Using Knowledge Base for Event-Driven Scheduling of Web Monitoring Systems

Yang Sok Kim[1,2], Sung Won Kang[2], Byeong Ho Kang[2], and Paul Compton[1]

[1] School of Computer Science and Engineering, The University of New South Wales, Sydney, 2001, New South Wales, Australia
{yskim,compton}@cse.unsw.edu.au
[2] School of Computing and Information Systems, University of Tasmania, Hobart, 7001Tasmania, Australia
{swkang,bhkang}@utas.edu.au

Abstract. Web monitoring systems report any changes to their target web pages by revisiting them frequently. As they operate under significant resource constraints, it is essential to minimize revisits while ensuring minimal delay and maximum coverage. Various statistical scheduling methods have been proposed to resolve this problem; however, they are static and cannot easily cope with events in the real world. This paper proposes a new scheduling method that manages unpredictable events. An MCRDR (Multiple Classification Ripple-Down Rules) document classification knowledge base was reused to detect events and to initiate a prompt web monitoring process independent of a static monitoring schedule. Our experiment demonstrates that the approach improves monitoring efficiency significantly.

Keywords: web monitoring, scheduling, MCRDR.

1 Introduction

Nowadays a large amount of new and valuable information is posted on the web daily and people wish to access this in a timely and complete fashion. This may be done manually, in that, people go to specific web pages and check whether information is new. However, this approach has limitations. For example, it can be very difficult to identify which objects have been changed on the web page since the last visit. Various web monitoring systems, sometimes called continuous query (CQ) systems, have been proposed by many researchers, including CONQUER [1], Niagara [2], OpenCQ [3] and WebCQ [4]. Even though they were proposed in the different contexts, they were designed to help users to keep track of continually changing web pages and identified changed information on the specific web pages by revisiting them frequently and comparing objects. Web monitoring systems may focus on different objects on the web pages, including hyperlinks, images, and texts.

There are two main goals in web morning systems. On the one hand, they should find changed objects on the target web pages without missing any information. The problem here is that they may miss information when the revisit interval for a specific

T. Di Noia and F. Buccafurri (Eds.): EC-Web 2009, LNCS 5692, pp. 169–180, 2009.

web page is longer than its change interval. On the other hand, they should find changed objects without significant delay that is the gaps between publishing and collecting time. These two main goals may be achieved by very frequent revisits to the target web pages. However, there are significant restrictions to the revisit frequency as web monitoring systems operate under resource constraints related to computing power and network capacity, and there may be restrictions on access to specific web pages by the web servers. The goal then is a scheduling algorithm that minimizes delay and maximizes coverage given the resource constraints.

Various statistical approaches have been proposed to improve web monitoring efficiency. CAM [5] proposes web monitoring with a goal of capturing as many updates as possible. CAM estimates the probability of updates by probing sources at frequent intervals during a tracking phase, and using these statistics to determine the change frequency of each page. However, CAM does not explicitly model time-varying update frequencies to sources and cannot easily adapt to bursts. The WIC algorithm [6] converts pull-based data sources to push-based streams by periodically checking sources for updates. The algorithm is parameterized to allow users to control the trade-off between timeliness and completeness when bandwidth is limited. The algorithm chooses the objects to refresh based on both user preferences and the probability of updates to an object. However, the algorithm does not consider how to determine the probability of an object update, which is an important aspect of any pull-based scheduling. Bright et. al [7] proposed adaptive pull-based policies in the context of wide area data delivery, which is similar to web monitoring. They explicitly aim to reduce the overhead of contacting remote servers while meeting freshness requirements. They model updates information on data sources using update histories and proposes two history-based policies to estimate when updates occur. In addition, they also presented a set of adaptive policies to cope with update bursts or to estimate the behaviour of objects with insufficient histories available. The experimental evaluation of their policies using trace data from two very different wide area applications shows that their policies can indeed reduce communication overhead with servers while providing comparable data freshness to existing pull-based policies.

Although these approaches provide sophisticated scheduling policies, these have the following limitations: Firstly, the statistical approaches ignore how the user uses the monitored information or how different users value it. The users are often interested in specific topics such as sports or financial news. Furthermore, the user may try use further processes related to their own personal knowledge management, such as filtering and/or classification of the collected information, to overcome information overload problems. Users do not give equal importance to the all collected information. For example, if a user is a fund manager, he will probably be interested in financial news. If the user is interested in specific topics, the system should give information about these topics greater priority and the schedules should satisfy this requirement. Secondly, the statistical approaches underestimate the fact that the volume of information published may be affected by specific events. For example, when the investment company, Lehman Brothers, collapsed, many online newspapers published articles related to this specific event. The previous methods which used statistic or mathematically-based schedules for web monitoring systems cannot properly react to these kinds of

event-based publication volatiles. Whereas the first issue is related to the information demand factor, this second issue is closely related to information supply.

