clustering algorithms with load data obeying Zipf-like distribution

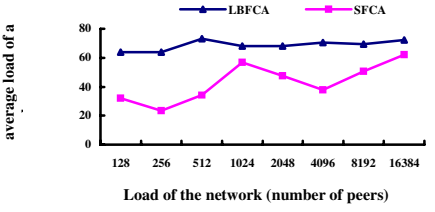


Fig. 2. Results of two clustering algorithms with load data obeying random distribution

Then we evaluate the degree of load balance which reflects the extent of load balance among clusters, computed by the standard deviation of cluster loads, denotes LD, as presented in Formula (1)

$$LD = \sqrt{\sum_{1}^{n} (x_i - \bar{x})^2 / n} \tag{1}$$

Where x_i is the load of the i-th cluster and $\bar{x}$ is the average load of clusters. Comparison between LDs is shown in Fig.3, where M is 100 and x is 10. LD of clusters of LBFCA fluctuates slightly in [0,19], and that of SFCA fluctuates in a relatively wide range in [0,36]. As the load increases, the fluctuating intervals of the two algorithms are gradually being narrowed and approximating each other. In view of LD, LBFCA is better than SFCA, but the advantage of LBFCA over SFCA becomes less and less as the load increases.

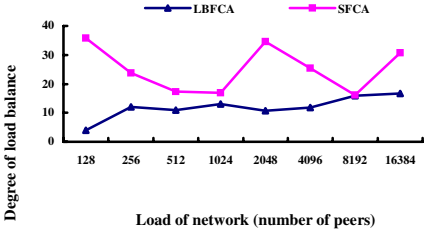


Fig. 3. Comparison between LDs of the two algorithms

Third, we evaluate the similar degree of data semantics within clusters generated by the two algorithms and the influence on clustering when the parameter x varies. For the limitation of paper size, the testing results will not be represented herein.

5 Conclusions

For current semantic clustering algorithms take little account of load balance and are not suitable for taxonomy hierarchy semantic space, two self-organizing clustering algorithms for taxonomy hierarchy are proposed in the paper. The two algorithms dynamically partition a network into distinct semantic clusters by the load and consider semantics of data within clusters and load balance among clusters synthetically. As a result, a good tradeoff is found where both semantics of data and

load balance among clusters are maintained. The experiment indicates considerable performance and scalability of these two algorithms, and thus they can fit well in practice. Future works will focus on the researches of semantic cluster maintenance, semantic routing and search algorithms among clusters.

Acknowledgments. This research was partially supported by the National Natural Science Foundation of China (Grant No. 60473074 and 60573089) and Natural Science Foundation of Liaoning Province (Grant No. 20052031).

References

1. W. Nejdl, B. Wolf, C. Qu etc. EDUTELLA: a P2P Networking Infrastructure based on RDF. In eleventh WWW Conference Proceedings, Hawaii, USA, May 2002.
2. L. Chen, M. T. zsu, V. Oria. Robust and Fast Similarity Search for Moving Object Trajectories, In Proceedings of 24th ACM International Conference on Management of Data (SIGMOD'05), Baltimore, MD, June 2005, pages 491-502.
3. KaZaA. http://www.kazaa.com, May 2003.
4. A. Crespo, H. Garcia-Molina. Semantic Overlay Networks for P2P Systems. Technical report, Stanford University, 2002.
5. A. Loeser. Taxonomy-based Overlay Networks for P2P Systems. In Proceedings of The Second Workshop on Semantics in Peer-to-Peer and Grid Computing, 2004.
6. W. Nejdl, M. Wolpers, W. Siberski etc. Super-Peer-based Routing and Clustering Strategies for RDF-based P2P Networks. In Proceedings of WWW'03, 2003.
7. L. Chen, R. Ng. On the Marriage of Lp-Norm and Edit Distance. In Proceedings of 30th International Conference on Very Large Data Base, Toronto, Canada, August 2004, pages 792-803.
8. Y. Liu , X.Liu , L. Xiao , Li. .Ni, and X. Zhang. Location-Aware Topology Matching in P2P Systems. IEEE INFOCOM, 2004.
9. M. Li, W. C. Lee, and A. Sivasubramaniam. Semantic Small World: An Overlay Network for Peer-to-Peer Search. In Proceedings of the International Conference on Network Protocols, October, 2004.
10. C. Wang, L. Xiao, Y. Liu. Distributed Caching and Adaptive Search in Multilayer P2P Networks. In Proceedings of ICDCS'04, 2004.

Using Categorial Context-SHOIQ(D+) DL to Migrate Between the Context-Aware Scenes

Ruliang Xiao[1,2], Shengqun Tang[1], Ling Li[1], Lina Fang[1], Youwei Xu[1], and Yang Xu[1]

[1] Wuhan University, State Key Lab of Software Engineering, Wuhan 430072, China
[2] Hunan Finance & Economics College, Department of Information & Management, Changsha 410205, China
xiaoruliang@163.com

Abstract. An important issue in semantic web ontology application is how to improve ontological evolvement to fit the semantics of the unceasingly changing context. This paper presents a context-based formalism-Context-SHOIQ(D+) DL which is under the frame of SHOIQ(D+) DL, a kind of description logic, from the category theory point of view. The core part of the proposed formalism is a categorial context based on the SHOIQ(D+) DL, that captures and explicitly represents the information about contexts. Additionally, this paper presents some meta languages about reasoning and knowledge representation, finally discusses context-aware migration between different scenes with the categorial Context-SHOIQ(D+)DL.

1 Introduction

As is known to all, ontologies are shared models of a domain that encode a view which is common to a set of different parties. While the environment keeps unceasingly changing, semantic information in the constant ontology can't reflect the dynamic context. Hence, an important issue in semantic web ontology application is how to improve the ontological evolvement to fit the semantics of the unceasingly changing context.

Category Theory is a mathematics theory, which studies "objects" and "morphisms" between them, and may summarize all knowledge and knowledge processing from the structural angle[1]. The category has the systematic characteristics, on the contrary, SHOIQ(D+) DL that is equivalent to the OWL-DL fragment of OWL (Web Ontology Language)(S stands for the basic ALC DL extended with transitive roles, H stands for hierarchies between roles, O stands for nominals classes whose extension is a single individual, D+ stands for datatypes)[2][3], is dispersed. The context is the structured knowledge that possesses the systematic characteristic. Just as the direction pointed in the literature [1], there exists close corelation between category and knowledge base. Hence, we will investigate the context based on the SHOIQ(D+) DL from the point of view of category theory.

There is not an explicit formalism definition of context in the semantic web so

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 454–459, 2006.

far. In the paper, we define a context as a category, a binary form $\gamma =< \mathcal{C}, \mathcal{R} >$, where $\mathcal{C}$ as a collection of concepts (objects) in the ontology (SHOIQ(D+) DL) which are usually organized in taxonomies, $\mathcal{R}$ as a collection of relations which represents a type of interaction between concepts of the domain, including functions, axioms and so on. If we use $\gamma_1, \gamma_2, \ldots, \gamma_n$ to represent serial contexts, then migrating from a context to another using categorial character, the context-awareness can improve the ontological evolvement. In the paper, we propose a novel context-based formalism– Context-SHOIQ(D+) DL which is under the frame of SHOIQ(D+) DL that leads to the solution of the problem at the bottom level of OWL from the combinative (i.e., category theory and description logic) point of view.

The remainder of the paper is as follows. In section 2, we describe categorial context based on SHOIQ(D+) DL. Section 3 shows Syntax and Semantics of Context-SHOIQ(D+)DL. Section 4 discusses the migration between context-aware scenes. Section 5 compares related works and further points out future work.

2 Categorial Context and Context Migration

General speaking, each conception, and each relation between the conceptions means them in possession of a concrete environment, which is usually called a context. We define a context as a (sub)categorial conceptualization as following:

Definition 1 (Context). *A context is a kind of category, which is given by a binary form* $< \mathcal{C}, \mathcal{R} >$, $\mathcal{C}$ *as a collection of conceptions (objects) in the SHOIQ (D+) DL, denoted by A,B,C,...;* $\mathcal{R}$ *as a collection of relations (roles)(arrows) in the SHOIQ(D+) DL, denoted by R, S,.... From the ontological point of view, a categorial context is showed itself a mini ontology.*

For example, $\gamma.C$, means conception C depends context γ . Supposed that γ represents "Cold-War-era" , C denotes "super country" , $\gamma.C$ means a kind of solid meanings "USA" , and "USSR" .

Definition 2 (Subcontext). *A context U is a subcontext of a context V, if*

(1) $U_0 \subseteq V_0$*, where* U_0,V_0 *is the conceptions set with respect to U, V respectively.*

(2) $U_1 \subseteq V_1$*, where* U_1,V_1 *are the relations (or roles) set with respect to U, V respectively.*

(3) composition and identities in U coincide with those of V.

Definition 3 (Functor). *Given two contexts U and V, a functor F :U→V consists of operations:*

$F_0 : U_0 \rightarrow V_0$ *and* $F_1 : U_1 \rightarrow V_1$*, where* U_0, U_1 *are the conceptions set and roles set of the context U respectively.* V_0, V_1 *are the conceptions set and roles set of the context V respectively, such that*

(1)Provided that f:A → B occurs in the context U, $F_1(f)$*:* $F_0(A) \rightarrow F_0(B)$ *must be in the V.*

(2)For every object A∈U, $F_1(id_A)$=id ($F_0(A)$).

(3)If $g \circ f$ is defined in the context U, then $F_1(g) \circ F_1(f)$ is defined in the context V , and $F_1(g \circ f) = F_1(g) \circ F_1(f)$.

Definition 4 (Context Sequence). *Provided that there is a serial of contexts that construct a concrete environment, we call this environment Context sequence. It can be denoted as $\gamma_1.\gamma_2.....\gamma_n$, where γ_j. as (j=1,2, , n) context. This kind of context sequence is called Context-Aware Scenes too.*

Definition 5 (Context Migration). *If there are two contexts that construct a context sequence, and they can be denoted as $\gamma_1.\gamma_2$, where γ_i (i=1,2) as context, then moving from a context γ_1 to another γ_2 is called Context Migration, it can be denoted as $\gamma_1 \sim \gamma_2$.*

Once context migration takes place, it makes possible to communicate between different scenes on a semantic level by the means of a kind of common understanding about specific terms. Furthermore, those scenes can be constructed with the relevant agents, and can become a context sequence. In section 4, we will discuss the migration between context-aware scenes.

3 Categorial Context-SHOIQ(D+)DL Syntax and Semantics

Based on these definitions of categorial context, we take the notion of context as the core mechanism for representing semantics, and combining with the SHOIQ(D+) DL of OWL, to get an extensive context-SHOIQ(D+) DL. We use "γ" to denote a context.

Definition 6. *Suppose that Γ is a set of contextual names, "γ" a context name within Γ, $\mathcal{C}$ a set of conceptions, $\mathcal{R}$ a set of roles, $\gamma.\mathcal{C}$ means whole conception set $\mathcal{C}$ depending on the context γ. Similarly, roles within context γ denotes $\gamma.\mathcal{R}$, $\mathcal{R}$ holds a subset of $\mathcal{R}^+$ with respected to transitive roles, $\mathcal{R}^+ \sqsubseteq \mathcal{R}$, context γ holds a role name set $\mathcal{R} \cup \{R^- | R \in \mathcal{R}\}$, in order to avoid operating R^{--}, we add a new kind of role inv(), and a new transitivable function tran(). $tran(R) = true$ if and only if $R \in \mathcal{R}^+$, or inv(R)$\in \mathcal{R}^+$. We say a role R, within a context γ, is a simple if the R is not transitive, and it is not compositive by a transitive role R and others, otherwise R is complex.*

General inclusion axioms about role are in possession of as following form:

(1) $\gamma.R \sqsubseteq \gamma.S$, where $R,S \in \gamma.\mathcal{R}$;

(2) $\gamma_1.R \sqsubseteq \gamma_2.S$, where $\gamma_1,\gamma_2 \in \Gamma$, $\gamma_1.R$, $\gamma_2.S \in \gamma.\mathcal{R}$ and $\gamma_1 \sqsubseteq \gamma_2$;

(3) $\gamma_1.R \sqsubseteq \gamma_2.R$, where $\gamma_1,\gamma_2 \in \Gamma$, $\gamma_1.R$, $\gamma_2.R \in \gamma.\mathcal{R}$ and $\gamma_1 \sqsubseteq \gamma_2$;

After added context conception into SHOIQ(D+), conception set within context-SHOIQ(D+) DL holds property as follows:

$$\gamma. < C, R > = < \gamma.C, \gamma.R >$$

"O" stands for nominals (classes whose extension is a single individual) within SHOIQ(D+)DL, and a single individual a can be denoted as a conception by an operator $\{a\}$. Therefore, every conception within SHOIQ(D+) DL must be a conception within Context-SHOIQ(D+) DL:

If $\gamma.C$, $\gamma.D \in \gamma.\mathcal{C}$, $\gamma.R \in \gamma.\mathcal{R}$, S is a simple role depending on a context, and n is a non-negative integer, then $\gamma.C \sqcup \gamma.D$, $\gamma_1.C \sqcup \gamma_2.D$, $\gamma.C \sqcap \gamma.D$, $\gamma_1.C \sqcap \gamma_2.D$, $\gamma_1.\neg C$, $\gamma.R.C$, $\leq_\gamma nR.C$, $\geq_\gamma nR.C$ all are conceptions.

General inclusion axioms about conception are in possession of as following form:

(1)$\gamma.C \sqsubseteq \gamma.D$, where γ is a categorial context, $\gamma \in \Gamma$, and C,$D \in \gamma.\mathcal{C}$;

(2)$\gamma_1.C \sqsubseteq \gamma_2.D$, where γ_1,$\gamma_2 \in \Gamma$, $\gamma_1.C$, $\gamma_2.D \in \gamma.\mathcal{C}$ and $\gamma_1 \sqsubseteq \gamma_2$;

(3)$\gamma_1.C \sqsubseteq \gamma_2.C$, where γ_1,$\gamma_2 \in \Gamma$, $\gamma_1.C \in \gamma.\mathcal{C}$ and $\gamma_1 \sqsubseteq \gamma_2$;

From the above general inclusion axioms, we all know that a Tbox is still a set of general inclusion axioms.

Suppose that I=$\{a,b,c,\ldots\}$, an assertion must be in the form of :

$$a : \gamma.C, (a, b) : \gamma.R, where\ a, b \in I,\ \ C \in \gamma.\mathcal{C}, R \in \gamma.\mathcal{R}.$$

An Abox consists of limited number of assertions.

Definition 7. *A Context-SHOIQ(D+) DL interpretation is a pair I=($\triangle^I$,$\cdot$I), where $\triangle^I$ contains a nonempty set of objects (the resources) of I , is called domain; and a function ($\cdot$I) maps every role to a subset of $\triangle^I \times \triangle^I$ such that, for conceptions C,D, roles R,S, non-negative integer n, #M means the cardinality of M.*

(1)$(\gamma.R)^I$=$\gamma^I.R^I$, for $R \in \mathcal{R}^+$

(2)$(\gamma.R^-)^I$= $\{< \gamma^I.x, \gamma^I.y > | < y, x > \in \gamma^I.R^I\}$

(3)$(\gamma.C \sqcap \gamma.D)^I = \gamma^I.C^I \cap \gamma^I.D^I$

(4)$(\gamma_1.C \sqcap \gamma_2.D)^I = \gamma_1^I.C^I \cap \gamma_2^I.D^I$

(5)$(\gamma.C \sqcup \gamma.D)^I = \gamma^I.C^I \cup \gamma^I.D^I$

(6)$(\gamma_1.C \sqcup \gamma_2.D)^I = \gamma_1^I.C^I \cup \gamma_2^I.D^I$

(7)$(\gamma_1.\neg C)^I = \gamma_1^I \setminus \gamma_1^I.C^I$

(8)$(\forall_\gamma R.C)^I = \{x | \forall y. < x, y > \in \gamma^I.R^I\ implies\ y \in \gamma^I.C^I\}$

(9)$(\exists_\gamma R.C)^I = \{x | \exists y. < x, y > \in \gamma^I.R^I and\ y \in \gamma^I.C^I\}$

(10)$(\leq_\gamma nR.C)^I = \{x | \#\{y. < x, y > \in \gamma^I.R^I and\ y \in \gamma.C^I\} \leq n\}$

(11)$(\geq_\gamma nR.C)^I = \{x | \#\{y. < x, y > \in \gamma^I.R^I and\ y \in \gamma.C^I\} \geq n\}$

An interpretation I satisfies a terminology set Tbox T, if only if every general inclusion axiom $\gamma_1.C \sqsubseteq \gamma_2.D, \gamma_1^I.C^I \subseteq \gamma_2^I.D^I$. we denote this I as model of $T, I \models T$.

For the context γ, we define it as follows that $\gamma^I \subseteq I$,$\gamma^I.C^I = \gamma^I \cap C^I, \gamma^I.R^I = \gamma^I \cap R^I$.

As far as Abox is concerned, single individual $a \in I$, an interpretation for it is $a^I \in \triangle I$, and satisfies some assertions as follows that:

(1)$a : \gamma.C$, if only if $a^I \in \gamma^I \cap C^I$;

(2)$(a, b) : \gamma$, if only if $(a^I, b^I) \in \gamma^I \cap R^I$;

(3)$a \neq b$,if only if $a^I \neq b^I$.

Definition 8 (global-I, local-I). *A **global interpretation** (for short,global-I) for context sequence* $\gamma_1.\gamma_2.\ \ldots\ .\gamma_n$*, is* $(\gamma_1.\gamma_2.\ \ldots\ .\gamma_n)^{global-I} = \gamma_1^I \cup \gamma_2^I \cup \ldots \cup \gamma_n^I$*; On the other hand, a **local interpretation** (for short,local-I) for it, is* $(\gamma_1.\gamma_2.\ \ldots\ .\gamma_n)^{local-I} = \gamma_n^I \backslash (\gamma_1^I \cup \gamma_2^I \cup \ldots \cup \gamma_{n-1}^I)$*, where* γ_j*(j=1,2, , n) as context and* $\gamma_j \in \Gamma$*.*

Definition 9. *A interpretation for context migration* $(\gamma_1.\gamma_2.\ \ldots\ .\gamma_n) \sim \gamma_{n+1}$*, includes global interpretation* $((\gamma_1.\gamma_2.\ \ldots\ .\gamma_n) \sim \gamma_{n+1})^{global-I}$ *and local interpretation* $((\gamma_1.\gamma_2.\ \ldots\ .\gamma_n) \sim \gamma_{n+1})^{local-I}$*.*

According to all these descriptions hereinbefore, there exists subsumption and satisfiability in our Context-SHOIQ(D+)DL as in the SHOIQ(D+)DL, and tableau arithmetic[3] also can be our an important tool for testing conception satisfiability in the new Context-SHOIQ(D+)DL.

4 Discussion of Context-Aware Migration Between Different Scenes

When agents migrating between context-aware scenes, the viewpoints of categorial context and SHOIQ(D+) DL can help us to solve the problem of ontological evolvement as follows: Firstly, during the migrating between different scenes, there must be semantic inconsistency. Each context works as a segment of ontology to encapsulate all information into its own category as if a namespace in a language. Hence, categorial segment envelops this kind of possible inconsistency so as to stop the pervasion of semantic inconsistency. Secondly, thinking of categorial context makes the context to be a structural segment of ontology. The migration (that includes immigration and emigration) between different scenes keeps the structure of a categorical context consistent with others because that the functor of two category is a homomorphism. Finally, there are a global interpretation and a local interpretation for the context sequence and the context migration in the formalism Context-SHOIQ(D+)DL. It enlarges the capability of information representation and knowledge reasoning with Context-SHOIQ(D+)DL.

In Context-SHOIQ(D) DL, because that any context is considered as a category, all concepts, roles (relations), and all assertions depend on a concrete context. Now we present a reasoning example with context-awareness.

Suppose that two categorial context γ_1, γ_2, and context γ_1 defines the concept "bird" as "feathered animal with two wings and two legs, usually able to fly"(see "Oxford Advanced Learner's English-Chinese Dictionary"), while there is an assertion "Ostrich is a kind of bird" in the context γ_2, then what can we infer in our Context-SHOIQ(D+) formalism ?

Because that the inference demands agent's migration between the different scenes, we can denote the current context as γ, and firstly, agent immigrates into scene γ_2, then immigrates into γ_1. Thus,we set up a context sequence $\gamma_1.\gamma_2.\gamma$. Using the *global-I*, *local-I* and some definitions of inference defined by our categorial Context-SHOIQ(D+) DL (see section 3), we can infer that:

(1)Ostrich is a kind of feathered animal.
(2)An Ostrich has two wings.
(3)An Ostrich has two legs.
But we can't draw a conclusion that (4)Ostrich can flies.

In this paper, We have shown how we can deal with the general context-aware case with the categorial Context-SHOIQ(D+) DL. Our goal is to support the unceasingly changing ontological evolvement environment by applications of context-aware mechanism with categorial viewpoint in description logic ontologies.

5 Related Works and Conclusion

Some researchers do much work in the ontological evolvement field. Their works can be classified as two classes: upper-ontology, mappings-OWL. Wang takes advantage of upper ontology for modeling context[4]. Bouquet in the reference [2] proposes that a local ontology in C-OWL is considered as a context. Mappings are made of bridge rules that express semantic relations, and they can be used for representing modular ontologies and combining different viewpoints. These approaches above are mainly at the angle of technology, while our work mainly focuses on the meta-level methodology.

In the paper, Context-SHOIQ(D+) DL, a novel categorial context formalism is proposed. The aim is to do some research for methodology of ontology evolvement. The practice proves that our research is valuable. This is only the beginning to explore the context of an evolving ontology, and a lot of challenges such as reducing and reasoning of knowledge while ontology collaborates with dynamic foreign context-aware agents, need to be done by our further efforts.

References

1. Ru-Qian Lu.: Towards a Mathematical Theory of Knowledge. J.Comput. Sci. & Tech. Nov. 2005, Vol.20, No.6, pp.751-757.
2. Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen et al.: C-OWL: Contextualizing Ontologies. D. Fensel et al. (Eds.): ISWC 2003, LNCS 2870, pp. 164-179.
3. Ian Horrocks.: Description Logics in Ontology Applications. TABLEAUX 2005, LNAI 3702, pp. 2-13.
4. Xiao Hang Wang, Da Qing Zhang, Tao Gu,et al.: Ontology Based Context Modeling and Reasoning using OWL. Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'2004).

SCEND: An Efficient Semantic Cache to Adequately Explore Answerability of Views*

Guoliang Li, Jianhua Feng, Na Ta, Yong Zhang, and Lizhu Zhou

Department of Computer Science and Technology, Tsinghua University, Beijing, China
{liguoliang, fengjh, dcszlz}@tsinghua.edu.cn,
zhangy@tsinghua.org.cn, dan04@mails.tsinghua.edu.cn

Abstract. Maintaining a semantic cache of materialized XPath views inside or outside the database, is a novel, feasible and efficient approach to accelerate XML query processing. However, the main problems of existing approaches are that, they either can not exploit sufficient potential cached views to answer an issued query or need too much time for cache lookup. In this paper, we propose, SCEND, an efficient Semantic Cache based on dEcompositioN and Divisibility, which adequately explores the answerability of views, and speeds up cache lookup dramatically. We decompose complex XPath queries into some much simpler and tractable ones to improve cache hit rate, moreover, we introduce a notion of the divisibility between two positive integers to accelerate cache lookup. In addition, we present a new replacement technique for SCEND to improve performance for caching. We experimentally demonstrate the efficiency of our caching techniques and performance gains obtained by employing such a cache.

1 Introduction

XML is increasingly being used in data intensive applications and has become de facto standard over the Internet. Major database vendors are incorporating native XML support in the latest versions of their relational database products. Data meant for web services and data exchange applications are often most conveniently stored directly as XML. In this scenario, the number and size of XML databases are rapidly increasing, and XML data becomes the focus of query evaluators and optimizers. In a relational database system, the in-memory buffer cache is crucial for good performance. A similar buffer cache can be employed in XML systems. However, XML query processing presents a different set of challenges. Query execution on semi-structured data is intrinsically harder to optimize. The buffer cache reduces the disk I/O cost, but not the computational cost. Maintaining a semantic cache of query results has been proposed [DFJ+96, MS05].

* Supported by the National Natural Science Foundation of China under Grant No. 60573094, Tsinghua Basic Research Foundation under Grant No. JCqn2005022, Zhejiang Natural Science Foundation under Grant No. Y105230, and 973 Program under Grant No.2006CB303103.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 460 – 473, 2006.

The cached queries are basically materialized views, which can be used in query processing. Thus, at any moment, the semantic cache contains some views $\{V_1,V_2,\ldots,V_n\}$. When the system has to evaluate a new query Q, it inspects each view V in the cache and determines whether it is possible to answer Q from the result of V. We say that view V answers query Q if there exists some other query *CQ* which, when executed on the result of V, gives the result of Q. We write this as $CQ \bullet V \equiv Q$, and we call *CQ* the Compensating Query. When some cached view answers an issued query, we have a hit; otherwise we have a miss.

There are several applications for such a semantic cache. First, consider its use inside the XML database system. Suppose some query Q is answered by view V, with compensating query being *CQ*. Then Q is answered by executing *CQ*, which is simpler than Q, on the result of V that is a much smaller XML fragment than the original data instance. This can result in a significant speedup, as we show in our experiments. The semantic cache can also be maintained at the application tier, and there will be additional savings for a hit, from not having to connect to the backend database. For a heavily loaded backend server, these savings can be large. This kind of middle-tier caching has become popular for web applications using relational databases [LKM+02]. Finally, the semantic cache can also be employed in a setting like distributed XQuery [RBH+04] where sub-queries of a query might refer to remote XML data sources, connected over a WAN. Here, a sub-query that hits in the local cache will not have to be sent over the network, and the savings can be huge. Checking query/view answerability requires matching operations between the tree patterns of the query and view. Looking up the semantic cache by iterating over all views will be very inefficient when the number of views is large. [MS05] maintains XML views in relational database, and it needs string match and exhaustive test for cache lookup, and when the cached views are complicated, it is too inefficient and even infeasible, further it can not exploit sufficient views to answer an issued query.

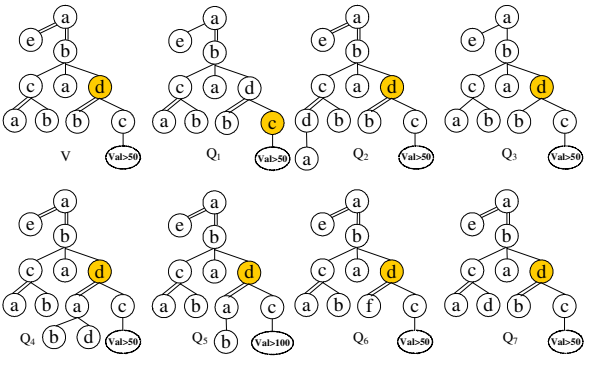


Fig. 1. One cached view V and seven queries Q_1-Q_7

We present some examples of how queries are answered from the cache and disadvantage of existing studies. They will make clearer the challenges in doing efficient lookup in a large cache, and also illustrate query rewriting for cache hits. Suppose there are one cached view V and seven queries $Q_1,\ldots,Q_7$ as shown in Fig. 1.

It is obvious that the result of V contains the results of Q_4 and Q_5 respectively. For Q_4, CQ_4 is d[.//a[b][d]], and we only check whether element d in the result of V satisfies CQ_4, moreover, querying CQ_4 on V is much easier than querying Q_4 on original instance; for Q_5, CQ_5 is d[.//a[b]][c>100], that is, we only check whether element d in the result of V satisfies CQ_5. In this way, we need not query Q_4 and Q_5 through general XML query processing method, which is more complicated than the operation upon the view result, and it saves many times of I/O. It's easy to find out that the result of V also contain the results of Q_1,Q_2,Q_3 respectively. But in general naïve cache system based on exact string match of the query and view, neither of Q_1,…,Q_5 can be answered by V, since they are not string match with V. Even if in [MS05], Q_1,Q_2,Q_3 can't be answered by V. Moreover, the method in [MS05] is not efficient, since Q_6,Q_7 are also string match with V in its first step, but it's obvious that V cannot answer Q_6, Q_7. Further, it can't support // and * in the XPath queries well.

In this paper, we will demonstrate how to speed up cache lookup and how to adequately explore the answerability of views to improve cache hit rate. In our method, V can answer Q_1,…,Q_5, but Q_6, Q_7 can be excluded easily.

Our Contributions:

- We describe a novel cache organization and demonstrate that cache lookup in our method is more efficient, even when there are several hundreds of thousands of cached views.
- We propose SCEND, an efficient Semantic Cache based on dEcompositiN and Divisibility, to improve query/view answerability through two techniques: checking the divisibility of two positive integers to speed up view selection and decomposing complicated XPath queries into some much simpler ones to explore the answerability of views. The performance of SCEND is more efficient than the existing state-of-the-art approach.
- An efficient replacement strategy for SCEND is proposed, which can improve performance dramatically for caching.

The rest of this paper is organized as follows: we start with related work in Section 2. Section 3 introduces some preliminaries and describes how to exploit query/view answerability. In section 4, a strategy for efficient cache lookup is presented. We propose a method for cache replacement in section 5. In Section 6, we present our experimental results, and then conclude in Section 7.

2 Related Work

A related problem is that of containment between XPath queries. In [MS02], this problem is shown to be coNP-complete. A polynomial time algorithm is also presented for checking containment, which is sound but not complete. Balmin et al. [BOB+04] also propose using materialized XPath views to answer queries. However, they do not address the problem of speeding up cache lookup when there are a large number of views to consider. Further, their criterion for query/view answerability is exactly containment between them. Application-tier caching for relational database has received a lot of attention lately, in the context of database-driven web sites [LKM+02, YFI+00]. Chen et al. [CR02] propose a semantic cache of XQuery views. It focuses on various aspects of the query/view matching problem, which is harder for

XQuery. Having XQuery views will result in smaller cached results and concise rewritten queries, which will speed up cache hits. However, optimizing lookup is harder due to the more complex matching involved, and lookup is likely to become the bottleneck when there are a large number of cached views to consider. Mandhani et al. [MS05] propose a method about how to find V to answer Q, but when there are large numbers of views in cache, it is inefficient. Further, it may involve cache miss for some queries, which can be answered by views in cache, thus in this paper we will demonstrate how to speed up view selection, and explore the answerability of views.

3 Preliminaries

We now look at query/view answerability. The question that we consider is this: Given a view V and a query Q, does V answer Q (a view V contains the cached query and its corresponding cached result, and when there is no ambiguity, we may also refer to V as the cached query), and if yes, then what should *CQ* be, so that $CQ \bullet V \equiv Q$.

More importantly, note that the result of V contains that of Q doesn't imply V can answer Q. For example, suppose V=*a*[c]//*b*/*d*[e], and Q=*a*[c]/*b*/*d*[e], the result of Q is contained in that of V. If only the result of *d (*result node*)* of V is cached, we can not answer Q using the result of V, because we don't know which *d* element in cache satisfies *a*/*b*/*d*, since there are no results of nodes *a, b* in the cache. However, if the results of nodes *a, b* in V are also cached, we can use the result of V to answer Q by employing some policies: first element sets S_a, S_b are obtained by selecting elements that satisfy *a*/*b* on the cached results of *a*, *b* respectively, and then the result set S_d is obtained by selecting the element, which has a parent in S_b, in the cached result of *d*.

3.1 Tree Pattern Inclusion

Definition 1 TP (Tree Pattern). *A tree pattern is a labeled tree TP = (V, E), where V is the vertex set, E is the edge set. Each vertex v has a label, denoted by v.label, whose value is in tagSet∪{"*"}, where tagSet is the set of all element names in the context. An edge can be a child edge representing the parent-child relationship (/) or a descendant edge representing the ancestor-descendant relationship (//).*

In this paper, XPath queries are naturally represented as tree patterns, and we are going to reason using these, to derive a sound procedure for answering this question, whether V can answer Q and how to construct *CQ*.

We present an example showing how an XPath query is represented as a tree pattern. Fig.1 shows the tree pattern for V=a[.//e]//b[c[.//a][b]][a]/d[.//b][c>50]. Child and descendant axes are respectively shown by single slash (/) and double slashes (//). The ellipse-shaped nodes are predicates qualifying their parent nodes. Note that the result node *d* of the query is marked by solid circle.

For any view V and query Q, V can answer Q only if the result of Q is contained in that of V. Query P is contained in query Q if and only if there is a homomorphism from Q to P [CM77], and there is a classical characterization result for conjunctive queries against relational databases. Similar characterizations can also be given for some tree patterns. For simplicity we define homomorphism only via tree patterns.

Definition 2 (Homomorphism). *A homomorphism h from Q to P maps each node of Q to a node of P such that the following conditions hold:*

i) The root of Q must be mapped to the root of P.
ii) If (u, v) is a child-edge of Q then (h(u),h(v)) is a child-edge of P.
iii) If (u, v) is a descendant-edge of Q then h(v) has to be below h(u) in P.
iv) If u is labeled with e≠ then h(u) also has to carry label e.*

Definition 3 (Tree Pattern Inclusion). *Let P and Q be two Tree Patterns, if there is a homomorphism h from Q to P, then P is included in Q.*

Definition 4 (Restrictive Tree Pattern Inclusion). *Let P and Q be two Tree Patterns, P is restrictively included in Q, denoted by P⊆Q, if satisfy: ∃h, which is a homomorphism from Q to P, and ∀u,v∈Q, u≠v, then h(u)≠h(v).*

Theorem 1. *If Q is restrictively included in V, the result of V contains that of Q.*

Theorem 1 means that, if Q is restrictively included in V, we can use the result of V to answer Q, however, it is a sufficient but not necessary condition. Moreover, Tree Pattern Inclusion is very complicated to check, thus we introduce Restrictive Tree Pattern Inclusion. The difference of them is: the latter must assure *h* is an injection.

