LinkSelector: A Web Mining Approach to Hyperlink Selection for Web Portals

XIAO FANG
University of Toledo
and
OLIVIA R. LIU SHENG
University of Utah

As the size and complexity of Web sites expands dramatically, it has become increasingly challenging to design Web sites where Web surfers can easily find the information they seek. In this article, we address the design of the portal page of a Web site, which serves as the homepage of a Web site or a default Web portal. We define an important research problem—hyperlink selection: selecting from a large set of hyperlinks in a given Web site, a limited number of hyperlinks for inclusion in a portal page. The objective of hyperlink selection is to maximize the efficiency, effectiveness, and usage of a Web site's portal page. We propose a heuristic approach to hyperlink selection, LinkSelector, which is based on relationships among hyperlinks—structural relationships that can be extracted from an existing Web site and access relationships that can be discovered from a Web log. We compared the performance of LinkSelector with that of the current practice of hyperlink selection (i.e., manual hyperlink selection by domain experts), using data obtained from the University of Arizona Web site. Results showed that LinkSelector outperformed the current manual selection method.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*data mining*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms: Web Portal, Hyperlink Selection, Metrics, Algorithms

Additional Key Words and Phrases: Web mining

1. INTRODUCTION

As the size and complexity of Web sites expands dramatically, it has become increasingly challenging to design Web sites where Web surfers can easily find the information they seek. To address this challenge, we formally define a research problem in this area, hyperlink selection, and present a Web mining-based approach, LinkSelector, as a solution.

Authors' addresses: X. Fang, Department of Information, Operations and Technology Management, The University of Toledo, OH 43606; email: xiao.fang@utoledo.edu; O. R. Liu Sheng, School of Accounting and Information Systems, The University of Utah, UT 84112.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 1515 Broadway, New York, NY 10036 USA, fax: +1 (212) 869-0481, or permissions@acm.org.
© 2004 ACM 1533-5399/04/0500-0209 $5.00

There are two dominant ways that Web surfers find the information they seek [Chakrabarti 2000]: using search engines and clicking on hyperlinks. Research on the former is concerned with improving recall and precision [Lawrence and Giles 1998, 1999; Chakrabarti 2000] of search engines. Our research, however, concentrates on improving the efficiency of the second way of searching for Web information. As Web surfers click on a group of hyperlinks to find the information they seek, placing appropriate hyperlinks in Web pages is critical to improving their Web search efficiency. In particular, this article focuses on placing appropriate hyperlinks in the portal page of a Web site, the entrance to a Web site.

The homepage of a Web site is one type of portal page. Homepages that easily guide users to the information they seek create a good first impression and attract more users, while homepages that make the information search difficult, give a bad first impression that results in the loss of users [Nielson and Wagner 1996]. A default Web portal is another type of portal page. Recently, Web portals that serve as personal entrances to Web sites have attracted more and more attention. Universities such as UCLA have built educational Web portals (e.g., My UCLA, http://my.ucla.edu) and corporations such as Yahoo! have developed commercial Web portals (e.g., My Yahoo!, http://my.yahoo.com). For practical purposes, portal service providers (e.g., Yahoo!) provide portal users with a standard default Web portal which the users can personalize (e.g., add or remove hyperlinks from the default Web portal). As the first version of a Web portal encountered by portal users, the default Web portal plays an important role in the success of a Web portal. According to My Yahoo!, most users never customize their default Web portals and a great deal of effort should go into the design of the default Web portal [Manber et al. 2000].

A portal page consists of hyperlinks selected from a hyperlink pool which is a set of hyperlinks pointing to top-level Web pages.[1] Usually, the hyperlink pool of a Web site consists of hyperlinks listed in the site-index page or the site-directory page. As shown in Figure 1, hyperlinks in the portal page of the University of Arizona Web site (http://www.arizona.edu) are selected from its hyperlink pool. The hyperlink pool consists of hyperlinks in its site-index page (http://www.arizona.edu/index/webindex.shtml). Hyperlinks in the portal page of My Yahoo! (http://my.yahoo.com) are also selected from its hyperlink pool. The pool, in this case, consists of hyperlinks in its site-directory page.

Given the Web design principle that scrolling must be avoided in portal pages [Nielson 1999], a well-designed portal page normally contains several dozen (i.e., usually less than 4 dozen) hyperlinks.[2] However, the hyperlink pool of a typical Web site has at least several hundred hyperlinks. For example, the portal page of the University of Arizona Web site consists of 32 hyperlinks, while

[1]Web pages in a Web site are organized in a hierarchy in which a high-level Web page is an aggregation of its low-level Web pages [Nielson 1999]. For example, the Web page of faculty list is one level higher than its corresponding faculty homepages and it is an aggregation of its corresponding faculty homepages. For a university Web site, top-level Web pages include the Web page of department list and the Web page of computing resources, and so on.

[2]Placing too many hyperlinks in a portal page will cause some hyperlinks to be visible only when scrolling down the window of the page. Unfortunately, according to Neilson's research [Nielson 1999], Web surfers rarely scroll down the window of a portal page.

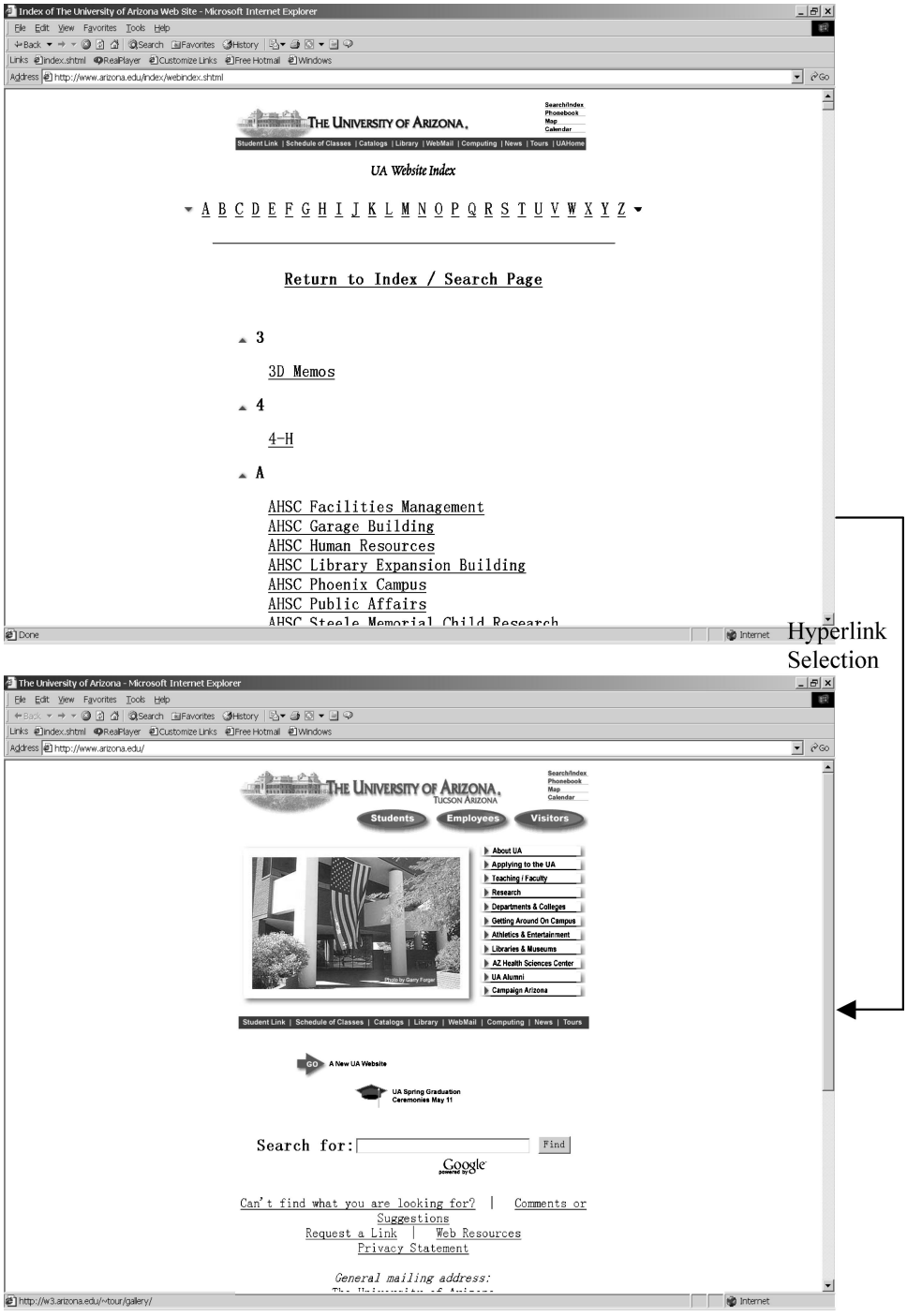

Fig. 1. (a) The hyperlink pool (top) and the portal page (bottom) of the Web site of the University of Arizona.