This paper focuses on these two issues and tries to suggest solutions for them. Our research, however, does not deny the importance of the previous research efforts. Instead, our research aims to improve statistical scheduling approaches by complementing them with an event-driven scheduling approach. Our event-driven scheduler detects new events on the Web using document classification knowledge and then initiates new monitoring process. Section 2 explains our document classification method which was employed to construct a knowledge base for our study. Section 3 proposes our event driven scheduling method. The experimental design employed for our scheduling method evaluation is discussed in Section 4 and experimental results are summarized in Section 5. Conclusions and further study is in Section 6.

2 MCRDR Document Classification System

We base our approach on document classification. Multiple Classification Ripple-Down Rules (MCRDR)[8], an incremental knowledge acquisition method, was employed to develop a document classification system, called an MCRDR classifier. The system acquires classification knowledge incrementally, because documents are provided continually and classification knowledge changes over time. Figure 1 illustrates an example of the knowledge base structure of the MCRDR classifier. As illustrated in the right tree of Figure 1, the user's domain knowledge is managed by a category tree, which is similar to a common folder structure and represents hierarchical relationships among categories. It can be easily maintained by domain experts for managing a conceptual domain model through simple folder manipulation.

The user's heuristic classification knowledge is maintained by an n-ary rule tree. The left tree of Figure 1 represents a rule tree, which has hierarchical relationships. A child rule refines its parent rule and is added as an exception of its parent rule. For example, Rule 3 is an exception rule of Rule 1. One special exception rule is the stopping rule, which has no indicating category (null), in the conclusion part. Rule 5 is an example of a stopping rule. In the inference process, the MCRDR classifier evaluates each rule node of the knowledge base (KB). For example, suppose that a document that has a set of keywords with T = {a, b, d, k} and B = {f, s, q, r} is given to the MCRDR classifier whose knowledge base is the same as in Figure 1. The inference takes places as follows. The MCRDR classifier evaluates all of the rules (Rule 1 and Rule 4) in the first level of the rule tree for the given case. Then, it evaluates the rules at the next level which are refinements of the rule satisfied at the top level and so on. The process stops when there are no more children rules to evaluate or when none of these rules can be satisfied by the given case in hand. In this instance, there exist two satisfied rule paths (Path 1: Rule 0 – Rule 1 – Rule 3, Path 2: Rule 0 – Rule 4 – Rule 5), but there is only one classification folder (*C5*), because Rule 3 is a stopping rule (see below). The MCRDR classifier recommends *C5* as a destination folder for the current case.