Example: in Fig. 1, as $Q_1,Q_2,\ldots,Q_5$ are restrictively included in V, so we can use the result of V to answer them. ∀i, $1\le i\le 5$, we need construct compensating queries CQ_i, which satisfies $CQ_i \bullet V \equiv Q_i$, to answer Q_i, and querying CQ_i on V is much easier to process than querying Q_i directly on original instance.

3.2 Criteria for Answerability

Definition 5 (Tree Pattern's MainPath). *A Tree Pattern's MainPath is the path from the root node to the result node, in the query tree pattern. Nodes on this path are* ***axis nodes****, while the others are* ***predicate nodes****. The Query* ***Depth*** *is the number of axis nodes.*

Definition 6 Prefix(Q,k). *is the query obtained by truncating query Q at its k-th axis node. The k-th axis node is included, but its predicates are not.* ***Preds****(Q,k) is the set of predicates of the k-th axis node of Q.* ***Infix****(Q,k) is the k-th axis node and its predicates.* $\boldsymbol{Q_k}$ *denotes the subtree of Q rooted at its k-th axis node.*

To improve cache hit rate and avoid huge storage, we cache the results of all the axis nodes in this paper. For example: consider the query Q=*a*[v]/*b*[@w="val1"][x[.//y]]//*c*[z>val2]. Depth of Q is three, its *MainPath* is *a*/*b*//*c*, and we cache the results of its axis nodes *a*, *b*, *c*. Prefix(Q, 2)=*a*[v]/*b*; Q_2=*b*[@w="val1"][x[.//y]]//*c*[z>val2]; Preds(Q,2)={@w="val1",x[.//y]}; Infix(Q,2)=*b*[@w="val1"][x[.//y]].

Theorem 2. *If ∀k, $1\le k\le D$, Infix(Q,k)⊆Infix(V,k), MainPath(Q,D)⊆ MainPath(V)*[1]*, and the k-th axis nodes of Q and V have the same label, then Q ⊆ V.*

Proof: As ∀Infix(Q,*k*)⊆Infix(V,*k*), so $\exists h_k$, h_k is a homomorphism from Infix(V,*k*) to Infix(Q,*k*), and h_k satisfies Definition 4. As MainPath(Q,*D*)⊆MainPath(V), so ∃*h*′, *h*′

[1] *MainPath(Q, k) is the path of Q from the 1-st axis node to the k-th axis node, and MainPath(V) is the MainPath of V. In this paper, the depth of V is denoted by* ***D****, and Infix(Q,* $\boldsymbol{D}$*)=*Q_D.

is a homomorphism from MainPath(V) to MainPath(Q,*D*) and satisfies Definition 4. $\forall u$, which is a node in MainPath(V), u must be an axis node, without loss of generality, suppose u is the *k-th* axis node, thus u is the root node of Infix(V,k), as the *k-th* axis node of Q and V have the same label, so $h'(u)=h_k(u)=u$. Thus, we can construct h: $\forall v \in$ V, there must exist only one k, $v \in$ Infix(V, k), let $h=h_k$. It is obvious that h is a homomorphism from V to Q and satisfies Definition 4, therefore, Q⊆V. □

Definition 7. *V→Q (V answers Q) iff. V and Q satisfy:*

i) MainPath(Q, D)⊆ MainPath(V).
*ii) ∀k,*1≤*k*≤*D, Infix(Q,k)⊆Infix(V,k), the k-th axis nodes of Q, V have the same label.*

Corollary 1. *If V→Q, the result of V contains that of Q.*

Corollary 1 is very easy to prove through Theorem 1 and 2. Therefore, in this paper, if ∃V in the semantic cache, which satisfies V→Q, we say that Q can be answered by the cached views; otherwise Q can not be answered. For example, in Fig. 1, ∀i, 1≤i≤5, V→Q_i, thus Q_1, Q_2,…,Q_5 can be answered by V, but Q_6, Q_7 can't.

For any tree patterns V and Q, it is much easier to check whether V→Q is true according to Definition 7 than whether Q⊆V is true according to Definition 4. Moreover, *MainPath* only contains /, // and *, but not [], and the complexity of the containment of $XPath^{\{/,//,*\}}$ is proved to be Polynomial[MS99], thus it is easy to check whether MainPath(Q,*D*)⊆MainPath(V) is true. In addition, Infix(Q,k) is much simpler than Q, thus it is easy to check whether Infix(Q,k)⊆Infix(V,k) is true.

To answer Q, we must construct compensating query, *CQ*, which satisfies *CQ*•V≡Q, and we will present how to construct *CQ* in section 3.3. However, if there are more than one cached quires that satisfy V→Q, it is better to select V, which has the longest depth, and this leads to less additional operations on V to answer Q.

3.3 Compensating Query

Suppose ∀k,1≤k≤D, VR^k is the cached result of the *k-th* axis node of V. CQ^k=Infix(Q, k), and CQ^k is considered as a query taking the *k-th* axis node as result node, CQ^{MP} = MainPath(Q,*D*), which takes the *D-th* axis node as result node. $Q^D=Q_D$, Q_D takes the *D-th* axis node as result node, but Q^D takes the result node of Q as its own result node.

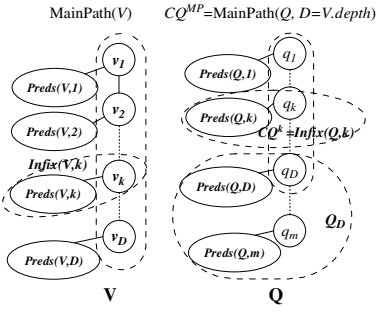


Fig. 2. Tree patterns V and Q

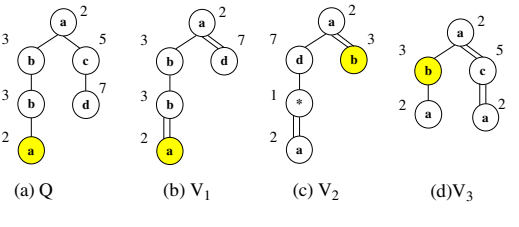


Fig. 3. One query and three views

Theorem 3. $(CQ^{MP}\bullet(CQ^1\bullet VR^1, CQ^2\bullet VR^2, \ldots, CQ^D\bullet VR^D))\bullet Q^D \equiv Q$.

Proof(sketch): As Infix(Q,*k*)⊆Infix(V,*k*), so $CQ^k\bullet VR^k \equiv$Infix(Q,*k*). As MainPath(Q, *D*)⊆MainPath(V), so $CQ^{MP}\bullet(CQ^1\bullet VR^1, CQ^2\bullet VR^2, \ldots, CQ^D\bullet VR^D) \equiv$**Prefix**(Q,*D*), thus, $(CQ^{MP}\bullet(CQ^1\bullet VR^1, CQ^2\bullet VR^2, \ldots, CQ^D\bullet VR^D))\bullet Q^D \equiv Q$.

Theorem 3 tells how to construct *CQ* to answer Q on the cached result of V as shown in Fig. 2. $CQ^k\bullet VR^k$ denotes the result of Infix(Q,*k*) by querying CQ^k on VR^k. As MainPath(Q,*D*) and MainPath(V) may be not the same, so we need to get the result of *D-th* node of Q by querying CQ^{MP}, and $CQ^{MP}\bullet(CQ^1\bullet VR^1, CQ^2\bullet VR^2, \ldots, CQ^D\bullet VR^D)$ denotes the result of the *D-th* axis node of Q by querying CQ^{MP} on each $CQ^k\bullet VR^k$, lastly since *D-th* axis node of Q may not be the result node of Q, thus the final result of Q can be gotten by querying Q^D on the result of the *D-th* axis node of Q.

Example: in Fig.1, for Q_3, CQ_3^1=a[e], CQ_3^2=b[c[a][b]][a], CQ_3^3=d[b][c>50], and CQ_3^{MP}= a/b/d, $Q_3^3 = Q_{3_3}$ =***d***[b][c>50]; for Q_1, Q_{1_3} =***d***[.//b][c>50], Q_1^3=d[.//b]/***c***[>50].

4 Cache Organization and Cache Lookup

4.1 Tree Pattern's Prime ProducT

We present how to check whether V can answer Q in section 3, however, if there are hundreds of thousands of views in semantic cache, Definition 7 is not so easy to verify. To accelerate view selection, we introduce Definition 8 and Theorem 4.

Definition 8 ***PPT (Tree Pattern's Prime ProducT).*** *Different nodes in tree patterns are assigned with distinct prime numbers (nodes with the same label are assigned with a same prime number). A Tree Pattern TP's Prime ProducT(PPT) is defined as:*

$TP_{PPT} = \prod_{e\in TP} v_p * v_c$ *,where e is any edge of TP, and* v_p, v_c *are assigned prime numbers of e's two vertexes.*

Example: in Fig. 3, we assign distinct prime numbers as: a(2), b(3), c(5), d(7), *(1), and wildcard * is always assigned with 1, since * can be matched by any label. Q_{PPT}= (2*3)*(3*3)*(3*2)*(2*5)*(5*7) = 113400; V_{1PPT} = (2*3)*(3*3)*(3*2)*(2*7)=4536; V_{2PPT}=(2*7)*(7*1)*(1*2)*(2*3)=1176; V_{3PPT}=(2*3)*(3*2) *(2*5)*(5*2)=3600.

Theorem 4. ∀*Two tree patterns P, Q, if P is restrictively included in Q(P⊆Q), then* $Q_{PPT} | P_{PPT}$ *must be true, where X|Y denotes that,* ∀*integers X,Y, Y can be exactly divided by X, that is, there is a third integer Z, Y=XZ).*

Proof: Suppose EQ^i={($h(u)$, h(v)) | h is a homomorphism from Q to P, which satisfies: if $u\neq v$, $h(u)\neq h(v)$}. ∀u, if u=*, $u.prime$=1; else $u\neq$*, $h(u)=u$, so $u.prime | h(u).prime$, thus ∀$(u,v)\in Q$, $u.prime*v.prime | h(u).prime*h(v).prime$. So, ∃ γ′, $h(u).prime*h(v).prime = \gamma' * u.prime*v.prime$. Thus, ∃ γ, which satisfies,

$$\prod_{((h(u),h(v))\in EQ^i} h(u).prime * h(v).prime = \gamma * (\prod_{(u,v)\in Q} u.prime * v.prime)$$

$$P_{PPT} = (\prod_{((h(u),h(v))\in EQ^i} h(u).prime * h(v).prime) * (\prod_{(h(u),h(v))\notin EQ^i} h(u).prime * h(v).prime)$$

$$= \gamma * \left(\prod_{(u,v) \in Q} u.prime * v.prime \right) * \left(\prod_{(h(u),h(v)) \notin EQ^i} h(u).prime * h(v).prime \right)$$

$$= Q_{PPT} * \gamma * \left(\prod_{(h(u),h(v)) \notin EQ^i} h(u).prime * h(v).prime \right)$$

Therefore, $Q_{PPT} | P_{PPT}$ is true. □

Corollary 2. *If $V_{PPT} | Q_{PPT}$ is false, V can not answer Q.*

Theorem 4 gives a necessary but not sufficient condition for P is restrictively included in Q. Corollary 2 tells which V can't answer Q, and in this way, we can exclude many views that cannot answer Q through V and Q's PPT according to Corollary 2.

Example: in Fig.3, as $V_{2PPT} | Q_{PPT}$ and $V_{3PPT} | Q_{PPT}$ are both false, so Q is not included in V_2 and V_3, in other words, V_2 and V_3 can't answer Q. As Q is included in V_1, so $V_{1PPT} | Q_{PPT}$ is true. In the same way, in Fig. 1, as $V_{PPT} | Q_{6PPT}$ and $V_{PPT} | Q_{7PPT}$ are false, so it is easy to exclude Q_6 and Q_7 directly since they don't satisfy Corollary 2.

4.2 SCEND: An Efficient Semantic Cache

To improve cache hit rate and avoid huge storage, we cache the results of all the axis nodes. In this way, it can improve the performance of the semantic cache. For example: in Fig. 1, if we only cache the result of *d* (result node) of V, V only can answer Q_4 and Q_5. However, if we cache the results of *a*, *b*, *d* in cache replacing the result of only *d*, V can answer $Q_1, Q_2, \ldots, Q_5$, and cache hit rate is improved. Moreover, it doesn't involve huge storage, and storage space of this is linear to cache only *d*. To accelerate cache lookup, Theorem 5 is introduced to check whether V can answer Q.

Theorem 5. *V can answer Q, if V and Q satisfy,*

i) $V_{PPT} | Q_{PPT}$ is true and $MainPath(V)_{PPT} | MainPath(Q,D)_{PPT}$ is true;

ii) $MainPath(Q,D) \subseteq MainPath(V)$;

iii) $\forall\, k, 1 \le k \le D = V.depth$

1) the k-th axis-nodes of Q and V have the same label;

2) $Infix(V,k)_{PPT} \mid Infix(Q,k)_{PPT}$ is true;

3) $Infix(Q,k) \subseteq Infix(V,k)$;

Theorem 5 is easy to prove through Definition 7, Corollary 1 and Theorem 4. However, checking whether V can answer Q through Theorem 5 is much easier than through Definition 7, because if one of the three conditions is false, V can't answer Q, thus, some views that can't answer Q can be excluded in advance.

In this way, SCEND, an efficient Semantic Cache based on dEcompositioN and Divisibility is proposed. To accelerate view selection, we build index on V, which can speed up cache lookup.

Suppose V^{MStr} is the string of MainPath(V), and $V^{MStr} = s_1 s_2 \ldots s_D$, $\forall 1 \le k \le D$, s_k is the *k-th* axis node of V, where *D* is the depth of V. We can define an equivalence relation ~ on V, $V_i \sim V_j$ iff. $V_i^{MStr} = V_j^{MStr}$. It is obvious that ~ satisfies reflexive, symmetric and transitive, therefore, we can partition views in the semantic cache into different equivalence classes according to ~, and there are no common elements between different classes. We build indices for those strings (such as V^{MStr}), and each string V^{MStr} is associated with some identifiers of views, which are in the same class as V.

Suppose str(Q, k)=$s_1s_2\ldots s_k$, $\forall 1\le i\le k\le$Q.*depth*, s_i is the *i-th* axis node of Q. For any query Q, we search the index to find whether there is a string which is the same as str(Q,k), if there is, and there is a view V, which can answer Q, in its associated views, then we will use V to answer Q; otherwise, we find whether there is a string which is the same as str(Q,k-1), and so on.

In this way, the number of views in each equivalence class is less than that of all views in the semantic cache, and V that has the longest depth is chosen to answer Q.

SCEND operates in two phases:

i) view selection, select V which has the longest depth to answer Q (Fig.4)

ii) reconstruct view, construct CQ for V, and CQ• V ≡ Q (Fig.5)

Algorithm 1 SCEND: ***View-Selection***
Input: views in the semantic cache and a query Q
Output: V can answer Q, which has the longest depth.
Initialization: indices is built on V
1. **for** k=Q.*depth* to 1 **do**
2. PVset={V|V^{MStr}=str(Q,k)};
3. **for** each V in PVset **do**
4. **if** $V_{PPT}|Q_{PPT}$ is false||$(V^{MP})_{PPT}|(Q^{MP})_{PPT}$ is **false then**
//V^{MP}=MainPath(V); Q^{MP}=MainPath(Q,D);
5. **continue**; *//go to 3*
6. **if** MainPath(Q,D)⊆MainPath(V) **then**
7. **for** i=1 to D=*V.depth* **do**
8. **if** Infix(V,i)$_{PPT}$ | Infix(Q,i) is **false**||the *i-th* axis nodes of Q and V are not same **then**
9. **break**; *//go to 3*
10. **else if** Infix(Q,i)⊆Infix(V,i) **then**
11. **continue**; *//go to 7*
12. **else break**; *//go to 3*
13. **if** i>*V.depth* **then** *//V can answer Q*
14. **return** V;
15. **return** null.

Fig. 4. *View-Selection* Algorithm

Algorithm 2 SCEND: ***Reconstruct-View***
Input: V: the selected view, Q: a query;
Output: the query result of Q on the result of V.
Initialization: CQ^k= Infix(Q,k); VR^k=the cached result of the *k-th* axis node of V; D=*V.depth*.
1. **for** k=1 to D **do**
2. **if** StringMatch(Infix(V,k),Infix(Q,k))=**true then**
3. $QR^k=VR^k$;
4. **else** QR^k=GeneralQuery(CQ^k,VR^k);
//$QR^k=CQ^k\bullet VR^k$
5. FVR^D=PathStack(CQ^{MP},QR^1,QR^2, **…**,QR^D);
//$FVR^D=CQ^{MP}\bullet(QR^1=CQ^1\bullet VR^1,\ldots, QR^D=CQ^D\bullet VR^D)$
6. **if** Q.*depth*==D **then return** FVR^D;
7. **else return** FVR=GeneralQuery(Q^D, FVR^D);
//$FVR=Q^D\bullet FVR^D$

Function StringMatch(V,Q)
Input: a cached query V; a query Q
Output: Boolean: if V and Q have the same normal form, then return true, else return false.
1. Normalize-Tree(V); Normalize-Tree(Q);
2. **if** V==Q **then return true**;
3. **else return false**;

Fig. 5. *Reconstruct-View* Algorithm

View-Selection(Fig.4) presents how to select V that can answer Q and *Reconstruct-View* (Fig.5) describes how to construct *CQ* to answer Q. *View-Selection* can skip views that can't answer Q in advance. *View-Selection* adds views that have the same string as str(Q,k) into PVset in line 2 through index, skips V which doesn't satisfy $V_{PPT}|Q_{PPT}$ and MainPath(V)$_{PPT}$|MainPath(Q,D)$_{PPT}$ in line 4, where it only needs to check the divisibility of two positive integers, then skips V which doesn't satisfy MainPath(Q,D)⊆MainPath(V) in line 6, lastly skips V which doesn't satisfy iii) of Theorem 5 in line 7-12, if there is a V that can answer Q, returns V in line 13-14.

Example: in Fig.1, when Q_6, Q_7 are issued, as $V_{PPT}|Q_{6PPT}$, $V_{PPT}|Q_{7PPT}$ are false, thus we can make sure that V can not answer Q_6, Q_7 immediately through Corollary 2 (line 4 of *View-Selection*). In Fig. 3, when querying Q, Q.*depth*=4, only $V_1{}^{MStr}$= str(Q,4), so we only check whether V_1 can answer Q, but skip V_2 and V_3 through the index.

Table 1. Normalize the tree pattern

Procedure Normalize-Tree(T)	**Procedure** Normalize-Node(x)
1. Let T have axis(x_1,…,x_k);	1. Let x have predicate node children p_1,…,p_n;
2. For i=1,…,k Normalize-Node(x_i);	2. For i=1, …, n Normalize-Node(p_i);
3. Concatenate the node labels of x_1,…,x_k, with appropriate axes in between.	3. Sort p_1,…,p_n lexicographically by their labels; 4. For i=1,…,n append [p_i.label] to x.label;

There may be some redundancy in the result of V which is selected in *View-Selection*, thus we need to construct *CQ*, which satisfies $CQ \bullet V \equiv Q$. To present *Reconstruct-View*, we introduce the normal form, which is a unique XPath query string of a tree pattern, since there are some tree patters which are equivalent but in different expressions [MS05]. We can get the normal form of a tree pattern through calling Normalize-Tree. Procedure Normalize-Tree in Table 1 is used to normalize a tree pattern to its normal form. For example, P=a[c[d]/e]/b[e[f]/g], Q=a[c[d][e]]/b[e[f][g]], although P and Q are not the same, they are equivalent, and Q is the normal form of P.

To construct *CQ*, we first construct CQ^k, CQ^{MP} and Q^D as Theorem 3, and then reconstruct them to get *CQ*. If Infix(V,*k*) and Infix(Q,*k*) are in the same normal form, then CQ^k=Infix(Q,*k*), so $QR^k=CQ^k \bullet VR^k=VR^k$ as shown in line 2-3 in *Reconstruct-View*, where VR^k is the result of *k-th* axis node of V, and QR^k is the result of querying CQ^k on VR^k; otherwise, *Reconstruct-View* will query each CQ^k on corresponding sub-view VR^k to get QR^k by calling GeneralQuery in line 4 (for CQ^k, its result node is the *k-th* axis node). GeneralQuery(CQ^k,VR^k) is used to get the result of querying CQ^k on VR^k, which is similar to general XML query processing method, but it is far easier than directly querying Q, and in this paper we use a holistic twig join algorithm [JLW+03] to implement it. Then, we synthesize the final result of the path in Q from the *1-st* axis node to the *k-th* axis node through algorithm PathStack[BKS02] in line 5, which only costs ($\sum|QR^k|+|FVR^D|$). Lastly, if Q.*depth*=*D*, that is, *D-th* axis node is the result node of Q, thus *Reconstruct-View* returns FVR^D directly in line 6; otherwise it gets the result of Q by querying Q^D on FVR^D through calling GeneralQuery in line 7.

Example: in Fig. 1, if Q_1,Q_2,…,Q_7 are issued, we find that V can answer Q_1,…,Q_5, but can't answer Q_6 and Q_7. ($V_{PPT}|Q_{6PPT}$ and $V_{PPT}|Q_{7PPT}$ are both false, so they are excluded directly in line 4 of *View-Selection*). Then we call *Reconstruct-View* to construct *CQ* to answer Q_1,…,Q_5. For Q_1, StringMatch(Infix(V,1),Infix(Q_1,1)) is not true, so we get QR_1^1 by querying CQ_1^1=Infix(Q_1,1) on VR^1, and StringMatch (Infix(V,2), Infix(Q_1,2)) and StringMatch(Infix(V,3),Infix(Q_1,3)) are both true, so $QR_1^2=VR^2$, $QR_1^3=VR^3$, and then we synthesize QR_1^k to get FVR^D in line 5. As $Q_1.depth(4) \neq D(D=V.depth=3)$, so we call GeneralQuery to get the result of ***c*** (result

Table 2. Compensating queries of Q_1-Q_5

Query	CQ^{MP}	CQ^1	CQ^2	CQ^3	Q^D
Q_1	a//b/d	a[e]	b[c[.//a][b]][a]	d[.//b][c>50]	d[.//b]/***e***[>50]
Q_2	a//b/d	a[.//e]	b[c[.//d[a]][b]][a]	d[.//b][c>50]	***d***[.//b][c>50]
Q_3	a/b/d	a[e]	b[c[a][b]][a]	d[b][c>50]	***d***[b][c>50]
Q_4	a//b/d	a[.//e]	b[c[.//a][b]][a]	d[.//a[b][d]][c>50]	***d***[.//a[b][d]][c>50]
Q_5	a//b/d	a[.//e]	b[c[.//a][b]][a]	d[.//a[b]][c>100]	***d***[.//a[b]][c>100]

node) of Q_1. For Q_2, $QR_2^2=CQ_2^2 \bullet VR^2$, $QR_2^1=VR^1$, $QR_2^3=VR^3$. For Q_3, CQ_3^1=a[e], CQ_3^2=b[c[a][b]][a], CQ_3^3=d[b][c>50], CQ_3^{MP}=a/b/d. Table 2 shows each CQ^k, CQ^{MP} and Q^D for Q_1-Q_5, and the underlined ones need not be constructed in our algorithm.

5 Cache Replacement

If the space for admitting a new query and its result is not sufficient, some cached queries and their corresponding results need to be replaced. We integrate LFU and LRU into LFRU in this paper, that is, we always replace the cached query which is the least frequently and recently used query. In our approach, the cached queries are classified into two parts via the latest visited time, one part is the recent 20% visited queries and the other part is the other 80% queries. Each of the two parts is assigned with an importance ratio α,β. Suppose the queries in the database is $\{q_1, ..., q_n\}$, we have record visited frequency vf_i, latest visited time lvt_i, fetch delay time fdt_i (execution time without cache) and occupied result size rs_i for each query q_i. The query q_i that $(\gamma_i * vf_i * fdt_i)/rs_i$ is minimal will always be first replaced, where γ_i is α/β if q_i is in 20% recent query, otherwise γ_i is 1. Recent queries are more important generally, thus, α/β should be larger than 1.

6 Experiments

This section evaluates the performance of our method and demonstrates the efficiency of our approach. Mandhani et al. [MS05] also proposed a technique for view selection, but their method was based on string match, and didn't fully explore the answerability of views. We call it SCSM, Semantic Cache based on String Match.

In this paper, we compare SCEND with SCSM and naïve cache, which requires exact string match between the query and view. The datasets we used are XMark [SWK+01] and DBLP(http://dblp.uni-trier.de/xml/). According to the DTDs of these two datasets, some "//" and "*" nodes are added to construct the queries and views as the input of our experiments. All the datasets follow the default Zipf distribution with exponent z, where z is a parameter, and the probability of choosing the *i-th* query is $\propto 1/i^z$. We cache the views in the semantic cache as [MS05], but there are some // and * in our XPath queries. All the experiments are carried out on a computer with Pentium III 1.14 GHz and 1G RAM by implementing in C++.

6.1 Cache Lookup Performance

We now see how much time it takes to lookup the semantic cache. Note that lookup doesn't include actually obtaining the result of Q, by executing *CQ* (for a cache hit) or Q (for a cache miss). Fig.6 and Fig.7 show how the hit rate varies with the zipf exponent z used for generating queries. As z increases, the locality of the queries increases, and thus the hit rates for both caches increase. Cached views and test queries we used are 20,000 and 200,000 respectively, for each z value. SCEND gives hit rates which are more than 20% higher than SCSM [MS05], and 50% higher than naïve cache on both XMark and DBLP. Thus, the query/view answerability that we capture is much richer than SCSM and naïve cache. Moreover, different DTDs will

not influence the performance of caching, therefore, for the remaining experiments, we use queries generated with DTD being XMark and z=1.2.

Fig. 8 shows how the average cache lookup time varies with the number of queries. Here we are looking to determine how well lookup scales to a large number of stored views. In all cases, the same cached view size 20,000 is used. We can see that the lookup time for SCEND remains constant at around 6.5 ms, even as the number of queries increases to 6 million. This time is very small compared to the time taken to execute a typical XPath query. This is exactly what we would like. Moreover, SCEND is better than SCSM, which take more than 12ms per lookup. The naïve cache takes a mere 0.5 ms per lookup. However, in query processing performance, this difference will be offset by the higher hit rate of the semantic cache, as we will see later. Fig. 9 shows how the hit rate varies with the number of queries. The hit rate for SCEND doesn't shoot up as the number of queries increases, and this is because our cache replacement policy is efficient for caching. However, the hit rate for SCSM and the naïve cache is various with the different numbers of queries. This contrast reflects the better scalability of our method.

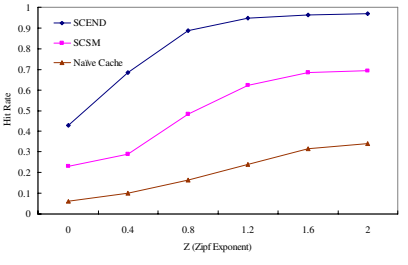


Fig. 6. Hit Rate vs Zip Exponent on XMark

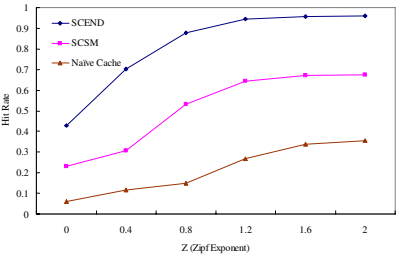


Fig. 7. Hit Rate vs Zip Exponent on DBLP

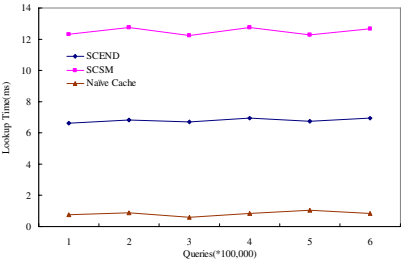


Fig. 8. Lookup Time(milliseconds) vs Number of Queries

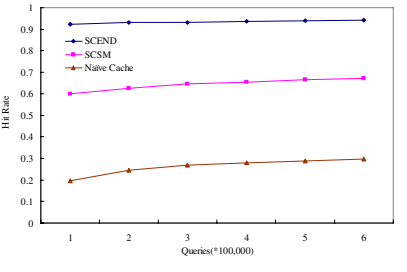


Fig. 9. Hit Rate vs Queries

6.2 Query Processing Performance

We now show the speedup obtained in query processing, by employing our semantic cache. The experimental setup is the same as before, with a difference: if cache hit, we query *CQ* on V to answer Q; otherwise, if cache miss, we will query Q directly.

The zipf exponent z, used in generating queries, is again 1.2, and DTD is also XMark. The number of cached queries and test queries are 20,000 and 200,000 respectively.

Fig.10 shows the average time per query for four different configurations. When not caching, the queries take 1100 ms (milliseconds) each. Having the naïve cache brings this down to 900 ms, while employing SCSM brings this down to 340 ms, but employing SCEND brings this down to 200 ms, which is a speedup by factors of 5.5, 4.5, 1.7 for no cache, naïve cache and SCSM respectively.

Further, Fig. 11 shows how the average time per query varies with the number of queries. The average time for SCEND and no cache doesn't shoot up as the number of queries increases, however, the average time for SCSM and the naïve cache increases with the increase of queries. As the number of queries increases, the locality of the queries also changes. This will not influence the method for no cache, but influence the other three methods. However, because our cache replacement policy is efficient for caching, so the performance of SCEND will not fall with the increase of the number of queries. This reflects the better scalability of our method.

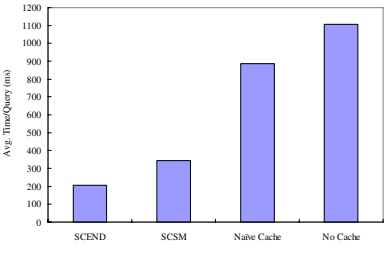


Fig. 10. Avg. Time/Query

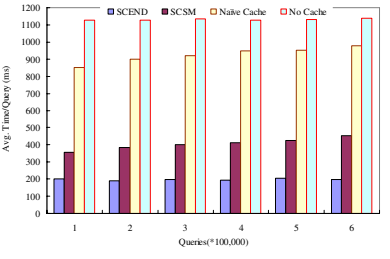


Fig. 11. Avg. Time/Query vs Number of Queries

7 Conclusion

We devise an efficient semantic cache, SCEND, to adequately explore answerability of views and accelerate cache lookup. SCEND decomposes the complex queries into some simpler ones, and transforms the query/view answerability of complex queries to those simpler ones. Moreover, we assign each query with a positive integer, and prove that the divisibility of two assigned positive integers of the query and cached view is a necessary condition for query/view answerability, and through the divisibility, cache lookup can be further speeded up.

The thorough experimental results give us rich confidence to believe that SCEND is more efficient than existing methods on answerability as well as scalability.

References

[BOB+04] A. Balmin, F. Ozcan, K. Beyer, R. Cochrane, and H. Pirahesh. A framework for using materialized xpath views in xml query processing. In VLDB, pages 60-71, 2004.

[BSK02] N. Bruno, D. Srivastava and N. Koudas. Holistic twig joins: Optimal XML pattern matching. In SIGMOD, pages 310-321, 2002.

[CLR90] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. Introduction to Algorithms. McGraw- Hill, 1990.

[CM77] A.K. Chandra and P.M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In STOC 1977, pages 77-90.

[CR02] L. Chen and E.A. Rundensteiner. Ace-xq: A cache-aware xquery answering system. In WebDB, pages 31-36, 2002.

[DFJ+96] S. Dar, M.J. Franklin, B.Jonsson, D. Srivastava, et al. Semantic data caching and replacement. In VLDB, pages 330-341, 1996.

[FFM03] S. Flesca, F. Furfaro, and E. Masciari. On the minimization of xpath queries. In VLDB, pages 153-164, 2003.

[JLW+03] H.F. Jang, H.J. Lu, W. Wang and J. Xu Yu. Holistic Twig Joins on Indexed XML Documents. In VLDB 2003.

[LKM+02] Q.Luo, S.Krishnamurthy, C.Mohan, et al. Middle-tier database caching for e-business. In SIGMOD, 2002.

[MS99] Tova Milo and Dan Suciu. Index structures for path expressions. In ICDT, pages 277-295, 1999.

[MS02] G. Miklau and D. Suciu. Containment and equivalence for an xpath fragment. In Proceedings of PODS, pages 65-76, 2002.

[MS05] B. Mandhani, D. Suciu. Query Caching and View Selection for XML Databases, In VLDB, pages 469-480, 2005.

[PCS+04] S.Pal, I.Cseri, G.Schaller, et al. Indexing xml data stored in a relational database. In VLDB, pages 1134-1145, 2004.

[RBH+04] C. Re, J. Brinkley, K. Hinshaw, and D. Suciu. Distributed xquery. In IIWeb, 2004.

[SWK+01] A.R.Schmidt, F.Waas, M.L. Kersten, et al. The XML Benchmark Project. Technical Report INS-R0103,CWI, 2001.