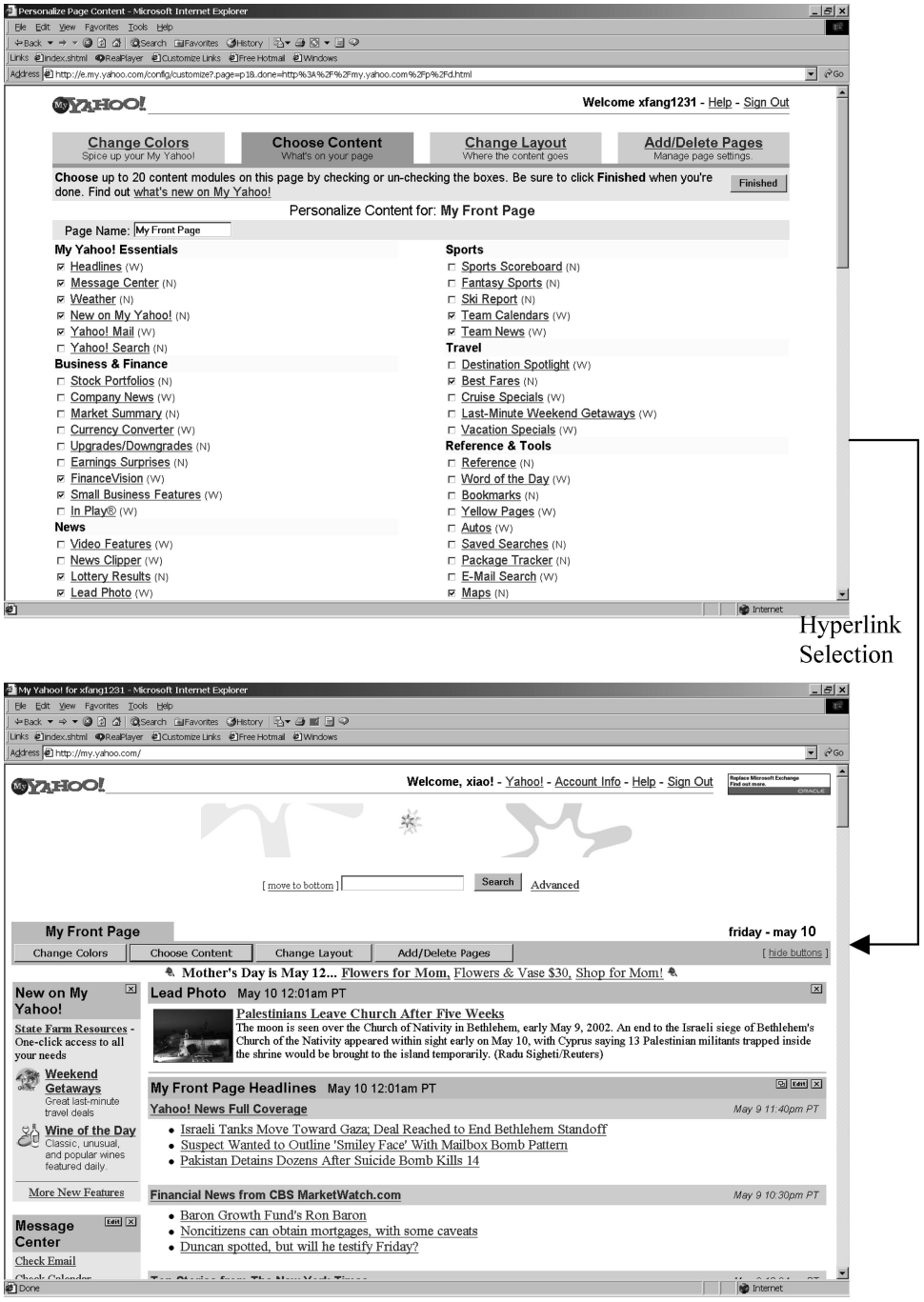

Fig. 1. (b) The hyperlink pool (top) and the portal page (bottom) of My Yahoo!

the hyperlink pool has 743 hyperlinks. It is computationally too expensive to exhaust all combinations of several dozen hyperlinks from a hyperlink pool with several hundred hyperlinks and find the one that is the most efficient in guiding Web surfers to the information they need. In this particular example, the number of combinations of selecting 32 hyperlinks from 743 hyperlinks is $1.44\mathrm{E}+56$ (i.e., C_{743}^{32}).

The current practice of hyperlink selection relies on the experience of domain experts (e.g., Web site designers). Obviously, such selection is subjective. In addition, it reflects only Web site designers' perspectives on what hyperlinks should be selected, not Web surfers' perspectives. The second perspective should be emphasized, as the purpose of hyperlink selection is to reduce Web surfers' information search efforts, not Web designers' efforts.

In comparison, our hyperlink selection method, LinkSelector, incorporates both patterns extracted from the structure of a Web site and those discovered from a Web log, which records Web surfers' information search behavior. LinkSelector first employs Web mining techniques [Kosala and Blockeel 2000] to extract the above-mentioned patterns and calculates preferences of hyperlinks and hyperlink sets defined in this article from these patterns. LinkSelector then selects hyperlinks from a given hyperlink pool by running the calculated preferences through a greedy algorithm developed in this article. There are two major contributions in this article.

—We have formally defined an important research problem—hyperlink selection.
—We have proposed and shown that a Web mining-based hyperlink selection approach outperforms other hyperlink selection approaches.

The rest of the article is organized as follows. We review related work in Section 2. In Section 3, we propose metrics to measure the quality of a portal page and formally define the hyperlink selection problem. A Web mining based-approach for hyperlink selection, LinkSelector, is presented in Section 4. In Section 5, we evaluate the performance of LinkSelector, using data obtained from the University of Arizona Web site and the metrics proposed in Section 3. We conclude the article in Section 6.

2. RELATED WORK

In this section, we first review works on Web mining on which LinkSelector is based. In Cooley et al. [1997], Web mining is defined as the process of discovering and analyzing useful information from the Web. A good survey on Web mining research can be found in Kosala and Blockeel [2000]. Srivastava et al. [2000] classified Web data into content, structure, and usage:

—Content is the data in Web pages. It usually consists of text and graphics.
—Structure is the data describing the organization of the Web, such as hyperlinks.
—Usage is the data that describe Web surfers' information search behavior. Web usage data can be found in Web logs.

IP Address[3]	Time	Method/URL/Protocol	Status	Size
128.196.133.16	[01/Sep/2001:05:38:33 -0700]	"GET /working/index.shtml HTTP/1.0"	200	7134
128.196.133.16	[01/Sep/2001:05:38:34 -0700]	"GET /working/images/head-employ.gif HTTP/1.0"	200	765
128.196.133.16	[01/Sep/2001:05:38:34 -0700]	"GET /working/images/work.jpg HTTP/1.0"	200	8864
128.196.133.16	[01/Sep/2001:05:38:34 -0700]	"GET /working/images/staff-quicklinks1.gif HTTP/1.0"	200	1618

Fig. 2. A sample Web log collected by a Web server at the University of Arizona.

For different types of Web data, corresponding Web mining methods are developed. Web content mining is the process of automatically retrieving, filtering, and categorizing Web documents, a good survey can be found in Chakrabarti [2000]. As it typically makes use of only texts on Web pages, valuable information implicitly contained in hyperlinks is overlooked. Web structure mining [Chakrabarti et al. 1999] infers useful patterns from the Web's link topology to help retrieve high quality Web pages. HITS and PageRank are two widely used Web structure mining algorithms. In the HITS algorithm [Kleinberg 1998], authority pages were defined as high quality pages related to a particular topic. Hub pages were those that were not necessarily authorities themselves, but provided pointers to other authority pages. HITS calculated hub weights and authority weights of Web pages using an iterative procedure and returned Web pages with the highest hub weights or authority weights. The PageRank algorithm [Brin and Page 1998] weighted each in-link to a page proportionally to the quality of the page containing the in-link. A Web page had a high PageRank score if it was linked from many other pages, and the scores were even higher if these referring pages were also good pages. The hyperlink vector voting algorithm [Li 1998] ranked Web pages based on both the number of hyperlinks to it and the anchor texts of the hyperlinks. Web usage mining [Srivastava et al. 2000] is the process of applying data mining techniques to discover Web access patterns from a Web log. Due to its relevance to this research, we review works on Web usage mining in more detail.

The data used for Web usage mining are Web logs. A Web log is a collection of data that explicitly records Web surfers' information search behavior in a Web site. Figure 2 shows a sample Web log collected by a Web server at the University of Arizona. Useful attributes for Web usage mining in a Web log include IP address, time, and URL, which explicitly describe who, at what time, accessed which Web page. Additional attributes include status of a HTTP request and the count of bytes returned by a Web server.

Projects of Web usage mining can be classified into two groups: general-purpose projects and specific-purpose projects [Srivastava et al. 2000]. General-purpose projects, such as Chen et al. [1996] and Cooley et al. [1999a], focused

[3]To protect the privacy of Web users, IP addresses in Figure 2 are artificial IP addresses.

on Web usage mining in general. Cooley et al. proposed an architecture and specific steps for Web usage mining, and presented a method to identify potentially interesting patterns from mining results (e.g., patterns in which unlinked Web pages are visited together frequently). Chen et al. explored a new data mining capability to mine path traversal patterns from Web logs. Specific-purpose projects focused on applications of Web usage mining. Web usage mining can be used to improve organizations of Web sites. Adaptive Web site project [Perkowitz and Etzioni 2000] used Web visiting patterns, learned from Web logs, to automatically improve the organization and presentation of Web sites. Spiliopoulou and Pohle [2001] exploited Web usage mining to measure and improve the success of Web sites. Lee and Podlaseck [2001] presented an interactive visualization system that provides users with the ability to actively interpret and explore Web log data of online stores in order to evaluate the effectiveness of Web merchandising. Web usage mining can also be used to personalize users' Web surfing experience. In Yan et al. [1996], clusters of visitors who exhibited similar information needs (e.g., visitors who accessed similar Web pages) were discovered via Web usage mining. These clusters could be used to classify new visitors and dynamically suggest hyperlinks for them. Mobasher et al. [2001] presented techniques to learn user preferences from Web usage data using data mining techniques, such as association rule mining. Based on the learned preferences, dynamic hyperlinks could be recommended for active visiting sessions. Anderson et al. [2001] developed MINPATH, an algorithm that automatically suggests useful shortcut links in realtime to improve wireless Web navigations. A complete survey of Web usage mining research by year 2000 can be found in Srivastava et al. [2000].

Research on browsing agents and recommendation systems is also related to this work. WebWatcher [Armstrong et al. 1995] and Letizia [Lieberman 1995] are two early examples of browsing agents. WebWatcher first asked a user's Web visiting goal. It then predicted and highlighted hyperlinks that the user would click based on models learned from previous Web visiting behavior. Inspired by WebWatcher, personal WebWatcher [Mladenic 1999] modeled Web visiting interests for a specific Web surfer based on her/his previously visited Web pages. Another Web browsing agent, Letizia learned a user's interests from the user's past behavior. It explored the Web ahead of the user and used the interests learned to recommend Web pages that the user could visit next. Content-based filtering and collaborative filtering are approaches to realizing recommendation systems. The content-based filtering approach has its root in information retrieval [Salton 1968]. Recommendation systems using this approach, such as NewsWeeder [Lang 1995], recommended objects to a user based on the comparison between the contents of the objects and the user's profile. Recommendation systems using the collaborative filtering approach, such as Ringo [Shardanand and Maes 1995] and GroupLens [Resnick 1994], recommended objects to a user because other users with similar tastes liked these objects. The Fab system described in Balabanovic and Shoham [1997] is based on the combination of the content-base filtering approach and the collaborative filtering approach.