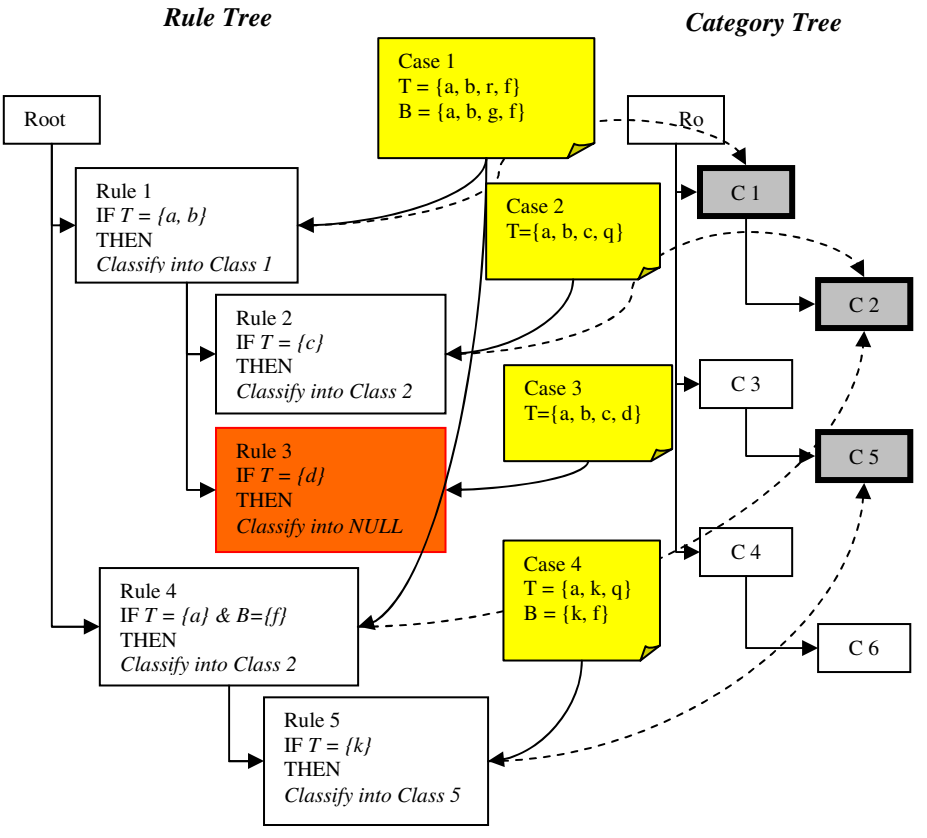


Fig. 1. Knowledge Base of MCRDR Classifier

The knowledge acquisition (KA) and inference processes are inextricably linked in an MCRDR classifier, so some KA steps depend on the inference structure and vice versa. Hereafter, Figure 1 will be used as an example to show how a knowledge base is constructed by the user. At the beginning, there is no rule in the knowledge base. The KA process begins when a case has been classified incorrectly or has no classification. If Case 1 is given to the MCRDR classifier, the system does not suggest any recommendation, because there is no rule in the knowledge base. The user creates two rules – Rule 1 and Rule 4 – using the current case. Therefore, the current case is classified into *C1* by Rule 1 and *C2* by Rule 2. This type of rule is called a refining rule in MCRDR, because it refines the default rule (Rule 0). This is a special type of refining rule because there is no recommendation. A general refining rule is exemplified by Case 2. If Case 2 is given to the system, the MCRDR-classifier suggests Class 1 as a recommendation according to Rule 1. Suppose the user does not satisfy this result and he/she wishes to classify this case into *C2*. After the user initiates the knowledge acquisition process, a new refining rule creation process is summarised in Figure 2:

Step 1: The user selects a destination folder from the category tree and the MCRDR classifier generates case attributes;

Step 2: The user selects keywords from the case attributes, for example 'c' in Title;

Step 3: The system generates document lists satisfying rules in this new rule path (Rule0 – Rule 1 – Rule 3 (new rule));

Step 4: If the user selects one or more of the documents in these lists to exclude them, the MCRDR–Classifier presents the difference lists instead of the case attributes; and

The user performs Step 2 ~ 4 iteratively until the remaining document lists do not include any irrelevant documents.

Fig. 2. Rule Creation Process

A stopping rule is exemplified by Case 3. If Case 3 is given to the MCRDR classifier, it suggests *C1* as a recommendation. Suppose the user does not classify this current case into this folder, but also does not want to classify it into any other folders, as a result, a stopping rule is created under the current firing rule (Rule 1). The stopping rule creation process is the same as the refining rule creation process, except that a stopping rule has no recommending folder. Prior studies show that this guarantees low cost knowledge maintenance[8]. The MCRDR classifier has been successfully used in various situations. Domain users can construct classification knowledge within a very short time and without any help from the knowledge engineer. Several papers have been written on performance evaluation of an MCRDR classifier [9-13].

3 Method

The MCRDR classifier was used to determine the similarity of web pages and identify the occurrence of particular events. The similarity between web pages can be defined by comparing the number of articles which have been classified into the same categories. For example, assume there is a monitoring system which has implemented an MCRDR classifier to classify collected articles and web pages, identified as A, B and C, are registered to be monitored. While the system is running, the classifier classify 17 articles from web pages A, 15 articles from web pages B and 4 articles from web page C into the same folder D. Clearly web pages A and B can be considered more likely to provide more similar information than web page C. Although web page C provides a few articles similar to web pages A and B, only a few articles have been classified into the same category.

We should be able to identify whether an event has been occurred by analysing the classification history. That is, an event can be defined as an occurrence of an abnormal pattern in the classification history of a particular web page. In this research, ***average publication frequency per hour per day*** was used to detect events, because publication patterns change according daily and even hourly basis[14]. For example, assume that normally an average of three articles from a web page are classified to a

- Register monitoring web pages of interest
- Set a schedule for monitoring each web page using naïve or statistical policies.
- Generates a day-hour average classification tables for each web page as 7 days × 24 hours matrices for each category.
- $A(K)_{ij}$ means web page K's average classifications at *j* hour on *i* day of week (e.g., 14:00 on Monday)
- Get the current classifications of each web page ($C(K)_{ij}$) (e.g., web page K's classification at 14:00 on Monday)
- If $C(K)_{ij} > A(K)_{ij} + \theta$ (classification threshold), then the system finds other monitoring web pages that provide similar contents and executes web monitoring regardless of original schedules.

Fig. 3. Event-Driven Scheduling Algorithm

particular category between 12:00pm and 2:00pm on Monday. However, if seven articles from the same web page are classified to the folder in the same time period and day, it can be considered that the some events may have occurred in the real world. To make this judgement, we need to set up a reasonable threshold for each web page to determine whether or not the web page is referring to a significant event. The event-driven scheduling algorithm is summarized in Figure 3.

There are three decision factors in the above algorithm. Firstly, it is necessary to determine how to generate a day-hour average classification tables. Each average value ($A(K)_{ij}$) can be calculated for an overall period or specific time span (e.g. the last five week). In this study, we used the overall experiment period to calculate this average. Secondly, it is necessary to determine the threshold value (θ) that is used for event detection. This value may be determined by experience and in this study this value was set at $0.2 \times A(K)_{ij}$. Lastly, it is necessary to determine which web pages are related to a specific web page. This research uses the classification history of web pages to determine similarity between web pages. That is, web page similarity is decided by the classification frequency for a specific category.