[YFI+00] K. Yagoub, D. Florescu, V. Issarny, and P. Valduriez. Caching strategies for data intensive web sites. In The VLDB Journal, pages 188-199, 2000.

[DBLP] http://dblp.uni-trier.de/xml/

Clustered Chain Path Index for XML Document: Efficiently Processing Branch Queries*

Hongqiang Wang, Jianzhong Li, and Hongzhi Wang

School of Computer Science and Technology,
Harbin Institute of Technology, Harbin, 150001
hqwang@hit.edu.cn, Lijz@mail.banner.com.cn,
wangzh@hit.edu.cn

Abstract. Branch query processing is a core operation of XML query processing. In recent years, a number of stack based twig join algorithms have been proposed to process twig queries based on tag stream index. However, each element is labeled separately in tag stream index, similarity of same structured elements is ignored; besides, algorithms based on tag stream index perform worse on large document. In this paper, we propose a novel index Clustered Chain Path Index (CCPI for brief) based on a novel labeling scheme: Clustered Chain Path labeling. The index provides good properties for efficiently processing branch queries. It also has the same cardinality as 1-index against tree structured XML document. Based on CCPI, we design efficient algorithms KMP-Match-Path to process queries without branches and Related-Path-Segment-Join to process queries with branches. Experimental results show that proposed query processing algorithms based on CCPI outperform other algorithms and have good scalability.

1 Introduction

XML data is often modeled as labeled and ordered tree. Queries on XML data are commonly expressed in the form of tree patterns, which represent a very useful subset of XPath[1] and XQuery[2].

Query processing on XML document are usually based on some kind of index. 1-index[7] uses Bi-Simulation relationship between elements to classify them into many sets. 1-index is brilliant for its very small size against tree structured XML document. It's efficient to process source path (path from root to the element containing only Parent-Child relationship) queries based on 1-index. While 1-index discourages us when evaluating branch queries because (1) the exact PC relationships between elements are lost; (2) the elements are indiscriminating in the same set.

Bruno et al.[3] proposed the holistic twig matching algorithms TwigStack which is I/O optimal for queries with only ancestor-descendant edge. TwigStack uses tag

* This paper is partially supported by Natural Science Foundation of Heilongjiang Province, Grant No. zjg03-05 and National Natural Science Foundation of China, Grant No. 60473075 and Key Program of the National Natural Science Foundation of China, Grant No. 60533110.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 474 – 486, 2006.

streams which group all elements with the same tag together and assign each element a region-encoding. The element streams can be regarded as a trivial index of the document. Recently, many works [4, 5, 9, 11] are proposed to improve TwigStack to reduce intermediate results. Enumerative indexing in tag stream index lost similarity of elements and brings large cardinality of index. Queries are processed slower against larger XML document.

Based on observation of the structure similarity of elements with same source path, we propose a novel labeling scheme Clustered Chain Path labeling scheme (CCP for brief) which groups all elements with the same source path into one labeling in index. With CCP labeling scheme, the cardinality of index are extremely reduced to a very small "skeleton". Actually, CCPI has the same cardinality as 1-index against tree structured XML document. The difference between CCPI and 1-index is that each CCP in CCPI contains all its ancestors; besides, each element in a CCP distinguishes itself from others. These properties help CCPI to support branch query processing.

The main contributions of this paper include:

- We propose a novel labeling scheme Clustered Chain Path (CCP for brief) labeling which can cluster all elements with the same source path into only one CCP.
- We propose a novel index structure CCPI based on CCP labeling by group all CCPs with same leaf tag together. The cardinality of the index equals to the cardinality of 1-index against tree structured document.
- Based on CCPI, we develop efficient algorithms, KMP-Match-Path to process queries without branch and Related-Path-Segment-Join to process queries with branch.
- We perform a comprehensive experiment to demonstrate the benefits of our algorithms over previous approaches.

Organization. The rest of the paper proceeds as follows. We first discuss preliminaries in Section 2. The CAPI index structure is presented in Section 3. We present query processing algorithms in Section 4. Section 5 is dedicated to our experimental results and we close this paper by the related works and a conclusion in Section 6 and Section 7.

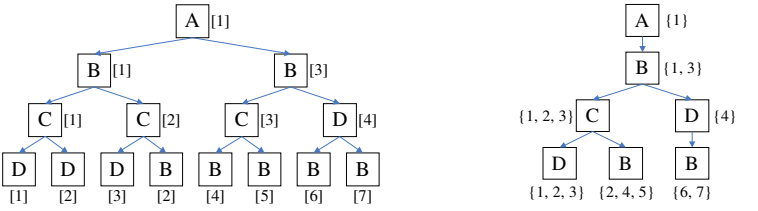
2 Preliminaries

2.1 Data Model and Pre-ordered Path

In this paper, we model XML document as a rooted, ordered and labeled tree. We only focus on element since it is easy to generalize our methods to the other types of nodes. We focus on a commonly used subset of XPath queries consisting of child axis navigation (/), descendant axis navigation (//), wildcard (*), and branches ([ ]).

We first introduce *pre-ordered element* (Fig.1 (a)). An element is labeled with its tag T and a number p where p is the pre-order of elements with tag T by left-to-right pre-order traversal of document tree. Then we introduce *pre-ordered path.* A

pre-ordered path of an element is a sequence of pre-ordered elements from root to given element. For example, element B[3] in Fig.1 (a) has the source path: "/A/B", its pre-ordered path is "/A[1]/B[3]".

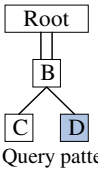


(a) sample.xml (b) 1-index of sample.xml

Fig. 1. A sample XML tree and 1-index of sample XML

2.2 Query Pattern Matching

Query is usually considered as a pattern tree containing branches and related paths. Branch is a root to leaf path in the query pattern while related path is the conjunction of two or more branches. Among all the branches, there is one branch which contains the element to be returned, this branch is *trunk*.



(a) Query pattern

Query: //B[/C]/D
Related path: //B
$Branch_1$: //B/C
$Branch_2$: //B/D (trunk)

(b) Analyzed query

Matched pre-ordered paths:
{(/A[1]/B[3]/D[4],
/A[1]/B[3]/C[3])}

(c) Match in sample.xml

Fig. 2. Example query pattern and match

Given an XPath query Q and an XML document D, a match of Q in D is identified by a mapping from a pre-ordered paths set of D to query Q, such that (1) for each root to leaf branch B in Q, there is a pre-ordered path in the set matches B; (2) for each related path RP in Q, the pre-ordered paths which matched each branch separately shares the related path segment that matches RP.

A *related path segment* (RPS for brief) in a pre-ordered path is the part of the path that matches related path in Q. The source path of a RPS is denoted as *SRPS*.

Example 2.1. Query Q="//B[/C]/D" has two branches (Fig. 2), which are "//B/D" and "//B/C". The two branches are related by related path "//B". Pre-ordered path set OPS={/A[1]/B[3]/C[3], /A[1]/B[3]/D[4]} is a match from sample.xml to query Q because pre-ordered path "/A[1]/B[3]/C[3]" and "/A[1]/B[3]/D[4]" match branch "//B/C" and "//B/D" separately, besides RPSs of "/A[1]/B[3]/C[3]" and "/A[1]/B[3]/D[4]" are "/A[1]/B[3]" which matches the related path "//B" in Q. Since "//B/D" is trunk, we keep "/A[1]/B[3]/D[4]" only as a match to Q.

Finding all matches of a query pattern in an XML document is a core operation in XML query processing. In this paper, we solve the problem in three steps: (1) we cluster pre-ordered paths into CCP sets; (2) for each branch B_i in query pattern Q, we find all CCPs matches B_i; (3) if B_i and B_j are related by related path RP, we join CCP_i and CCP_j. Then we return solutions to Q.

3 CCPI Data Structure

In this Section, we present the structure of CCPI. We first introduce CCP labeling scheme and its properties in 3.1; next we introduce the procedure of constructing CCPI and analyze the cardinality of CCPI in 3.2, finally, we introduce the physical storage of CCPI in 3.3.

3.1 Clustered Chain Path Labeling Scheme

Considering lots of elements sharing a same source path can match/deny a query altogether, we try to cluster them into a clustered pre-ordered path. To avoid losing Parent-Child relationship, we use registry table on each pre-ordered element to record its children's pre-orders.

Definition 3.1. A **Clustered Chain Path** is an array of Registered Elements. A **Registered Element** in a CCP is a tuple <T, O, RT, *f*> where

- T is the tag of Registered Element.
- O is the pre-order set of Registered Element, corresponding to some pre-ordered elements with tag T.
- RT is the registry tables set of Registered Element. A registry table corresponding to pre-order *o* in O contains children's pre-orders of T[*o*].
- f : O → RT is a function from pre-order to its registry table.

For example, C[1], C[2] and C[3] share the same source path "/A/B/C". The CCP of {C[1], C[2], C[3]} is an array RE where RE[1]=<A, {1}, {(1, 3)}, f_1(1)=(1, 3) >; RE[2]= <B, {1, 3}, {(1, 2), (3)}, f_2(1)=(1, 2), f_2(3)=(3) >; RE[3]= <C, {1, 2, 3}, {(ε), (ε), (ε)}, f_3(any)=(ε) > (ε means null). We can just write the CCP as a string "/A[1{1,3}]/B[1{1, 2},3{3}]/C[1{ε},2{ε},3{ε}]" and the CCP is considered to be the union of pre-ordered paths "/A[1]/B[1]/C[1]", "/A[1]/B[1]/C[2]" and "/A[1]/B[3]/ C[3]" .

CCP labeling has good properties (some properties are proven in 3.2):

1. For each source path, there is exactly one corresponding CCP in CCPI.
2. Each pre-order set O in registered element is in ascendant order.
3. Each registry table in registered element is in ascendant order.
4. Registry table set RT is in ascendant order by comparing pre-orders in each registry table.
5. Map function *f* is indeed a map from index of O to index of RT.

We omit the proof here for the sake of the space of the paper.

3.2 Constructing CCPI

Each element gets its pre-ordered path when it's parsed. The pre-ordered path is transformed to a CCP. If two CCPs share a same source path, they are clustered with algorithm Cluster-Path. For large XML file, we use buffer to contain CCPs in memory and write them to disk if buffer is full, then we organize the CCPI after parsing to confirm there is exactly one CCP corresponding to a source path. The algorithm Cluster-Path is present in algorithm 1.

In algorithm 1, CCP_1 and CCP_2 are input CCPs sharing the same source path, CCP is the output path containing CCP_1 and CCP_2. RES_1, RES_2 and RES are registered elements arrays of CCP_1, CCP_2 and CCP.

Algorithm 1. Cluster-Path

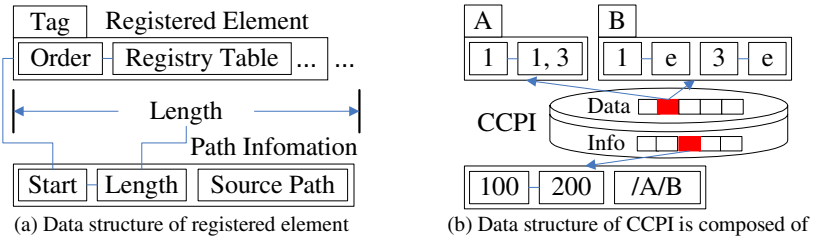
Input: CCP_1, CCP_2 sharing the same source path
Output: $CCP=CCP_1 \cup CCP_2$
1 for each RES[*i*] in CCP
2 RES[*i*]= Sort-Merge-Union ($RES_1[i]$, $RES_2[i]$)
3 return CCP

When clustering, we Sort-Merge-Union pre-order set as well as registry tables if they associate with the same tag in two CCPs. After clustering, the pre-order sets are well sorted as well as registry tables. Then property 2 and 3 of CCP labeling is proved here. These properties contribute to efficiently processing queries with branches.

Theorom 2.1. The cardinality of CCPI equals to the cardinality of 1-index against tree structured XML document.

We omit the proof here for the sake of the space of the paper.

Analysis of algorithm Cluster-Path. The number of elements clustered in a CCP equals to the size of leaf registered element in the CCP. Let CCP_1 and CCP_2 are input CCPs of algorithm 1. Let the number of pre-ordered paths clustered in CCP_1 is N_1 and N_2 for CCP_2. For each registered element in CCP_1, it contains at most N_1 pre-ordered elements (the worst case), the sum of total pre-orders in its registry table is less than or equal to N_1 (N_1 elements have N_1 parents at most). The time complexity of Sort-Merge-Union of two registered elements sets is $O(N_1+N_2)$. Let the length of CCP_1 is L, The time complexity of Cluster-Path in the worst case is $O(L\times(N_1+N_2))$.



(a) Data structure of registered element and path info object

(b) Data structure of CCPI is composed of data part and info part.

Fig. 3. Data structure of CCPI

3.3 Physical Storage of CCPI

We separate CCPI data into two parts: info part and data part. A CCP is represented as sequenced tuples {<Pre-Order, Registry Table>} as data part and <Source Path, Start Pointer, Length> as info part where Start Pointer is the start position of data part, and Length is the bytes of data part. We cluster all CCP with same leaf tag together and save them in two files, one for data part and one for info part. (Fig. 3)

4 Query Processing Based on CCPI

In this Section, we present query processing algorithms based on CCPI. We introduce algorithm KMP-Match-Path to process queries without branch in 4.1 and Related-Path-Segment-Join to process queries with branches in 4.2.

4.1 Processing Query Without Branches

The basic idea of processing query Q without branches is to match source paths of CCPs in CCPI to Q. The info part of CCPI is preloaded into memory as a hash table from tags to an array of path info objects before queries arrived. The reason of preloading info part is that (1) path matching algorithm uses source path instead of CCP and (2) info part is usually quite small although the document may be large.

When query Q arrived, we use string match algorithm to match source path to Q. Although it's a co-NP problem to judge the containment of two XPath expressions [8], the problem in our paper is to judge if a source path matches a general XPath expression without branch.

To make the problem simple, we first introduce broken XPath expression sequence. A *broken XPath expression sequence* is a sequence of sub strings decomposed from query XPath expression using delimiter "//". For example, XPath expression "//A//B" is decomposed into (Root, A, B), while "/A//B/C" is decomposed into (Root/A, B/C). After decomposing, there is no "//" in broken XPath expressions.

We use Knuth-Morris-Pratt string match algorithm [19] to perform matching from a source path to query XPath expression. (Algorithm 2) Each time when function KMP-Match-Broken is called, if brokenXPath is matched by SP, the sub string from beginning to the matched point of SP is abandoned, the rest part of SP is returned. If brokenXPath can not be matched by SP, function KMP-Match-Broken returns null which means matching failed.

Example 4.1. Suppose we match "//B//C/D" with source path "/A/B/B/C/C/D". The broken XPath sequence is {Root, B, C/D}. We add "Root" element to source path and perform matching. The first time function KMP-Match-Broken is called, since root matches root, it returns "/A/B/B/C/C/D". The second time when it is called, since "/A/B" matches "B", it returns "/B/C/C/D" and the last time it returns "" while the broken XPath sequence are all matched, and KMP-Match-Path returns true.

Since the algorithm is based on string matching, it's easy to match query expression with wildcard by add the rule that any element can match wildcard element "*" in query.

Algorithm 2. KMP-Match-Path

```
     Input: Source Path SP, XPath
     Output: true if SP matches XPath or false if else
1    brokenXPaths= sub strings of XPath using delimiter "//"
2    for each brokenXPath in brokenXPaths {
3      SP=KMP-Match-Broken(SP, brokenXPath)
4      if (SP=null)  return false
5      else if (all brokenXPaths are matched & SP="")
6        return true }

     Function KMP-Match-Broken(SP, brokenXPath)
     //see Knuth-Morris-Pratt algorithm for detail
```

Notice that if there is no match in path info part for a query Q, then we can allege no solutions for Q without any disk IO.

Analysis of KMP-Match-Path. Time complexity of KMP-Match-Broken is $O(L+QL_i)$ where L is length of source path SP and QL_i is the ith length of broken XPath expression, let QL is length of query XPath, then $QL=\sum_i QL_i$. The time complexity of KMP-Match-Path is O(L+QL).

4.2 Processing Query with Branches

The basic idea of processing query with branches is to split the query into several root-to-leaf branches and corresponding related paths. Then we evaluate each branch as a query without branch, finally we join the intermediate solutions to final solutions by their related paths.

Algorithm 3. Related-Path-Segment-Join

```
     Input: CCP1, CCP2 whose SRPSs are same, CCP1 is trunk
     Output: CCP satisfy both query branch
1      RES[1]= Sort-Merge-Join (RES1[1], RES2[1])
2      for each RES[i] (i>1)
3        if (RES[i-1]==null)    return null
4        RES[i]=Filter(RES1[i], RES[i-1])
5      return CCP

     Function Filter(RES[i], RES[i-1])
     //RES[i-1] must has participated Sort-Merge-Join
     // or has been filtered
1    for each RT in RES[i-1]
2        Referring=Referring ∪RT
3    return Sort-Merge-Join(RES[i], referring)
```

4.2.1 Algorithm Related-Path-Segment-Join

Algorithm Related-Path-Segment-Join joins two input CCPs whose SRPSs are same is present in algorithm 3. In algorithm3, RES_1, RES_2 and RES are arrays of registered elements in CCP_1's RPS, CCP_2's RPS and output CCP's RPS.

Algorithm Related-Path-Segment-Join operates in two phases. In the first phase (lines 1), the first registered element in both CCPs are joined. In the second phase (lines 2-5), for $RES_1[i]$ (i>1) in CCP_1, it has two join choices, join with $RES_2[i]$, or filtered by $RES_1[i\text{-}1]$. The latter choice is better because the Referring in function Filter comes from $RES_1[i\text{-}1]$ is definitely smaller than $RES_2[i]$. Referring is a sub set of registry tables of $RES_2[i\text{-}1]$ and it's a sub set of $RES_2[i]$.

Analysis of Related-Path-Segment-Join. Because numbers in pre-order set O and registry set RT are all distinct, time complexity of Sort-Merge-Join on these sets is linear to the input. Suppose there are N_1 pre-ordered paths in CCP_1 and N_2 for CCP_2, there are at most $N=\min(N_1, N_2)$ elements in each registered elements in joined CCP. Each time when function filter is called, there are at most 2N numbers participate Sort-Merge-Join. Let L is the length of trunk or the length of SRPS if both CCP_1 and CCP_2 are not trunk, the time complexity of algorithm 3 in the worst cast is $O(L\times N)$.

Theorom 4.1. Algorithm 3 reports all correct join results.

Proof. Suppose a pre-ordered path $P_1=RPS+P_3$ is a solution to Q. Then there must exist a pre-ordered path $P_2=RPS+P_4$ matching another branch in Q. P_1 is clustered in CCP_1 and P_2 is clustered in CCP_2 in algorithm 3. When we perform join on the first element, the first pre-ordered element in P_1 is kept because P_2 in CCP_2 has the same RPS with P_1. Next, by filtering one by one, P_1 is outputted. □

4.2.2 Processing Query with Nested Branches or Multiple Branches

If query Q has nested branches or multiple branches, we first split Q into multiple branches and their related paths. Q is divided as several sub queries; each sub query contains one related path and corresponding branches. A related path can only appear in one sub query of Q, but a branch can appear in many sub queries of Q. Each branch is alleged solvable using algorithm KMP-Match-path. After that, a query plan is created, and then we perform Related-Path-Segment-Join for each sub query (Fig. 4). The problem of join order selection in query plan is an import issue; however we left it as our future work. In this paper, we select the sub query with the longest related path inside as the processing sub query.

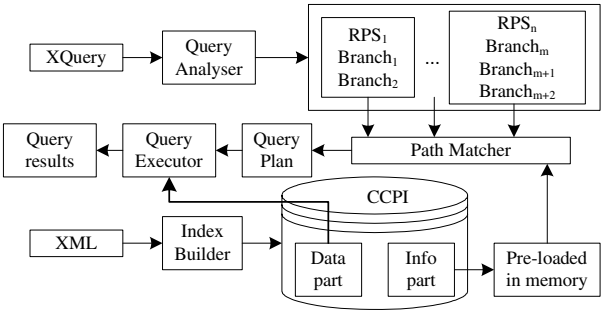


Fig. 4. The architecture of query processing based on CCPI

Example 4.3. Consider query "//B[/C[/D]/B]/D" to sample.xml (Fig. 1). There are two sub queries with three branches and two related paths. One sub query is $subQ_1$ whose branches are "//B/C/D" and "//B/C/B" with related path "//B/C"; the other is $subQ_2$ whose branches are "//B/C/D", "//B/C/B" and "//B/D" with related path "//B". Since "//B/C" is the longest related path, we process $subQ_1$ first. Because the branches in $subQ_1$ are not trunk, we return only joined RPS of the branches since only RPS is useful to join with other sub queries. The result RPS is directly the input data part of "//B/C/D" and "//B/C/B", we just need to join it with data part of "//B/D" using related path "//B" and return query results.

Likewise, query with multiple branches such as "//B[/D]/C[/B]/D" consist of two sub queries with three branches and two related paths which are "//B/C/D" and "//B/C/B" with related path "//B/C"; "//B/C/D", "//B/C/B" and "//B/D" with related path "//B". The processing procedure is just like example 4.3.

Since query with branches is broken into many sub queries without branches and all branches will be checked if it's solvable before access data part in CCPI, we can allege that a query has no solutions if any branch are not matched in info part of CCPI.

5 Experiments

5.1 Experimental Setup

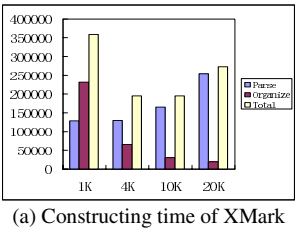
All experiments are run on a PC with Pentium IV 3.0G processor, 2G DDR400 memory and 20G SCSI hard disk. The OS is Windows 2000 Server. We implement our system using JDK1.5. We obtained the source code of Nok [18] and TJFast [9] from the original authors.

We used three different datasets, including one synthetic and two real datasets. The first synthetic dataset is the well-known XMark [16] benchmark data (with factor 1). The two real datasets are DBLP and TreeBank [18]. (Table 1)

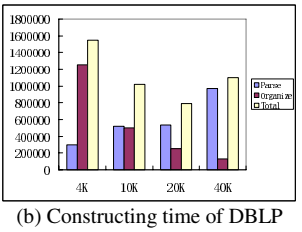
We can see that CCPI is so highly clustered that only a small amount of CCPs against large amount of elements.

Table 1. Datasets of experiments

Dataset	Elements	File Size	Max Depth	Tags	CCPs
XMark	1666315	116M	12	74	514
DBLP	3795138	156M	6	35	127
TreeBank	2437666	84M	36	250	338748



(a) Constructing time of XMark



(b) Constructing time of DBLP

Fig. 5. Constructing time of CCPI with different size of buffer

5.2 Constructing Time of CCPI

First, we study the constructing time of CCPI with different size of buffer. If we use bigger buffer when constructing CCPI, the parsing time will be longer because more Cluster-Path is called and the organize time will be shorter. We test different size of buffer on XMark and DBLP in Fig. 5.

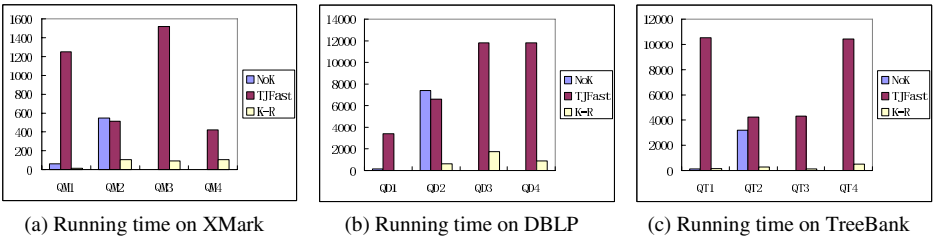
From the experimental result (Fig. 5), we can see that it's not true that constructing CCPI with bigger buffer is faster than that with smaller buffer. Also, it's hard to know the exact buffer size with which the constructing time is the shortest.

5.3 Queries and Performance

We choose 4 queries on each dataset including: (1) a source path query; (2) a branch query without "//"; (3) a path query with "/" and "//" but no branch; (4) a query with "/","//" and branches. The running times of the queries are shown in Fig. 6.

Table 2. Query expressions

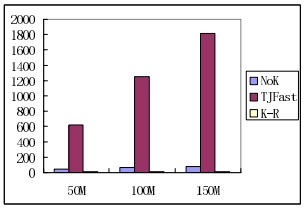
Query	Query Expressions
QM1	/site/regions/africa/item/description/parlist/listitem/text/keyword
QM2	/site/people[/person/profile[/education]/age]/person/phone
QM3	/site/closed_auctions//emph
QM4	/site/people/person[//age]//education
QD1	/dblp/mastersthesis/year
QD2	/dblp/article[/ee]/year
QD3	/dblp//author
QD4	/dblp/article[//author]//year
QT1	/FILE/_QUOTES_/S/VP
QT2	/FILE/EMPTY/S[/VP]/SBAR
QT3	/FILE/EMPTY/S/NP//SBAR
QT4	/FILE/_QUOTES_/S[//NP]//VP



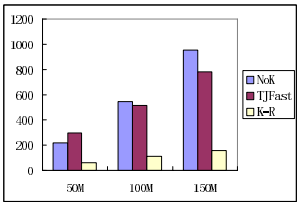
(a) Running time on XMark (b) Running time on DBLP (c) Running time on TreeBank

Fig. 6. Experimental results, K-R is KMP-Match-Path or Related-Path-Segment-Join

From the experimental results of Fig.6, we see that the algorithms based on CCPI outperform Nok and TJFast on all datasets. Since CCPI has small cardinality compared with other index labeling scheme; algorithms based on CCPI have less disk accesses than Nok and TJFast.



(a) Running time of QM1



(b) Running time of QM2

Fig. 7. Scalability of algorithms, K-R is KMP-Match-Path or Related-Path-Segment-Join

5.4 Scalability

We test scalability of our algorithms on 50M, 100M and 150M XMark datasets, the queries we used is QM1 and QM2 in table 2. The running times of queries are shown in Fig. 7. Compared with Nok and TJFast, the scalability of algorithms based on CCPI is better because the cardinality of CCPI is almost invariable when document grows to larger size.

6 Related Works

N.Bruno et al. [3] proposed a holistic twig join algorithm based on region encoding, namely TwigStack, TwigStack is I/O optimal for queries with only A-D edge but it is sub optimal for queries with P-C edge. Lu et al [5] proposed a look-ahead method to reduce the number of redundant intermediate paths. Jiang et al [11] used an algorithm based on indexes built on containment labels. The method can "jump" elements and achieve sub-linear performance for twig pattern queries. Lu et al [9] proposed TJFast based on a new labeling scheme called extended Dewey. TJFast only needs to access labels of leaf nodes to answer queries and significantly reduce I/O cost. BLAS by Chen et al. [10] proposed a bi-labeling scheme: D-Label and P-Label for accelerating P-C edge processing. Their method decomposes a twig pattern into several P-C path queries and then merges the results. Zhang et al [17] proposed Nok pattern tree and algorithm for efficiently evaluating path expressions by NoK pattern matching.

1-index [7] is based on the backward bi-simulation relationship. It can easily answer all simple path queries. F&B Index [15] uses both backward and forward bi-simulation and has been proved as the minimum index that supports all branching queries. These "exact" indexes usually have large amount of nodes and hence size, therefore, a number of work has been devoted to find their approximate but smaller counterparts. $A(k)$-index [12] is an approximation of 1-index by using only k-bi-simulation instead of bi-simulation. $D(k)$-index [13] generalizes $A(k)$-index by using different k according to the workload. $M(k)$-index and $M^*(k)$-index [14] further optimize the $D(k)$-index by taking care not to over-refining index nodes under the given workload.

7 Conclusions and Future Work

In this paper, we have proposed a novel index structure CCPI. It can extremely reduce the size of index and disk IO when processing queries. Based on CCPI, we design efficient algorithms to process queries with or without branches. We implement the algorithms and compare the performance with other algorithms. From the experimental results, we conclude that our algorithms perform better than others on queries with or without branch. The advantage comes from less disk access due to high clustering of CCPI. Besides, our algorithms based on CCPI have good scalability.

Join order selection is important when there are many branches and related paths in query. We need to build a cost model based on CCPI to create query plan. It's one of our future works.

Acknowledgement. Special thanks to Ning Zhang of University of Waterloo, and Jiaheng Lu of National University of Singapore for providing support for our experiments.

References

[1] XML Path Language (XPath) 2.0. http://www.w3.org/TR/xpath20/.
[2] XQuery 1.0: An XML query language. http://www.w3.org/TR/xquery/.
[3] N. Bruno, D. Srivastava, and N. Koudas. Holistic twig joins: optimal XML pattern matching. In SIGMOD Conference, pages 310-321, 2002.
[4] H. Jiang et al. Holistic twig joins on indexed XML documents. In Proc. of VLDB, pages 273-284, 2003.
[5] J. H. Lu, T. Chen, and T. W. Ling. Efficient processing of XML twig patterns with parent child edges: a look-ahead approach. In Proceedings of CIKM Conference 2004, pages 533–542, 2004.
[6] Q. Li and B. Moon. Indexing and querying XML data for regular path expressions. In Proc. of VLDB, pages 361-370, 2001.
[7] T. Milo and D. Dan Suciu. Index structures for path expressions. In ICDT, pages 277-295, Jerusalem, Israel, 1999
[8] G. Miklau and D. Suciu. Containment and equivalence for an XPath fragment. In PODS, pp. 65–76, 2002.
[9] Jiaheng Lu, Tok Wang Ling, Chee Yong Chan, Ting Chen: From Region Encoding To Extended Dewey: On Efficient Processing of XML Twig Pattern Matching. 193-204. In Proc. of VLDB, 2003.
[10] Y. Chen, S. B. Davidson, and Y. Zheng. BLAS: An efficient XPath processing system. In Proc. of SIGMOD, pages 47-58, 2004.
[11] H. Jiang, W. Wang, H. Lu, and J. X. Yu. Holistic twig joins on indexed XML documents. In In Proceeding of VLDB 2003, pages 273–284, 2003.
[12] R. Kaushik, P. Shenoy, P. Bohannon, and E. Gudes. Exploiting local similarity for efficient indexing of paths in graph structured data. In ICDE 2002.
[13] C. Qun, A. Lim, and K. W. Ong. D(k)-index: An adaptive structural summary for graph-structured data. In ACM SIGMOD, pages 134-144, 2003.

[14] H. He and J. Yang. Multi resolution indexing of XML for frequent queries. In ICDE 2004.
[15] R. Kaushik, P. Bohannon, J. F. Naughton, and H. F. Korth. Covering indexes for branching path queries. In SIGMOD 2002.
[16] XMark: The XML-benchmark project. http://monetdb.cwi.nl/xml.
[17] N. Zhang, V. Kacholia, and M. T. Özsu. A succinct physical storage scheme for efficient evaluation of path queries in XML. In ICDE 2004, pages 54–65.
[18] U. of Washington XML Repository. http://www.cs.washington.edu/research/ xmldatasets/.
[19] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, Introduction to Algorithms, Second Edition, The MIT Press © 2001.

Region-Based Coding for Queries over Streamed XML Fragments

Xiaoyun Hui, Guoren Wang, Huan Huo, Chuan Xiao, and Rui Zhou

Northeastern University, Shenyang, China
wanggr@mail.neu.edu.cn

Abstract. Recently proposed Hole-Filler model is promising for transmitting and evaluating streamed XML fragments. However, by simply matching filler IDs with hole IDs, associating all the correlated fragments to complete the query path would result in blocking. Taking advantage of region-based coding scheme, this paper models the query expression into query tree and proposes a set of techniques to optimize the query plan. It then proposes XFPR (XML Fragment Processor with Region code) to speed up query processing by skipping correlating adjacent fragments. We illustrate the effectiveness of the techniques developed with a detailed set of experiments.

1 Introduction

As an emerging standard for data representation and exchange on the Web, XML is adopted by more and more applications for their information description. Recently, several applications of XML stream processing have emerged, and many research works focus on answering queries on streamed XML data, which has to be analyzed in real-time and by one pass. In order to decrease the transmission and evaluation cost, Hole-Filler model [1] is proposed. As a result, Xstream-cast [2], XFrag [3], and other approaches focus on dealing with fragmented XML data based on Hole-Filler model. Figure 1 gives an XML document and its DOM tree, which acts as an example of our work .

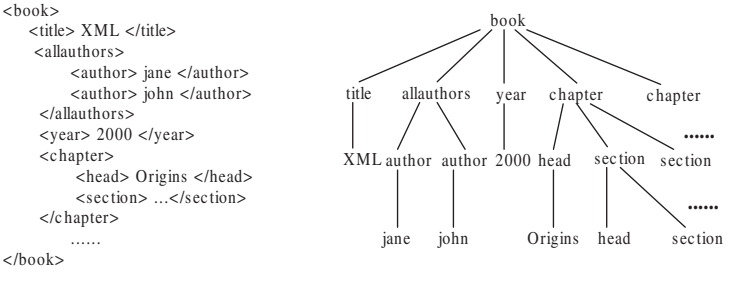


Fig. 1. An XML Document and its DOM Tree

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 487–498, 2006.

Data fragmentation offers various attractive alternatives to organize and manage data. In this way, infinite XML streams turn out to be a sequence of XML fragments, and queries on parts of XML data require less memory and processing time. Furthermore, changes to XML data may pose less overhead by sending only fragments corresponding to the changes, instead of sending the entire document.