Research employing only usage information, such as the work described in Chen el al. [1996], and research employing content information, such as the

work described in Armstrong et al. [1995] and Lang [1995], did not consider the information contained in the structure of a Web site. In Cooley [1999a] and Perkowitz and Etzioni [2000], the structure of a Web site was used to filter out uninteresting Web visiting patterns (i.e., patterns in which directly linked Web pages are visited together frequently). Although it is proper to exclude these uninteresting patterns in Cooley [1999a] and Perkowitz and Etzioni [2000], these excluded, uninteresting patterns provide valuable insights into hyperlink selection. We will discuss this in Section 4.1. The hyperlink selection approach proposed in this article is based on both Web usage information and Web structure information, and considers both interesting and uninteresting Web visiting patterns. Research in Web structure mining that combines Web content and Web structure information is also related to this research. For example, HITS favors Web pages (i.e., hub pages) that point to many other good pages (i.e., authority pages). LinkSelector also favors Web pages that point to many other good pages. Unlike HITS, good pages in LinkSelector are measured using Web usage information (i.e., the frequency that the pages are visited together). Besides structurally related Web pages, LinkSelector also considers structurally unrelated Web pages.

3. PROBLEM DEFINITION—HYPERLINK SELECTION

It is desirable to design a portal page of a Web site that: (a) can effectively facilitate Web surfers in locating information stored in the Web site, (b) has a limited number of hyperlinks placed in the portal page, and (c) is frequently used/visited by Web surfers when they search for information in the Web site. We propose three metrics to measure the quality of a portal page—effectiveness, efficiency, and usage, which address (a), (b), and (c) above. All three are calculated from Web logs. A Web log can be broken down into sessions with each session representing a sequence of consecutive Web accesses by the same visitor. For the convenience of readers, important notations used in this article are summarized in Table I.

The effectiveness of a portal page is measured as the degree of ease in finding user-sought, top-level Web pages[4] from the portal page. We denote s as the number of sessions in a Web log, S_j as a session, for $j = 1, 2, \Lambda\ s$, and H as a hyperlink pool, which is a set of hyperlinks pointing to top-level Web pages. In S_j, user-sought, top-level Web pages are Web pages pointed to by hyperlinks in UHL(S_j), which is the intersection of the hyperlinks clicked in S_j and H. Usually, Web pages that are 1–2 clicks away[5] from a portal page can be easily found from the portal page. We denote l as the number of hyperlinks in a Web

[4]We believe that a levelwise approach is appropriate for the design of a Web site. In this approach, the portal page is designed to find user-sought, top-level Web pages easily. Top-level Web pages are then designed to easily locate user-sought Web pages one level below. As the hyperlink selection approach proposed in this article can be applied to every subsequent level, consequently, Web sites designed from this approach will be easy to navigate, to find user-sought information. In this article, we concentrate on designing the portal page to facilitate the search of top-level Web pages.

[5]Web pages that are 1-click away from a portal page, refer to Web pages that are directly pointed to by hyperlinks in the portal page. Web pages that are 2-clicks away from a portal page, are Web pages that are pointed to by hyperlinks in Web pages 1-click away.

Table I. Notation Summary

Notation	Description
s	the number of sessions in a Web log
S_j	a session, for $j = 1, 2, \ldots, s$
H	the hyperlink pool of a Web site
$UHL(S_j)$	user-sought top-level Web pages in S_j
l	the number of hyperlinks in a Web site
L_j	a hyperlink, for $j = 1, 2, \ldots, l$
P_{L_j}	the set of hyperlinks in the Web page pointed to by L_j
PHL	the set of hyperlinks in a portal page
EHL	$EHL = \bigcup_{\forall L_j \in PHL} P_{L_j}$
HL	$HL = PHL \cup EHL$
N	the number of hyperlinks to be placed in a portal page
$effectiveness(S_j)$	$effectiveness(S_j) = \frac{\lvert UHL(S_j) \cap HL \rvert}{\lvert UHL(S_j) \rvert}$
$effectiveness(\log)$	$effectiveness(\log) = \frac{\sum_{j=1}^{s} effectiveness(S_j)}{s}$
$efficiency(S_j)$	$efficiency(S_j) = \frac{\lvert UHL(S_j) \cap HL \rvert}{\lvert PHL \rvert}$
$efficiency(\log)$	$efficiency(\log) = \frac{\sum_{j=1}^{s} efficiency(S_j)}{s}$
$usage(\log)$	$usage(\log) = \sum_{j=1}^{s} \lvert UHL(S_j) \cap HL \rvert$
k-HS	a set of k hyperlinks L_i, where $L_i \in H$
σ(k-HS)	the support of a k-HS
SR	the set of structure relationships between hyperlinks in H
AR	the set of access relationships among hyperlinks in H
PRE_{L_i}	the preference of a hyperlink L_i, where $L_i \in H$
PHP	a set of pairs, in which, each pair consists of a hyperlink pair with Group II relationship and its preference
PHS	a set of pairs, in which, each pair consists of a hyperlink set with Group II relationship and its preference

site, L_j as a hyperlink, for $j = 1, 2, \ldots, l$, P_{L_j} as the set of hyperlinks in the Web page pointed to by L_j and PHL as the set of hyperlinks in the portal page of a Web site. EHL is the set of hyperlinks, where

$$EHL = \bigcup_{\forall L_j \in PHL} P_{L_j}. \tag{1}$$

EHL consists of hyperlinks that are contained in Web pages directly pointed to by the portal page. Web pages pointed to by hyperlinks in HL are 1–2 clicks away from the portal page and can be easily found from the portal page, where

$$HL = PHL \cup EHL. \tag{2}$$

Example 1. As shown in Figure 3, the portal page of a Web site contains hyperlinks L_1, L_2, and L_3. Web pages pointed to by hyperlinks L_1, L_2, and L_3 contain hyperlinks L_2, and L_4, hyperlink L_3, and hyperlinks L_5, and L_8, respectively.

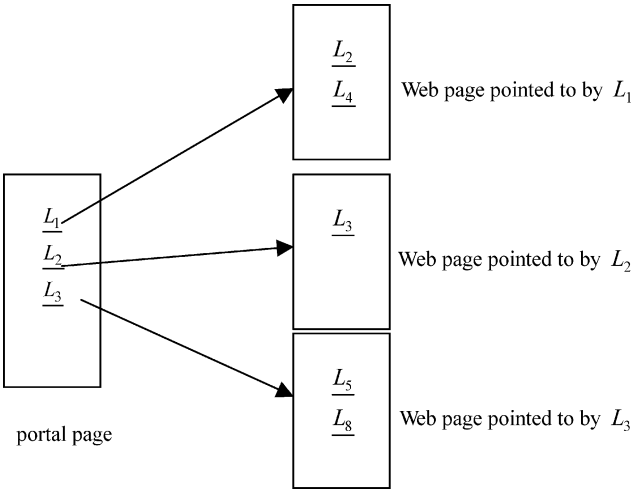

Fig. 3. *PHL* and *EHL*.

In this example, $PHL = \{L_1, L_2, L_3\}$, $P_{L_1} = \{L_2, L_4\}$, $P_{L_2} = \{L_3\}$, and $P_{L_3} = \{L_5, L_8\}$. According to (1), $EHL = \cup_{\forall L_j \in PHL} P_{L_j} = (P_{L_1} \cup P_{L_2} \cup P_{L_3}) = \{L_2, L_3, L_4, L_5, L_8\}$; according to (2), $HL = PHL \cup EHL = \{L_1, L_2, L_3, L_4, L_5, L_8\}$.

The effectiveness of a portal page can be measured in terms of the recall rate of the portal page at two different levels—session level and Web log level. For a session S_j, the more hyperlinks in UHL(S_j) found in HL, the more user-sought, top-level Web pages are easily found from the portal page; hence, the higher the effectiveness of the portal page.

Definition 1. For a session S_j, the session level effectiveness of a portal page is defined as:

$$effectiveness(S_j) = \frac{\left|UHL(S_j) \cap HL\right|}{\left|UHL(S_j)\right|}, \tag{3}$$

where $j = 1, 2, \ldots, s$, and $|X|$ denotes the cardinality of a set X. And, the log level effectiveness of a portal page is defined as:

$$effectiveness(\log) = \frac{\sum_{j=1}^{s} effectiveness(S_j)}{s}. \tag{4}$$

Example 2. In a session S_1, hyperlinks $L_1, L_{10}, L_{11}, L_2, L_{13}, L_{14}, L_5, L_9, L_7$, and L_{12} were clicked. Hyperlinks L_1, L_2, L_5, and L_7 are also elements of the hyperlink pool H. Hence, UHL(S_1) is $\{L_1, L_2, L_5, L_7\}$. For the portal page given in Example 1, its session level effectiveness is,

$$effectiveness(S_1) = \frac{|UHL(S_1) \cap HL|}{|UHL(S_1)|} = \frac{|\{L_1, L_2, L_5\}|}{|\{L_1, L_2, L_5, L_7\}|} = \frac{3}{4} = 0.75.$$

The result states that 75% of the user-sought, top-level Web pages in session S_1 can be easily found from the portal page.

Given the limited number of hyperlinks that can be placed in a portal page, it is desirable to have more user-sought, top-level Web pages easily found from the portal page (i.e., more hyperlinks in $UHL(S_j) \cap HL$) with fewer hyperlinks placed in the portal page (i.e., fewer hyperlinks in PHL). Motivated by this consideration, we define the efficiency of a portal page as the following.