4 Experiment Design

4.1 System Development

The Java programming language and MySQL database were used to develop our event-driven monitoring system. Figure 4 illustrates the system architecture of the event-based web monitoring system, which consists of five modules. The static scheduler initiates each static monitoring process. There are many previous researches on static scheduling methods, but they are not included in this research, because this research mainly focuses on the event-driven scheduling. In this system the user can specify fixed revisit intervals such as every 2 hours. This simple static scheduling may be replaced by more sophisticated methods, and the methods proposed here would go on top of these more sophisticated methods. As computing resources are generally limited, it is necessary to manage system resources efficiently and a thread pool may be used for this purpose. In our system, a Java thread pool was implemented

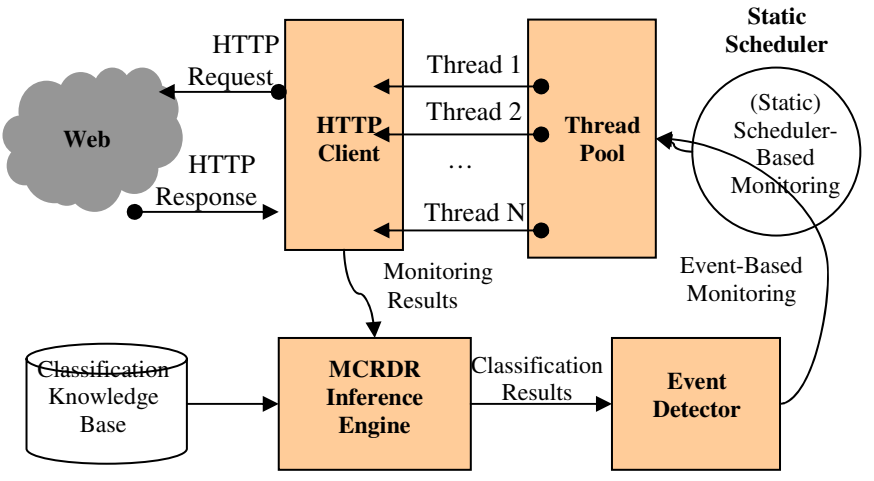


Fig. 4. Event-Based Web Monitoring System Architecture

to manage large scale HTTP requests, which protects against running out of resources. The Apache httpclient (http://hc.apache.org/httpclient-3.x/) library was used in this project because this is stable and commonly used in the HTTP network programming. A HTML parser, called htmlparser (http://htmlparser.sourceforge.net/), library was used to manipulate HTML/XHTML documents. The MCRDR (Multiple Classification Ripple-Down Rules) inference engine automatically classified the collected documents. The event detector finds anomalous document classification results from each monitoring session and initiates additional event-driven monitoring processes when an abnormal increase of the publications occurs.

4.2 Data Collection

In order to collect web history data, 60 web pages were monitored every two hours for about a month, from 8th August to 3rd September, 2008. This data collection period corresponds with the Beijing Olympic period. The Olympic period provides an excellent opportunity to examine event-driven scheduling performance compared with the normal web environment as the Games sporting events are exactly the type of events that event-driven scheduling should pick up. Web pages to be monitored are selected as follows: First of all, we attempted to choose the countries which more tend to be interested in Olympic Games, because web pages from those countries may publish more articles which are related to Olympic Game than others. We selected Australia, the United States and Great Britain as the countries to monitor. They were in the top 10 countries from the medal table for the 2004 Olympics in Athens and would be expected to have a keen interest in the 2008 Olympics. In addition, articles published in these countries will be in English. This is important because with the MCRDR classifier we used a user can create rules for the MCRDR classifier only in English. After selecting these countries, ten generally "well-known" news web sites were chosen from three different countries. We decided to monitor the sports page of each

Fig. 5. Category Structure

selected web site as well as their homepages, because articles about Olympics may be updated more frequently in the sports page. As a result, a total of 60 web pages (3 countries × 10 web sites × 2 web pages (homepage and sports page) were used to collect web history data. Obviously, focusing on the "well-known" news web sites may bias our results to a certain degree, but we believe these sites are popular sites for many people. During this period, a total 131,831 web pages; 46,724 pages from Australia, 43,613 pages from United States and 41,494 pages from Great Britain were downloaded and stored into the database. The size of all collected data was around 580MB. Each data includes the information about its link (stored in the form of absolute URL), the link name (title), data origin (to indicate which Web site it was published from), its contents (HTML source associated with absolute URL) and the time it was captured by the system.

4.3 Classification with MCRDR Classifier

After collecting documents from the above web pages, a master's degree student classified the collected documents with the MCRDR classifier for about one month (between 5th September 2008 and 13th October 2008). We decided to define two major categories first, *Countries* and *Sports*, because these two concepts are the main subjects in Olympic Games. Under *Countries* category, the name of the top ten countries in the previous Olympic 2004 were created and 28 categories of summer sports, referred to on the official Web site of the Beijing 2008, were created as sub-categories of *Sports* category. Figure 5 summaries the category structure used in this experiment. A total 3,707 rules were manually created using 7,747 condition words. Each rule generally contains an average 2 condition words. The number of rules for the conclusion categories was 1,413 and the number of stop rules was 2,294. There are more of these as initially we made rules which were not specific enough and many articles that were not really related to a particular category were assigned to it, so the participants inevitably had to create many stop rules to correct errors. A total of 29,714 articles, about 22% of the entire articles from the dataset, were classified into the each classification category.