However, recently proposed frameworks have not fully exploited the advantage of Hole-Filler model. In XFrag, a novel pipelined framework is presented for processing XQueries to achieve processing and memory efficiency. XML fragments are processed as and when they arrived and only those messages that may effect on the query results are kept in the association table. However, the XFrag pipeline is still space consuming in maintaining the links in the association tables and time cost in scheduling the operations for each fragment. And it can not avoid "redundant" operations when dependence occurs between adjacent operators.

This paper presents a new framework and a set of techniques for efficiently processing XPath queries over streamed XML fragment. We make the following contributions: (i)we adopt the region-coding scheme for XML documents and adapt it to streamed XML fragment model. This coding scheme simplifies the method checking structural relationship (such as parent-child relationship "/" and ancestor-descendant relationship "//") between fragments and speeds up complex query operations, especially for nested-loop query and twig pattern query. (ii)we propose techniques for enabling the transformation from XPath expression to optimized query plan. We model the query expression using query tree and enable further analysis and optimizations by eliminating the "redundant" path evaluations. (iii)based on optimized Query tree, we propose query plan directly into an XML fragment query processor, named XFPR, which speed up query processing by skipping correlating adjacent fragments. Note that, we assume the query clients cannot reconstruct the entire XML data before processing the queries.

The rest of this paper is organized as follows. Section 2 introduces our region-based coding scheme for streamed XML fragments. Section 3 gives a detailed statement of our XML fragment processing framework. Section 4 shows experimental results from our implementation and shows the processing efficiency of our framework. Our conclusions are contained in Section 5.

2 Region-Based Coding Scheme on Hole-Filler Model

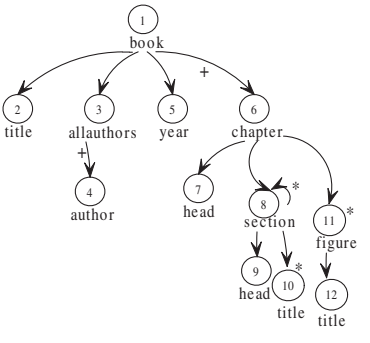
In our approach, we employ hole-filler model [1] to describe XML fragments. In order to specify the relationships between fragments, we adopt the region-based coding scheme of XML documents and adapt it to hole-filler model that holds both the data contents and structural relationships.

2.1 Preliminary Hole-Filler Model

In the Hole-Filler model, document is pruned into many fragments. Those fragments are correlated to each other by fillers and holes. Since fragments may

arrived in any order, the definition of filler and hole can maintain the context of fragments.

The main concept of hole-filler is that every fragment is treated as a "filler" and is associated with a unique ID (denoted as fid). When a fragment needs to refer to another fragment in a parent-child relationship, it includes a "hole", whose ID (hid) matches the fid of the referenced filler fragment. Another important information transmitted by the server is Tag Structure. It is transmitted in the stream as structural summary that provides the structural make-up of the XML data and captures all the valid paths in the data. A tag corresponds to an XML tagged element and is qualified by a unique id ($tsid$), a name (the element tag name), and a type. For an element with type "Filler", we prune the element in the document and make it the root of the subtree. For embedded type, the element is embedded within its parent element, that means it is inside the same fragment. This information is useful while expanding wildcard path selections in the queries. The DTD and the corresponding tag structure of the XML document (given in Figure 1) are depicted in Figure 2.




```
<stream: structure>
 <tag name="book" id="1" Filler="true">
    <tag name="title" id="2" / >
    <tag name="allauthors" id="3" Filler="true">
       <tag name="author" id="4"/>
    </tag>
    <tag name="year" id="5" />
   <tag name="chapter" id="6" Filler="true">
      <tag name="head" id="7" />
      <tag name="section" id="8" Filler="true">
         <tag name="head" id="9" />
         <tag name="title" id="10" />
      </tag>
     <tag name="figure" id="11" Filler="true">
        <tag name="title" id="12" />
     </tag>
   </tag>
 </tag>
</stream: structure>
```

Fig. 2. Tag Structure of Hole-Filler Model

2.2 Region-Based Encoding Representation

We can associate fillers with holes by matching fids with hids. However, it does not suffice since fid and hid in a fragment alone cannot directly capture the ancestor-descendant structural relationships between fragments.

We extend the fid with one widely accepted encoding approach [7], where the position of an element occurrence is represented as a 4-tuple (*DocId, StartPos, EndPos, Level*): (i) *DocId* is the identifier of the document, which can be omitted with one single document involved (ii) *StartPos* is the number given in a pre-order traversal of the document and *EndPos* is the number given in a post-order traversal of the document.(iii) and *Level* is the nesting depth of the fragment's

root element (or string value) in original document, helping to identify "parent-child" relationship between fragments. In figure 2, "section" and "head" are in the same filler and the level of this filler equals the level of "section" in the original document. We use the 3-tuple of the root element in the fragment representing fid. As for *hid*, we use the number of *StartPos*. Figure 3 gives three fragments of the document in Figure 1 after coding fid with (*StartPos, EndPos, Level*).

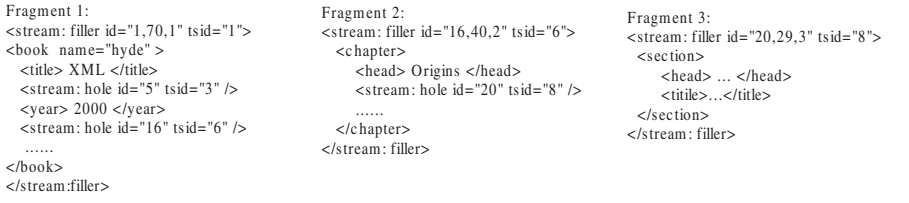


Fig. 3. XML Document Fragments

Taking (*StartPos, EndPos, Level*) as fid, the fragment not only retains the link information between correlated fragments, but also indicates the descendant fragments within the region from *StartPos* to *EndPos*. Given the region codes, the interval (StartPos, EndPos) of two arbitrary fragments are either inclusive or exclusive, and we can get the ancestor-descendant relationship between nodes by testing their region codes.We suppose that f_1 with region code (S_1, E_1, L_1),f_2 with region code (S_2, E_2, L_2) are fragments.

Ancestor-Descendant: f_2 is a descendant of f_1 iff $S_1 < S_2$ and $E_2 < E_1$. e.g. Fragment 3 with code (20,29,3) is a descendant of Fragment 1 with code (1,70,1) in Figure 3.

Parent-Child: f_2 is a child of f_1 iff (1) $S_1 < S_2$ and $E_2 < E_1$; and (2) $L_1 + 1 = L_2$. e.g. Fragment 2 with code (16,40,2) is a child of Fragment 1 with code (1,70,1) in Figure 3.

3 XFPR Query Handling

Based on hole-filler model, infinite XML streams turn out to be a sequence of XML fragments, which become the basic processing units of the query. From the analysis of simple code, we can find that getting the ancestor-descendant relationship needs more steps of assuredness of parent-child relationship. So, waiting for a fragment to come to complete the information necessary for execution would result in blocking. Taking advantage of region coding scheme for fragments, we can skip evaluating the structural relationship between the fragments to expedite processing time, especially for nested path expressions.

This section focuses on the techniques based on region coding scheme. We first introduce the pruning polices on query expressions to eliminate "redundant" path evaluations. Then we present the query plan transformation techniques for efficient query handling with XML fragments.

3.1 Query Plan Generation

Linear Pattern Optimization. Let T be an optimized query tree after dependence pruning and TS a tag structure complying with the DTD, we can transform the queries on XML elements to queries on XML fragments. Since T captures all the possible tsids involved in the query according to tag structure, we only need to locate the corresponding fragments, which are presented by element nodes with type "filler" in query expressions. Note that T can also capture all the valid paths in query path. For linear pattern optimization, we only need to handle operators which can output results.

For example, the end element of $Query1$ "*/book/chapter/section/head*" is "head", and its type is not "filler". However, the type of "section" element which is in the same filler with "head" is "filler".We only need to handle particular fragment "section" in output operator by matching its tsid 8 and level 3 in region code without inquiring the parent fragments.

A simple query with "/*", "//", for example, Query 2 "/book//author", we can catch the relationship with "book" fillers and the fillers including "author" element since region coding can quickly and directly locate the ancestor-descendant relationship without knowing the intermediate fillers. However, such case cannot be directly used for queries including predicates.

Twig Pattern Optimization. It is not common for path expressions to have only connectors such as "/" and "//". The location step can also include one or more predicates to further refine the selected set of nodes.We can simplify a path computation into an XML fragment matching operation after determine the key nodes in the query tree to speed up the query. In this way, queries involving one or more predicates or twig patterns can also be optimized.

Definition 3.1. *Let TS be a tag structure and T an optimized query tree after dependence pruning. For $t_i \in T$, if the successor of t_i is more than one nodes , or the predicate of t_i is not null, then t_i is defined as the key node in the query tree, in short $KN(T) = t_i$.*

Taking advantage of the region code of the fragments, we can quickly judge the ancestor-descendant relationship between fragments by comparing *(StartPos, EndPos)* of the nodes. If $a.StartPos < d.StartPos$ and $d.EndPos < a.EndPos$, fragment a is the ancestor of fragment d. Since the content of fragment is guaranteed by tag structure, we can prune off the intermediate nodes in query tree, which are not key nodes and only keep the key nodes and the output nodes in the query tree. As for nested-loop step, we can apply the same policy and check the level number with the number of repetition steps.

Considering the Query 3: /book/chapter[/head]/figure/title, whose query tree is presented in Figure 4. In (a), all kinds of fragments involved in this query are indicated. In (b), according to definition 3.1, we only keep the "chapter" and "figure" nodes since "chapter" has predicate and "figure"can output the result. In this way, we can compare ancestor-descendant relationship between "chapter" fillers and "figure" fillers without inquiring the "book" fillers.

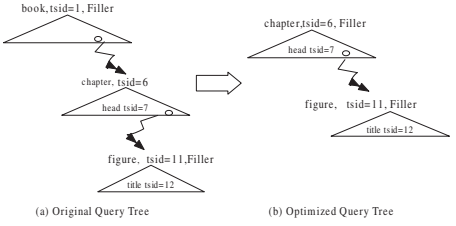


Fig. 4. Query Tree of Query 3

Nested Pattern Optimization. Given an XPath p, we define a simple subexpression s of p if s is equal to the path of the tag nodes along a path $< v_1, v_2, \cdots v_n >$ in the query tree of p, such that each v_i is the parent node of $v_{i+1} (1 \leq i < n)$ and the label of each v_i (except perhaps for v_1) is prefixed only by "/". If each v_i shares the same tsid and the same predecessor, we define it as repetition step. For example, *Query 4: /book/chapter/section/section/section/head* is such a query involving repetition step, whose query tree is shown in Fig. 5 (a).

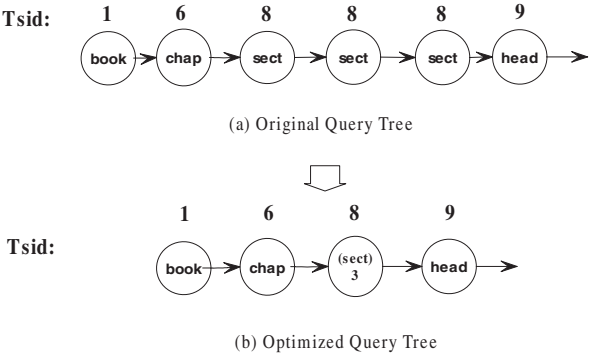


Fig. 5. Query Tree of Query 4

Query 4 involves three types of fragments with tsid 1, tsid 6 and tsid 8. Since '/section' is a nested path step, the query tree includes three nodes with the same tsid 8. Repetition step in XPath expression degrades the performance significantly, especially when the repeated path is highly nested. Taking advantage

of level number, we can simplify such repetition path evaluation. Since parent-child operator '/' indicates the nodes at adjacent level, we can optimize the query tree by pruning off repeated steps and recording the number of repetition step for further processing. In Query 4, '/section' occurs three times, so we keep only one of them in query tree and embrace it with "( )", at the right corner of which we mark 3. The query plan for Query 4 after pruning is shown in Figure 5 (b). According to the linear pattern optimization, we only need to handle "section" filler with level 5.

Considering Query 2:*/book//head* fragments with tsid 1, tsid 6, and tsid 8. Since "/section" is a nested path step, there can be many repetition steps between "/book" and "//head". Taking advantage of region coding, we can simplify such repetition path evaluation by checking only ancestor-descendant relationship between fragments with tsid 1, tsid 6 and tsid 8. In the optimized query plan, we can embrace "/section" and mark * that means we can handle "section" fillers in any level.

3.2 The XFPR Matching Algorithm

XFPR is based on optimized query plan after pruning off the "redundant" operations in query tree. In this section, we focus on the main algorithm of query evaluation in XFPR framework and then analyzing its efficiency comparing to previous work.

The transform from query tree to query plan is a mapping from XPath expression and the tag structure to XFPR processor. For each node in query tree, the tsid of the element node with type "Filler" is corresponding to an entry of the hash table, which is characterized by a predecessor p, a bucket list b, and the tag structure corresponding to the fragment. The predecessor p resolves fragments relationships and predicate criteria. The bucket list b linked each fragment with the corresponding tsid of the node together, and each item is denoted as a *f-tuple* $(fillerid, \{holeid\}, value)$, in which $fillerid$ denotes *(StartPos, EndPos, Level)* in XFPR, $holeid$ denotes *(StartPos, tsid)* in XFPR, value can be set to $true$, $false$, undecided ($\perp$), or a result fragment corresponding to predicates. While the former three values are possible in intermediate steps that do not produce a result, the latter is possible in the terminal step in the query tree branch.

Algorithm 1 describes the processing method, which is based on the SAX event-based interface that reports parsing events. When a fragment is processed by XFPR, it first needs to verify if the predecessor operator has excluded its parent fragment due to either predicate failure or exclusion of its ancestor. If the ancestor fragment has arrived, the value of the *f-tuple* copies the status of its ancestor's value, otherwise the value is tagged with an "$\perp$". And it has to trigger the descendant fragments and pass the status value on to its descendants as fragments may be waiting on operators to decide on their ancestor eligibility.

Algorithm 1 startElement()

```
1: if (isFragmentStart()==true) then
2:   tsid=getTsid();
3:   fid=getFid(); // including fid.startPos, fid.endPos, fid.level
4:   if ( hashFindOperator(tsid)!=null) then
5:     fillInformation();
6:     if (isQueryFragment()==true) then
7:       p=findAncestorOperator(tsid);
8:       for each f-tuple ft of p do
9:         if ( ft.fid.startPos < fid.startPos && ft.fid.endPos > fid.endPos
           && isSatisfied(ft.fid.level,fid.level) && ft.fid.value! = ⊥) then
10:          currentValue=ft.value;
11:        end if
12:      end for
13:    else
14:      q=findDescendantOperator(tsid);
15:      for each f-tuple ft of q do
16:        if ( fid.startPos < ft.fid.startPos && fid.endPos > ft.fid.endPos
           && isSatisfied(ft.fid.level,fid.level) && fid.value! = ⊥) then
17:          ft.value=currentValue;
18:        end if
19:      end for
20:    end if
21:  end if
22: end if
```

3.3 Algorithm Analysis

In this section, we illustrate the advantage of our region-coding scheme adopted in XFPR with different types of queries, and compare with the query efficiency with simple fid and hid numbering scheme in previous frameworks [8, 3].

Hierarchical Matching. As the illustrative example, consider Query 5:*//chapter/section/section/title*, which returns the "title" of the "section" nested in other "sections".In the previous frameworks with simple fid and hid numbering scheme, when fragment with fid 2, hids 4, 13 arrives, it is accepted by "chapter" operator by hashing to the corresponding entry of the hash table, and the link information are recorded as f-tuple (2, {4, 13}, undecided). The value identifies the fragment's state which is decided by its parent fragment. If its parent fragment arrives, the value copies the parent fragment's value, otherwise the fragment's value is set "undecided". Similarly, there are three "section" fragments with fid 4, hid 6, fid 6, hid 10 and fid 13, hid 15 arrive successively. After inquiring the predecessor operator and triggering the successive operator, the "section" fragments with fid 4 and 13 relate to the "chapter" fragment with fid 2, and fragment with fid 6 is the child fragment with fid 4. However, not all the "section" fragments can be output as the results, since only those "section" fragments matching the second "/section" steps in the query path contribute to

the results. As the fragment with fid 6 matches the second location step "/section" in the query path, it is output as the result after tagging "true" for the corresponding value of the fragment's f-tuple.

We can find out from Query 5 that the simple *fid,hid* numbering scheme is not efficient for some queries. In the hash table, each entry recording the arrived fragment information needs to execute two steps. One is inquiring predecessor, the other is triggering successive operators. It takes too much time to find the related fragments before query processing under such kind of manipulation. Moreover, after many such manipulations, filler still cannot output the result due to not matching the particular location step in query path.

However, with region-coding scheme, we can easily identify the element level and speed up such hierarchical relationship operations. In our framework, we use nested pattern optimization and linear pattern optimization to generate the query plan for Query 5, we only need to check each arrived "section" fragment whether its tsid equals to "4" and its level equals to "4". When "section" fragment with region code arrives, we directly output the element "title" as the result, which is in the "section" fragment representing the same element type and the fourth level in XML document tree.

From the analysis of the query example, we can conclude that simple numbering scheme is not suitable for hierarchical relationship evaluation. Especially, if the path expression inquiries the element in a nested hierarchical step, it complexes the processing and costs more time. Obviously, region-based coding scheme shows strong superiority in hierarchical relationship matching.

Skipping Fragments. Consider Query 6:*//Chapter/section[/figure]/*/figure/title*, which is a twig pattern expression with "*" involved. Tag structure defines the structure of data and captures all the valid paths. We can use it to expand wild-card path selections in queries to specify query execution. So after we specify the "*"node, Query 6 equals to " book/chapter/section [/figure1] /section/figure2/title". In order to distinguish the "figure" in branch expression and the "figure" in main path expression, we denote the former one as figure1 and the latter as figure 2.

We know that the simple numbering scheme for fid and hid can only handle parent-child relationship between fragments. In this way, it might decelerate the query evaluation because it would result in blocking to wait for all the the correlated fragments to come to complete the query path. However, taking advantage of twig pattern optimization, XFPR can skip the intermediate fragments and output the results as soon as possible. For fragments with tsid 4 and tsid 7 satisfying the condition $section.StartPos < figure2.StartPos \wedge section.EndPos > figure2.EndPos \wedge figure2.Level = 5 \wedge section.Level = 3$, algorithm outputs the results immediately if "figure1" fragment also satisfies the following condition:$figure1.StartPos > section.StartPos \wedge figure1.EndPos < section.EndPos \wedge figure1.Level = 4$.

The handle process adopted twig pattern optimization is that : assume the "section" fragment with region code $(6, 27, 3)$ has already arrived and its information is recorded in the association table. When "figure" fragment with

region code $(15, 26, 5)$ arrives, it is compared with the "section" fragment by their *StartPos, EndPos*. Since $15 > 6$, $26 < 27$ and figure.Level=5, section.level=3 the "figure" fragment is one of the descendant fragments of the "section" fragment. However the "figure" fragment (mapping in the twig pattern) as the child of the "section" has not arrived yet, the "section" fragment can not verifies its value. So we can not output the "figure" fragment as the result. When the "figure" fragment with region code $(9, 14, 4)$ arrives, its f-tuple value is set "true" since it satisfies the condition (i.e. $Level = 4 \wedge 6 < 9 \wedge 27 > 14$). So the "section" fragment triggers its descendant "figure" fragment with region code $(15, 26, 5)$. Then the "title" element in the same fragment is output.

From the analysis of the query example, we can conclude that by taking advantage of region coding scheme, checking an ancestor-descendant structural relationship is as easy as checking a parent-child structural relationship. Hence we can skip intermediate fragments along the path and produce the query results as soon as possible without waiting for all the correlated fragments to arrive.

4 Performance Evaluation

In this section, we present the results of performance evaluation of various algorithms over queries with different types, depths and document sizes on the same platform. All the experiments are run on a PC with 2.6GHz CPU, 512M memory. Data sets are generated by the xmlgen program. We have fragmented an XML document into fragments to produce an stream, based on the tag structure defining the fragmentation layout. And we implemented a query generator that takes the DTD as input and creates sets of XPath queries of different types and depths. We have used 4 queries on the document and compared the results among the following algorithms: (1) XFrag [3], (2) XFPro [8] and (3) XFPR. Figure 6 shows the queries that we used.

NO	Path expression
Q1	book/section/title
Q2	book/section/*/title
Q3	book//section/title
Q4	book/section[/figure/title]/section/title

Fig. 6. Path expression

In Figure 7 (a), (b) three kinds of processing strategies over various query types are tested and compared. From the result, we can conclude that for any query type, XFPR outperforms its counterparts, and query performance doesn't vary much on different query types. This because the XFPR only need to process the fragments that include key nodes or the output results. Furthermore, XFPro outperforms XFrag in time, because it deletes the dependent operations. But it

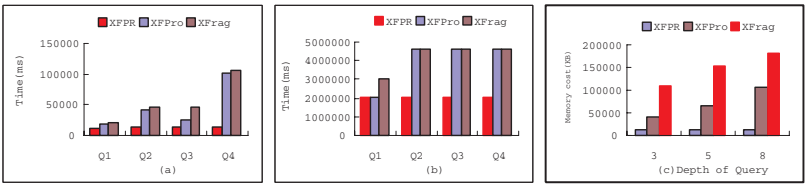


Fig. 7. Time with Different Queries

is not better than XFPR since it has to specify the query paths when queries including "*" or "//" and it cannot eliminate intermediate fillers. For XFrag, each fragment needs to be passed on through the pipeline and evaluated step by step. Therefore the performance of XFrag is affected by the character of query. For memory usage, complex queries will result in an increase in the number of operators joining in the query processing, along with more information in the association table and additional space consuming.

Figure 7 (c) shows the time of various query depths, 3, 5 and 8 respectively on the three methods. When the depth increases, the time of XFrag and XFPro increases due to the increased path steps. While with region coding , XFPR greatly reduces intermediate path steps' evaluation, thus time cost of deep queries is almost the same with that of short queries.

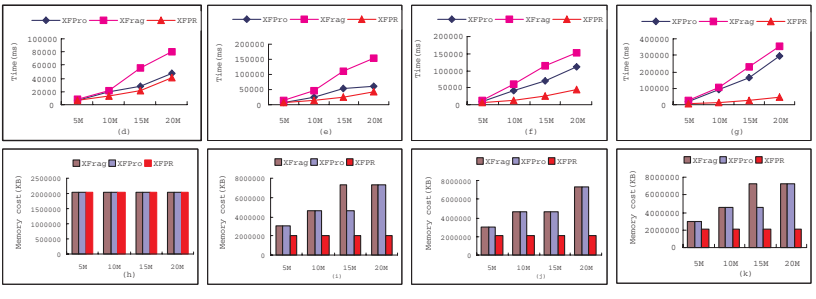


Fig. 8. Time with Different Documents

In Figure 8 (d),(e),(f) and (g), as the document size increases, XFrag observes the most costly, for much time is wasted in inserting fragments and finding relationship between them. In XFPro, with dependence pruning, it omits some operators corresponding the query path. The work XFPR performs best for the reason that quite a number of intermediate fillers are out of regard, and query path is greatly shortened. Figure 8 (h),(i),(j) and (k), illustrate the memory usage for different document size. For XFPR, memory usage is less impact with size increasing since many intermediate fillers are omited. For XFPro and XFrag,

this case is in reverse. However, XFPro performs a bit better than XFrag. XFPro only considers subroot nodes in tid tree while XFrag handles all operators in pipeline. With the document size increasing, more operators means more redundant information, so space cost becomes large.

5 Conclusions

This paper adopts the region coding scheme for XML documents and adapts it to streamed XML fragment model. Taking advantage of region coding scheme, we model the query expressions into *query tree* and propose a set of techniques, which enable further analysis and optimizations. Based on this optimized query tree, we map a query tree directly into an XML fragment query processor, named XFPR, which speed up query processing by skipping correlating adjacent fragments. Our experimental results over XPath expressions with different properties have clearly demonstrated the benefits of our approach.

Acknowledgement. This work is partially supported by National Natural Science Foundation of China under grant Nos. 60573089 and 60273074 and supported by Specialized Research Fund for the Doctoral Program of Higher Education under grant SRFDP 20040145016.

References

1. Fegaras, L., Levine, D., Bose, S., Chaluvadi, V.: Query processing of streamed XML data. In: Eleventh International Conference on Information and Knowledge Management (CIKM 2002), McLean, Virginia, USA (November 4–9, 2002)
2. Bose, S., Fegaras, L., Levine, D., Chaluvadi, V.: A query algebra for fragmented XML stream data. In: Proceedings of the 9th International Conference on Data Base Programming Languages, Potsdan, Germany (September 6–8, 2003)
3. Bose, S., Fegaras, L.: XFrag: A query processing framework for fragmented XML data. In: Eighth International Workshop on the Web and Databases (WebDB 2005), Baltimore, Maryland (June 16–17,2005)
4. Liu, Y., X.Liu, Xiao, L., .Ni, L., Zhang, X.: Location-aware topology matching in p2p systems. In: IEEE INFOCOM, Hongkong (2004)
5. Chen, L., Ng, R.: On the marriage of lp-norm and edit distance. In: Proceedings of 30th International Conference on Very Large DataBase, Toronto, Canada (August, 2004)
6. L. Chen, M.T.O., Oria, V.: Robust and fast similarity search for moving object trajectories. In: Proceedings of 24th ACM International Conference on Management of Data (SIGMOD'05), Baltimore, MD (June 2005)
7. Zhang, C., Naughton, J., DeWitt, D., Luo, Q., Lohman, G.: On supporting containment queries in relational database management systems. In: Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, Santa Barbara, CA (May, 2001)
8. Huo, H., Wang, G., Hui, X., Zhou, R., Ning, B., Xiao, C.: Efficient query processing for streamed XML fragments. In: The 11th International Conference on Database Systems for Advanced Applications, Singapore (April 12–15,2006)

PrefixTreeESpan: A Pattern Growth Algorithm for Mining Embedded Subtrees

Lei Zou[1], Yansheng Lu[1], Huaming Zhang[2], and Rong Hu[1]

[1] HuaZhong University of Science and Technology, 1037 Luoyu Road, Wuhan, P.R. China
[2] The University of Alabama in Huntsville, Huntsville, AL, 35806, USA
{zoulei,lys}@mail.hust.edu.cn, hzhang@cs.uah.edu,
hurong@smail.hust.edu.cn

Abstract. Frequent embedded subtree pattern mining is an important data mining problem with broad applications. In this paper, we propose a novel embedded subtree mining algorithm, called *PrefixTreeESpan* (i.e. **Prefix-Tree**-projected **E**mbedded-**S**ubtree **pa**ttern), which finds a subtree pattern by growing a frequent *prefix-tree*. Thus, using divide and conquer, mining local length-1 frequent subtree patterns in *Prefix-Tree-Projected database* recursively will lead to the complete set of frequent patterns. Different from *Chopper* and *XSpanner* [4], *PrefixTreeESpan* does not need a checking process. Our performance study shows that *PrefixTreeESpan* outperforms *Apriori-like* algorithm: *TreeMiner* [6], and *pattern-growth* algorithms :*Chopper* , *XSpanner* .

1 Introduction

Mining frequent structural patterns from a large tree database [2, 4, 6] and a graph database [5] is a new subject in frequent pattern mining. Many mining subtree pattern algorithms have been presented. Generally speaking, these algorithms can be broadly classified into three categories [3]. The first category is Apriori-like, such as *TreeMiner*; the second category of algorithms is based on enumeration tree, such as *FREQT* [1] and *CMTreeMiner* [3]. The above two categories are based on candidate-generation-and-test framework. The last category is based on pattern-growth.,such as *Chopper* and *XSpanner* [4]. In this paper, we present a novel subtree pattern mining method based on pattern-growth, called **PrefixTreeESpan** (i.e.**Prefix-Tre**e-projected **E**mbedded-**S**ubtree **pa**ttern). Its main idea is to examine the prefix-tree subtrees and project their corresponding project-instances into the projected database. Subtree patterns are grown recursively by exploring only local frequent patterns in each projected database. Using divide and conquer, this algorithm finds the complete set of frequent patterns correctly.

To summarize our contributions in this paper: 1) we define *prefix-tree* and *postfix-forest* for tree databases. The importance of *prefix-trees* and *postfix-forests* is that for tree databases, they play the role of *prefix* subsequences and *postfix* subsequences as in sequence databases; 2) we define *GE* (i.e ***G****rowth* ***E****lement*) for a *prefix-tree*. Because of *GE*, any local frequent pattern in some projected database must

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 499 – 505, 2006.

correspond to one frequent subtree pattern in the original database. So no candidate checking is needed, which is different from *Chopper* and *XSpanner*; 3) we propose a pattern growth algorithm *PrefixTreeESpan* for mining frequent *embedded* subtrees.

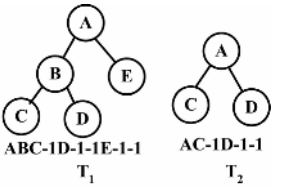


Fig. 1. Embedded Subtree and Pre-Order-String

2 Preliminaries

Trees and **Embedded Subtree:** Our trees are always **ordered, labeled, and rooted,** which can be denoted as T=(V,v_0,E ,$\sum$,L), where (1) V is the set of nodes ; (2) v_0 is the root; (3)E is the set of edges; (4)$\sum$ is an alphabet; (5) L is an function: V−>$\sum$ that assigns labels to nodes.

For a tree T with node set V, edge set E , we say that a tree T′ with node set V′, edge set E′, is an embedded subtree of T if and only if (1) V′ is a subset of V, (2) the labeling of nodes of V in T is preserved in T′, (3) (v_1,v_2)∈E′, where v_1 is the parent of v_2 in T′, if and only if v_1 is an **ancestor** (including parent) of v_2 in T and (4) for v_1, v_2 ∈V′, *preorder*(v_1)<*preorder*(v_2) in T′ if and only if *preorder*(v_1) < *preorder*(v_2) in T. If T′ is an embedded subtree of T, it will be denoted as $T'\in T$. Obviously, in Fig.1, T_2 is embedded subtree of T_1.

Mining Frequent Subtree Patterns Task: Given a trees database $D=\{T_i|$ i in a index set$\}$, where T_i is a tree , and a minimal support count *min_sup*≥*0,* and a pattern tree t_i, we define $d(t_i,T)=1$ if and only if $t_i\in T$, otherwise $d(t_i,T)=0$. The problem of mining frequent embedded subtrees is defined as to discover all pattern trees t_i, such that $Sup_D(t_i)=\sum_T\in_D d(t_i,T)\geq min\_sup$, where $\boldsymbol{Sup_D(t_i)}$ is denoted as the support count of the pattern tree t_i in the tree database D.

Pre-Order-String: According to [2], we have the recursive definition for the *pre-order string* : (1) for a rooted ordered tree T with a single node r , the pre-order string of T is $S(T)= l_r -1$, where l_r is the label for the single node r, '–1' is called *end flag*; and (2) for a rooted ordered tree T with more than one node, assuming the root of T is r (with label l_r) and the children of r are r_1……r_k from left to right then the pre-order string for T is $S(T)= l_rS(T_{r1})…S(T_{rk}) -1$.

3 PrefixTreeESpan: Mining Embedded Subtree Patterns by Prefix Tree Projections

Definition 1 (**Prefix-Tree**)**.** Let T be a tree with m *nodes*, S be a tree with n nodes*, where* n≤m. When pre-order scanning the tree T from its root until to the *n-th* node,

the scanning path forms a tree *M*. If the tree *S* is isomorphic to the tree *M*, we call *S* a ***Prefix-Tree*** of the tree *T*.

An example of prefix-trees is illustrated in Fig. 2. Given a tree *T* on the left, its five prefix trees are shown on the right, which have 1, 2,3,4,5 nodes respectively.

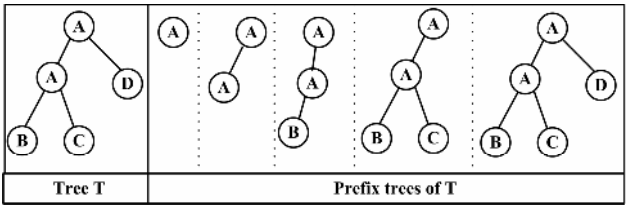


Fig. 2. An example of Prefix trees

Example 1 (Running Example). Let *D* be a tree database as in Fig.3 (a) and *min_support*=2. The set of node labels is {A, B, C, D,E}. Frequent subtrees can be mined by a **prefix-tree-projection** method in the following steps.

Step 1: Find length-1 frequent embedded subtree patterns. Scan the tree database *D* to find all frequent labels in trees. Each frequent label is a length-1 subtree patterns. They are <A^1 –1>, <B^1 –1>, <C^1 –1>, <D^1 –1> (*pre-order-string* of a tree). Obviously they are reported since they are frequent subtrees, as shown in Fig.3 (b). Notice that Arabic numeral in the superscript of 'A' in <A^1 –1 > denotes the position of 'A' in pre-order traversing the pattern tree <A^1 –1 >.

Step 2: Divide search space. The complete set of frequent subtrees patterns can be partitioned into the following four subsets according to the four prefix trees: (1) the ones having prefix-tree <A^1 –1>; (2) the ones having prefix-tree <B^1 –1>; (3) the ones having prefix-tree <C^1 –1>; (4) the ones having prefix-tree <D^1 –1>.