Definition 2. For a session S_j, the session level efficiency of a portal page is defined as:

$$efficiency(S_j) = \frac{\left|UHL(S_j) \cap HL\right|}{\left|PHL\right|}, \tag{5}$$

where $j = 1, 2, \ldots, s$, and $|X|$ denotes the cardinality of a set X. And the log level efficiency of a portal page is defined as:

$$efficiency(\log) = \frac{\sum_{j=1}^{s} efficiency(S_j)}{s}. \tag{6}$$

Example 3. For the portal page given in Example 1, and the session S_1 given in Example 2, the session level efficiency of the portal page is,

$$efficiency(S_1) = \frac{\left|UHL(S_1) \cap HL\right|}{\left|PHL\right|} = \frac{|\{L_1, L_2, L_5\}|}{|\{L_1, L_2, L_3\}|} = \frac{3}{3} = 1.$$

A well designed portal page should attract Web surfers to use/visit it. Usage of a portal page measures how often a portal page is visited. As the portal page constructed by LinkSelector has not been used by Web surfers, we measure its usage by counting the number of user-sought, top-level Web pages that can be easily found from the portal page (i.e., the number of hyperlinks in $UHL(S_j) \cap HL$). The measurement is an approximation of usage because ease of finding these Web pages from the portal page will attract users to visit the portal page. We define usage measured at the Web log level as follows.

Definition 3. The log level usage of a portal page is defined as:

$$usage(\log) = \sum_{j=1}^{s} \left|UHL(S_j) \cap HL\right|, \tag{7}$$

where $|X|$ denotes the cardinality of a set X.

Based on the three metrics presented above, we formally define the hyperlink selection problem.

Given: (1) the hyperlink pool H of a Web site;
(2) the number of hyperlinks to be placed in a portal page of the Web site—N, where $N < |H|$.

Select N hyperlinks from the hyperlink pool H to include in a portal page.
Objective: maximize the effectiveness, efficiency, and usage of the resulting portal page.

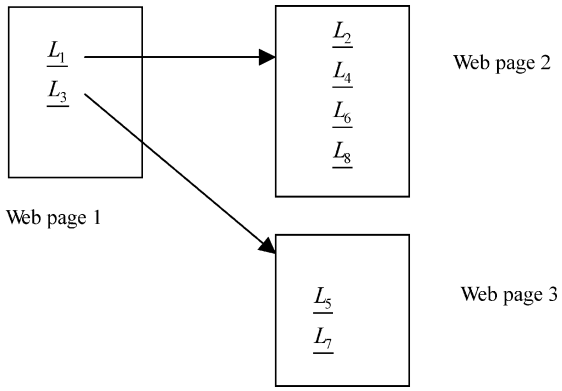

Fig. 4. Structure relationship.

4. THE LINKSELECTOR APPROACH

As discussed in Section 1, it is computationally too expensive to find the optimal solution for the hyperlink selection problem. In this section, we present a heuristic solution method named LinkSelector. Compared with the current practice of hyperlink selection based on the experience domain experts, our method incorporates patterns extracted from the structure of a Web site and those extracted from a Web log, which record Web surfers' information search behavior in the Web site. Hence, LinkSelector is more objective and reflects the Web surfers' perspectives on which hyperlinks should be selected, while the current practice of hyperlink selection is subjective and reflects only the domain experts' perspectives. We introduce LinkSelector in Section 4.1. A detailed description of LinkSelector is illustrated step-by-step in Section 4.2 through Section 4.6. We discuss the time complexity of LinkSelector in Section 4.7.

4.1 Overview of LinkSelector

Hyperlinks in a hyperlink pool may be directly connected to each other (i.e., one hyperlink is contained in a Web page pointed to by another hyperlink), or accessed together in a session. Accordingly, we categorize relationships among hyperlinks in a hyperlink pool into two types—structure relationship and access relationship. For any pair of hyperlinks L_i and L_j in H, L_i has a structure relationship with L_j, denoted as $L_i \rightarrow L_j$, if and only if $L_j \in P_{L_i}$ (i.e., L_j is a hyperlink on the page pointed to by L_i).L_i is the initial hyperlink and L_j is the terminal hyperlink in this structure relationship.

Example 4. As shown in Figure 4, Web page 1 contains hyperlinks $L_1$and L_3; Web page 2 , which is pointed to by hyperlink L_1, contains hyperlinks L_2, L_4, L_6, and L_8; and Web page 3 , which is pointed to by hyperlink L_3, contains hyperlink L_5 and L_7. All hyperlinks are elements of the hyperlink pool H. In this example, $P_{L_1} = \{L_2, L_4, L_6, L_8\}$ and $P_{L_3} = \{L_5, L_7\}$. Structure relationship $L_1 \rightarrow L_2$ exists, in which $L_1$is the initial hyperlink and L_2 is the terminal hyperlink. Similarly, structure relationships $L_1 \rightarrow L_4, L_1 \rightarrow L_6, L_1 \rightarrow L_8, L_3 \rightarrow L_5$, and $L_3 \rightarrow L_7$ also hold.

Structure Relationship / Access Relationship	YES	NO
YES	I	II
NO	III	IV

Fig. 5. Pair-wise relationships for hyperlinks in a hyperlink pool.

For a hyperlink that acts as the initial hyperlink in M structure relationships, placing it in a portal page makes it an element of PHL, and the M terminal hyperlinks in all the structure relationships elements of EHL. As discussed in Section 3, Web pages pointed to by hyperlinks in HL (i.e., $PHL \cup EHL$) can be easily found from a portal page. Therefore, the more structure relationships a hyperlink participates in as the initial hyperlink, the more top-level Web pages can be easily found from a portal page, if the hyperlink is placed in the portal page. In Example 4, L_1 participates in four structure relationships (i.e., $L_1 \rightarrow L_2, L_1 \rightarrow L_4, L_1 \rightarrow L_6$, and $L_1 \rightarrow L_8$) as the initial hyperlink. Hence, placing L_1 in a portal page enables five top-level Web pages (i.e., pages pointed to by L_1, L_2, L_4, L_6 and L_8) to be easily found from the portal page.

We use access relationships to reflect the Web visiting pattern that some hyperlinks are visited together in a session. In a Web log, a session is identified using an IP address and a timeout (see Section 4.3.1). The LinkSelector approach is based on both access relationships and structure relationships. Before introducing access relationships, we introduce hyperlink sets first. A hyperlink set k-HS is a set of k hyperlinks L_i, where $L_i \in H$. For a Web log, the support of a k-HS (denoted as σ (k-HS)) is the ratio of the number of sessions in which the k-HS is accessed over the total number of sessions in the Web log. For a k-HS , where $k \geq 2$, there exists an access relationship among elements in the k-HS if and only if its support, σ (k-HS), is greater than a predefined threshold. σ (k-HS) is called the support of the access relationship.

For hyperlinks in a hyperlink pool, their pair-wise relationships can be categorized into four groups as shown in Figure 5.

Group I relationships indicate that both a structure relationship and an access relationship hold between two hyperlinks. As we have discussed, hyperlinks participating in more structure relationships as initial hyperlinks will be selected for the portal page over other hyperlinks, with respect to increasing the number of hyperlinks in HL. For a structure relationship $L_i \rightarrow L_j$, the support of the access relationship between L_i and L_j $-$ $\sigma(\{L_i, L_j\})$, reflects the quality of the structure relationship. The higher the support $\sigma(\{L_i, L_j\})$, the higher the possibility that L_i and L_j will be accessed together in future visits.[6] Hence, hyperlinks participating in higher quality structure relationships as initial hyperlinks will be selected for the portal page over other hyperlinks, with respect to increasing the quality (i.e., possibility to be visited) of hyperlinks in HL. Increasing the number and the quality of hyperlinks in HL could increase the hits of user-sought, top-level Web pages in HL (i.e., $|UHL(S_j) \cap HL|$), which, in turn,

[6]This is based on an assumption that Web visiting patterns are coherent in past and future visits [Perkowitz and Etzioni 2000].

could increase the effectiveness, efficiency, and usage of a portal page, according to Equations (3), (5), and (7). In this regard, a Group I relationship that was regarded as uninteresting in previous research [Perkowitz and Etzioni 2000; Cooley 1999a], provides us with two indicators of hyperlink preference in hyperlink selection: the number of structure relationships a hyperlink participates in as the initial hyperlink, and the qualities of these structure relationships, measured as supports of access relationships between the hyperlink and its terminal hyperlinks. We describe how to calculate the preference of a hyperlink using these two indicators in Section 4.4.

Group II relationships indicate that an access relationship, but not a structure relationship, exists between two hyperlinks. As hyperlinks with a Group II relationship are structurally unlinked, in order to navigate from the Web page pointed to by one hyperlink to the Web page pointed to by the other, Web visitors have to explore the Web site to find the path. This creates inconvenience for Web surfing, and the situation becomes worse when two hyperlinks are accessed together frequently (i.e., access relationship between the hyperlinks). In contrast, if these two hyperlinks were placed together in a portal page, they could easily be found from the portal page, which could increase the hits of user-sought, top-level Web pages in HL (i.e., $|UHL(S_j) \cap HL|$), and the effectiveness, efficiency, and usage of a portal page (i.e., according to Equations (3), (5) and (7)). Hence, hyperlink pairs with a Group II relationship are preferred to hyperlink pairs without such a relationship in hyperlink selection. For a hyperlink pair with a Group II relationship, the support of the access relationship between its hyperlinks is the determining factor for its preference over other hyperlink pairs with a Group II relationship in hyperlink selection. We describe how to calculate the preference of a hyperlink pair for a Group II relationship in Section 4.5.

Group III relationships indicate that a structure relationship, but not an access relationship, exists between two hyperlinks. This relationship reveals that the Web page pointed to by the initial hyperlink in a structure relationship contains a rarely-visited hyperlink, which is the terminal hyperlink, in the structure relationship. Hence, a Group III relationship reveals a design problem with internal pages. As hyperlink selection focuses on choosing hyperlinks for a portal page; we do not discuss Group III relationships in this article.

Group IV relationships do not reveal any patterns between hyperlinks; thus, they are not considered in this research.