4.4 Simulated Web Monitoring

Simulated web monitoring was conducted to evaluate our event-driven scheduling method using the above data set. Our simulation system periodically retrieves the

collected web pages with given intervals and calculates each article's delay time between the capturing time and the retrieving time. For example, if the system starts to retrieve the collected articles every four hours from 8th August 2008 00:00:00 and there was an article collected at 8th August 2008 02:10:00, its delay is 110 minutes. The top five sub-categories of *Countries*, in respect to the amount of the classified documents, were chosen for this experiment. Then the top five monitoring web pages of each category were selected according to their classified article count. Two types of simulated web monitoring were conducted with these five categories and their five web pages. Firstly, a static simulation was conducted to calculate a default delay with given intervals (2, 4, 8, 12, and 24 hours). Then three types of event-driven monitoring simulations were conducted with different **scheduling interval assignment strategies**. There are three possible ways to assign intervals based on the number of classified documents in a category and it is necessary to examine whether or not these assignment strategies affect on the performance of the event driven scheduling methods. Suppose that there are five web pages, called P_1 (100), P_2 (90), P_3 (80), P_4 (70), and P_5 (60), where the numbers show the classified documents in a category. Firstly, the shorter interval can be assigned to the web page that has higher number of documents. This is called the "top-down" strategy in this research. According to this strategy, each web page has the following intervals: P_1 (2 hours), P_2 (4 hours), P_3 (8 hours), P_4 (12 hours), and P_5 (24 hours). Secondly, the intervals may be assigned to the inverse order of the "top-down" strategy, which is called the "bottom-up" strategy. In this strategy, each web page has the following intervals: P_1 (24 hours), P_2 (12 hours), P_3 (8 hours), P_4 (4 hours), and P_5 (2 hours). Lastly, the intervals may be assigned randomly, which is called the "random" strategy.

5 Experimental Results

Figure 7 illustrates simulated web monitoring results with static scheduling method, where the horizontal axis represents simulated monitoring intervals in hours scale, the vertical axis represents each category's average delay time of five web pages by minute scale, and each bar represents the selected category number.

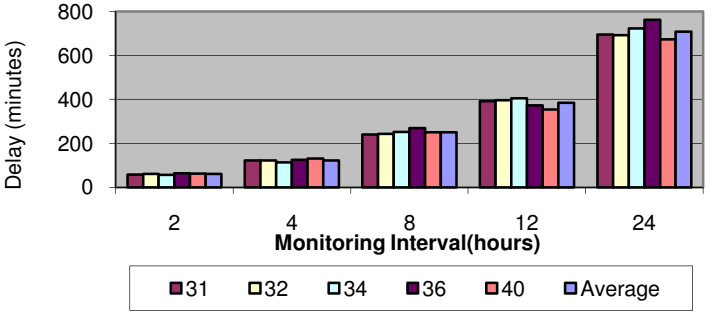
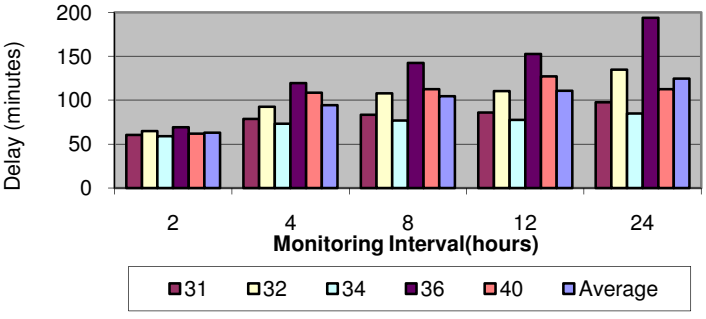
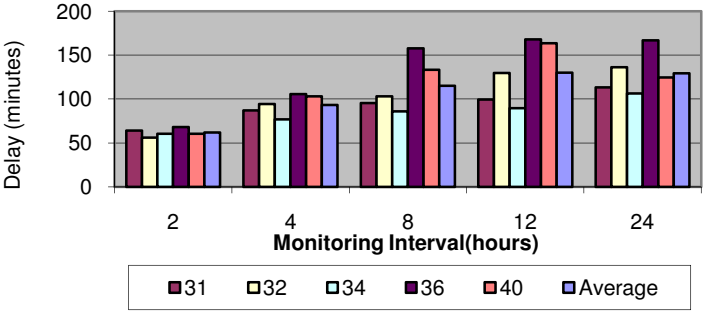