Step 3: Find subsets of subtrees. Construct corresponding *projected database* and mine each recursively to find the subsets of patterns subtrees.

How to construct the projected database?
For example, in order to construct <A^1 –1>-projected database, we scan the database *D* to find all **occurrences** of node 'A'. For **each occurrence** of 'A', the nodes after the occurrence, according to **pre-order traversal**, are formed to the **project-instance**. Obviously, one project-instance is corresponding to one occurrence. All project-instances are collected as <A^1 –1>**-project-database**, as shown in Fig.4 (a). There are three **occurrences** of the node 'A' in the database D, as shown in Fig.3 (a). For the fist occurrence of 'A', the nodes 'C' and 'E' are after the occurrence. They are formed the first project-instance. Though other nodes, such as 'A', 'B', 'D', are also after the first occurrence, they cannot be attached to the first occurrence directly or indirectly.

Definition 2 (Growth Element). Suppose that we have a tree *T* having *m* nodes, another tree T′ having (m+1) nodes, and *T* is the **prefix-tree** of T′. The node ***n*** which exists in T' but not exists in *T*, denoted as (T′ *T*), is called as the ***G**rowth **E**lement* (**GE** for short) of *T* w.r.t T′. Actually, the node ***n*** is the (m+1)-*th* node of T′ by

pre-order travel. Please notice that a *GE* represents not only the label of ***n***, but also its attaching position in *T*. A GE is always denoted as (*L*, *n*), where *L* is label, and *n* means that the GE is attached to the *n-th* node of the prefix-tree.

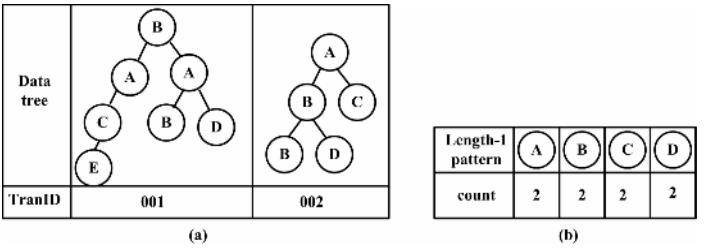


Fig. 3. (a) Database D and (b) frequent length-1 subtree patterns

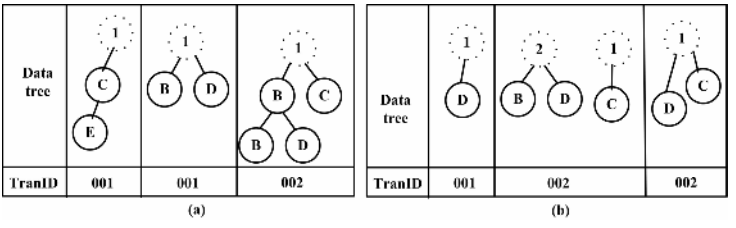


Fig. 4. (a) <A^1 –1>-projected database and (b) <A^1 B^2 –1 –1>-Projected database

Algorithm PrefixTreeESpan
Input: A tree database D, minimum support threshold *min_sup*
Output: All frequent subtree patterns
Methods:
1) Scan D and find all frequent label *b*.
2) **For each** frequent label *b*
3) Output pattern tree <b –1>;
4) Find all **Occurrences** of b in Database D, and construct <b –1>-projected database through collecting all corresponding *Project- Instances* in *D*;
5) **call** *Fre*(<b –1>, 1 , ProDB(D,<b -1>) , *min_sup*).

Function Fre(S, *n* , ProDB(D,S), *min_sup*)
Parameters: *S*: a subtree pattern ; *n*: the length of S; *ProDB(D,S)* : the <S>-projected database; *min_sup*: the minimum support threshold.
Methods:
1) Scan *ProDB(D,S)* once to find all frequent ***GEs*** *b*.
2) **For each** GE *b*
3) extent *S* by *b* to form a subtree pattern S′, and output S′.
4) Find all **Occurrences** of *b* in ProDB(D,S), and construct < S′ > -projected database through collecting all corresponding *Project- Instances* in *ProDB(D,S)* ;
5) **call** *Fre* (S′, *n+1* , *ProDB(D,* S′),*min_sup*).

Fig. 5. Algorithm PrefixTreeESpan

Scanning the <A^1 –1>-projected database, as shown in Fig.4 (a), we will find all **GEs**. They are {(C,1),(E,1),(B,1),(D,1)}. But (E,1) is not frequent. So frequent subtree patterns having prefix tree <A^1 –1> can be partitioned into 3 subsets: (1) those having prefix tree <A^1 B^2 –1 –1>; (2) those having prefix tree <A^1 C^2 –1 –1>; (3) those having prefix tree <A^1 D^2 –1 –1>. These subsets can be mined through constructing respective projected databases and mining each recursively.

For example, in order to construct <A^1 B^2 –1 –1 >-projected database, we scan the <A^1 –1>-projected database to find all occurrences of GE (B, 1). Each occurrence is corresponding to a project-instance. All project-instances are formed <A^1 B^2 –1 –1 >-projected database. The GEs in <A^1 B^2 –1 –1 >-projected database are {(D,1),(B,2),(D,2),(C,1)}. Notice that, the GE (D, 2) is different from the GE (D, 1). Since (D,2) means that the GE is attached to the second node of prefix-tree <A^1 B^2 –1 –1 >, while (D,1) means that the GE is attached to the first. We will find that only GE (D,1) is frequent. Recursively, all frequent pattern subtrees having prefix-tree <A^1 B^2 –1 –1> can be partitioned into only 1 subset: those having prefix-tree<A^1 B^2 –1 D^3 –1 –1>. We have to omit the mining process due to space limited.

Our algorithm *PrefixTreeESpan* is given in Fig.5.

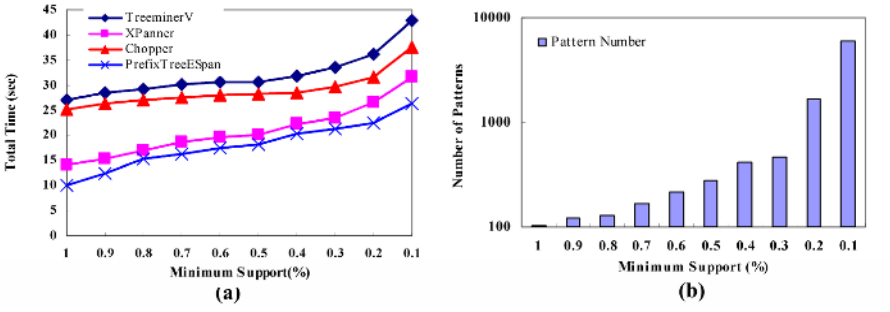


Fig. 6. (a) The running time of experiments on T1M and (b) The number of patterns on T1M

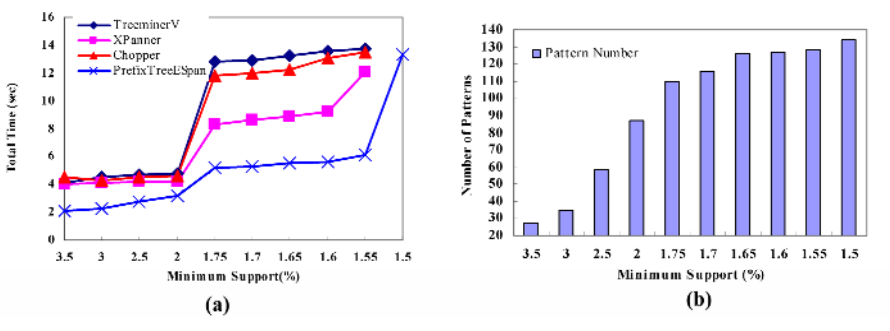


Fig. 7. (a) The running time of experiments on CSLOG and (b) The number of patterns on CSLOG

4 Experiments

In this section, we evaluate the performance of *PrefixTreeESpan*, and compare it with some other important algorithms. All experiments are done on a 1.7GHz Pentium Ⅳ PC with 1 GB main memory, running RedHat Fedora Core 3. All algorithms are implemented in standard C++ with STL library support and compiled by g++ 3.4.2 compiler. We compare our method *PrefixTreeESpan* with other important algorithms, such as *TreeMiner*, *Chooper* and *XSpanner* on 2 datasets, i.e. *T1M* and *CSLOG*[*]. *T1M* is a synthetic dataset, which is generated by the tree generation program[1] provided by Zaki [6].For *T1M*, we set parameters T=1,000,000 , *N*=100, *M*=10,000,*D*=10, *F*=10. *CSLOG* is a real dataset, which is also provided by Zaki [6]. The details about the tree generation program, parameters and *CSLOG* are given in [6]. Fig.6a and Fig.7a show the total running time, and the number of patterns is shown in Fig.6b and Fig.7b. Generating candidate patterns and testing in algorithm *TreeMiner* consume too much time, which leads to not good performance. In *Chopper* and *XSpanner*, we have to check *l-patterns* discovered in the first step whether there exit subtree patterns corresponding to them, which is a computationally expensive task. But our method overcomes this problem.

5 Conclusions

In this paper, we present a novel algorithm *PrefixTreeESpan* (i.e. **Prefix-Tre**e-projected **E**mbedded-**S**ubtree **pa**ttern) for mining embedded ordered subtree patterns from a large tree database. Mining local length-1 frequent subtree patterns in *Prefix-Tree-Projected database* recursively will lead to the complete set of frequent patterns.

Acknowledgments

We acknowledge to some valuable suggestions from Professor Jian Pei at Simon Fraser University. We also express our sincere thanks to Professor Mohammed J. Zaki at Rensselaer Polytechnic Institute for providing the synthetic data generator, CSLOGS data set and source codes of algorithm *Treeminer*. We thanks to Professor Wei Wang and his students at Fudan University. They provide us with the executable codes of algorithm *XSpanner*.

References

1. T. Asai, K. Abe, S. Kawasoe, H. Arimura, H. Sakamoto, and S. Arikawa. *Efficient Substructure Discovery from Large Semi-structured Data*. in *Proc. Second SIAM Int'l Conf Data Mining, Apr.2002*. 2002.
2. Y. Chi, S. Nijssen, R.R. Muntz, and J.N. Kok, *Frequent Subtree Mining -An Overiew*. Fundamenta Informaticae, 2005.

[1] Available at http://www.cs.rpi.edu/~zaki/software/

3. Y. Chi, Y. Xia, Y. Yang, and M. Richard R, *Mining Closed and Maximal Frequent Subtrees from Databases of Labeled Rooted Trees.* IEEE Transactions on Knowledge and Data Engineering, 2005.
4. C. Wang, M. Hong, J. Pei, H. Zhou, W. Wang, and B. Shi. *Efficient Pattern-Growth Methods for Frequent Tree Pattern Mining*. in *PAKDD 2004*. 2004.
5. X. Yan and J. Han. *CloseGraph: Mining Closed Frequent Graph Patterns*. in *KDD'03*. 2003.
6. M.J. Zaki. *Efficiently Mining Frequent Trees in a Forest*. in *Proceedings of the Int. Conf. on Knowledge Discovery and Data Mining*. 2002. Edmonton, Alberta, Canada.

Evaluating Interconnection Relationship for Path-Based XML Retrieval*

Xiaoguang Li[1], Ge Yu[2], Daling Wang[2], and Baoyan Song[1]

[1] School of Information Sci. and Tech., Liaoning University, Shenyang, P.R. China
[2] School of Information Sci. and Eng., Northeastern University, Shenyang, P.R. China
{xgli, bysong}@lnu.edu.cn, {yuge, dlwang}@mail.neu.edu.cn

Abstract. As one of popular solutions for meaningful result determination, interconnection relationship has been accepted widely by search engines. However, it suffers from heavy storage overhead and serious time complexity of index construction and query evaluation. We extend the retrieval syntax to support path query for partial structural information, propose a variant of interconnection relationship, and present an efficient evaluation algorithm with the extended structural summaries. The experiments show that with the promise of meaningful answers, our approach offers great performance benefits of evaluating interconnection relationship at run time.

1 Introduction

Unlike fully structured query, keyword based retrieval allows users to describe their querying intention by the partial structure information, even keywords only. The critical issue of keyword based retrieval is to determine meaningful results. One of popular methods to determine meaningful results is *interconnection relationship* proposed by XSEarch [1]. Interconnection relationship has been proved good retrieval quality and suitable for most situations in the real world. However, it suffers from the following drawbacks:

Firstly, XSEarch provides an offline index to record whether all pairs of nodes in an XML data satisfy the interconnection relationship or not. It suffers from serious storage overhead as the index size is much larger than the original file, and the time complexity of building index is $O(|T|^2)$ where $|T|$ is the node number of the XML data T. It is far beyond a user's tolerance for large-scale XML data. XSEarch proposes an online index to make the index size smaller, but it increases the cost of query evaluation greatly. Secondly, the syntax of XSEarch doesn't support path information, but tag and keyword only. In practice, users always know the partial structure information. The query with the partial structure information can convey more precise intention than the query of tag and keyword presented only.

Our contributions in this paper are (1) the extension of retrieval syntax, which supports both keyword and path to describe more precise intention; (2) the variant of

* Supported by the National Natural Science Foundation of China (No. 60573090).

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 506 – 511, 2006.

interconnection relationship, which forms the basis for efficient determination of interconnection relationship; (3) the integration of interconnection relationship evaluation into query processing with the technique of structural summaries, which speeds up interconnection computation.

2 Query Syntax and Semantics

We model XML data as a rooted, ordered tree $T = (N, E, r)$. The set of nodes $N = NE \cup NV$, where NE is the set of *inner nodes*, and NV is the set of *leaf nodes*. For a tree T, let $Label: NE \rightarrow String$ be the function that assigns the label to the node of NE. A *node path* is a sequence of nodes $u_1, u_2,\ldots,u_n$ such that for every pair u_i, u_{i+1} of consecutive nodes there is an edge $(u_i, u_{i+1}) \in E$. A *label path* $lp = l_1.l_2\ldots l_n$ is a sequence of label names of a path $p = u_1.u_2,\ldots,u_n$. A node path matches a label path if $Label(u_i)=l_i$, for $1 \le i \le n$. A node u matches a label path if a node path ending in u matches the label path.

The query language in this paper is mainly based on XSEarch, while extended to support path information. Formally, a search term has the form $p: k$, $p:$ or $:k$, where k is a keyword, and p is a regular label path expression. A query has the form $Q = (t_1,\ldots, t_n)$, a sequence of required and optional search terms. We say the node u satisfies the search term t if and only if

- when t has the form of $p:$, then the node u matches the label path p;
- when t has the form of $:k$, then the node u has the leaf whose value contains the keyword k;
- when t has the form of $p: k$, then u matches the label path p and has the leaf whose value contains the keyword k.

Furthermore, we say a result $r = (v_1, v_2,\ldots, v_n)$ satisfies a query Q if and only if each node v_i satisfies the corresponding search term t_i. And r is meaningful when any pair of v_i and v_j $(1 \le i, j \le n, i \ne j)$ is meaningful.

3 Variant of Interconnection Relationship

Definition 1 (***LCA of two label paths***)**.** For the label paths lp_1 and lp_2, a label path $lp = l_1 l_2\ldots l_m$ is said to the maximal prefix path of lp_1 and lp_2 if and only if:

- lp is the prefix path of both lp_1 and lp_2, and
- $\forall lp'$, lp' is a prefix path of both lp_1 and lp_2, then lp' is the prefix path of lp

and then l_m is the LCA of lp_1 and lp_2.

Definition 2 (***full label path***)**.** For the node u, suppose $np=u_0.u_1.u_2\ldots u_n.u$ is the node path from the root u_0 to u, the full label path of u is the label path matching the node path np..

Definition 3. (***k-ancestor***)**.** For the node u, suppose $u_0.u_1.u_2\ldots u_n.u$ is the path from the root u_0 to u, the node u's k-ancestor is the node u_{k-1}, for $k \le n$.

***The variant of Interconnection Relationship*:** For a query $Q = (t_1, t_2,\ldots, t_n)$, let $r = (v_1, v_2,\ldots, v_n)$ be one of results satisfying Q. if r is meaningful, then for any pair of nodes v_i and v_j ($1 \le i, j \le n$, $i \ne j$), the following conditions hold:

1. Let lp_i and lp_j be the full label paths of v_i and v_j respectively, let l be the LCA of lp_i and lp_j. Then there doesn't exist two distinct labels with the same name among the labels from l to lp_i and lp_j; and
2. Let k be the length from the root of an XML data to l, and let the node v_i' and v_j' be the k-ancestor of v_i and v_j respectively, then $v_i' = v_j'$; otherwise
3. $Label(v_i) = Label(v_j)$.

Theorem 1. The variant of interconnection relationship is equivalent to interconnection relationship.

4 Evaluating Interconnection Relationship for Path Query

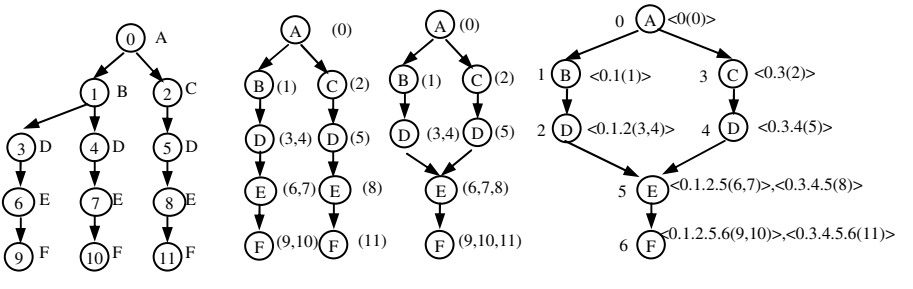
Here we extend two typical summaries: Strong DataGuide [3] and $A(k)$-index [4] to support computing the k-ancestor and the full label path, and propose an algorithm to evaluate the variant efficiently.

4.1 Extended Structural Summaries

Strong DataGuide. Strong DataGuide as illustrated in Fig. 1(*b*), preserves all complex paths, and promises precise and safe results for any length of query. With the definition of DataGuide, we have the property below.

Property 1. For a strong DataGuide I, there must be the unique full label path of a node in I corresponding to the full label path in T.

***A*(*k*)-index.** $A(k)$-index is a kind of k-bisimilarity based summaries. The full label path of a node can not be obtained though $A(k)$-index because the nodes are merged according to local similarity.



(*a*) An XML tree (*b*) DataGuide (*c*) $A(1)$-index (*d*) extended $A(1)$-index

Fig. 1. Illustration of DataGuide, $A(1)$-index and extended $A(1)$-index

Now, we extended an $A(k)$-index I to $I' = (V, E, UID, EXT')$, where UID is the set of *unique identifier of index node* (*uid*). The node in $ext^I(A)$ of an index node A is associated with a sequence of index node identifiers, called *path identifier* (*pid*),

which represents the node's full label path in I'. Furthermore, the nodes in $ext^I(A)$ are grouped by their *pid*, and the set of groups are denoted as $G(A)$. Then the element e of $ext^{I'}(A)$ consists of the nodes $g(A) \in G(A)$ and their path identifier *pid*, i.e. e=(*pid*, $g(A)$). Fig. 1(*d*) shows the extended $A(1)$-index of Fig. 1(*c*), where the number in the left of an index node is its *uid*, and the right is its extent. Obviously, if a node u is the ancestor of a node v, then u's *pid* is the prefix of v's *pid*. Moreover, we have the property:

Property 2. The label path of a node obtained by traversing the extended $A(k)$-index I' backward with the node's *pid* is its full label path corresponding to that in T.

Theorem 2. Given an index node A in an extended $A(k)$-index I', $Succ(A)$= $\{u | u \in e.g(B)$, where $e \in ext^{I'}(B)$, B is the child of A and there exists $e' \in ext^{I'}(A)$ that $e'.pid$ is the prefix of $e.pid$ and $|e'.pid| = |e.pid|+1$.

Extended $A(k)$-index construction. Our algorithm for the extended $A(k)$-index construction is a variant of the standard algorithm [4]. In our algorithm, we needn't to compute the successors for a node, but the successors of the elements in an index node are computed only through the *pid* itself with Theorem 2. Instead of resorting to searching in the original data, the summary construction can be speeded up by this means.

4.2 Computing *k*-Ancestor

Here we examine two typical numbering schemes, path-based numbering and region-based numbering, and study *k*-ancestor computation with the numbering scheme and structural summaries.

Property 3. For a path-based numbering scheme, let $i_0.i_1 \ldots i_n$ be the code of a node u, a node v is k-ancestor of the node u iff v'code is the prefix with the length k of u's code, that is $i_0.i_1 \ldots i_{k-1}$.

Property 4. For a region-based numbering scheme, a node v is the k-ancestor of a node u iff $v.start < u.start < u.end < v.end$, and $v.level = k$.

Unlike path based numbering scheme, the k-ancestor can not be computed only by the region-based code itself. The theorems bellow show k-ancestor computing can be restricted to a small range for the structural summaries in Section 4.1 used.

Theorem 3. For two index nodes A and B in a DataGuide I on tree model, if A is the k-ancestor of B, then the k-ancestor of nodes in $ext^I(B)$ must be in $ext^I(A)$.

Theorem 4. For an extended $A(k)$-index I', given a node u, let A be the index node that has the unique element $e \in ext(A)$ and $u \in e.g(A)$, let lp be the full label path obtained by $e.pid$, and let A' be the k-th index node from the beginning in the lp, then u's k-ancestor must be in the element e' of $ext^{I'}(A')$, where $e'.pid$ is the prefix of $e.pid$ and $|e'.pid| = k$.

4.3 Evaluating Algorithm

Property 5. For a set of nodes $N=\{u_1,\ldots,u_i,\ldots,u_m\}$, suppose $u_1 < \ldots u_{i-1} < u_i < u_{i+1} \ldots < u_m$, where "<" represent the order of numbering scheme, such as the preorder or post

order for region-based numbering scheme, and the lexicographic order for path-based numbering scheme, then we have that the nodes in the set $N'=\{u'_1,\ldots,u'_i,\ldots,u'_m\}$, where u'_i is the k-ancestor of u_i for $1\le i\le m$, satisfy $u'_1\le\ldots u'_{i-1}\le u'_i\le u'_{i+1}\ldots\le u'_m$, where "≤" represents the order "<" or equal.

The evaluating algorithm firstly verifies the elements, instead of nodes, according to Condition 1 and 3, and further verifies the nodes in the elements passing through in a manner of merge join where the join condition is the identical k-ancestors of two nodes. Note that the nodes in an element should be sorted on the numbering order, which can be achieved at the stage of summary construction. With Property 5, all results satisfying Condition 2 can be retrieved safely. Due to the limitation of space, the detail of the algorithm is not given here.

Theorem 5. The time complexity of the algorithm of evaluating meaningful results is $O\left(\left(\prod_{i=1}^{n} n_e^i\right)\left(n^2(n_{\max}+n'_{\max})+d^2\right)\right)$, where n_e^i is the number of elements for the term t_i, n is the number of terms in the query Q, n_{max} is the maximal number of nodes in all elements, n_{max}' is the maximal number of nodes in the immediate results and d is the maximal length of full label path.

5 Performance Analyses

We test the construction time of the extended A(k)-index as shown in Fig. 2, and the efficiency of evaluating interconnection relationship as shown in Fig.3 and Fig. 4. the experimental results indicate our approach outperforms the original one.

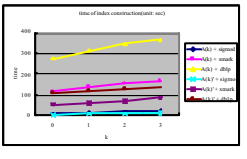


Fig. 2. Time of index construction

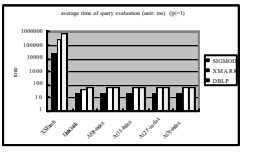


Fig. 3. Average time of query evaluation (|p|=1)

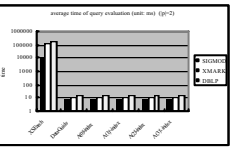


Fig. 4. Average time of query evaluation (|p|=2)

6 Related Works

Numerous works have been done about IR-style query on XML document. XIRQL [8] and XXL [9] propose the query language to support keyword query, but don't consider the notion of meaningful results. Meet [5] operator returns LCAs as query answers without consideration of the meaningfulness too. EquiX [10] and XRANK [6] propose a ranking mechanism, but just return the most special fragment as answers, of which parts maybe semantically unrelated. Schema-Free XQuery [7] put forwards a concept of meaningful lowest common ancestor and shows a promising precision of query that outperforms other approaches. But Schema-Free XQuery uses and extends XQuery as its query language, which is not as easy to write a query as

IR-style keyword search for naïve users. Furthermore it requires accessing all the nodes with the same entity type, even a query only contains keywords, not any element. The syntax of XSEarch doesn't support path information, but tags and keywords only. The offline index in XSEarch leads to serious storage overhead, while the online index increases greatly the cost of evaluating query. In previous works, we have proposed an efficient approach to evaluate the interconnection relationship for keyword query with the enhanced inverse index [2].

7 Conclusions

In this paper, we introduce the retrieval syntax extending the traditional one to support the path query, and propose a variant of interconnection relationship, which is equivalent to interconnection relationship and makes the basement for efficient interconnection evaluation. We extend two typical structural summaries, strong DataGuide and $A(k)$-index, for path evaluation and the variant evaluation efficiently. Our approach need not create additional relationship index but just utilize the existing structural summary, and extensive experiments show that with the promise of returning meaningful answers, our approach offers great performance benefits of evaluating interconnection relationship at run time.

References

1. Cohen. S, Mamou. J, Kanza. Y, Sagiv. Y. XSearch: a semantic search engine for XML. Proc. of VLDB, 2003, pp:45-56
2. Li. X.G, Gong. J, Wang. D.L, Yu. G. An effective and efficient approach for keyword-based XML retrieval. Proc. of WAIM, 2005, pp:56-67.
3. Goldman. R, Widom. J. DataGuides: enabling query formulation and optimization in semistructured databases. Proc. of VLDB, 1997, pp: 436-445.
4. Kaushik. R, Shenoy. P, etc. Exploiting local similarity for indexing paths in graph-structured data. Proc. of ICDE, 2002, pp: 129~140.
5. Schmidt. A, Kersten. M, Windhouwer. M. Querying XML document made easy: nearest concept queries. Proc. of ICDE, 2001, pp: 321
6. Guo. L, Shao. F, Botev. C, Shanmugasundaram. J. XRank: ranked keyword search over XML documents. Proc. of SIGMOD, 2003, pp: 16-27
7. Li. Y. Y, Yu. C, Jagadish. H.V. Schema-free XQuery. Proc. of VLDB, 2004, pp: 72-83.
8. N. Fuhr and K. Grobjoham. XIRQL: a query language for information retrieval in XML document. Proc. of SIGIR, 2001.
9. Theobald. A and Weikum. G. The index-based XXL search engine for querying XML data with relevance ranking. Proc. of EDBT, 2002.
10. Cohen. S, Kanza. Y, Kogan. Y, et al. EquiX: a search and query language for XML. Proc. of JASIST, 2002.

User Models: A Contribution to Pragmatics of Web Information Systems Design

Klaus-Dieter Schewe[1] and Bernhard Thalheim[2]

[1] Massey University, Information Science Research Centre
Private Bag 11222, Palmerston North, New Zealand
k.d.schewe@massey.ac.nz
[2] Christian Albrechts University Kiel, Department of Computer Science
Olshausenstr. 40, D-24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract. On a high level of abstraction a Web Information System (WIS) can be described by a storyboard, which in an abstract way specifies who will be using the system, in which way and for which goals. While syntax and semantics of storyboarding has been well explored, its pragmatics has not. A first step towards pragmatics of storyboards is the observation and documentation of life cases in reality. These, however, have to be complemented by user models. This paper presents an approach to capture user models and to specify the various facets of actor profiles that are needed for them. We analyse actor profiles and present a semi-formal way for their documentation. We outline how these profiles can be used to specify actors, which are an important component of a storyboard.

1 Introduction

A Web Information System (WIS) is an information system that can be accessed through the world-wide-web. On a high level of abstraction a WIS can be described by a storyboard [15], which in an abstract way specifies who will be using the system, in which way and for which goals. In a nutshell, a *storyboard* consists of three parts:

- a *story space*, which itself consists of a hierarchy of labelled directed graphs called *scenarios*, one of which is the main scenario, whereas the others define the details of *scenes*, i.e. nodes in a higher scenario, and a *plot* that is specified by an assignment-free process, in which the basic actions correspond to the labels of edges in the scenarios,
- a set of *actors*, i.e. abstractions of user groups that are defined by *roles*, which determine obligations and rights, and *user profiles*, which determine user preferences,
- and a set of *tasks* that are associated with *goals* the users may have.

In addition, there are many constraints comprising static, dynamic and deontic constraints for pre- and postconditions, triggering and enabling events, rights

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 512–523, 2006.

and obligations of roles, preference rules for user types, and other dependencies on the plot. Details of storyboarding have been described in [15].

Storyboarding and also the preceding strategic modelling of WIS [11] are unique to our approach to WIS modelling. Other approaches to WIS engineering such as the object oriented OOHDM [4,13,16], WebML [1], HERA [6] and variants of UML [2,9] concentrate on providing models of content, navigation and interaction by means of extended views, which in our own work is captured by so-called *media types* [15]. WSDM [3] emphasises the additional need for a mission statement and a high-level description of processes and users in the WIS.

While syntax and semantics of storyboarding has been well explored, its pragmatics apart from the use of metaphors [17] has not. In a parallel submission [14] we introduced life cases, which capture observations of user behaviour in reality. The idea is that these observations give rise to both a concrete scenario for the usage of the WIS including roles and user profiles. We then derive the candidate scenarios by abstraction. Further integration and refinement of these scenarios gives rise to the story space.

In this paper we deal with the problem of extracting user models. For this we specify the various facets of actor profiles. We then analyse actor profiles and present a semi-formal way for their documentation. Finally, we outline how these profiles can be used to specify actors.

Explicit modelling of users has a long tradition in artificial intelligence, and has emerged as a field in its own right [5,7,8,10]. However, the focus is usually slightly different from ours, as we do not intend to adapt system behaviour to observed user behaviour, but use classification of users as a modelling means for capturing WIS pragmatics. This is similar to the focus in [12], though we focus on WISs.

2 User Models

User modelling has changed the development to human-computer interfaces, and allows to tailor systems to the users, their needs, abilities, and preferences. User modelling is based on the specification of the *user profile* that addresses the characterization of the user, and the specification of the *user portfolio* that describes the user's tasks, involvement, and collaboration on the basis of the mission of the WIS. In general, user profiles are specified through the *education profile* based on an insight into the knowledge, skills, and abilities of potential users, the *work profile* with a specification of the specific work approaches of users, and the *personality profile* representing the specific properties of users.

Education Profiles. The *education profile* is characterized by properties obtained during education, training, and other educational activities, i.e. education, capabilities, and application knowledge.

The *education* is characterised by the technical and professional training a user got. Technical training emphasises the understanding and practical application of basic principles of science and mathematics. Professional training places major emphasis upon the theories, understanding, and principles in such fields as science, and engineering. It results in erudition, knowledge, and literacy.

Capabilities of the user for task solutions are based on the attainment of proficiency in skills such as understanding the problem area, reasoning capabilities on analogy, realizing variations of the problem solution, solving and handling problems, communication abilities, abilities for explaining results and solutions, and abilities for integration of partial problem solutions.

Application knowledge depends on the application type, the application domain, the application structuring by subjects, the direct structure, the description, the context and environment of the description, the parameters of the application, and application functions such as accumulation, observation, structuring, summarizing, etc.

Work Profiles. The *work profile* is mainly characterized by the task solving knowledge and skills in the application area, i.e. task expertise and experience, system experience, and information and interaction profiles.

The *task expertise* describes the exact and partial knowledge ofdata, procedures, algorithms, functions, constraints, associations, etc. Partial knowledge is characterised by vagueness, variants of representations, calculi, comments, constraints, etc.

The *task experience* identifies both positive experience, e.g. applicable knowledge, strategies, models and theories, and negative experience, e.g. development or knowledge deficits, cooperation and planning support reduction, capability gaps, insufficient competences, etc. In addition, the users' experience in coping with errors, self-organization, planning, and abstraction can be specified.

The system experience depends on the systems to be explored and used. It is limited by the user's abilities to cope with information delivered by a system and system-supported collaboration, and to work with systems in parallel to working without system support.

The information profile is based on the information needs of a user discussed below, i.e. the intentions in approaching the system, the amount of information a user can cope with, and the complexity of information a user can handle. The information profile can be modelled by database schemata that are extended by a user's preferred way of browsing through information.

The interaction profile of a user is determined by his frequency, intensity, and style of utilization of the WIS. Users might be experienced in working with several workplaces or with only one, e.g. a single browser. The interaction profile supports a flexible communication, cooperation, and coordination of a given user.

Personality Profiles. The *personality profile* of a user characterizes his *general properties* and his preferences in dealing with the WIS.

General properties of the user or of a user group are the *status* in the enterprise, community etc., the *context* for performing a task such as position,allocated tasks, task area and customer properties, the *psychological and sensory properties* capturing vision, hearing, motoric control, information processing, speed of information uptake, attitude and anxiety, the *background and personality factors* capturing intentions, motivation, expectations, emotions, etc., *training and education*, *behavioural patterns* such as acceptance of system properties and methods, *required guidance* by help and tutoring systems, and *user type*, i.e. whether is casual, user, knowledgeable or expert.