LinkSelector employs Group I and Group II relationships to calculate preferences of hyperlinks and preferences of hyperlink pairs, respectively. Based on preferences calculated, an algorithm is developed to extract N hyperlinks from the given hyperlink pool. We outline LinkSelector in Figure 6. Steps in this algorithm are described in Section 4.2 through Section 4.6.

4.2 Discovering Structure Relationships

LinkSelector discovers structure relationships between hyperlinks in a hyperlink pool by parsing the Web pages the hyperlinks point to. The Web page pointed to by each hyperlink L_i in H, is retrieved and parsed. A structure

Input: H : the hyperlink pool of a website
N : the number of hyperlinks to be placed in the portal page of the website
Output: PHL: the set of hyperlinks in the portal page of the website
Discover structure relationships;
Discover access relationships;
Calculate preferences of hyperlinks based on group I relationship;
Calculate preferences of hyperlink pairs based on group II relationship;
Select hyperlinks.

Fig. 6. The sketch of LinkSelector.

relationship, $L_i \rightarrow L_j$, is added to structure relationships SR, if L_j appears in the Web page L_i points to, L_j and L_i are different, and L_j is an element of H.

4.3 Discovering Access Relationships

4.3.1 *Web Log Preprocessing.* Access relationships among hyperlinks can be extracted from a Web log. A raw Web log, collected from a Web server, needs to be preprocessed before meaningful access relationships can be extracted [Cooley et al. 1999b]. In this research, we divide the preprocessing task into two steps—Web log cleaning and session identification.

In Web log cleaning, two types of Web log records are removed. First, Web log records with the value of the status attribute greater than 400 are deleted because they record failed Web access. Second, Web log records recording accessory requests to a Web page request, such as picture requests, are also removed. As a raw Web log records every file request sent to a Web server, one Web access could result in several Web log records. For example, an access to a Web page with two pictures will result in three Web log records: one for the Web page, and the other two for the pictures. Web log records recording accesses to Web pages are sufficient to describe the Web surfers' information search behaviors.

The basic processing unit for extracting access relationships is a session. A Web log needs to be divided into sessions before the extraction of access relationships. However, in HTTP protocol, as connections between Web clients and a Web server are stateless, there is no notion of session in a Web log. One method of dividing a Web log into sessions is based on timeout. If the time between page requests from the same user exceeds a certain limit, it is assumed that the user has started another session [Cooley et al. 1999b]. Some commercial tools use 30 minutes as a default timeout. Catledge and Pitkow [1995] calculated a timeout of 25.5 minutes based on empirical data. Another method is to modify a Web server to encode session identifiers in Web pages, transferred between clients and a Web server [Yan et al. 1996]. As the second method requires modification of a Web server, it is not convenient and practical for many Web sites. We adopt the timeout method to identify sessions. Web log records are first sorted by client IP address, then by access time. We use a 30-minute timeout to divide Web log records, generated by the same IP address, into sessions.

4.3.2 *Mine Access Relationships From The Preprocessed Web Log.* Once a raw Web log has been cleaned and divided into sessions, large itemset discovery algorithms, proposed in association rule mining, can be applied to extract access relationships from it. Association rule mining [Agrawal and Srikant 1994] is defined on a set of items $L = \{i_1, i_2, \Lambda, i_k\}$. Let D be a set of transactions, where each transaction T is a set of items, such that $T \subseteq L$. The support of an itemset (i.e., a set of items) in D is the fraction of all transactions containing the itemset. An itemset is called large if its support is greater than a user-specified support threshold. The most important step in association rule mining is to find large itemsets and their supports. Various algorithms for finding large itemsets, such as Apriori [Agrawal and Srikant 1994], are in use.

In the case of mining access relationships, sessions correspond to transactions, hyperlinks correspond to items, and hyperlink sets correspond to itemsets. Applying the Apriori algorithm on the preprocessed Web log, all hyperlink sets that have access relationships among their elements can be found, and their corresponding supports can be calculated. The Apriori algorithm can be found in Agrawal and Srikant [1994] and is skipped in this article. The output of mining access relationships is a set of pairs, denoted as AR, in which each pair contains a hyperlink set and its corresponding support.

4.4 Calculating Preferences of Hyperlinks

As discussed in Section 4.1, with respect to increasing the effectiveness, efficiency, and usage of a portal page, the preference of a hyperlink in hyperlink selection is determined by two factors: the number of structure relationships it participates in as the initial hyperlink, and the qualities of these structure relationships, measured as supports of access relationships between it and its terminal hyperlinks. Based on these two factors, we define the preference of a hyperlink as below.

Definition 4. For a hyperlink L_i, where $L_i \in H$, and L_i participates in M structure relationships as the initial hyperlink, $(L_i \rightarrow L_{j_m}) \in SR$, for $m = 1, 2, \Lambda\, M$, the preference of the hyperlink L_i is defined as:

$$PRE_{L_i} = \sum_{m=1}^{M} (\sigma(\{L_i, L_{j_m}\})^* coeff) \tag{8}$$

$$\text{where} \quad coeff = \begin{cases} 1 & \text{if} \quad (\{L_i, L_{j_m}\}, \sigma(\{L_i, L_{j_m}\})) \in AR \\ 0 & \text{otherwise.} \end{cases}$$

Example 5. Given a hyperlink pool $H = \{L_1, L_2, L_3, L_4, L_5, L_6, L_7\}$, structure relationships SR, the minimum threshold for access relationship -0.002, and access relationships AR,

$$SR = \{L_1 \rightarrow L_2, L_1 \rightarrow L_3, L_1 \rightarrow L_4, L_2 \rightarrow L_5, L_3 \rightarrow L_4, L_3 \rightarrow L_6, L_4 \rightarrow L_5, L_5 \rightarrow L_6, L_5 \rightarrow L_7, L_6 \rightarrow L_3, L_7 \rightarrow L_3, L_7 \rightarrow L_4\}$$

$$\begin{aligned} AR = \{&(\{L_1, L_2\}, 0.022), (\{L_1, L_3\}, 0.018), (\{L_1, L_4\}, 0.01), (\{L_1, L_5\}, 0.002),\\ &(\{L_1, L_6\}, 0.014), (\{L_1, L_7\}, 0.014), (\{L_2, L_5\}, 0.007), (\{L_2, L_6\}, 0.006),\\ &(\{L_2, L_7\}, 0.008), (\{L_3, L_6\}, 0.008), (\{L_3, L_7\}, 0.012), (\{L_4, L_5\}, 0.01),\\ &(\{L_5, L_6\}, 0.03), (\{L_6, L_7\}, 0.005), (\{L_1, L_6, L_7\}, 0.003)\}. \end{aligned}$$

Hyperlink L_1 participates in three structure relationships as the initial hyperlink. According to Equation (8),

$$\begin{aligned} PRE_{L_1} &= \sum_{m=1}^{3} (\sigma(\{L_1, L_{j_m}\})^* coeff) = \sigma(\{L_1, L_2\})^*1 + \sigma(\{L_1, L_3\})^*1 + \sigma(\{L_1, L_4\})^*1 \\ &= 0.022 + 0.018 + 0.01 = 0.05 \end{aligned}$$

Similarly, we get,

$$\begin{aligned} PRE_{L_2} &= 0.007, PRE_{L_3} = 0.008, PRE_{L_4} = 0.01, PRE_{L_5} = 0.03, \\ PRE_{L_6} &= 0.008 \text{and } PRE_{L_7} = 0.012. \end{aligned}$$

4.5 Calculating Preferences of Hyperlink Pairs

As discussed in Section 4.1, with respect to increasing the effectiveness, efficiency, and usage of a portal page, hyperlink pairs with a Group II relationship are preferred to hyperlink pairs without it in hyperlink selection. For a hyperlink pair with a Group II relationship, the support of the access relationship between its hyperlinks is the determining factor for its preference over other hyperlink pairs with a Group II relationship in hyperlink selection. We set preferences of hyperlink pairs without a Group II relationships to be 0, and those of hyperlink pairs with Group II relationships to be the supports of the access relationships between its hyperlinks.

Both hyperlink pairs with Group II relationships and their preferences can be extracted from access relationships AR. For a 2-HS (i.e., a hyperlink set with two hyperlinks as elements) in AR, if there exists no structure relationship between its elements, then the 2-HS is a hyperlink pair with a Group II relationship, and its support σ (2-HS) is the preference of the hyperlink pair. We denote PHP as a set of pairs in which each pair consists of a hyperlink pair with a Group II relationship, and the preference of the hyperlink pair.

Example 6. For a 2-HS $\{L_1, L_5\}$ in access relationships AR, given in Example 5, since there is no structure relationship between hyperlinks L_1 and L_5, $\{L_1, L_5\}$ and its support become an element of PHP, shown below. Preferences of hyperlink pairs outside PHP are set to be 0.

$$\begin{aligned} PHP = \{&(\{L_1, L_5\}, 0.002), (\{L_1, L_6\}, 0.014), (\{L_1, L_7\}, 0.014), (\{L_2, L_6\}, 0.006),\\ &(\{L_2, L_7\}, 0.008), (\{L_6, L_7\}, 0.005)\}. \end{aligned}$$

LinkSelector considers not only hyperlink sets with 2 elements (i.e., hyperlink pairs) that have a Group II relationship, but also hyperlink sets with more than 2 elements that have a Group II relationship. Discussions on hyperlink pairs with Group II relationships can be extended to include hyperlink sets with more than 2 elements.

Definition 5. A hyperlink set k-HS, where $k \geq 2$, is a hyperlink set with a Group II relationship if and only if ,

—there exists an access relationship among its hyperlinks, and
—there is no structure relationship between any pair of its hyperlinks.

Similarly, we set preferences of hyperlink sets without a Group II relationship to be 0. And, the preference of a hyperlink set with a Group II relationship is set to be the support of the access relationship among its hyperlinks. We denote PHS as a set of pairs in which each pair consists of a hyperlink set with a Group II relationship, and the preference of the hyperlink set.