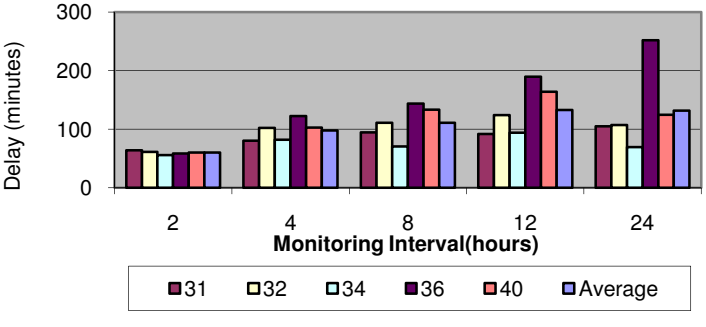
Fig. 6. Simulated Monitoring Results with Static Scheduling



(a) Top-down Scheduling Strategy



(b) Bottom-down Scheduling Strategy



(c) Random Scheduling Strategy

Fig. 7. Simulated Monitoring Results with Event-Driven Scheduling

The static scheduling based results show that average delay time of each category's five web pages increase as the monitoring intervals increase and shows similar levels of delay for the same monitoring intervals. For example, whereas there is about 60 minutes delay when the monitoring interval is set two hours, there is about 700 minutes delay when the monitoring interval is set 24 hours. These results were used as default delay time for each category in the following discussion.

Table 1. Experimental Results

Monitoring Methods		Monitoring Intervals					
		2	4	8	12	24	Average
Static Monitoring		60.8	122.6	251.3	384.2	708.8	305.5
Event-Driven Monitoring	Top-down	63.1	94.5	104.6	110.7	124.8	99.5
	Bottom-up	61.8	93.4	115.2	130.1	129.4	106.0
	Random	60.0	97.9	110.7	132.7	131.6	106.6
	Average	61.6	95.3	110.2	124.5	128.6	104.0
	Improvements	101%	78%	44%	32%	18%	34%

Three event-driven simulated web monitoring results with different scheduling interval assignment strategy are illustrated in Figure 7(a), Figure 7(b), and Figure 7 (c).The main findings are as follows: Firstly, the results show that the event driven scheduling significantly improves monitoring performance compared to the static scheduling method. Table 1 summaries average delays of all categories and it demonstrates that the delay time significantly improves as the event-driven scheduling methods applied to the monitoring. Secondly, the results show that the event-driven monitoring system improves more when the monitoring intervals are longer. For example, when the monitoring interval is four hours, the event-driven monitoring delay is 78% of the static monitoring delay, but when the monitoring interval is 24 hours, it is only 18%. Lastly, the results show that there is no significant difference between different event-driven monitoring strategies. Although overall performance of the top-down strategy is slightly better than those of other strategies, it is not so significant.

6 Conclusions and Further Work

In this paper, we pointed out the limitation of existing scheduling approaches in capturing event-based changes on the Web and introduced a possible scheduling algorithm to cover those changes for the monitoring system. A knowledge base scheduling algorithm is able to trigger the scheduler for other web pages, if an abnormal pattern has been detected on a particular web page. Our experiment was perform with 60 selected news web pages from three different countries; Australia, the United States and Great Britain. The results show that the monitoring system can significantly reduce the delay time, by implementing an event-driven scheduling algorithm. However, several issues still need to be addressed. First of all, the experiments were done by simulation; and the system has not yet been tested in a real situation. Secondly, we have not attempted to find the most appropriate threshold to define an event and the time span between the time when an event is detected and the time the system activates an event-based scheduler. Lastly, our rules were written specifically to classify information about the Olympics. We believe that for any specialist or even general monitoring system enough knowledge will gradually be entered to pick up all

important events related to either general or particular specialist interests. A further interesting question is whether it would be useful to use such an approach not just for scheduling but providing alerts (e.g., flagging a user that many web pages were publishing a lot of new information about a particular topic).

References

1. Liu, L., Pu, C., Han, W.: CONQUER: a continual query system for update monitoring in the WWW. Computer Systems Science and Engineering 14(2), 99–112 (1999)
2. Naughton, J., et al.: The Niagara internet query system. IEEE Data Engineering Bulletin 24(2), 27–33 (2001)
3. Liu, L., Pu, C., Tang, W.: Continual Queries for Internet Scale Event-Driven Information Delivery. IEEE Transactions on Knowledge and Data Engineering 11(4), 610–628 (1999)
4. Liu, L., Pu, C., Tang, W.: WebCQ: Detecting and delivering information changes on the Web. In: CIKM 2000. ACM Press, Washington D.C (2000)
5. Pandey, S., Ramamritham, K., Chakrabarti, S.: Monitoring the dynamic web to respond to continuous queries. In: WWW 2003, Budapest, Hungary (2003)
6. Pandey, S., Dhamdhere, K., Olston, C.: WIC: A General-Purpose Algorithm for Monitoring Web Information Sources. In: 30th VLDB Conference, Toronto, Canada (2004)
7. Bright, L., Gal, A., Raschid, L.: Adaptive pull-based policies for wide area data delivery. ACM Transactions on Database Systems (TODS) 31(2), 631–671 (2006)
8. Kang, B., Compton, P., Preston, P.: Multiple Classification Ripple Down Rules: Evaluation and Possibilities. In: 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, University of Calgary (1995)
9. Kim, Y.S., et al.: Adaptive Web Document Classification with MCRDR. In: International Conference on Information Technology: Coding and Computing ITCC 2004, Orleans, Las Vegas, Nevada, USA (2004)
10. Park, S.S., Kim, Y.S., Kang, B.H.: Web Document Classification: Managing Context Change. In: IADIS International Conference WWW/Internet 2004, Madrid, Spain (2004)
11. Kim, Y.S., et al.: Incremental Knowledge Management of Web Community Groups on Web Portals. In: 5th International Conference on Practical Aspects of Knowledge Management, Vienna, Austria (2004)
12. Kim, Y.S., et al.: Knowledge Acquisition Behavior Anaysis in the Open-ended Document Classification. In: 19th ACS Australian Joint Conference on Artificial Intelligence, Hobart, Australia (2006)
13. Kang, B.-h., Kim, Y.S., Choi, Y.J.: Does multi-user document classification really help knowledge management? In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS, vol. 4830, pp. 327–336. Springer, Heidelberg (2007)
14. Brewington, B.E., Cybenko, G.: Keeping Up with the Changing Web. Computer 33(5), 52–58 (2000)