For characterization of *user preferences* we may concentrate on those that are important for WIS development such as preferences for input and output devices and dialogues. For input and output devices this captures the *handling of types* such as text, picture, graphics, audio and video and the required support for these, the *preference of specific forms*, e.g. character-, line-, window- or form-orientation with presentation preferences for colour, shape, size, font, format, contrast, etc., the *required guidance and help during input or output* capturing tutoring, explanations, alternatives, etc., *command preferences*, e.g. the command style, and the *understanding of the input and output tasks* depending on the level and capabilities.

Preferences for the design of dialogues capture *dialogue properties* such as the information load, flexibility, complexity, expressive power, initiated actions, consistency, feedback, etc., *dialogue forms and styles* such as conversation, question/answer form, input/act forms and various presentation styles, *dialogue structuring*, e.g. closed dialogues, open discussions, undirected querying, interrogation, etc., *dialogue control* on amount, quantity, relevance, representation, etc., and *dialogue support*, e.g. by tools.

Representation of User Profiles. User profiles can also be described in a semi-formal way as follows:

`User profile`:	⟨user profile name⟩
`Education profile`:	⟨general description⟩
`Education`:	⟨list of degrees⟩
`Capabilities`:	⟨list of skills⟩
`Knowledge`:	⟨description of knowledge in the application⟩
`Work profile`:	⟨general description⟩
`Task expertise`:	⟨description of knowledge⟩
`Task experience`:	⟨positive and negative experience⟩
`System experience`:	⟨experience with infrastructure planned⟩
`Information profile`:	⟨information need⟩
`Interaction profile`:	⟨interaction properties⟩
`Personality profile`:	⟨general description⟩
`General properties`:	⟨list of user properties⟩
`Preferences`:	⟨list of input/output/dialogue preferences⟩
`Based On`:	⟨*user goals*⟩
`Based On`:	⟨*user types*⟩

3 Actor Profiles

We group users by their information demand and requested content, their utilisation patterns, and their specific utilisation and context. The abstraction of a group of users is called an *actor*. Actors are characterized by profiles and portfolios. Same as for users actor profiles consist of the *education*, *work* and *personality* profiles. Profiles are used for the derivation of preferences. These preferences are used for adaptation of scenes to specific actors.

An actor takes part in some scenes of the storyboard. S/he is characterised by the character, location,actions, intention, and circumstances, i.e. what happened before s/he came there. He must discern the actions he is taking as well as the rhythms of the work as a whole, and he must determine what adjustments must be made in his/her storyboard for each of the other characters. Though actors are abstractions of user groups, we may characterize them by these abstract properties.

Example 1. Let us consider an edutainment site. Based on the profile we can derive a number of preferences of learners. The background knowledge leads to different speed and reception of content by the learner. Work abilities and habits influence current work. Learning styles have many many facets depending on the profile of the learner. The social environment with cultural and psychological differences, the support for treatment of history of the learning process without annoying repetitions, and the the content change management with or without refresh are mapped to preferences we apply to the storyboard.

The profiles of actors can be specified using the following template:

`Actor profile`:	⟨actor profile name⟩
`Grouping criteria`:	⟨characteristics of grouping of users⟩
`Information demand`:	⟨general description⟩
`Utilisation pattern`:	⟨general description⟩
`Specific utilisation`:	⟨general description⟩
`Actor context`:	⟨general description⟩

4 Actor Portfolios

A portfolio is determined by the responsibilities one has and is based on a number of targets. The *actor portfolio* within an application is thus based on a set of tasks an actor has or intends to complete and for which solution the actor has the authority and control, and a description of involvement within the task solution.

Task modelling means to understand what a user wants to accomplish while visiting the WIS. At the same time, task analysis may lead to a reorganisation of the work processes to be supported. The WIS should support streamlining and

augmenting what users do. Task analysis leads to a description of things users do and those they need to know. It does not specify how the task is accomplished. The tasks supported by a WIS need to be representative for the application, important within the application, and completely supported. Task support can be tailored depending on the profile and the context of the actors.

4.1 Tasks

A *task* is a usually assigned piece of work, which often has to be finished within a certain time by a user or a set of users whose duty is its completion. It implies work imposed by a user in authority and obligation or responsibility to perform. A task may consist of subtasks, so we assume that tasks can be constructed on the basis of elementary tasks. Thus, a task is characterized by:

Problem statement: Tasks are associated to problems, for which often a class of solution strategies is provided. Additionally, problems often require collaboration with the local and global systems and with other actors.

Target states: After successfully completing a task we may observe a change in the state. Target states are specified by means of target conditions. Some of the target conditions can be optional. If no optional conditions are given then all conditions are obligatory. Target states correspond to intents.

Initial states: The necessity for tasks enactment is based on the insufficiency of the current state of affairs. Additionally, task enactment conditions may specify, under which circumstances we can start task execution.

Profile presupposed for task completion: The completion of a task requires skills, experience and knowledge that must be presupposed by the user, whenever the task is going to be activated. Tasks may be embedded into a certain organizational context. The profile also presupposes a certain technical environment, e.g. communication, information, and workspace systems.

Instruments for task completion: Task enactment is supported by instruments such as actions and data. Problems are solved on the basis of an information demand and within a class of functions that might be used for task solution. Leteron the information demand is mapped to database views or media objects. The function utilization is organized on the basis of workflows.

Collaboration profile: A task may be allocated to a single user or a group of collaborating users. In the latter case, the subtasks for each user, the obligations, the time and other restrictions must be given.

Auxiliary conditions: The settling of tasks may be restricted. Typical auxiliary conditions are based on rights for the direct handling and retrieval, roles of the antagonist and the protagonist, and obligations required whenever settling a task.

We extend tasks by a *general task context* that consists of the releasing actor(s), the actors to which task completion and results of completion are

reported, the organisational unit that calls for task completion, the activities that are required by users, the aids that can be used for task completion, and the workspace that is supporting task completion. The latter is mapped to session support or to temporary workplaces of users.

Tasks are associated with intents a user or a group of users have. The intents are either objects or targets [14]. In the latter case, we can directly map the task specification to corresponding targets.

Example 2. In a learning WIS for data mining learners have very different skills, very different background knowledge, very different approaches to learning, and want to use learning systems whenever this is really necessary. Let us consider tasks to understand the essentials of data mining and seeks for an algorithm that serves his needs.

1. The learner searches first algorithms that might be applicable within the application under consideration.
2. Next the user needs to understand the algorithm that seems to be the most appropriate. Understanding also includes learning the interpretation of results obtained by applying one of the algorithms to the learners data. For that, the learner may first look over demonstrations and illustrations for the chosen algorithm.
3. Now the learner experiments with the data. Data are selected and configured for the algorithm. The algorithm is executed and results are obtained.
4. Finally, the results are explained to the learner. S/he may now continue by selecting another method or algorithm and obtain additional insight into the data to be analyzed.

We may map these subtasks to the intents: exploring data mining algorithms, understanding how to prepare data, learning to interpret results obtained.

At the same time this task may be required by a life case [14]. For instance, the learner works as a clerk in a bank and tries to figure out which loans have become faulty and what is the reason for it. In this case, the clerk becomes a learner and tries to learn data mining as much as it is needed for the life case.

4.2 Task Execution

The *task execution model* defines what, when, how, by whom, and with which data work has to be done:

- Task activities define how the work is actually done, e.g. by instructing the user or by invoking an application.
- The control flow defines the sequence of activity execution. Depending on the specific WIS several control constructs are available such as decision, alternative, parallelism, loops, start/exit conditions, and different kinds of constraints such as deadlines.

– The data flow specifies how data flows through the task. Depending on the WIS different concepts exist such as input/output-container of activities or local/global variables.

The *result of execution* is the state of affairs reached and the satisfaction of target conditions. Task enactment is supported by pre-determined plans. These plans are scenarios which have a number of stories they are supporting. We may use blackboards for recording open targets. Whenever a target is completed and a task is considered to be successfully fulfilled then we delete this task from the blackboard. The *involvement* of actors within the task solutions is based on the specification of the *role* an actor plays, the *part* the actor plays within the scenario, and the *rights* and *obligations* an actor has within the given role.

The role specifies the behaviour expected of an actor. A role is a comprehensive pattern of behaviour and serves as a strategy for coping with recurrent situations and dealing with the roles of others. A role remains relatively stable, even though different users occupy the position. A user may have a unique style of role execution, but this is exhibited within the boundaries of the expected behaviour of the actor. Role expectations include both actions and qualities: for instance, in real life a teacher may be expected not only to deliver lectures, assign homework, and prepare examinations but also to be dedicated, concerned, honest, and responsible.

There are two types of roles: declarative and contextual ones. The former ones declare that an actor is playing a particular role, e,g, an actor being identified and as an employee. Contextual roles show how an actor acts within the context of a scenario and how an actor is involved within the context of another scenario. Declarative roles may be modelled by associating the actor to a role type. Contextual roles are modelled by associating an actor with the work effort the actor is assigned to and a role type describing the involvement of the actor. The role type provides a description of the role and can be hierarchically structured. Roles may also be hierarchically structured. At the same time roles may be played in collaboration.

Example 3. In an e-business application we can distinguish roles such as worker, customer, and roles within the distribution scenario. The worker role entails roles such as contractor, contact, employee, and sponsor. The customer role takes part in bills to customer, ship to customer, and end-user customer. Within the distribution scenario we may distinguish active and pro-active roles, i.e. the distributor and an agent. These roles can again be distinguished into association, competitor, partner, prospect, referrer, regulatory agency, shareholder, supplier, and website visitor.

At the same time several roles are played by people or organizations within the context of project management. A person or organization may be the sponsor, worker, manager, lead, advisor or quality assurance manager for a project.

Users usually occupy several positions, which may or may not be compatible with one another. So actors may play several roles at the same time. In this

case, they must have a combined view of the application and a combined access to their capabilities. Therefore, the WIS must provide the ability to *set-up* and *manage* these roles, and to *combine* portfolios so that the application maintains to be consistent for each actor.

The *part* associates the atmosphere, mission, objectives, and objects with the involvement of the actor in a scenario. It is refined, when context of the involvement is taken into consideration. The part is based on a categorization of the behavior of users. We may derive that users tackling some problem do not like to wait. The part is described through the kind of interaction actors are preferring for the current tasks and the behaviour of actors we need to support. Therefore, the behavioral stereotype is described through the part.

A role entails certain *obligations* and *rights*. Obligations are obligatory tasks, conduct, service, or functions that arise from the users role under specified conditions. Rights are granted as a peculiar benefit, advantage, or favor to a role. We specify right and obligations as conditions that are mapped to conditions on actions an actor can call in a scenario. Obligations and rights include such for functionality, for collaboration, for content, and for dissemination of such to other actors. Obligations and rights may include also the specification of obligations, permissions and prohibitions in the case of exceptional situations.

4.3 Actor Portfolios and Life Cases

Portfolios refine the specification of life cases, which are collected by observing real life situations and the data and action flow in them. The life case description allows to deduct the demands for data, functions, performance, and supporting aids such as *workspace* and *workplace support*. The workspace is determined by the portfolio, its support by the profile of the actors involved.

Example 4. Let us consider a life case *relocation*. The portfolio for this life case is based on a number of tasks such as: change basic data of issuer, provide a view on the data to all other interested parties, initiate tasks of associated life cases, archive old data and current changes, handle exceptional tasks, and change data in official documents. The life case is complex due to the large variety of subtasks, the initiation of which depends on the issuer. These tasks are associated with subtasks such as checking, whether all necessary documents have been supplied, special handling must be applied, etc.

At the same time we need to consider the involvement of at least four different actors. The issuer raises the change of data task handled by civil servants. If the issuer raises exceptions for data transfer then special support is required by an actor that acts as a data protection official. Finally, the data on the relocation are delivered to a number of actors that play the role of official bodies or support organizations. The security profiles of the issuer and the civil servants require a sophisticated data management, e.g. tax and social security data must be kept in isolation from each other.

For this life case we get tasks related to basic changes, to changes of associated parties, to change of data ownership, and to tasks of associated life cases. The

actors participating in the life case are restricted to the four main types of actors. Their subtasks are different from the tasks of the issuer.

During analysis of user behaviour a number of principles needs to be observed. These principles form the *portfolio restrictions*.

Non-determinism of users behaviour: We need to cope with all possible options that are plausible. We do not assert that an option for execution will be followed. The various user intents are stored on a blackboard that is used to record tasks and subtasks to be completed.

Intention-based behaviour: We may assume that a user intermittently terminate actions as soon as their aims and their targets have been achieved. A typical implementation support for this principle is realized through the submit button. After submitting data or after hitting the submit button the user starts a new scenario or another scene under the assumption that the current intention has been achieved.

Task-oriented termination: We may assume that a user will terminate the current interaction when their whole task has been achieved. In achieving a target, subsidiary tasks might be generated. Termination is based on termination conditions. For instance, the user of an ATM machine expects that with termination the bank card is returned. Therefore, the user will have the card after termination of the ATM scenario.

Reactive behaviour: Users of a WIS are reacting to stimuli or messages of the WIS the user is observing. Rather than specifying each reaction in a separated form we use generic functions for the specification of this behaviour. Reactive behaviour can be modelled on the basis of observation-reaction pairs.

Collaboration target behaviour: The user enters an interaction with the knowledge of the task dependent on the targets that must be discharged. The collaboration target is based on a pre-determined plan that is known in advance to the user. Once a target has been achieved the user does not expect to need to repeat the actions again.

Separation of mental and physical actions: Whenever the user commits to taking a physical action, then this action cannot be revoked after a certain point. Before doing the physical action the user generates the signal sent from the brain to the motoric system. We supply the user model by a guard-action pair linking mental actions with physical actions.

No-option-based completion: A user may abort a scenario when there is no apparent action the user can take that would help to complete the task. If e.g. the user of a railway ticketing system does not find the appropriate option, then the user might give up and might assume that the intentions cannot be achieved. We model this opportunity by a no-option-based addendum that is added to each scene where a user may leave the story.

Relevance: A user chooses an action only, if this action seems to be relevant for achieving the currently considered targets.

4.4 Representation of Portfolios

Portfolios can be specified in a semi-formal way using the following template:

`Actor portfolio:`	⟨actor portfolio name⟩
`Task:`	⟨general description⟩
`Extension of:`	`General characterisation of tasks`
`Characterisation:`	⟨general description⟩
`Initial state:`	⟨characterisation of the initial state⟩
`Target state:`	⟨characterisation of the target state⟩
`Profile:`	⟨profile presupposed for solution⟩
`Instruments:`	⟨list of instruments for solution⟩
`Collaboration:`	⟨specification of collaboration required⟩
`Auxiliary:`	⟨list of auxiliary conditions⟩
`Execution:`	⟨list of activities, control, data⟩
`Result:`	⟨final state, satisfied target conditions⟩
`Actors involvement:`	⟨general description⟩
`Role:`	⟨description of role⟩
`Part:`	⟨behavioural categories and stereotypes⟩
`Collaboration:`	⟨general description⟩
`Communication:`	⟨protocols and exchange⟩
`Coordination:`	⟨contracts and enforcement⟩
`Cooperation:`	⟨flow of work⟩
`Restrictions:`	⟨general description⟩
`Actor restrictions:`	⟨general description⟩
`Environment:`	⟨general description⟩
`Based On:`	⟨*life cases*⟩
`Based On:`	⟨*intentions*⟩
`Based On:`	⟨*general tasks, audience, mission, goal*⟩

5 Conclusion

In this paper we analysed the extraction of user models as a contribution to supporting pragmatics of storyboarding, which closes a gap in our development methodology. We described facets of user profiles capturing education,work and personality features. For this we introduced a semi-formal specification template. Furthermore, we discussed actor portfolios as a way to capture responsibilities and targets. This is closely related to the specification of tasks. We demonstrated how profiles can be used in the development of storyboards.

Though user modelling has been intensively studied in Artificial Intelligence our work addresses user models from a novel angle. Our interest is the refinement of life cases in order to set up an adequate storyboard, not the inference of online system adaptation on the basis of observed user interaction. In this sense our work is intrinsically connected to the work in [14], a parallel submission on life cases. Both user models and life cases define the core of storyboarding pragmatics. In addition, a deeper understanding of contexts is needed, which will be addressed next and published in due time.

References

1. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., and Matera, M. *Designing Data-Intensive Web Applications.* Morgan Kaufmann, San Francisco, 2003.
2. Conallen, J. *Building Web Applications with UML.* Addison-Wesley, Boston, 2003.
3. De Troyer, O., and Leune, C. WSDM: A user-centered design method for web sites. In *Computer Networks and ISDN Systems – Proceedings of the 7th International WWW Conference.* Elsevier, 1998, pp. 85–94.
4. Güell, N., Schwabe, D., and Vilain, P. Modeling interactions and navigation in web applications. In *Conceptual Modeling for E-Business and the Web*, S. W. Liddle, H. C. Mayr, and B. Thalheim, Eds., vol. 1921 of *LNCS.* Springer-Verlag, 2000, pp. 115–127.
5. Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., and von Wilamowitz-Möllendorff, M. Gumo - the general user model ontology. In *User Modeling* (2005), vol. 3538 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 428–432.
6. Houben, G.-J., Barna, P., Frasincar, F., and Vdovjak, R. HERA: Development of semantic web information systems. In *Third International Conference on Web Engineering – ICWE 2003* (2003), vol. 2722 of *LNCS*, Springer-Verlag, pp. 529–538.
7. Kobsa, A. User modeling and user-adapted interaction. *User Modeling and User-Adapted Interaction 14*, 5 (2004), 469–475.
8. Kobsa, A. User modeling and user-adapted interaction. *User Modeling and User-Adapted Interaction 15*, 1-2 (2005), 185–190.
9. Lowe, D., Henderson-Sellers, B., and Gu, A. Web extensions to UML: Using the MVC triad. In *Conceptual Modeling – ER 2002*, S. Spaccapietra, S. T. March, and Y. Kambayashi, Eds., vol. 2503 of *LNCS.* Springer-Verlag, 2002, pp. 105–119.
10. Magnini, B., and Strapparava, C. User modelling for news web sites with word sense based techniques. *User Modeling and User-Adapted Interaction 14*, 2-3 (2004), 239–257.
11. Moritz, T., Schewe, K.-D., and Thalheim, B. Strategic modelling of web information systems. *International Journal on Web Information Systems 1*, 4 (2005), 77–94.
12. Mylopoulos, J., Fuxman, A., and Giorgini, P. From entities and relationships to social actors and dependencies. In *Conceptual Modeling - ER 2000* (Berlin, 2000), Springer-Verlag, pp. 27–36.
13. Rossi, G., Schwabe, D., and Lyardet, F. Web application models are more than conceptual models. In *Advances in Conceptual Modeling*, P. Chen et al., Eds., vol. 1727 of *LNCS.* Springer-Verlag, Berlin, 1999, pp. 239–252.
14. Schewe, K.-D., and Thalheim, B. Life cases: An approach to address pragmatics in the design of web information systems. submitted for publication.
15. Schewe, K.-D., and Thalheim, B. Conceptual modelling of web information systems. *Data and Knowledge Engineering 54*, 2 (2005), 147–188.
16. Schwabe, D., and Rossi, G. An object oriented approach to web-based application design. *TAPOS 4*, 4 (1998), 207–225.
17. Thalheim, B., and Düsterhöft, A. The use of metaphorical structures for internet sites. *Data & Knowledge Engineering 35* (2000), 161–180.

XUPClient - A Thin Client for Rich Internet Applications

Jin Yu[1], Boualem Benatallah[1], Fabio Casati[2], and Regis Saint-Paul[1]

[1] School of Computer Science and Engineering
University of New South Wales
Sydney, NSW 2052, Australia
{jyu,boualem,regiss}@cse.unsw.edu.au
[2] HP Laboratories
1501 Page Mill Road, MS 1142
Palo Alto, CA 94304, USA
fabio.casati@hp.com

Abstract. With the help of rich web client technologies, developers are creating rich internet applications in response to end users' growing demand in richer web experiences. However, most of these technologies are fat client based. That is, to enable rich user interfaces, application code, whether binary or script, must be downloaded and executed on the client side. In this paper, we propose a thin client based approach - XUPClient, a rich web client aimed at closing the gap between web and desktop user interfaces. It allows end users to interact with rich UI components only found in desktop environment, while remaining thin in terms of application logic; i.e. all application code resides on the server side, and the client only renders declarative UI definitions. XUPClient is built on top of the Extensive User Interface Protocol (XUP), a SOAP-based protocol for communicating events and incremental user interface updates on the web.

1 Introduction

Traditionally, the user interfaces of web applications are built with HTML with the aid of some JavaScript performing data validation and action confirmation. We call this approach the classic HTML approach. However, this approach lacks usability and rich user interactivity, as HTML was primary designed for representing textual information with embedded images, not highly interactive user interfaces. As end users are increasingly demanding their applications to be much more interactive, web developers are facing the challenge of creating *rich internet applications*, which are web applications with dynamic and highly responsive user interfaces.

To provide a highly usable and productive web application experience, web user interfaces need to be brought closer to desktop user interfaces, with the aid of *rich web clients*. Specifically, for end users, we believe rich web clients should satisfy the following key requirements:

- To enable high usability and productivity, users should be able to interact with a set of rich UI components similar to those found in desktop applications.
- Updates to the user interfaces should be applied incrementally, not one page at a time. Therefore, users will no longer perceive slow and annoying page refreshes.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 524 – 535, 2006.

- User interfaces should be very responsive as end users expect fast response time from their applications. This implies the use of asynchronous mechanism to perform UI operations while computing in the background.
- The client-side environment should only execute safe UI code, without compromising the security of the end user's computer.
- Maintaining the web's deployment advantage, business applications should not be installed on the client side; they should be centrally managed on the server side.

There are quite a few existing rich web client technologies, such as Java Applet, ActiveX, Flash, AJAX (Asynchronous JavaScript + XML) [1], that satisfy some of the above requirements. However, they all share one thing in common: application specific code is downloaded and executed on the client side to create the rich user interface. The application code could be in either binary (Applet, ActiveX) and script (Flash, AJAX), and is downloaded and executed for *each web application* the end user interacts with.

It is well-known that the execution of downloaded code by the client may impose security risks to the end user's computer. Although both .NET and Java have sophisticated security sandbox to restrict the execution of downloaded code, security remains to be a major concern. As a result, many end users choose not to install downloaded code when prompted, even though they are certified by public CAs (certification authorities). In addition, due to the same security concerns, many organizations do not allow the installation of downloaded code, with a few exceptions for well-known applications such as Acrobat PDF Reader. Finally, executing downloaded scripts (as in AJAX) has similar security problems; for example, cross site scripting and script injection are two common techniques to compromise browser's security.

We use the term *fat client* to describe client software that executes application specific code. On the other hand, if a client only renders UI definitions, we call it a *thin client*. All the above rich web clients are fat clients, since they all download and execute application code. In the classic HTML approach, we consider web browser to be a thin client. However, in the AJAX approach, the browser is no longer thin since a large amount of application code (i.e. JavaScript) is downloaded and executed.

To satisfy the above requirements and overcome limitations in the fat client based approaches, we propose an alternative rich web client, XUPClient, which allows users to interact with rich UI components without downloading and executing application-specific code. Being a thin client, XUPClient has the following advantages:

- There are no client-side security issues since the client does not execute downloaded code.
- It provides a better security model for sensitive business data, since only UI definitions, not application data, are transported over the network and/or manipulated on the client side. This is especially important for applications with high security requirements, such as government and financial applications.
- It facilitates centralized application management. As a result, there is no client-side administration cost.

In addition, XUPClient is based on the Extensible User Interface Protocol (XUP) [4,15], a SOAP-based protocol for communicating events and incremental user interface updates on the web. As a result, XUPClient updates the user interfaces

incrementally, not one page at a time, and users will no longer perceive slow and annoying page refreshes. Further, XUPClient employs an asynchronous communication mechanism to avoid blocking user interactions, thereby giving end users a more pleasant and responsive user interface.

XUPClient is part of our overall OpenXUP rich internet application framework. In this paper, we focus on the client side of the framework. More information about OpenXUP can be found in [5]. In the next section we describe the design of XUPClient. Section 3 illustrates its runtime behavior using an example. We then briefly outline implementation issues and the application development process in Section 4 and 5. Finally, we discuss related work and conclusions in Section 6 and 7.

2 Thin Client Design

2.1 Overview

Our primary design goal is to provide a rich web user interface to end users while letting developers keep their application logic on the server side. To avoid executing application code, XUPClient interprets XML-based declarative UI markups and then let end users interact with the resulting user interface model (UI model).

A UI model is a representation of the user interface which the end user perceives and interacts with. Given an XML markup fragment, XUPClient builds a UI model by mapping XML elements to UI components, such as buttons and panels, and by mapping XML attributes to UI properties, such as color and size. Based on this abstraction, we can provide support to any XML-based declarative languages, such as SUL [9], XUL [14], XAML [12], XHTML, and WML. However, it is more advantageous to use UI language such as SUL and XUL which focus on fluid, desktop-style user interfaces. These languages provide a rich set of UI components suitable for building highly interactive applications; whereas XHTML and WML are more appropriate for page-based hypertext applications.

In addition to UI components and their properties, a UI model also includes events, which are fired when the end user interacts with the UI components. Some UI events are common to most UI components, whereas certain events are only fired by specific UI components. For examples, events like "mouse click" and "key press" are available to most components, while the "window closed" event is specific to the UI component "window". A rich UI language such as SUL or XUL includes a large set of UI events, which allow for rich user interactions; whereas a page-based language such as XHTML only provides a limited number of UI events.

Having a set of rich UI components and events alone is insufficient to provide a highly interactive user interface. What is needed is the UI behavior, which defines how to handle user actions and update the UI model. In fat client based approaches, such as AJAX and Java Applet, the UI behavior is provided by the downloaded application code which handles UI events and manipulates the UI model. However, a thin client does not download or execute application code, so some other mechanism is needed to provide the UI behavior.

XUPClient leverages the Extensive User Interface Protocol (XUP) to provide the necessary UI behavior. XUP is a SOAP-based protocol that delivers UI events and

incremental UI updates. Since it delivers UI events from the client to the server, application code on the server side now has the opportunity to handle the events (i.e. user actions) and behave appropriately. In addition, since XUP delivers incremental UI updates from the server to the client, XUPClient can render the changes to its UI view without executing application code.

Table 1. Protocol elements for incremental UI updates

<xup:addUIElement>	Add a UI component under a specified parent component
<xup:updateUIElement>	Update the content of the specified UI component
<xup:removeUIElement>	Remove the specified UI component
<xup:updateUIAttr>	Update the specified UI property of the component

As shown in Table 1, XUP offers fine-grained updates to the UI model. For example, <xup:addUIElement> may be used to add a list item to a list box, and <xup:updateUIAttr> may be used to update the background color of a button. Therefore, with the protocol level support of the delivery of UI events and incremental UI updates, XUPClient can provide rich and dynamic UI behaviors without downloading or executing application code.

2.2 Architecture

XUPClient's architecture is depicted in Figure 1. We take a layered approach to ensure its modularity and extensibility.

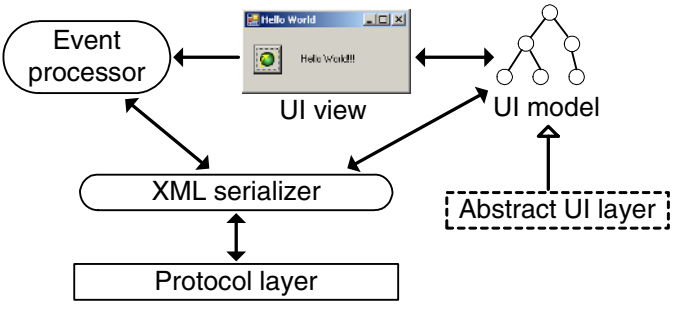


Fig. 1. XUPClient architecture

First, XUPClient makes a clear separation between the model and view of the user interface. The model is data representation of the user interface in an XML-based UI language such as SUL or XUL; whereas the view corresponds to the rendering of model using the underlying native user interfaces toolkit offered by the platform. The decoupling of the model and view of a user interface allows XUPClient to effectively support XML-based UI languages.

In addition, XUPClient separates the UI model into two layers: the abstract UI layer and the concrete UI model. The abstract UI layer models UI components as XML

elements and UI properties as XML attributes; it interfaces with the XML serializer to generate XML from UI components and to create UI components from XML. The concrete UI model contains a tree of UI components from a rich UI language such as SUL or XUL. For example, the root of the UI tree is a window, with two panels as its children; and one of the panels in turn contains two children, a button and a label. The abstract UI layer allows XUPClient to support multiple UI languages.

The event processor is responsible for mapping native UI events to XML-based UI events defined in the appropriate UI languages. It listens to native UI events from the UI view and sends the mapped XML UI events to the server via the XML serializer.

Finally, the XML serializer is responsible for the parsing and serialization of the UI components and events into XML. It communicates with the protocol layer to deliver UI events to and receive UI updates from the server.

At runtime, when an end user manipulates the UI view (e.g. click on a button, input some text), a native UI event is fired from the view. The event processor captures that event and translates it to an XML UI event in the UI language used by the current UI model. It then passes that to the XML serializer which delivers the UI event to the server through the protocol layer.

Similarly, when the web application on the server side modifies the user interface, the server will send the corresponding UI updates to XUPClient, which will in turn update its UI model appropriately. After that XUPClient pushes the updates to the UI view by rendering the user interface changes.

2.2.1 UI State Management

Unlike web browsers, XUPClient maintains the UI state (both model and view) of the web application it interacts with. This is the key to displaying incremental UI updates, since without UI state, a thin client would have to redisplay the entire screen / page of the application for each UI update. In addition, since the client now maintains the UI state, the server-side application code no longer needs to regenerate screen / page every time.

Through the protocol elements in Table 1, XUPClient allows application code on the server side to manipulate its client-side UI state. In this case, UI updates are marshaled through XUP protocol responses, and each response may contain multiple additions, removals, and updates of UI components and properties. For example, UI updates in a response may instruct XUPClient to add a button to a panel and change the text of the window title.

To allow application code on the server side to process user inputs, XUPClient also sends UI updates to the server. The UI updates represent the changes to the UI components and properties caused by user actions; for example, text entered in a text box and a check box been unchecked. Unlike the UI updates originated from the server side, the UI updates caused by end user actions typically include updates but not additions or removals of UI components and properties. Note that XUPClient does not send UI updates immediately; they are queued and sent to the server within an event request. This accommodates for data entry scenarios where the user enters text in a number of fields and then clicks a button to submit the information when done.

In summary, changes to the UI model are communicated bi-directionally. The UI values sent by the client (in event requests) are originated from end user actions;

whereas the UI updates received by the client (in server responses) are originated from application code.

2.3 Event Processing

It is very important for a thin client to minimize network traffic caused by UI events, while at the same time provide a rich and interactive user interface. First, the thin client must not send every UI event to the server; it should filter out high frequency native events such as mouse movements. Second, it should only send *events of interest*. That is, it should only deliver events that the server-side application wants to receive. For example, a mouse click on a menu bar will display a submenu. This behavior can be handled locally at the client side, so the application has no interest in receiving this mouse click event. Another example, typing in a text field generates a key press event for each key stroke. But the application may not care about each key stoke; it only wants to receive the entire text when the user clicks on a button labeled "submit". In this case, the application is interested in the mouse click event on the "submit" button, but not interested in the key press events in the text field.

To support this *event selection* concept, we introduce the notion of *event selector*, which allows an application to selectively receive events of interest. Event selectors are defined by applications and sent to the client in XUP responses. Specifically, an event selector defines the type of event and the UI component that fires the event; e.g. mouse click on a button. On the client side, whenever the event processor captures a native UI event, it will first identify the event type and the specific UI component that fired the event, and then search for a matching event selector. If a match is found, the event will be sent to the server; otherwise, the event will be discarded.

To further reduce unwanted event traffic, a thin client should be able to filter events based on their content. For example, an application may want to receive mouse click event on an "OK" button, but it only cares when the left mouse button is used, not the middle or right mouse buttons. Another example, the application may be only interested in specific key press events, such as delete or enter keys, from a tree component; so other unwanted key press events should be filtered out by the client. Therefore, to support the *event filtering* concept, we introduce the notion of *event mask*, which may be attached to an event selector to define fine-grained event filtering criteria specific to a particular event type.

Listing 1. Event selector example

```
<xup:selector id="s1" event="sul:key-down" element="tree1" async="false"
  xmlns:xup="http://www.openxup.org/2004/02/xup"
  xmlns:sul="http://www.openxup.org/2004/02/sul">
  <xup:mask sul:keys="del enter"/>
</xup:selector>
```

Listing 1 shows an example event selector. Here the application is interested in the "sul:key-down" event from a tree component with the ID "tree1". To ensure extensibility, event types are XML namespace qualified, so "sul:key-down" means the "key-down" event from the SUL language. In addition, the event selector also contains an event mask, which specifies that only "del" and "enter" key events should be sent to

the application on the server side. The "sul:keys" attribute in the event mask is qualified by the SUL namespace because the keys are defined in the UI language (SUL), not in the protocol (XUP).

The combination of event selection and filtering mechanisms allows for dynamic and versatile UI behavior. Applications can decide what, where, and when to receive UI events. For example, a user tries to fill in an account creation form, with text fields such as username, password, credit card number, and a "create" button. The application may simply add a single event selector to select the mouse click event on the "create" button. In this case, no events will be generated while the user is entering text; the values of the text fields will sent together with the mouse click event on the "create" button. Alternatively, the application may wish to add two additional event selectors, for selecting the "tab" key event on the username and credit card number text fields. This allows the application to immediately check duplications of usernames and validate the credit card number when the user tabs away from the respective text fields. Essentially, with the two additional event selectors, the application has enabled a more fluid user interface by providing immediate feedback to the end user.