Example 7. PHS extracted from access relationships AR in Example 5 is,

$$PHS = \{(\{L_1, L_5\}, 0.002), (\{L_1, L_6\}, 0.014), (\{L_1, L_7\}, 0.014), (\{L_2, L_6\}, 0.006), (\{L_2, L_7\}, 0.008), (\{L_6, L_7\}, 0.005), (\{L_1, L_6, L_7\}, 0.003)\}.$$

According to Definition 5, hyperlink set $\{L_1, L_6, L_7\}$ is added to PHS. Preferences of hyperlink sets outside PHS are set to be 0.

The algorithm to extract PHS from access relationships AR is straightforward and skipped in this article.

4.6 Selecting Hyperlinks

In the preceding sections, we have defined preferences of hyperlinks and preferences of hyperlink sets that reflect the contribution of the selected hyperlinks to the three metrics of a portal page. In this section, we define an approximation of the hyperlink selection problem using the preferences as follows.

Given: (1) the hyperlink pool H of a Web site;
(2) the number of hyperlinks to be placed in a portal page of the Web site—N, where $N < |H|$,

Select N hyperlinks from the hyperlink pool H to construct PHL (i.e., the set of hyperlinks in a portal page).
Objective: maximize the following objective function,

$$SCORE(PHL) = \sum_{L \in PHL} PRE_L + \sum_{k-HS \subseteq PHL} (\sigma(k - HS)), \tag{9}$$

where $(k - \text{HS}, \sigma(k - \text{HS})) \in PHS$ and $2 \leq k \leq |PHL|$. SCORE(PHL), which is the score of PHL, includes two parts: the preferences of hyperlinks in PHL, and the preferences of hyperlink sets in PHL.

We propose an algorithm, namely, the greedy aggregation algorithm, to address the above problem heuristically. The algorithm is based on an average-score matrix whose indexes are hyperlink sets and whose elements are average scores of the merged indexes (i.e., hyperlink sets), as defined below.

```
Input: H: the hyperlink pool of a website
       PRE_L : preferences of hyperlinks, where L ∈ H
       PHS : hyperlink sets and their preferences
       N: the number of hyperlinks to be placed in a portal page
Output: PHL : the set of hyperlinks in the result portal page

distribute each hyperlink in H into a unique hyperlink set;
max_size = 1; /*max_size: maximum number of hyperlinks in
                hyperlink sets*/
initialize AVG; /*AVG: the average-score matrix*/
While (max_size < N)
   merge hyperlink sets with the highest average score into C;
     /*C: merged set*/
   If (size(C) > max_size) /*size(C): number of hyperlinks in C*/
      max_size = size(C);
   End if
   update AVG ; /*only need to re-compute average scores
                  related to the merged set*/
End while
PHL = C;
```

Fig. 7. The greedy aggregation algorithm.

Definition 6. For any two hyperlink sets C_i and C_j, the average score of $C_i \cup C_j$ (i.e., the merge of C_i and C_j), $AVG(C_i \cup C_j)$ is,

$$AVG(C_i \cup C_j) = \frac{\sum_{L \in |C_i \cup C_j|} PRE_L + \sum_{k-HS \subseteq |C_i \cup C_j|} (\sigma(k - HS))}{|C_i \cup C_j|}, \tag{10}$$

where $(k - \text{HS}, \sigma(k - \text{HS})) \in PHS$ and $2 \leq k \leq |C_i \cup C_j|$.

Example 8. Given preferences of hyperlinks and hyperlink sets calculated in Example 5 and Example 7, for two hyperlink sets $C_1 = \{L_2, L_3\}$ and $C_2 = \{L_7\}$, according to Equation (10),

$$AVG(C_1 \cup C_2) = \frac{PRE_{L_2} + PRE_{L_3} + PRE_{L_7} + \sigma(\{L_2, L_7\})}{3} = 0.0117.$$

$AVG(C_i \cup C_j)$ (see Equation (10)) is designed, based on the following two considerations: (a) its nominator is used to reflect the objective function (see Equation (9)); and (b) its denominator is introduced to avoid the possibility that the number of hyperlinks in hyperlink sets could impact the value of $AVG(C_i \cup C_j)$. Using the greedy aggregation algorithm, each hyperlink in H is initially placed in a unique hyperlink set and the initial average-score matrix is constructed based on (Equation 10). The algorithm merges the two hyperlink sets having the highest average score. After merging the two hyperlink sets, the average-score matrix is updated by recomputing average scores related to the merged set. This process is repeated until the number of hyperlinks in a hyperlink set is greater than or equal to N.[7] This hyperlink set is output as PHL.[8] We summarize the greedy aggregation algorithm in Figure 7.

[7]The proposed greedy algorithm is practical, but generates a local optimal solution.

[8]If the number of hyperlinks in the result hyperlink set is larger than N, then N hyperlinks are selected from it according to their individual preferences, from high-to-low, to construct *PHL*.

4.7 Time Complexity of LinkSelector

To compute the time complexity of LinkSelector, time complexity for each step in the algorithm is considered first. We denote the number of hyperlinks in the given hyperlink pool H as n_l, the number of sessions in a Web log as n_s, the number of elements in access relationships AR as n_{AR}, and the number of elements in structure relationships SR as n_{SR}. Structure relationships are discovered in Step 1. In this step, pages pointed to by all hyperlinks in H are retrieved and parsed to extract structure relationships. If the average number of hyperlinks in all retrieved pages is β, time complexity for Step 1 is $O(n_l \times \beta)$. Usually, β is much less than n_l. Hence, time complexity for Step 1 is $O(n_l)$. We employ the Apriori algorithm [Agrawal and Srikant 1994] to discover access relationships AR[9] in Step 2. Apriori goes through all sessions in a Web log α rounds to discover access relationships AR. Therefore, time complexity for Step 2 is $O(n_s \times \alpha)$. As α is much less than n_s (e.g., in our experiment discussed in Section 5, α equals 6, while n_s is 262K), time complexity for Step 2 is the order of $O(n_s)$. Preferences of hyperlinks are computed in Step 3. If the average number of structure relationships a hyperlink participates in as the initial hyperlink is γ, time complexity for Step 3 is $O(n_l \times \gamma)$. Usually, γ is much less than n_l. Hence, time complexity for Step 3 is $O(n_l)$. In Step 4, each hyperlink set in access relationships AR is checked to see whether there is a structure relationship between its elements. Hence, time complexity for Step 4 is $O(n_{AR})$. One, $n_l \times n_l$, average-score matrix is created in Step 5, which needs $O(n_l^2)$ time. At worst, it also needs $O(n_l^2)$ time to re-compute the similarity matrix. Hence, time complexity for Step 5 is the order of $O(n_l^2)$. Combining time complexities of all 5 steps, we get the time complexity of LinkSelector as $O(n_l^2 + n_s + n_{AR})$.

5. EXPERIMENT RESULTS

To evaluate the performance of LinkSelctor, we compare it with other commonly used hyperlink selection approaches. Section 5.1 describes the experimental data, while Section 5.2 presents and analyzes the comparison results.

5.1 Experimental Data

We performed the experiments on the University of Arizona Web site because it is large enough and generates sufficient Web logs to permit comparisons of different hyperlink selection approaches. Three months of Web logs (September 2001, February 2002 and July 2002), collected from the University Web server, were used separately in the experiments. The three months of Web logs can be considered as representative Web logs for three major time periods at the University—Fall semester (Sep. 2001), Spring semester (Feb. 2002), and Summer break (July 2002). As visitors to the University Web site changes from time period to time period, visitors' Web access patterns could also change. Furthermore the University Web site was redesigned and went live in July, 2002. The three months of Web logs and Web structure information thus provide an ideal testbase for LinkSelector, under different situations.

[9]We assume that the Web log is cleaned and sessions are identified before applying LinkSelector.

Table II. Summary of the Experimental Data

Time Span	Total Data	Training Data	Testing Data	Number of Access Relations	Number of Structure Relationships
Sep. 2001	344K sessions	262K sessions (23 days)	82K sessions (7 days)	184	594
Feb. 2002	282K sessions	208K sessions (21 days)	74K sessions (7 days)	209	594
July 2002	323K sessions	245K sessions (24 days)	78K sessions (7 days)	167	578

The raw Web log of Sept. 2001 contained 10 million records and the cleaned Web log had 4.2 million records, sufficiently rich for extracting Web visiting patterns. We used a 30-minute timeout to divide the cleaned Web log into 344K sessions. 23 days of the cleaned Web log, containing 262K sessions, were used as the training data for different hyperlink selection approaches, the last 7 days of the cleaned Web log which contained 82K sessions were used as the testing data. Table II summarizes the statistics of the three months of Web logs. By applying the Apriori algorithm to the three months of training data, we derived 184, 209, and 167 access relationships, respectively, for the training data in Sept. 2001, Feb. 2002 and July 2002. Access relationships shared by all three training data sets totaled 136.[10] For example, the access to /visiting/alumni.shtml was much more active in Feb. 2002, compared with the other two months. Specifically, the most active access relationship involving /visiting/alumni.shtml in Feb. 2002 was between /visiting/alumni.shtml and /spotlight/index.shtml, which points to campus news. A possible reason could be that several major alumni gatherings were held in Feb. 2002.

The hyperlink pool used in all three time periods consists of 743 hyperlinks in the index page of the University Web site (http://www.arizona.edu/index/webindex.shtml). Among these, 110 hyperlinks pointed to Web pages at the University Web server and the remaining 633 hyperlinks, pointed to Web pages at other Web servers. Because the Web log was collected from the University Web server, access relationships among the 633 hyperlinks pointing to Web pages at other Web servers could not be mined from the collected Web log. Therefore, the hyperlink pool used in our experiments consists of the 110 hyperlinks pointing to Web pages at the University Web server. From these, 594 structure relationships were extracted before the Web site redesign; 578 structure relationships were extracted after the Web site redesign. Between the structure relationships extracted before and after the Web site redesign, 533 structure relationships were the same. Some notable structure relationship changes after the redesign included the structure relationships between several hyperlinks and hyperlink /shared/aboutua.shtml, such as the structure relationship "/shared/athletics.shtml → /shared/aboutua.shtml". After the Web site redesign, some hyperlinks were renamed (e.g., /shared/sports-entertain.shtml

[10]Two access relationships are regarded as the same if, (1) hyperlinks in the two access relationships are the same, and (2) the deviation between the supports of the two access relationships is no more than 10%.