Lecture Notes in Computer Science 5692

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

Tommaso Di Noia Francesco Buccafurri (Eds.)

E-Commerce and Web Technologies

10th International Conference, EC-Web 2009
Linz, Austria, September 1-4, 2009
Proceedings

Springer

Volume Editors

Tommaso Di Noia
Politecnico di Bari, Dipartimento di Elettrotecnica ed Elettronica
Via E. Orabona 4, 70125 Bari, Italy
E-mail: t.dinoia@poliba.it

Francesco Buccafurri
University of Reggio Calabria, Department DIMET
Via Graziella, loc. Feo di Vito, 89122, Reggio Calabria, Italy
E-mail: bucca@unirc.it

Library of Congress Control Number: 2009932591

CR Subject Classification (1998): J.1, H.4, H.2, H.3, K.6.5, C.3, E.3

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

ISSN 0302-9743
ISBN-10 3-642-03963-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-03963-8 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12736850 06/3180 5 4 3 2 1 0

Preface

After the initial enthusiastic initiatives and investments and the eventual bubble, electronic commerce (EC) has changed and evolved into a well-established and founded reality both from a technological point of view and from a scientific one. Nevertheless, together with its evolution, new challenges and topics have emerged as well as new questions have been raised related to many aspects of EC. Keeping in mind the experience and the tradition of the past editions of EC-Web, we tried, for its 10th edition, to introduce some meaningful innovations about the structure and the scientific organization of the conference. Our main target was to highlight the autonomous role of the different (sometimes heterogeneous) aspects of EC, without missing their interdisciplinary scope. This required the conference to be organized into four "mini-conferences," each for a relevant area of EC and equipped with a corresponding Area Chair. Both the submission and the review process took into account the organization into four tracks, namely: "Service-Oriented E-Commerce and Business Processes," "Recommender Systems," "E-Payment, Security and Trust" and "Electronic Commerce and Web 3.0." Therefore, the focus of the conference was to cover aspects related to the theoretical foundation of EC, business processes as well as new approaches exploiting recently emerged technologies and scenarios such as the Semantic Web, Web services, SOA architectures, mobile and ubiquitous computing, just to cite a few. Due to their central role in any realistic EC infrastructure, security and privacy issues are widely considered, without excluding legal and regulatory aspects.

We received a broad spectrum of submissions and we are confident that the papers that were finally selected for publication and presentation will contribute to a better understanding of EC issues and possibilities in the Web 2.0 and Web 3.0 eras. We are grateful to all authors for their submissions. All papers were reviewed by at least three reviewers, either members of the Program Committee or external experts in the field. We received 61 papers and we accepted 20 of them for full oral presentation and 11 papers for short oral presentation. We received submissions from 26 countries (covering five continents), namely, Algeria, Australia, Austria, Brazil, Canada, Chile, China, Colombia, Croatia, Cyprus, Czech Republic, Denmark, France, Germany, Greece, Hungary, India, Iran, Italy, New Zealand, Romania, Serbia, Slovakia, South Korea, Spain, Taiwan, The Netherlands, Tunis, UK, USA and Vietnam.

Keynote talks further enriched EC-Web 2009. Edith Elkind gave the talk "Voting: A View Through the Algorithmic Lens" introducing recent developments in computational social choice and discussing the use of voting in practical applications. Martin Hepp in his talk "Product Variety, Consumer Preferences, and Web Technology: Can the Web of Data Reduce Price Competition and Increase Customer Satisfaction?" explained how to develop a Semantic Web enabled e-commerce application using the GoodRelations vocabulary.

We wish to thank Track Chairs Martin Hepp, Barbara Masucci, Giovanni Semeraro and Stefan Tai for their valuable contribution and support as well as all the PC members of each track and external reviewers. Our thanks also go to Roland Wagner and

to Gabriela Wagner for their great support in every single step of the organization. We do not forget Amin Anjomshoaa, who supported us with ConfDriver and fixed and changed the review system according to our needs. We are very grateful to them all.