Finally, to make the user interface more responsive, XUPClient supports *asynchronous* event delivery using event selectors. As shown in Listing 1, the "async" attribute indicates whether the event should be delivered synchronously (if false) or asynchronously (if true).

2.4 Comparing Thin Client and Fat Client

In this section we discuss some of the differences between thin client and fat client from architecture's point of view.

2.4.1 Code Location

In general, the user interface of an application can be separated into *definition* and *behavior*. By UI definition, we mean the description of the UI components and their properties. The UI definition of an application can be programmed via an imperative language such as Java or C#, or declared in an XML-based language such as XUL or XHTML. By UI behavior, we mean how the application changes its user interface in responses to user actions. Since UI behavior is dynamic in nature, it is usually programmed, not declared.

In fat client based approaches, UI behavior is always executed at the client side via downloaded application code. For instance, with Java Applet, both UI behavior and UI definition are programmed in Java and executed on the client side; and for AJAX, the UI definition is provided in HTML, but the UI behavior is programmed in JavaScript and executed on the client side. On the other hand, in our thin client based approach, the client only renders XML-based rich UI definition, leaving the execution of the UI behavior to the application code on the server side.

In addition, in fat client based approaches, the client must understand some business logic, in order to render business data and communicate with the server side business process. On the contrary, our thin client only knows about UI, not business logic or data.

In summary, in our thin client approach, the declarative portion of the application (i.e. UI definition) is processed by the client, whereas the programmed portion of the application (i.e. UI behavior and business logic) is executed on the server side. This separation allows XUPClient to provide a rich UI experience to the end user while at the same time ensuring a secure desktop environment.

2.4.2 Leveraging Desktop Computing Power

As desktop computers are very powerful nowadays, it is desirable to leverage that computing power; e.g. desktop machines can provide distributed computing resources in a scientific computing grid. Obviously, fat clients are more suitable for this type of data crunching applications, since application code can be executed on the client side.

On the other hand, for management and security reasons, most business applications keep data processing on the server side, and the desktop is only used for inputting data and displaying results. Hence XUPClient is ideal for these applications. Furthermore, although XUPClient does not take advantage of the desktop's data processing capability, it can fully leverage the desktop's graphical computing power to render rich UI and graphics.

2.4.3 Network Bandwidth Considerations

The event selection and filtering mechanisms introduced earlier eliminate most of unwanted event traffic. In addition, since XUPClient maintains UI state, many user actions are handled locally by the UI components themselves. For example, a tree component remembers its own content - the hierarchy of tree nodes. Therefore, once the tree is populated, expanding a tree node is entirely a local operation. Furthermore, simple data validations can be performed locally on the client side. For instance, a text field component may support declarative input patterns such as date format, number range, text length, and even regular expressions; all these can be supported by the UI component locally, without involving roundtrips to the server.

Finally, although a fat client does not send UI events to the server, it sends and receives application data to and from the server. The application data are usually serialized application objects, in XML or binary form. Therefore, the amount of traffic generated by a fat client is no less than the amount of event traffic from XUPClient.

3 Running XUPClient

This section shows the runtime behavior of XUPClient. To simplify the illustration, we use the classic "Hello World" example to demonstrate the user interactions. However, the same interaction patterns apply to rich internet applications in the real world.

3.1 Accessing Web Applications

The first step is to access the web application. XUPClient uses XUP URLs to identify and access web applications. XUP URLs are very similar to HTTP URLs, except that the URL scheme is either "xup" or "xups" (for SSL connection); for example,

"xup://www.example.org/xupws/XUPSrv.asmx?app=HelloWorld". Note that the part of URL after the hostname depends on how and where the application is deployed.

The URL can be directly entered into XUPClient; alternatively, since XUPClient supports web browser integration, the user may either type the XUP URL in the browser's address bar or click a hyperlink containing the URL. Finally, since an XUP URL can be stored in a shortcut file, the user can just double click on the file to launch the corresponding web application.

3.2 User Interactions

XUPClient sends a startup request to the server after the user types in the URL corresponding to the "Hello World" application. The application processes the request and then the server returns the following response:

Listing 2. Initial response

```
<soap:Envelope xmlns:xup="http://www.openxup.org/2004/02/xup" ...>
  <soap:Header>
    <xup:session application="HelloWorld" sid="X35759759"/>
  </soap:Header>
  <soap:Body>
    <xup:startupResponse xmlns:sul="http://www.openxup.org/2004/02/sul">
      <xup:addResources>
        <sul:image id="globe">R0lGODlhGAAYAMQAAP......</sul:image>
      </xup:addResources>

      <xup:addUIElement position="-1">
        <sul:window id="w-main" title="Hello World">
          <sul:button id="b1" position="..." image="globe"/>
          <sul:label id="l1" position="...">. . . . . .</sul:label>
        </sul:window>
      </xup:addUIElement>

      <xup:addEventSelectors>
        <xup:selector id="sel1" element="b1" event="sul:action"/>
      </xup:addEventSelectors>
    </xup:startupResponse>
  </soap:Body>
</soap:Envelope>
```

In the above SOAP header, the server returns a session ID to be used for identifying the client in subsequent requests. The <xup:addResources> element in the body specifies an image (the globe icon) to be made available to the client. The <xup:addUIElement> element adds a tree of UI components to the client's UI model. Here, the root of the tree is the main window of the application, which contains a button and a label. Finally, the <xup:addEventSelectors> element includes an event selector ("sel1") which selects the action (i.e. mouse click) event from the button.

XUPClient processes the initial response and displays the corresponding UI view. As shown in Figure 2a, the window contains a button with the globe icon and a label with "……" as its text.

Fig. 2a. Initial window

Fig. 2b. After button click

Now the end user clicks on the globe button, which causes XUPClient to send the following event request:

Listing 3. Event request for button click

```
<soap:Envelope xmlns:xup="http://www.openxup.org/2004/02/xup" ...>
  <soap:Header>
    <xup:session application="HelloWorld" sid="X35759759" .../>
  </soap:Header>
  <soap:Body>
    <xup:event xmlns:sul="http://www.openxup.org/2004/02/sul"
      type="sul:action" selector="sel1"/>
  </soap:Body>
</soap:Envelope>
```

Here, the request contains the SUL "action" event, which corresponds to the mouse click action. For a button, the action event fires when the button is activated (by mouse click, keyboard shortcut, etc.). Note that this event matches the event selector "sel1", which instructs XUPClient to listen to action events from the button.

After the application processes the above event, the server returns the following XUP response:

Listing 4. Event response for button click

```
<soap:Envelope xmlns:xup="http://www.openxup.org/2004/02/xup" ...>
  <soap:Body>
    <xup:eventResponse>
      <xup:updateUIElement element="l1">
        Hello World!!!
      </xup:updateUIElement>
    </xup:eventResponse>
  </soap:Body>
</soap:Envelope>
```

Here, the event response contains an <xup:updateUIElement> element, which specifies the new text for the label "l1". As a result, XUPClient will display the new label text "Hello World!!!", as shown in Figure 2b.

4 Implementation

XUPClient is implemented in .NET; it can be deployed as a standalone Windows program or as a browser plug-in (i.e. .NET Smart Client). Since the client is a thin

client, it needs to be deployed only once, not on a per-application basis. In addition, installation should be fast as the client has a small footprint, currently about 400K.

On the server side, we provide a toolkit which runs inside the ASP.NET application server. The server toolkit takes care of protocol issues and offers a set of event-driven .NET APIs for developing web applications. Both XUPClient and the server toolkit are part of our overall OpenXUP rich internet application framework.

5 Application Development

Our server toolkit offers a set of event driven APIs for developing rich internet applications. The APIs contain classes that model UI components, properties, and events. Both UI definition and behavior can be programmed using the APIs. However, the preferred way to create UI definitions is through XML-based templates, which simply contain markups in the appropriate UI languages (e.g. SUL). The use of XML-based UI templates greatly simplifies the user interface creation process.

More information about application development in the OpenXUP framework can be found in [5].

6 Related Work

Most rich web client technologies we know are based on the fat client approach. This also includes Flash-based frameworks such as Flex [6] and OpenLaszlo [7], both of which require the download and execution of application code on the client side.

XForms [13] improves upon traditional HTML Forms by providing better client-side events and data validations. However, XForms inherits HTML's page-based model: the response of a form submission either replaces the current form or the entire containing page. In addition, it only offers a limited set of user interface controls.

RemoteJFC [3] is a user interface toolkit that offers a remote version of the JFC (Java Swing) API. Application code on the server side uses this API to control the UI of the client viewer running on the end user's desktop computer. The client viewer does not contain any application code and is therefore thin. However, in order for the application code on the server side to update the client-side UI, the client viewer must run as a RMI server. This architecture has obvious security problems.

There are many mature remote display technologies, such as X11 [11], NeWS [2], VNC [10], RDP [8], etc. These are low level UI protocols, transporting graphics and low-level controls. As a result they are usually very verbose and only work well in a local area network environment where network resource is plenty. In addition, with protocols such as X11 and NeWS, the client and server locations are reversed. For example, in X11, the end user' desktop must open a server port and listen to incoming X11 traffic. This has obvious security implications; and as a result most corporate firewalls do not allow incoming X11 traffic.

7 Conclusion

Our thin client based design enables a rich web experience for end users while at the same time ensures a safe desktop environment. In addition, with XUPClient, end users will no longer experience frequent "page refreshes"; instead, they shall perceive much faster response time since UI updates are delivered to them incrementally, rather than one full page at a time. Finally, XUPClient is easy to deploy; it can be seamlessly integrated with any web browsers, or installed as a standalone program.

References

1. Garrett, J. J., Ajax: A New Approach to Web Applications. Feb. 2005. <http://www.adaptivepath.com/publications/essays/archives/000385.php>
2. Gosling, J., Rosenthal, D.S.H., Arden, M.J. *NeWS Book: An Introduction to the Network / Extensible Window System.* Springer Verlag, March 1990.
3. Lok, S., Feiner, S.K., Chiong, W.M., and Hirsch, Y.J. A Graphical User Interface Toolkit Approach to Thin-Client Computing. In *Proceedings of WWW 2002* (Honolulu, Hawaii, May 2002).
4. Yu, J. and Benatallah, B. XUP - a SOAP-based Simple User Interface Protocol. In *Proceedings of AusWeb 2005* (Gold Coast, Australia, July 2005).
5. Yu, J., Benatallah, B., Casati, F., and Saint-Paul, R. OpenXUP - an Alternative Approach to Developing Highly Interactive Web Applications. In *Proceedings of ICWE 2006* (Menlo Park, California, July 2006).
6. Macromedia Flex <http://macromedia.com/software/flex/>
7. OpenLaszlo <http://openlaszlo.org>
8. Remote Desktop Protocol (RDP) <http://msdn.microsoft.com/library/enus/termserv/termserv/remote_desktop_protocol.asp>
9. SUL <http://openxup.org/TR/sul.pdf>
10. Virtual Network Computing (VNC). <http://www.realvnc.com/>
11. X Window System. <http://www.x.org/X11.html>
12. XAML <http://www.xaml.net>
13. XForms 1.0 <http://www.w3.org/TR/xforms/>
14. XUL. <http://www.mozilla.org/projects/xul>
15. XUP - Extensible User Interface Protocol. <http://www.openxup.org/TR/xup/>, <http://www.w3.org/TR/xup>

2D/3D Web Visualization on Mobile Devices

Yi Wang[1,2,3], Li-Zhu Zhou[1,2], Jian-Hua Feng[1], Lei Xie[3], and Chun Yuan[2]

[1] Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, China
[2] Institute of Information Technology, Graduate School at Shenzhen,
Tsinghua University, Shenzhen 518055, GuangDong Province, China
[3] School of Creative Media, City University of Hong Kong,
Tatchi Avenue, Kowloon, Hong Kong, China
wangy01@mails.tsinghua.edu.cn, dcszlz@tsinghua.edu.cn,
fengjh@tsinghua.edu.cn, xielei@cityu.edu.hk,
yuanc@sz.tsinghua.edu.cn

Abstract. Visualization is able to present the rich Web information intuitively and make the Web search/mining more productive. Mobile computing is able to provide the flexibility of working anytime and anywhere. Therefore, it is natural to combine the two techniques for intriguing applications. However, the technical limitations of mobile devices make it difficult to port well-designed visualization methods from desktop computers to mobile devices. In this paper, we present what we learned on engineering 3D Web visualization on both high-end and low-end mobile devices as the MWeb3D framework, which forms a distributed pipeline that move intensive computation from the mobile devices to server systems. Some important issues of this strategy include: (1) separating visualization from graphics rendering, (2) encoding visual presentation for transmitting via bandwidth-limited wireless connections, (3) user interaction on mobile devices, and (4) highly efficient graphics rendering on the mobile devices. We will show fruitful experiments on both PDA and mobile phone with photos taken from both simulator and real mobile devices.

1 Introduction

Within recent years, mobile computing techniques evolve significantly fast and mobile devices advance in our daily life to a wide variety of applications. Mobile computing provides the capability of solving problems anytime and anywhere. But technical limitations, e.g., small screen, and the mobile environment, e.g., use when walk or drive, challenge traditional ways of information presentation and user iteration.

Because visualization can help people understanding information intuitively and solving problems effectively, also, visualization techniques are efficient on presenting rich information in limited display area, it becomes a natural idea to adapt visualization techniques on mobile devices to deliver rich information via the small screens and to support the users grasping at a glance. For example, Chittaro and et al. [1] designed a comprehensive visual metaphor to

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 536–547, 2006.

show hotspots on a 2D map. With this technique, drivers can easily understand both the direction and the distance to hotspots around and even out of the screen.

In this paper, we discuss visualizing the Web searching/mining results on mobile devices. Compared with prior works of mobile visualization, visualizing the Web involves computation-intensive jobs such as searching and mining. It also requires more complicated visual metaphors to present rich and constantly updating Web information in small display. So, instead to design specific visual metaphors, this paper emphasizes more on a distributed computing framework that has the flexibility to tailer and incorporate those existed and well-designed 2D/3D visual metaphors. The framework has the performance potential to provide fluent interactions in the mobile environment.

Visualization of the Web on desktop systems have been studies for years. Some typical works include [2] [3] [4] [5] [6]. Unfortunately, technical limitations of the mobile devices make it very difficult to port existed Web visualization techniques from desktop systems to mobile environment. Some major limitations include:

- Small display with low resolution and a few colors. In contrast, most existed Web visualization techniques were designed for desktop systems with large screen and millions of colors.
- Limited computing capability, such as CPUs without floating-point units, memory bus with limited bandwidth, and the lack of graphics hardware.
- Slow wireless connections, such as Bluetooth, which can not afford the traditional way of Web visualization that retrieve large amount of Web information to the mobile device before analysis and visualization.

For some specific visualization tasks, such as the 2D map visualization [1], the above difficulties can be addressed by modifying or designing visual presentations especially for mobile devices. However, a more general solution, especially for 3D visualization, requires systematic strategies that are applicable to various visualization tasks on mobile devices.

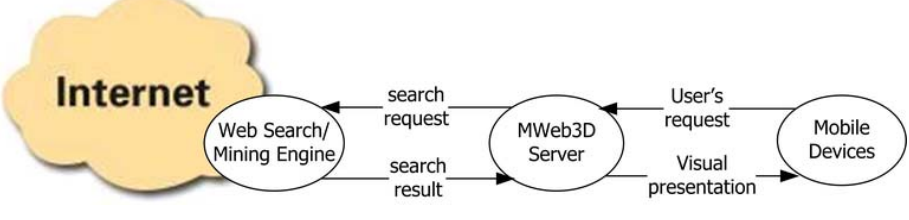


Fig. 1. The visualization pipeline of the MWeb3D framework

In this paper, we present our exploration of 2D/3D Web visualization engineering on both high-end and low-end mobile devices as a general solution, the *MWeb3D* framework, which features the following advantages:

- **Distributed visualization pipeline.** To release the mobile devices from the intensive computations, we separate traditional meaning of "visualization" into two parts: the generating 2D/3D *visual presentations* and the 3D graphics rendering. By moving the former part to MWeb3D server systems, we organize all computational components as a distributed pipeline, as shown in Figure 1. Only the last stage of the pipeline, graphics rendering and displaying, is performed on mobile devices.
- **Encoding and transmitting visual presentations.** Because the visual representation of the Web information generated by the MWeb3D server have to be encoded and transmitted to the mobile device for rendering and displaying, and because the limited wireless bandwidth, we discuss the designing of encoding schemas that can encode the visual presentations compactly.
- **User interaction on mobile devices.** Interactive visualization on mobile devices, especially the navigation in visual presentation, has to adapt to the small screen size and special input methods, such as the stylus of PDAs and numeric keypad of mobile phones. We discuss adapting interaction methods, such as the level-of-detail (LOD), to mobile devices and incorporating with the mobile input methods.
- **Graphics computing on mobile devices.** As graphics computing is well-known of its computational complexity, we discuss properties of the computational facilities on mobile devices and how to make full use of them to achieve efficient graphics rendering on mobile devices.

2 The Visualization Pipeline

Visualizing Web information includes several major computational tasks: (1) retrieving Web information via Web searching or Web mining; (2) analyzing the retrieved information and generate the visual representation of the information; (3) rendering the visual presentations into image; and (4) displaying the image onto screen.

Traditional information visualization approaches, designed for desktop systems with promising computational power, usually implement these tasks monolithically as a single piece of software running on a single computer. However, for mobile devices, any of these tasks may be over-challenging. Therefore, in the MWeb3D framework, we distribute these computational tasks to different computers, in particular, we move the former two tasks on powerful sever systems and leave the latter two on the mobile devices. The distributed tasks are organized as a pipeline, as shown in Figure 1, where, task (1) runs on the Web search/mining engine sever, such as Google or the VisMap server [6], task (2) runs on the MWeb3D server, and tasks (3) and (4) run on the mobile client.

2.1 Discussion on Separating Visualization from Rendering

It is arguable that not all visualization methods can have the rendering stage separated from the generating of visual presentation. However, because the Web

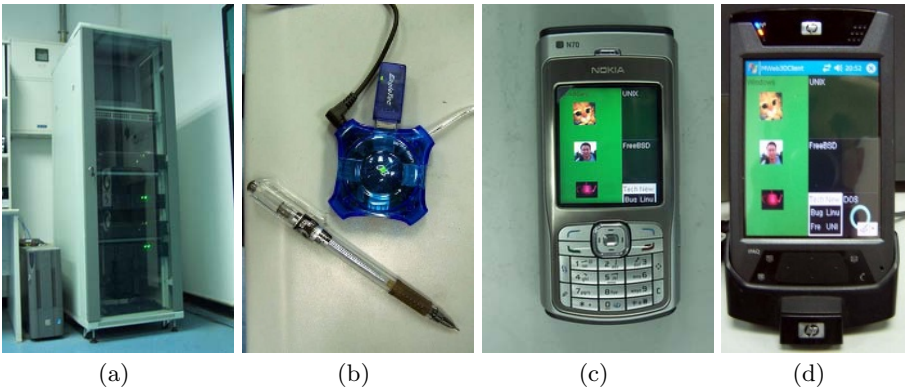
searching/mining is in fact a process that abstract knowledge domain [7] from Web information, visualization of Web searching/mining result is intrinsically a process that represents abstracted concepts or entities of the knowledge domain with graphics elements and organizes the layout of these elements to represent relationships among the concepts or entities, as the examples shown in Figure 3, 4 and 5. So, it is natural to encode the layout and transmit to mobile devices for rendering and displaying.

Another organization of the pipeline that even lightens the computational intensity of the mobile devices is to move also the task (3) from the mobile device to the MWeb3D server system. But this organization requires transmitting the rendering results as images or video to the mobile device, which is far from affordable by the limited bandwidth supported by the mobile devices, such as Bluetooth.

However, running task (2) as a separate stage in the pipeline is promising, because this step usually have to do layout with iterative optimization algorithms, such as the generation of treemap [8] [9], of cluster map [10] and of pathfinder networks [7], that are computationally intensive.

2.2 Serving Multiple Simultaneous Clients

Because the MWeb3D server usually have to generate visual presentations for multiple mobile clients simultaneously, to cope with the possibly large number of simultaneous clients, we implement the MWeb3D server as a Linux cluster (c.f. Figure2(a)) using OpenMOSIX, which enhances the Linux kernel with support to distribute multiple tasks over computers in a local area network. Once a mobile client connects to the MWeb3D server cluster, the OpenMOSIX is in



(a) (b) (c) (d)

Fig. 2. Experimental devices used to form the distributed rendering pipeline. (a) The MWeb3D server running on a cluster of 4 Linux-based computers. (b) The Bluetooth adapter that connects each computer in the MWeb3D server cluster with the mobile devices. (c) A Nokia N70 mobile phone that renders and displays a treemap visual presentation. (d) An HP iPAQ PDA that runs the same treemap visual presentation.

charge of allocating a certain computer to serve the client. In the MWeb3D framework, the basic serving unit for an MWeb3D server is called an *MWeb3D transaction*. A complete MWeb3D transaction is composed of the following steps:

1. the mobile client requests connection and passing the search request;
2. one computer in the MWeb3D cluster is allocated to serve the client and rely the search request to Web searching/mining engine;
3. after accepts retrieved Web information, the responsible MWeb3D computer generates and encodes the visual presentation,
4. after the visual presentation is transferred to the mobile device, the client program renders and displays the presentation.

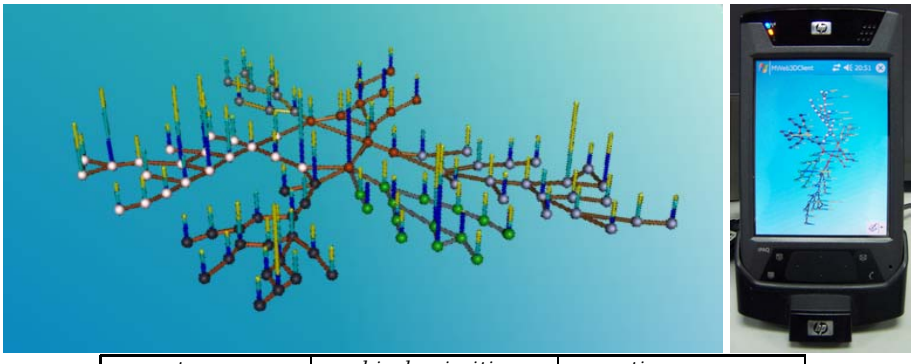
Because the MWeb3D transaction is a pure read operation, maintaining its consistency is easier than maintaining the transaction of database management systems. In our experimental system, when either side of the Web searching/mining engine, MWeb3D server and the mobile client times out, the whole transaction is aborted. This helps ensuring the high responsibility of the MWeb3D system.

We measure the computational intensity of a MWeb3D server computer in the cluster by the number of MWeb3D transactions that the computer is processing. By constraining the maximum number of simultaneous transactions that each computer allows and allocating the most disengaged computer to serve new transactions, we implement a simply and fast tasking balancing algorithm, which makes that the overall processing capacity of the MWeb3D cluster grows linear to the number of computers in the cluster.

3 Compact Encoding of Visualization Result

Because we distributed the task of generating visual presentation and the one of graphics rendering on the MWeb3D server and the mobile device, the generated visual presentation have to be encoded compactly and be passed through the limited wireless bandwidth to the mobile device for rendering. Because, generally, any visual presentation can be represented by a layout of graphics primitives, such as points, facets, and curves, we can design a general encoding schema that supports these primitives. Such a general schema will concise, because it only need to support a few types of graphics primitives. However, if we encode according to such schema, the encoding result may be lengthy. Take the 3D visual presentation shown in Figure 3 as an example:

This visual presentation is called *pathfinder network* [7], it describes the "virtual social relationship" between a group of users of a Web forum [6], where, each small sphere represents a user, the vertical cylinder on the sphere represents the number of posters that the user had submitted to the Web forum, and the edges that connects the spheres represent the intimate social relationships between pairs of users. These intimate relationships are mined by counting the frequency that two users post to the same topic.



concept	*graphical primitives*	*properties*
user	spheres	user-id, position, color
activity	cylinders on spheres	user-id, length
social relationships	edges between spheres	user-id-1, user-id-2

Fig. 3. Illustration of the *pathfinder network* and corresponding visual schema

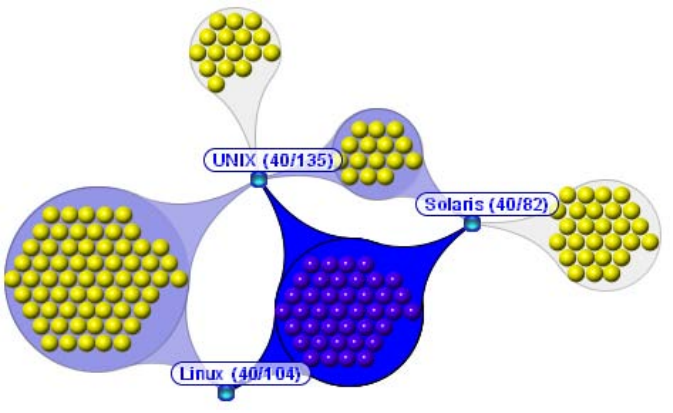
If we encode this visual presentation by the "general schema", we will have to describe the whole scene by a huge number of 3D triangular facets. Such encoding result is far from affordable by contemporary wireless connections such as Bluetooth and even Wi-Fi. If we define the encoding schema specifically for pathfinder networks, we can encode the visual presentation as the 3D position of the spheres (users), the length of vertical cylinders (the number of posters), and the connections between the users (the edges), as shown by the table in Figure 3. Although this new definition of encoding schema is specific to pathfinder networks, it brings in compact encoding result.

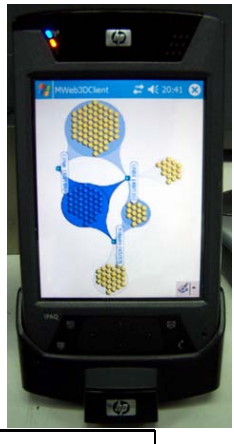
Specificity is one of the many approaches to incorporate semantics, or domain knowledge, into the encoding schema. When more domain knowledge is implicitly incorporated, the less information would have to be encoded. Moreover, incorporating semantics also makes customization easier. For example, the specific schema designed for the pathfinder network can allow mobile users to specify their favorite colors of the users, the vertical cylinders and the connections.

On the opposite side, as can be seen from the above example, the specificity also complicates the encoding schema. In the balancing of complex/concise encoding schema between compact/lengthy encoding result, we choose to design encoding schema for each kind of visual presentation. Beside the schema specific to the pathfinder network, we also defined and implemented visual encoding schema for the *treemap* [8] [9] and the *cluster map* [10] as shown in Figure 4 and Figure 5.

In Figure 4, the cluster map visualizes three discussion topics in a Web forum: UNIX, Solaris and Linux. Users are grouped as their participations to these topics. Each group are represented by a drop-shape, which contains a set of

spheres that correspond to the users. In Figure 5, the treemap visualizes the hierarchy of discussion categories. Each category is represented by a rectangle, whose area is proportional to the popularity. Users that often post in certain category have their photos shown as icons in corresponding rectangle. The photos that show the visual presentation rendered and displayed on HP iPAQ PDA are shown aside to the illustrations, and the definition of visual encoding schema for these visual presentations are listed below the illustrations.





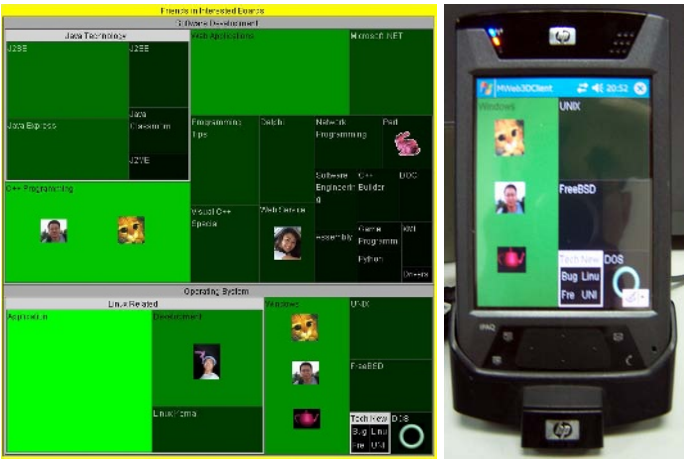
concept	*graphical primitives*	*properties*
topic	crystal spheres with text label	topic-id, text-label
user-group	drop-shapes	group-id, color, topic-id-list
user	spheres in drop-shapes	user-id, group-id

Fig. 4. Illustration of the *cluster map* and corresponding visual schema

4 Interactive Visualization on Mobile Devices

Rendering and displaying the encoded visual presentation on mobile devices must adapt to the small screen and special input methods of the mobile devices, such as stylus of PDA and numeric keypad of mobile phone.

Often, the small screen cannot hold all the details of the visual presentation, some previous works use interaction techniques to hide information that is currently out of users' focus. In [1], the authors draw maps with hot-spot on the PDA. For each hot-spot that is out of the screen area, a red aura is drawn around the hot-spot and the boundary of the aura is extended to the current display. So the users can approximate the relative position of the hot-spot. In [11], the authors use fish-eye technique to draw a calendar consisting of a grid of daily arrangements. The currently focused day will occupy the most area of the screen while other days will shrink to small spaces.



concept	*graphical primitives*	*properties*
category	rectangular area	category-id, color, text-label, parent-category-id
user	icons with user's photo or logo	user-id, category-id, icon-image

Fig. 5. Illustration of *treemap* and corresponding visual schema

In our work, we propose to use the level-of-detail (LOD) technique to adapt the tree map and cluster map to mobile devices. For the tree map (c.f. back to Section 3 and Figure 5), which is designed to visualize a hierarchy of categories, we merge too small rectangles in deep layers into a single larger rectangle. Figure 6 compares a tree map being visualized in a window of a desktop system and in a window of PDA. The screen area of PDA is about 1/4 of the window area, so if the user wants to have a overall view on the treemap, many text labels and icons are hidden on the PDA screen. However, we can notice that the icons on the PDA screen have the same size as shown on the desktop system. When the user want to focus on a certain category that is represented by a rectangle, she/he can simply tap the rectangle with the stylus, and the tapped rectangle will be enlarged to occupy the whole screen. If the user want a step back to upper level, only a double-tap is required.

Figure 7 shows different level of detail of the same cluster map (c.f. back to Section 3 and Figure 4) displayed on an iPAQ PDA and a Nokia N70 mobile phone. Because the screen size of Nokia N70 is about 1/5 of the iPAQ, details of the clustermap, including the spheres in the user groups are neglected on the mobile phone when the user is looking at an overview of the visual presentation. By pressing the keys 2, 6, 8 and 4 on the numeric keypad, users can switch among the topics (blue crystal spheres); by pressing 5, the user can enlarge the visual presentation to have a closer look to the selected topic and the user groups related with it. By pressing 0, the visual presentation is shrinked one level.

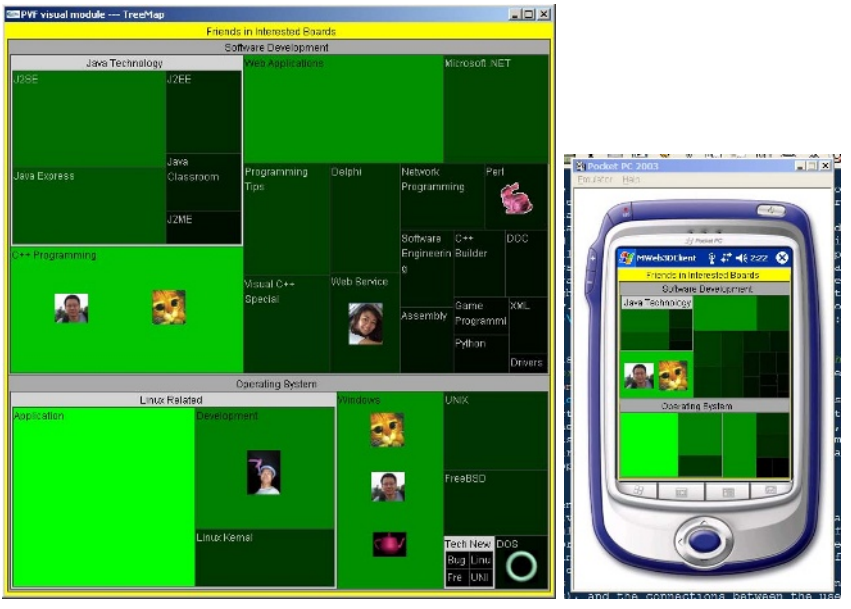


Fig. 6. Apply level-of-detail technique to rendering tree map

Fig. 7. Apply level-of-detail technique on rendering cluster map

5 Graphics Computing on Mobile Devices

Graphics computing is one of the most challenging tasks, even for desktop systems and workstations. On most current mobile devices, there is no graphics acceleration hardware. Only a few top-end mobile devices have equipped with fixed-function graphics processors. This means that most graphics operations have to be executed by the CPU, or as usually said in the graphics society, "by software". Unfortunately, most current mobile CPUs do not support floating-point computation, which, however, is heavily relied by graphics rendering. So developers have to use fixed-point techniques to do real number computation. What is potential to make use of is that, many mobile CPUs provide integer Single Instruction Multiple Data-stream (SIMD) instructions. For example, all Intel mobile CPUs with Xscale technique support integer MMX instructions that are originally designed for the Intel Pentium Pro desktop systems. These SIMD instructions can perform the same type of computation on multiple operands simultaneously. Although all operands are required to be integers, it is usually possible to use a sequence of integer SIMD instruction to perform real computation on several operands in a single instruction time. With the pseudo-floating-point SIMD technique, we successfully boosted the 3D graphics rendering operations of lighting and pixel interpolation.