Table III. Hyperlinks Selected by LinkSelector, Domain Experts, and Top-Link Selection (N = 6; Sept. 2001)

No.	LinkSelector	Expert Selection	Top-Link Selection
1	/index/alldepts-index.shtml	/student_link	/student_link
2	/shared/sports-entertain.shtml	/index/alldepts-index.shtml	/index/alldepts-index.shtml
3	/working/teaching.shtml	/newschedule/parse-schedule-new.cgi	/newschedule/parse-schedule-new.cgi
4	/shared/aboutua.shtml	/phonebook	/phonebook
5	/shared/getting-around.shtml	/shared/sports-entertain.shtml	/shared/sports-entertain.shtml
6	/spotlight/index.shtml	/shared/libraries.shtml	/shared/athletics.shtml

was renamed /home/athletics.shtml). We use the original names throughout the article to avoid confusion.

5.2 Performance Comparison with Expert Selection and Top-Link Selection

Current practices of hyperlink selection, namely, expert selection, rely on the experience of domain experts (e.g., webmasters). Hyperlinks, contained in the current portal page (http://www.arizona.edu) of the University Web site, are hyperlinks selected by domain experts. Another simple approach to hyperlink selection, namely, top-link selection, selects hyperlinks with top N access frequencies from the given hyperlink pool, where N is the number of hyperlinks to be placed in a portal page. Table III lists 6 hyperlinks selected by LinkSelector, domain experts,[11] and top-link selection, using the Sept. 2001 training data. Figure 8 illustrates the performance comparison among the three approaches, using the Sept. 2001 testing data.

On average, LinkSelector outperformed both expert selection and top-link selection with a 12.7% increase in the effectiveness. Given the large number of visiting sessions (e.g., 11.5k sessions per day at the Web site of the University of Arizona), this was a big improvement in ease of finding user-sought, top level Web pages. The improvement decreased from 22.1% to 8.4% as the selection ratio (i.e., the ratio of the number of the selected hyperlinks over the total number of hyperlinks in a hyperlink pool) increased from 1.8% (i.e., 2/110) to 9.1% (i.e., 10/110). However, even at the selection ratio of 9.1%, which is more than double the selection ratio for the portal page of the University of Arizona Web site (i.e., 32/743 = 4.3%), the improvement in effectiveness (i.e., 8.4%) was still apparent. Moreover, at a high selection ratio, such as 8.2% (i.e., N = 9), 4238[12] more user-sought, top level Web pages could be easily accessed from the portal page constructed using LinkSelector, than from that constructed utilizing expert selection. As the testing data covers a 7-day time period, the average saving per day was 605. Compared with expert selection and top-link selection,

[11]In this experiment, k hyperlinks, selected by domain experts, are k hyperlinks chosen from the portal page designed by domain experts and these chosen hyperlinks are top-k, frequently-accessed hyperlinks among all hyperlinks in the portal page.

[12]The number was derived by comparing $\sum_{j=1}^{s} |UHL(S_j) \cap HL|$ between LinkSelector and expert selection.

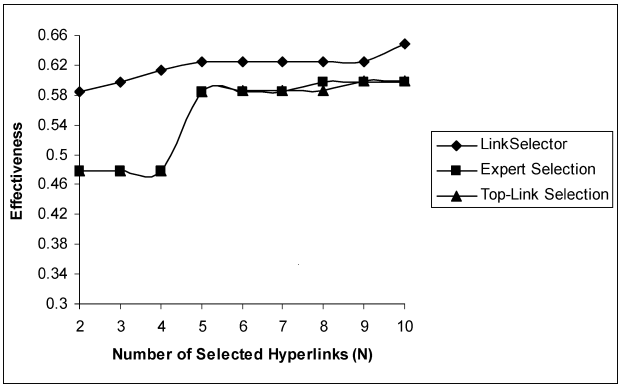

Fig. 8. (a) Effectiveness comparison among LinkSelector, expert selection, and top-link selection (Sept. 2001).

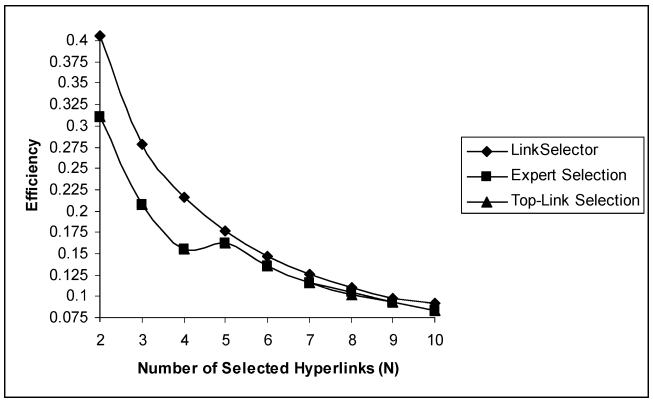

Fig. 8. (b) Efficiency comparison among LinkSelector, expert selection, and top-link selection (Sept. 2001).

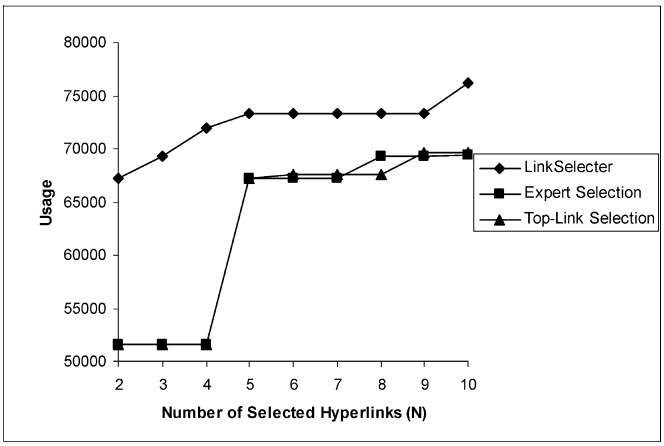

Fig. 8. (c) Usage comparison among LinkSelector, expert selection, and top-link selection (Sept. 2001).

LinkSelector improved the efficiency by 16.9% on average. Similarly, the improvement in the efficiency decreased from 30.2% to 9.3% as the selection ratio increased. Compared with expert selection and top-link selection, LinkSelector also improved the usage by an average of 17.0%. The improvement decreased from 30.2% to 9.4% as the selection ratio was increased.

The improvements were attributed to the relationships among hyperlinks considered in LinkSelector, but missed in the other two approaches. For example, hyperlinks /shared/sports-entertain.shtml and /shared/athletics.shtml were on a popular path to Web pages on sports and entertainments. The first hyperlink was the starting point of the path and the second one was the second link on this path. Therefore, both of them had high access frequency and were selected by top-link selection. However, top-link selection failed to consider that there was a structure relationship between these two hyperlinks. In this structure relationship, hyperlink /shared/sports-entertain.shtml was the initial hyperlink and hyperlink /shared/athletics.shtml was the terminal hyperlink. It was natural to navigate from /shared/sports-entertain.shtml to /shared/athletics.shtml and find Web pages on sports. Hence, it was unnecessary to put both of them in a portal page. Applying Group I relationships, LinkSelector selected only the starting point of the path—/shared/sports-entertain.shtml. Both top-link selection and expert selection failed to consider Group II relationships among hyperlinks (i.e., hyperlinks that are structurally unrelated, but access related). For example, hyperlinks /shared/sports-entertain.shtml and /shared/aboutua.shtml were structurally unrelated hyperlinks. However, a large number of sessions looking for information regarding sports and entertainment at the University (i.e., /shared/sports-entertain.shtml) also tried to learn something about the University (i.e., /shared/aboutua.shtml). Placing both hyperlinks in a portal page could save Web surfers effort in finding the path from one topic to the other. Applying the Group II relationship, LinkSelector selected both hyperlinks.

As described in Section 5.1, there were changes in Web visiting patterns and the Web site structure in Feb. and July 2002, compared with Sept. 2001. However, neither expert selection nor top-link selection were sensitive to these changes. Before and after the Web site redesign, the majority of the hyperlinks in the homepage of the University Web site were the same. Although there were more visits to hyperlink /visiting/alumni.shtml in Feb. 2002, top-link selection did not select it because it was not one of the top-visited hyperlinks. LinkSelector, on the other hand, was sensitive to these changes. Table IV lists hyperlinks selected by LinkSelector in the three time periods. As visits to /visiting/alumni.shtml were much more active in Feb. 2002 especially, there existed an access relationship between /visiting/alumni.shtml and /spotlight/index.shtml and no structure relationship between them, LinkSelector picked /visiting/alumni.shtml by considering the Group II relationship. In July 2002, as the structure relationship "/shared/sports-entertain.shtml → /shared/aboutua.shtml" emerged because of the Web site redesign, a Group II relationship between the two hyperlinks disappeared. Hence, LinkSelector did not pick /shared/aboutua.shtml using the July 2002 data.

Table IV. Hyperlinks Selected by LinkSelector (N = 6)

No.	Sep. 2001	Feb. 2002	July 2002
1	/index/alldepts-index.shtml	/index/alldepts-index.shtml	/index/alldepts-index.shtml
2	/shared/sports-entertain.shtml	/shared/sports-entertain.shtml	/shared/sports-entertain.shtml
3	/working/teaching.shtml	/working/teaching.shtml	/working/teaching.shtml
4	/shared/aboutua.shtml	/shared/aboutua.shtml	/student_link
5	/shared/getting-around.shtml	/spotlight/index.shtml	/spotlight/index.shtml
6	/spotlight/index.shtml	/visiting/alumni.shtml	/newschedule/parse-schedule-new.cgi

Figures 9 and 10 depict the performance comparison among the three approaches using the Feb. 2002 testing data and the July 2002 testing data, respectively.