September 2009

Francesco Buccafurri
Tommaso Di Noia

Organization

Program Chairs

Francesco Buccafurri	Università degli Studi Mediterranea di Reggio Calabria, Italy
Tommaso Di Noia	Politecnico di Bari, Italy

Track Chairs

Service Oriented E-Commerce and Business Processes
Stefan Tai, Karlsruhe University, Germany
Recommender Systems
Giovanni Semeraro, Università degli Studi di Bari, Italy
E-Payment, Security and Trust
Barbara Masucci, Università di Salerno, Italy
Electronic Commerce and Web 3.0
Martin Hepp, Bundeswehr University Munich, Germany

Program Committee

Service-Oriented E-Commerce and Business Processes

Marco Aiello	University of Groningen, The Netherlands
Christoph Bussler	Merced Systems, USA,
Schahram Dustdar	Vienna University of Technology, Austria
Holger Giese	HPI Potsdam, Germany,
Rania Khalaf	IBM Research, USA
Heiko Ludwig	IBM Research, USA
Ingo Melzer	Daimler Research, Germany
Christos Nikolaou	University of Crete, Greece
Thomas Sandholm	HP Labs, USA
York Sure	SAP Research, Germany
Vladimir Tosic	NICTA, Australia
Willem-Jan van den Heuvel	University of Tilburg, The Netherlands
Christian Zirpins	University of Karlsruhe, Germany

Recommender Systems

Gianbattista Amati	Fondazione Ugo Bordoni, Italy
Sarabjot Singh Anand	University of Warwick, UK
Liliana Ardissono	University of Turin, Italy
Giuliano Armano	University of Cagliari, Italy
Paolo Avesani	Fondazione Bruno Kessler, Italy

Pierpaolo Basile	University of Bari, Italy
Bettina Berendt	KU Leuven, Belgium
Shlomo Berkovsky	CSIRO, Australia
Robin Burke	De Paul University, USA
Rayid Ghani	Accenture Technology Labs, USA
Marco de Gemmis	Università degli Studi di Bari, Italy
Alexander Felfernig	University Klagenfurt, Austria
Michele Gorgoglione	Politecnico di Bari, Italy
Dietmar Jannach	Dortmund University of Technology, Germany
Pasquale Lops	Università degli sutdi di Bari, Italy
Bhaskar Mehta	Google Inc.
Stuart E. Middleton	University of Southampton, UK
Cosimo Palmisano	Fiat Group SpA, Italy
Michael Pazzani	Rutgers University, USA
Roberto Pirrone	University of Palermo, Italy
Francesco Ricci	Free University of Bozen-Bolzano, Italy
Shilad Sen	Macalester College, USA
Barry Smyth	University College Dublin, Ireland
Carlo Tasso	University of Udine, Italy
Eloisa Vargiu	University of Cagliari, Italy

E-Payment, Security and Trust

Anna Lisa Ferrara	University of Illinois at Urbana-Champaign, USA
Matthew Green	Johns Hopkins University, USA
Audun Jøsang	University of Oslo, Norway
Seny Kamara	Microsoft Research, USA
Gianluca Lax	Università Mediterranea di Reggio Calabria, Italy
Josè Maria Sierra	Universidad Carlos III de Madrid, Spain
Allan Tomlinson	University of London, UK

Electronic Commerce and Web 3.0

Hans Akkermans	Free University Amsterdam, The Netherlands
Alfredo Cuzzocrea	University of Calabria. Italy
Flavius Frasincar	Erasmus University Rotterdam, The Netherlands
Fausto Giunchiglia	University of Trento, Italy
Andreas Harth	DERI Galway, Ireland
Birgit Hofreiter	University of Vienna, Austria
Uzay Kaymak	Erasmus University Rotterdam, The Netherlands
Juhnyoung Lee	IBM Research, USA
Sang-goo Lee	Seoul National University, Korea
Andreas Radinger	Bundeswehr University Munich, Germany
Bernhard Schandl	University of Vienna, Austria
Gottfried Vossen	University of Münster, Germany
Peter Yim	CIM Engineering, Inc., USA

External Reviewers

Josè Francisco Aldana Montes
Claudio Baldassarre
Linas Baltrunas
Marco Brambilla
Steve Capell
Simona Colucci
Florian Daniel
Roberto De Virgilio
Eugenio Di Sciascio
Nicola Fanizzi
Fernando Ferri
Clemente Galdi
Christophe Guéret
Fedelucio Narducci
Cataldo Musto
Pasquale Pace
Azzurra Ragone
Davide Rossi
Michele Ruta
Jean-Claude Saghbini
Floriano Scioscia
Eufemia Tinelli
Alexander Totok

Table of Contents

Invited Talk

Design and Modelling of Enterprise and Distributed Systems

Electronic Commerce and Web 3.0

Collaboration-Based Approaches

Recommender Systems Modelling

Reputation and Fraud Detection

Recommender Systems and the Social Web

Recommender Systems in Action