For specific mobile platforms, we can also make use of special features to accelerate graphics computation. Our 3D graphics development on the HP iPAQ PDA uses the Vincent library (http://ogl-es.sourceforge.net/), which is a software implementation of the OpenGL/ES standard, the OpenGL standard tailored for mobile devices. Because Vincent uses the Intel Graphics Performance Primitive (GPP) library, we can easily implement many graphics computing part of the program by the fixed-point SIMD technique.

To implement 3D graphics engine on mobile phones is more challenging because of their CPUs even rarely support SIMD instructions. On the Nokia N70 (with Symbian operating system), we developed a *voxel engine* [1], which is originally designed by game developers for low-end home computers in 1980's. Different to the polygon-rasterizer engine as defined by the OpenGL/ES, the voxel engine is not capable of rendering full 3D objects represented by polygons, but it can render a height-map, which is a 2D function defined on the space of $\langle x, y \rangle$ and returning a height value z:

$$z \leftarrow h(x, y)$$

Because, generally, many 3D visual presentation are designed to layout most information on a 2D space to make the presentation easier to read by human users, other details are then added to the third dimension. For example, take a closer look at the 3D pathfinder networks (c.f. Figure 3), the spheres that represent users and the edges connecting these spheres are all defined on the $\langle x, y \rangle$ space. If we represent users by hemi-spheres instead of the spheres, as illustrated in Figure 8, we can simply model the visual presentation of pathfinder

[1] http://www.flipcode.com/articles/article_mobilegfx01.shtml

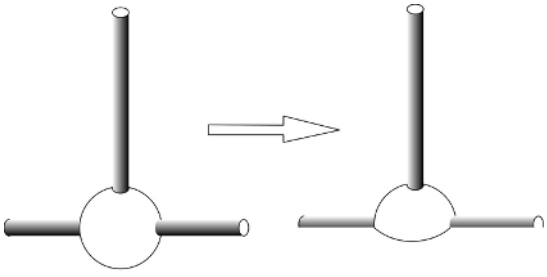


Fig. 8. Model the pathfinder network by a height map

network as a height-map. Similarly, the cluster map can also be represented by a height-map.

6 Conclusion and Discussion

In this paper, we present what we learned from engineering several Web visualization systems on mobile devices as the MWeb3D framework. Our discussion covered four major aspects: (1) to address the restricted computational power of mobile device, we separate the visualization into two steps of generating visual presentations and rendering them. By moving distributing the separately tasks onto server systems, we organized an efficient Web visualization pipeline, that can significantly lighten the computation intensity of mobile devices. (2) To address the slow network connectivity, we design compact encoding schema to compactly encode the generated visual representations before transmitting them to the mobile devices for rendering and displaying. (3) To adapt to the small screen size and specific input methods of mobile devices, we discuss general interaction methods, especially the level-of-detail techniques, to modify existed visual representations. (4) We also discussed how to make full use of mobile facilities to implement highly efficient graphics rendering engine.

From the perspective of delivering information from the Web to mobile devices, our work has similar objective as the kinds of wireless protocols, particularly, the Wireless Application Protocol (WAP). However, our protocol is specificially designed to convey the visual schemas.

From the perspective of distributing the generating and presenting of visual content onto servers and terminal devices respectively, our work has similar functions with the X-window system, Virtual Network Computing (VNC) technique, and Microsoft Remote Desktop service. However, our work is not designed to generally map window or screen to remote computers. Instead, our work focus on working with the bandwidth-limited wireless network environment.

Acknowledgment

This project is supported in part by the National Science Foundation of China No. 60520130299 and Hong Kong RGC Project No. 1062/02E.

References

1. Chittaro, L.: Visualizing information on mobile devices. IEEE Computer **39(3)** (2006) 40–45
2. Cugini, J., Laskowski, S.: Design of 3D visualization of search results: Evolution and evaluation. In: Proc. of IST/SPIE's 12th Annual International Symposium, San Jose, CA (2000) 23–28
3. Kules, B., Shneiderman, B.: Categorized graphical overviews for Web search results: An exploratory study using u.s. government agencies as a meaningful and stable structure. In: Proc. of the 3rd annual workshop on HCI Research in MIS. (2005)
4. Mann, T.M.: Visualization of www-search results. In: Proc. DEXA Workshop. (1999)
5. Roberts, J.C., Suvanaphen, E.: Visual bracketing for Web search result visualization. In et al, E.B., ed.: Proceedings Information Visualization (IV03), IEEE Computer Society (2003) 264–269
6. Wang, Y., Zhou, L.Z.: Extracting, presenting and browsing of Web social information. In: Proc. Web Age Information Management, LNCS 3739. (2005) 828–833
7. Chen, C., Paul, R.J.: Visualizing a knowledge domain's intellectual structure. IEEE Computer **34(3)** (2001) 65–71
8. Shneiderman, B.: Tree visualization with tree-maps: 2D space-filling approach. ACM Transactions on Graphics **11(1))** (1992) 92–99
9. Bederson, B.B., Shneiderman, B., Wattenberg, M.: Ordered and quantum treemaps: making effective use of 2D space to display hierarchies. ACM Transactions on Graphics **21(4)** (2002) 833–845
10. Fluit, C., Sabou, M., van Harmelen, F.: Cluter map — ontology-based information visualization. Chapter 3 of Visualizing the Semantic Web, editors Vladimir Geroimenko and Chaomei Chen (2002) 36–48
11. Bederson, B.B., Clamage, A., Czerwinski, M.P., Robertson, G.G.: Datelens: A fisheye calendar interface for pdas. ACM Transactions on Computer-Human Interaction **11(1)** (2004) 90–119

Web Driving: An Image-Based Opportunistic Web Browser That Visualizes a Peripheral Information Space

Mika Nakaoka, Taro Tezuka, and Katsumi Tanaka

Graduate School of Informatics, Kyoto University,
Yoshida-Honmachi, Sakyo, Kyoto, 606-8501, Japan
{nakaoka, tezuka, tanaka}@dl.kuis.kyoyo-u.ac.jp

Abstract. An image-based opportunistic Web browser called "WebDriving" is described that automatically and continuously visualizes the "peripheral information space". It extracts the images from the current Web page, its link-destination pages, and other related pages and uses them to dynamically construct a 3D space. This enables the user to concurrently browse not only the images on the current Web page but also the images in the peripheral information space. The user browses these images by "driving a car through the constructed 3D world". The user thus becomes aware of other relevant Web pages while browsing the current page, which is a form of "opportunistic browsing". An experiment in which elementary school pupils used the WebDriving browser demonstrated that they could use it to effectively obtain information because it enabled them to intuitively understand a Web space, the functions of a Web browser, and Web search.

Keywords: Image-based opportunistic web browser, Peripheral information space, Investigative learning.

1 Introduction

Search engines are used by many people on a daily basis to search for information on the Web. This can be problematic with conventional Web search engines because carefully chosen keywords must be entered in order to retrieve useful information. Many users have trouble identifying appropriate keywords due to an undeveloped ability to generalize concepts and to clearly specify for what they would like to search due to a lack of experience. One way to acquire information from the Web is by navigating through the link structure. Starting from a certain Web site, a user can search for the target information by navigating the links between pages. Users generally use a mixture of search and navigation to obtain information from the Web.

"Opportunistic searching" means to search for information of potentially interest without having a specific, precise target keywords or URLs. During an opportunistic search, the users may change his/her search interests. Conventional Web browsers are not sufficiently robust for opportunistic searching. During opportunistic searching, users are not so confident with the page currently being browsed. It would thus be helpful to make them aware of the current page's Peripheral Information Space" such as the link-destination pages and related (or similar) pages.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 548 – 553, 2006.

The goal of our research is to develop a navigation browser that makes users aware of the Peripheral Information Space while they are browsing the current page. Towards this goal, we developed a 3D Web browser called "WebDriving" that seamlessly integrates the information on the target page with that in the peripheral information space. After describing our motivation in Section 2, we describe related work in Section 3. The basic concept, particularly the two types of spaces that are constructed, is described in Section 4, and the features and functions are described in Section 5. The demonstration experiment and the evaluation are described in Section 6. We end the paper with a short summary.

2 Motivation

The development of the WebDriving browser was motivated while using conventional Web browsers for theme-based "investigative learning", which is currently being promoted at many Japanese elementary schools. The pupils are encouraged to find a topic of interest, and our WebDriving browser helps them do this by enabling them to "drive through" the 3D Web information space constructed from the images on the current Web page and on its peripheral information space. This approach suits them well because pupils are usually very curious about not only the current Web page but also what we call the "peripheral information space". When using the WebDriving browser, they look around the peripheral space restlessly until they eventually find a topic of interest.

Our Web navigation browser enables users to visualize the peripheral information space. This space includes all of the linked and related pages, which extend around the target page like a net. Web navigation and browsing are performed using two or more Web pages as the peripheral information space increases the likelihood of finding interesting pages.

3 Related Work

As an example from the opportunistic searching, a semantic search function was added to a fisheye interface [1], [2], [3]. A fisheye view presents an image while interactively guiding information with the closest relationship to the object at which the user was looking. The system is well suited for browsing when the user's intention is not distinctly determined.

Many methods have been proposed for using contents in the periphery to supplement the attributes of the current page, such as by employing texts in link anchors. [4], [5], [6] , [7].

Our research is different from them in the point of the immersive visualization system and the specialized representing only images.

4 Peripheral Information Space

The "peripheral Information space" for a given Web page p is defined as a collection of Web pages such that each member page is

- a page related to page *p* or
- a page that is reachable by link navigation from page *p* (as shown in Fig. 1).

During an opportunistic search, users search and browse Web pages, and they determine where to go next and what to search for next based on the contents they have viewed. By showing users the peripheral pages as well as the target page at the same time, we can improve the opportunistic search process.

When a user starts browsing page *p1*, the pages peripheral to *p1* are computed, and both *p1* and the peripheral pages are transformed and visualized depending on the users' intention. If the user clicks on a link or jumps to a related page, the selected link-destination page or the selected related page together with its peripheral pages are transformed and visualized into another peripheral information space seamlessly.

The size limitation of displays makes it difficult to show an entire Web page and all of the peripheral pages. Therefore, in our WebDriving browser, rather than show the entire pages, we show only the images extracted from the pages. The extracted images are placed in a 3D space. This is because the main purpose of WebDriving is to provide a way for users to find topics of interest. Of course, in WebDriving, a user can browse all of a visited page's content by "stopping the car".

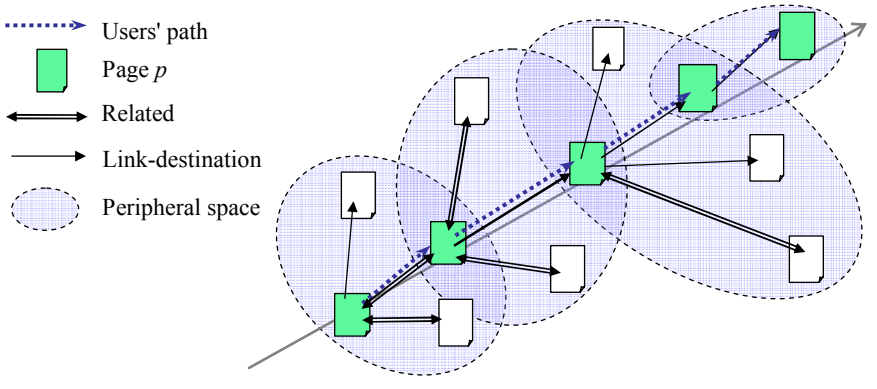


Fig. 1. Concept of "Peripheral Information Space" along the time-line

5 "Driving Through the 3D World"

The WebDriving browser has several features that enable users to visualize the peripheral information space, which is particularly useful to children.

- **Driving-game browsing:** WebDriving makes searching like a driving game, so only minimal encouragement is needed to get children to start using it. Because it works like a driving game, its operation is easy and intuitive.
- **Peripheral information space visualization:** WebDriving shows images collected from related pages in addition to those of the current page simultaneously. It enables users to navigate through the Web by presenting images from relevant pages and pages linked to the current page.

- **Browsing and closer viewing of Web pages:** In WebDriving, users can first skim through many Web pages and then view interesting pages more closely. These two phases are seamlessly connected.

WebDriving browser renders the linked and other related pages in the same 3D space as the driving course generated for the current page by reading the HTML code, and draws signboards, branches, and blinking markers. The signboards and surrounding terrain are based on the images contained in the collected pages. A side road appears a signboard with a triangular marker. A link-destination page appears with an orange blinking marker. Other related pages appear with green blinking markers, as shown in Fig. 2. Side roads are generated from the links in the current page and by searching similar Web pages using Google's "related" operator [8].

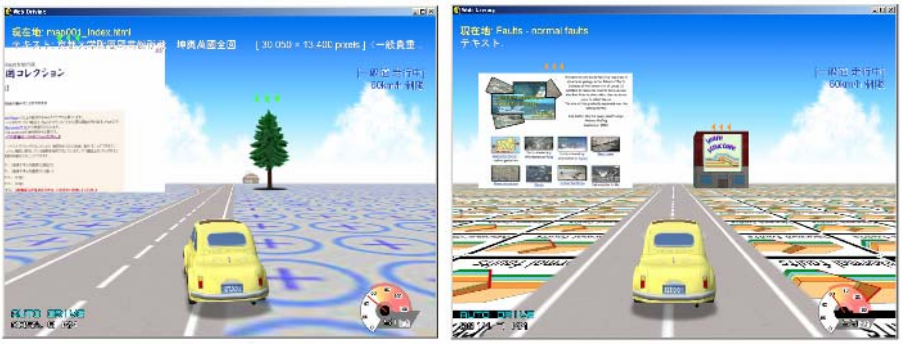


Fig. 2. Branches are indicated by signboards and "green" or "orange" blinking markers (to related pages or to link-destination pages)

6 Demonstration Experiment and Evaluation

We conducted a demonstration experiment in a public elementary school, in the following manner.

- **Task:** Find information about places to visit on school excursion trips.
- **Number of subjects:** 28 students.
- **Grade:** 5th grade.
- **Time for the experiment:** 45 minutes.

Following the experiment, we asked the 23 students to answer four questions.

Q1. Did you find WebDriving useful?
Q2. Which element was the most useful?
Q3. Which element was the least useful?
Q4. Did you find useful information?

There are results for Q1–Q4 as shown in Fig. 3.

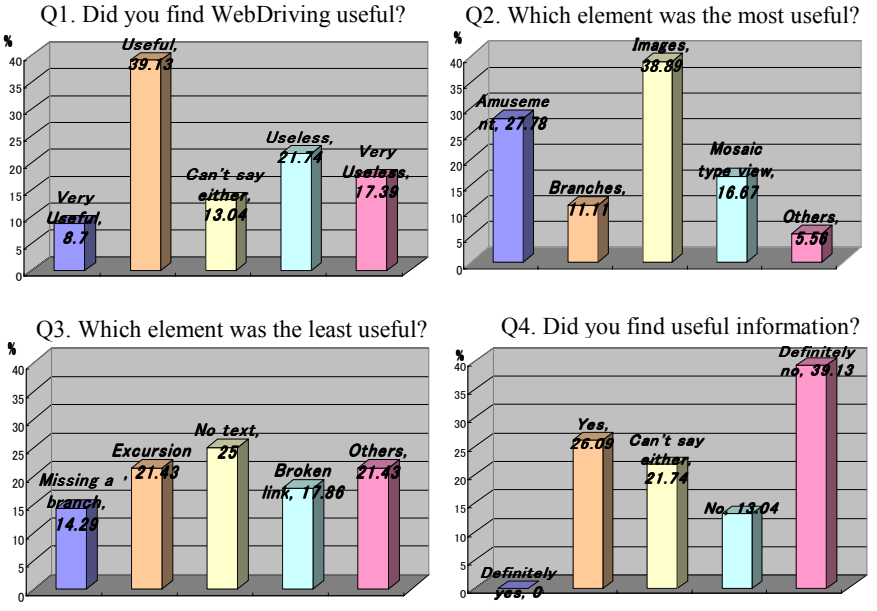


Fig. 3. Results for Q1–Q4

The responses to the first two questions indicate that the WebDriving browser is an entertaining and easy-to-use tool for "experiencing" Web searching and browsing. However, the responses to the last two questions indicate that while WebDriving can be a good motivator, users sometimes have trouble using it. One possible problem is that, since WebDriving allows the user to "drive" at high speed, a user may drive too fast to be able to turn at the desired branch. Another possible problem is that, since WebDriving allows the user to freely decide on the search topic, pupils may have trouble identifying a topic of interest as they are not accustomed to doing so.

7 Conclusion

We have described an image-based opportunistic Web browser called "Web Driving" that automatically visualizes the "Peripheral Information Space". Users can notice an innovative conception concurrently as they "drive through the 3D world". An experiment in which elementary school pupils used the WebDriving browser demonstrated that they could use it to effectively obtain information because it enabled them to intuitively understand a Web space, the functions of a Web browser, and Web search.

In concretely, too much adaptation of information might be intrusive way according to the circumstances. Therefore, by tabulating various information, WebDriving supports that the user can search actively.

Acknowledgments. This work was supported in part by the Japanese Ministry of Education, Culture, Sports, Science and Technology under a Grant-in-Aid for Software Technologies for Search and Integration across Heterogeneous-Media Archives, a Special Research Area Grant-In-Aid For Scientific Research (2) for the year 2006 under a project titled Research for New Search Service Methods Based on the Web's Semantic Structure (Project No. 16016247; Representative, Katsumi Tanaka), and the Informatics Research Center for Development of Knowledge Society Infrastructure (COE program by Japan's Ministry of Education, Culture, Sports, Science and Technology).

References

1. Janecek, P. and Pu, P.: Visual Interfaces for Opportunistic Information Seeking, Proceedings of the 10th International Conference on Human-Computer Interaction (HCII'03), Crete, Greece (2003) 1131–1135
2. Janecek, P. and Pu, P.: An Evaluation of Semantic Fisheye Views for Opportunistic Search in an Annotated Image Collection, Int. Journal of Digital Libraries, Vol. 5, No. 1, Special Issue on "Information Visualization Interfaces for Retrieval and Analysis", March (2005) 42–56
3. Janecek, P. and Pu, P.: Opportunistic Search with Semantic Fisheye Views, Proceedings of the 5th International Conference on Web Information Systems Engineering (WISE2004), Brisbane, Australia (2004) 668–680
4. Chakrabarti, S., Dom, B., Gibson, D., Kleinberg, J., Raghavan, P., and Rajagopalan, S.: Automatic resource list compilation by analyzing hyperlink structure and associated text, Proceedings of the 7th International World Wide Web Conference, April 14-18, Brisbane, Australia (1998)
5. Glover, E. J., Tsioutsiouliklis, K., Lawrence, S., Pennock, D. M., and Flake, G. W.: Using Web Structure for Classifying and Describing Web Pages, Proceedings of the 11th International World Wide Web Conference, Honolulu, Hawaii, USA (2002) 562–569
6. Attardi, G., Gulli, A., and Sebastiani, F.: Automatic Web Page Categorization by Link and Context Analysis, Proceedings of THAI-99, European Symposium on Telematics, Hypermedia and Artificial Intelligence (Hutchison, C. and Lanzarone, G. (eds.)), Varese, Italy (1999) 105–119
7. Pant, G.: Deriving link-context from HTML tag tree, Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, ACM Press, San Diego, California in conjunction with ACM SIGMOD International Conference on Management of Data, 13th June (2003) 49–55
8. Google: http://www.google.com

Blogouse: Turning the Mouse into a Copy&Blog Device

Felipe M. Villoria, Sergio F. Anzuola, and Oscar Díaz

The Onekin Group
University of the Basque Country
P^{o} Manuel de Lardizabal, 1 - 20018 San Sebastián (Spain)
{felipe.martin, jibfeans, oscar.diaz}@ehu.es

Abstract. Blogs are tools that put web publication into the layman's hands. Despite its simplicity, the publication process is a cumbersome task when the content to be published is already in desktop documents. In order to ease this process, we have created *Blogouse*, a user-friendly, editor-independent, and blog-independent publication tool, which applies annotation techniques to the publication system. To attain this aim, we have extended the mouse device functionality to be ontology-aware.

1 Introduction

Initially thought as personal diaries, blogs's scope has broadened to become a medium for professionals to communicate [4,5]. The content of blog posts can range from personal experiences to company news. Enterprises can use blogs for different purposes: as a content management system to manage the content of websites, as a bulleting board to support communication and document sharing in teams, as an instrument in marketing to communicate with Internet users or as a Knowledge Management Tool [4].

This implies that an increasing number of posts do not find their source in the personal experiences of the blog owner. Rather, the blog is used as a means to share information with others. And this information is currently kept in the user's desktop. Regardless of the origin of the information (e.g. a paragraph found in a PDF document or a slide in a PPT presentation), this content is liable to be blogged, i.e. to be discussed and shared with the blog participants.

This scenario leads to a decoupling between the blog as a sharing platform, and the desktop as a resource container. Currently, this gap is manually traversed by the blogger in a copy&paste manner, a cumbersome task that defeats the blog's easiness principle. Consider the following scenario. Reading the sport newspaper on the web, you come across with some awful comment on the performance of your favorite soccer player on the last match. You want to collect the opinions of your blogmates about this issue. To this end, you copy the corresponding paragraph from the newspaper and you blog it. The latter implies to go to the blog, create a new post, edit it, clean it[1] and finally, save it. Might be too cumbersome to be worth the effort!! Blog is an spontaneous action which can be refrained if the process is lengthy and unhandy.

Fortunately, some browsers (e.g. Firefox, Mozilla) and search engine (e.g. Google) are providing the *"BlogThis!"* functionality. This functionality achieves "copy&blog"

[1] A blog entry is basically text. Figures or rendering tags needs a special treatment.

K. Aberer et al. (Eds.): WISE 2006, LNCS 4255, pp. 554–559, 2006.

in an automatic way: posting a Web page is as easy as clicking on the *BlogThis!* icon. This opens a small window where a blog post is edited after the HTML page and a pointer is included to this source page.

Unfortunately, similar functionality is not available for resources other than HTML pages. This means that interesting quotes from PDF or Excel documents can only be posted through laborious copy&paste, defeating spontaneous blogging. And even if this were the case, a *BlogThis!* functionality on an editor basis would require the user to become familiarise with distinct GUI interfaces (for Word, Excel, or IE), putting an additional burden on the layman.

Based on this observation, this work describes a *BlogThis!* device for desktop resources that is editor independent: ***Blogouse***. *Blogouse* is a blog client that achieves editor independence by building on the mouse rather than the editor. By clicking on the mouse's middle button, a post can be created from no matter which editor. Specifically, this device aims to fulfil the following requirements:

- user-friendly. Usability is a main quality requirement in any blog interface to promote the participation of the layman. Sophisticated functionality should not be obtained at the cost of convoluted interfaces.
- blog-system independence. A myriad of tools are currently available for laymen to build up their blogs. *Blogouse* should be as independent as possible from the underlying blog system.
- editor independence. The device should work with any editor, no matter the format of the file.

The rest of the paper describes our contribution (Section 2), *Blogouse* blog client (Section 3), and, finally, evaluation and conclusions are given.

2 Blog Clients for Second-Generation Blogs

A blog is basically a sequence of **posts** that are arranged chronologically. The post structure includes a title, a description, and is uniquely identified by an anchor tag. Additionally, it is catching on for bloggers to categorise their posts. In this way, blogs become rudimentary annotation tools that permit users to annotate the post according with a very basic ontology. So far, the most common metadata for post description is "category".

Blog interaction can be achieved through graphical-user interface, RSS interface, and programmatic interface. We focus on the latter. The programmatic interface (i.e. the blog's API) allows access to the inner workings of the blog engine to be customised. Although there is not such thing as a standard for blog APIs, there is some consensus about the functionality to be supported. Some well-known APIs are *Blogger API*[2], *MetaWeblog API*[3], and *MovableType API*[4], which are built on top of the XML-RPC protocol[5]. These APIs permit the construction of desktop clients for blog engines.

[2] http://www.blogger.com/developers/api/1_docs/

[3] http://www.xmlrpc.com/metaWeblogApi

[4] http://www.sixapart.com/movabletype/docs/mtmanual_ programmatic

[5] http://www.xmlrpc.com/spec

A desktop client is an application which runs on your PC and communicates with a blogging system using the XML-RPC protocol. In this way, post editing and post publishing are decoupled, that is, the user types the post on the client side, and latter on, publishes it in one or several favorite blog engines. Unlike remote editing, local editing permits to incorporate additional content without caring about connexion problems. For shallow, semantic-less, typed post, this could not be regarded as an important advantage as editing a post just takes a few seconds. However, second-generation blogging oversees the initial use of blogs as personal diaries to become lightweight content management systems [4].

An additional argument in favour of blog clients is tight integration with the desktop resources. Web-based editing does not permit the sophistication GUI gadgets available in a desktop and to which users are accustomed. As pointed out in [3], *"access to other desktop applications and their data (e.g. through their public APIs), control of the clipboard, and techniques like drag-and-drop are difficult or impossible to implement in a web-based environment"* which make them also favour a desktop approach.

The *BlogThis!* functionality permits posts to be constructed out of existing desktop resources (e.g. a PDF file, an HTML page). The title, the content, and other properties of the post should be mostly derived from the resource itself. Hence, posting is equated with annotating the resource along "the blog ontology".

Some of the information can be automatically extracted when the mapping from the document content is clear. However, this is not always the case. For example, the title can be extracted from different places such as the document title, a section name or, even, the beginning of a phrase. This situation deteriorates when distinct formats and editors are considered. Automatic extraction could sacrifice editor/format independence, a hallmark of our approach.

Incorporating a *BlogThis!* button into each editor (i.e. as Google) will certainly lead to an integrated solution but at the price of coupling editing and blogging. The myriad of formats which can be found in current desktop (e.g. *.doc*, *.xml*, and *.txt*), and the corresponding editors, vindicate the use of an editor-independent solution. Our bet is to use the mouse to attain this aim.

Rather than using the extensibility technology provided by each editor (e.g. *ActiveX* controls in *Word*), we move down to the operating system so that the solution can be available to no matter which editor. The result is *Blogouse*, a *BlogThis!* device that achieves editor-independence by working at the operating-system level. By clicking on its middle button, the post is obtained from the resource regardless of the editor you are working with. In this way, the user does not have to move to a new editor when posting (e.g. desktop clients), nor has to learn a new GUI when resources from different formats are edited (e.g. *Google*). In this way, we strive to enhance the functionality without loosing the simplicity and usability that make blogs so popular.

3 Blogouse at Work

BlogThis! can be regarded as an extractive process that obtains a post out of an external resource. If all posts have the same characterisation, the only source of variation in this process will be the external resource as such, i.e. whether the post is obtained from an

HTML or a PDF document. This source of variation is faced by abstracting from the editor at hand, and working at the operating-system level.

However, we envisage a second source of variation, namely, the characterisation of the post. It can vary depending on the blog ontology. So far, conventional tags are used to describe no matter which post regardless of its content. However, authors envisage a promising scenario where posts can be annotated to surface their semantic content, and become integral parts of the semantic Web [2]. In this setting, post extraction depends on the ontology at hand, and the blog client needs to be configured with the blog ontology.

3.1 Blogouse Configuration

By clicking on the middle button of the mouse, the user is prompted to indicate a profile. If a new one is to be created, the user is prompted for the blog main page URL, the user name and the password.

When a blog connexion is created, *Blogouse* configures itself by retrieving the HTML page which is pointed at main page. Then, the following data are obtained:

- the blog's title, which is used to name the profile.
- the RSD (Really Simple Discovery) URL, which holds the URI for a RSD file[6]. RSD aims at simplifying *"the discovery of setting information by reducing the information a user must supply to three well known elements"*.
- the ontology URL, which points at the blog ontology file. This file is used to configure dynamically the mouse menus for the annotation process.

This process only occurs at configuration time. Once the configuration is set, the user deploys this configuration by just selecting the corresponding profile. The profile fixes where the post is going to be published, and which ontological terms are available for annotation during the *BlogThis!* process.

3.2 The *BlogThis!* Process

Once a profile is selected, *Blogouse* is ready for guiding the annotation process through drop-down menus. All the post properties are now obtained from the resource by selecting some text chunk, and next, clicking on the appropriate menu. For instance,

- *title* is obtained by selecting some text chunk, and next, clicking on the menu. Since *"title"* only admits one value, successive selections have a substitution effect. Moreover, this element is mandatory (denoted by the asterisk), therefore, the user has to annotate it before publishing content in the blog.
- *description* is also obtained through selection. However, this element is multivalued (denoted by *"m"*), so successive selections have a cumulative effect.
- *subject* and *rating* are metadata elements which are directly provided by the user by selecting the appropriate menus (see figure 1). As the metadata depends on the blog at hand, this GUI is generated on the fly based on the blog profile. For instance, *rating* only admits five values, namely *"none"*, *"low"*, *"medium"*, *"high"*, and *"outstanding"*.

During this process, the (inconclusive) post is kept in a special clipboard. Once concluded, the user can publish the bright-new post by clicking on *BlogThis!*.

[6] http://archipelago.phrasewise.com/spec

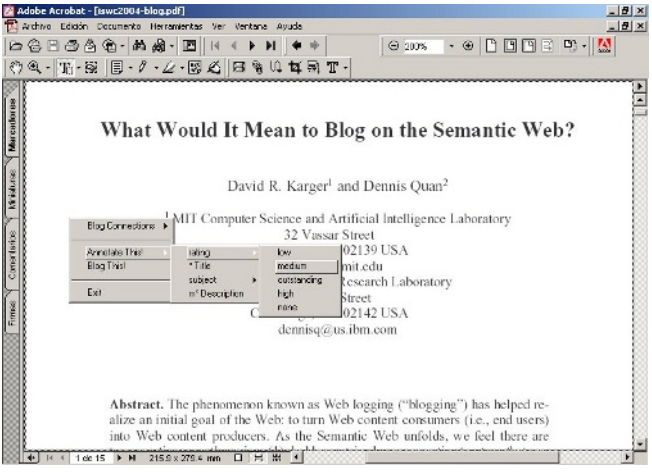


Fig. 1. Document rating annotation

4 Evaluation and Conclusions

An experiment has been conducted to use blogs for teaching material sharing and revision among six lecturers. Some lecturers were reluctant to participate as they look down blogs as mere diary's entries, and producing posts out of teaching material was quite labour intensive. On the other hand, most of them recognised the advantages of having a central hub where documents as well as comments can be widely and easily available. As pointed out in [4] *"weblog can reduce the volume of e-mails received by the participants; reduce searching time looking for teaching material; effectively archive project documentation"*.

All participants agree these advantages were effective, and that *Blogouse* facilitated the ready publication of their desktop resource. Editor independence was regarded as the most outstanding feature of *Blogouse*. The variety of formats used for storing teaching material as well as the heterogeneous range of editors in such a free environment as the University, where staff is free to choose the editor he or she enjoys most, makes this feature really a must. Furthermore, participants appreciate the easiness of the interface provided by *Blogouse*.

On the down-side, the drop-down menus were found cumbersome. The annotation of some metadata require three mouse clicks just for simple ontologies. This really poses a scalability problem for sophisticated ontologies where a longer "drill-down" process could be required to locate the appropriate concept.

Some authors note that semantic metadata should be produced *"as a by-product of tasks that a user is already used to perform on a day-to-day basis, such as entering people in an address book application, organizing events in a calendar or managing publications in a bibliographic database"* [1,3], and that *"while the metadata added to a blog entry could in principle be hand coded or added through specific form fields in*

a web-based blog editor, we believe that this would be far too complicated to appeal to a non-technical user, like the average employee we are aiming at" [3].

We agree on this statement. However, its feasibility is limited to structured resources (e.g. bibliographic entries, calendar entries and the like) where wrapping techniques can be used to extract the appropriate metadata without user intervention. However, it is well-known that most of the desktop resources are far from being structured, and annotation techniques and tooling are required here. In this sense, *Blogouse* offers a compromise between simplicity and more powerful approaches to metadata extraction.

Acknowledgements. This work is partially supported by the Spanish Science and Technology Ministry (MCYT) under contract TIC2005-05610.

References

1. James A. Hendler. Agents and the Semantic Web. *IEEE Intelligent Systems*, 16(2):30–37, March/April 2001.
2. David R. Karger and Dennis Quan. What Would It Mean to Blog on the Semantic Web? In *Proceedings of 3rd International Semantic Web Conference (ISWC2004)*, pages 214–228, November 2004.
3. Knud Möller and Stefan Decker. Harvesting Desktop Data for Semantic Blogging. *ISWC 2005 Workshop*, November 2005.
4. Martin Röll. Business Weblogs - A pragmatic Approach to introducing Weblogs in medium and large Enterprises. *BlogTalk*, May 2003.
5. Martin Röll. Distributed KM - Improving Knowledge Workers' Productivity and Organisational Knowledge Sharing with Weblog-based Personal Publishing. *BlogTalk 2.0*, July 2004.

Author Index