Using the Feb. 2002 data, compared with the other two approaches, LinkSelector improved effectiveness by an average of 16.2%, efficiency by an average of 20.1%, and usage by an average of 21.3%. Using the July 2002 data, LinkSelector improved effectiveness by an average of 5.9%, efficiency by an average of 7.2%, and usage by an average of 5.1%. In all the time periods, LinkSelector outperformed the other two approaches. The performance of expert selection and top-link selection was improved after the Web site redesign. The performance improvement was because some previously mentioned problems, such as the failure to consider Group II relationships between /shared/sports-entertain.shtml and /shared/aboutua.shtml, were fixed after the Web site redesign (e.g., Group II relationships between the above mentioned hyperlinks disappeared as the structure relationship between the hyperlinks emerged after the redesign). On the other hand, if LinkSelector had been adopted, these problems could have been fixed before the Web site redesign. Hence, the performance improvement also suggests that LinkSelector could be a good supporting tool in designing better portal pages. Even after the Web site redesign, the performance improvement of LinkSelector was still notable. On average 3068 more user-sought, top level Web pages could be easily accessed from the portal page constructed using LinkSelector than from those constructed utilizing the other two approaches. As the testing data covers a 7-day time period, average saving per day was 438. From the three experiments, we conclude that: (1) LinkSelector outperforms expert selection and top-link selection because of Group I and II relationships considered in LinkSelector, but missed in the other two approaches; (2) the more Group I and II relationships considered in LinkSelector that were missed in the other two approaches, the higher the performance improvement of LinkSelector.

6. CONCLUSION AND FUTURE WORK

In this article, we have formally defined the hyperlink selection problem and proposed a heuristic solution method named LinkSelector. The proposed method is based on relationships among hyperlinks—structure relationships

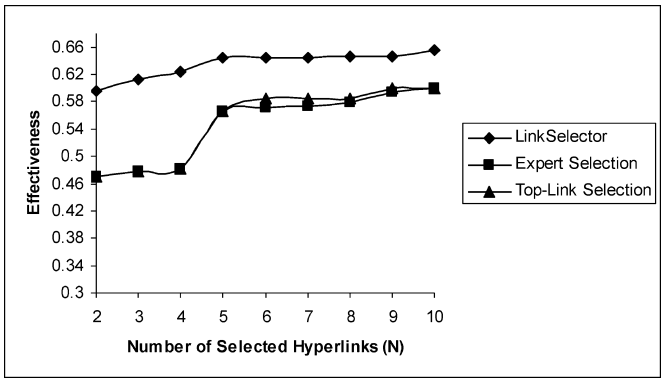

Fig. 9. (a) Effectiveness comparison among LinkSelector, expert selection, and top-link selection (Feb. 2002).

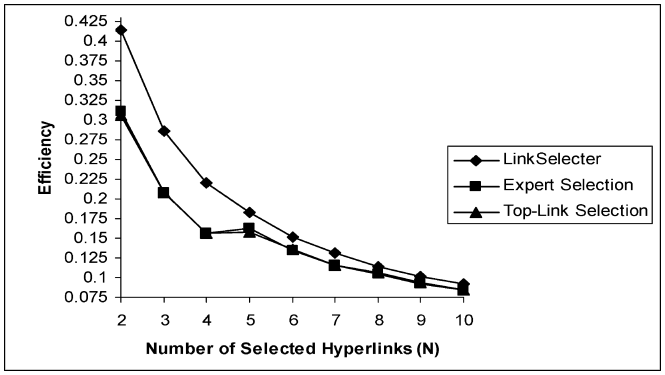

Fig. 9. (b) Efficiency comparison among LinkSelector, expert selection, and top-link selection (Feb. 2002).

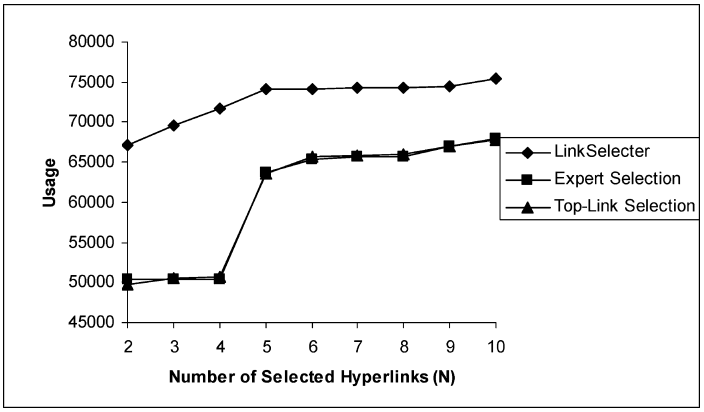

Fig. 9. (c) Usage comparison among LinkSelector, expert selection, and top-link selection (Feb. 2002).

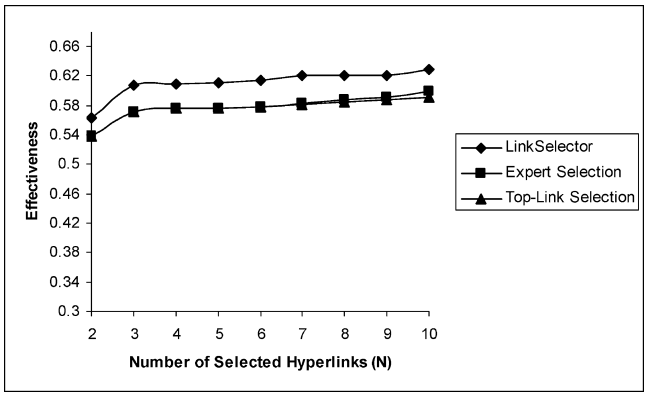

Fig. 10. (a) Effectiveness comparison among LinkSelector, expert selection, and top-link selection (July 2002).

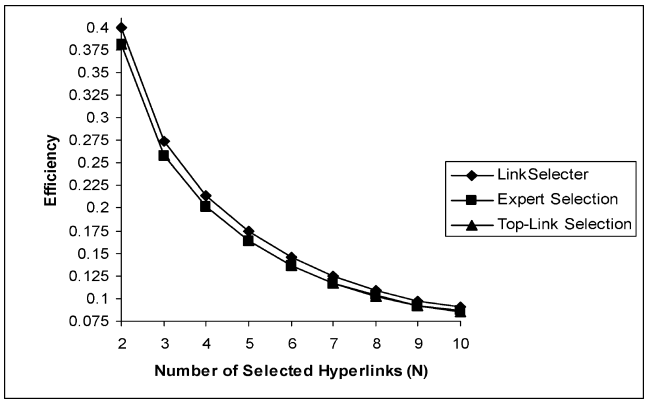

Fig. 10. (b) Efficiency comparison among LinkSelector, expert selection, and top-link selection (July 2002).

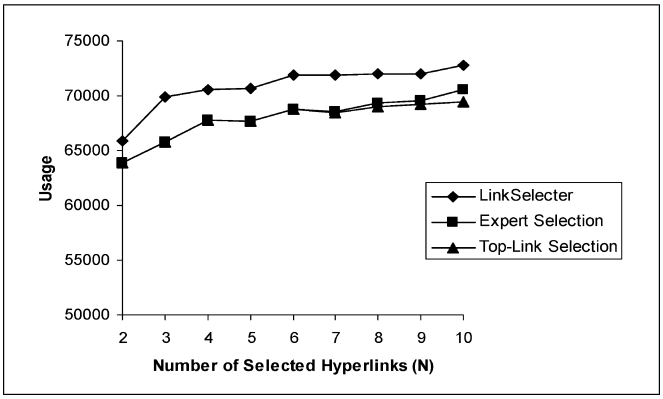

Fig. 10. (c) Usage comparison among LinkSelector, expert selection, and top-link selection (July 2002).

extracted from an existing Web site, and access relationships discovered from a Web log. Preferences of hyperlinks and hyperlink sets are calculated from these relationships and a greedy algorithm is developed to extract hyperlinks from a given hyperlink pool, using the preferences calculated. We also compare LinkSelector with the current practice of hyperlink selection and top-link selection, using data obtained from the University of Arizona Web site. Results show that LinkSelector, which integrates the structure and access information of a Web site, outperforms the other hyperlink selection approaches.

Future work is needed in three areas. First, we plan to conduct an empirical user study to examine the properties of different hyperlink selection approaches and compare their performances using the metrics proposed in Section 3. Second, how to collect cached Web logs and combine them with server logs to mining access relationships and how to extract structure relationships from dynamic Web pages needs to be researched. Third, we plan to work on making LinkSelector adaptive to changes both in the structure of a Web site and in Web visiting patterns of users. The former leads to changes in structure relationships and the latter causes changes in access relationships. As a result, hyperlinks selected by LinkSelector based on old structure relationships and access relationships could be out-of-date. To keep the selected hyperlinks up-to-date, an obvious solution is to rerun LinkSelector every time a change occurs. Apparently, for Web sites with frequent changes, the cost of frequent rerun is prohibitive. An efficient solution needs to be developed to monitor both types of changes and to trigger the rerun of LinkSelector only when necessary.

ACKNOWLEDGMENTS

The authors are grateful for the comments and suggestions from the Editors and the anonymous reviewers. They would like to thank the Center for Computing and Information Technology at the University of Arizona for providing Web logs used in the article.

REFERENCES

ANDERSON, C., DOMINGOS, P., AND WELD, D. 2001. Adaptive Web navigation for wireless devices. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, Seattle, WA, 879–884.

ARMSTRONG, R., FREITAG, D., JOACHIMS, T., AND MITCHELL, T. 1995. WebWatcher: a learning apprentice for the World Wide Web. In *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, Stanford, CA, March.

AGRAWAL, R. AND SRIKANT, R. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, Santiago, Chile, Sept., 487–499.

BALABANOVIC, M. AND SHOHAM, Y. 1997. Fab: Content-based collaborative recommendation. *Commun. ACM 40*, 3, 66–72.

BRIN, S. AND PAGE, L. 1998. The anatomy of a large-scale hypertextual Web search engines. In *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, April.

CATLEDGE, L. AND PITKOW, J. 1995. Characterizing browsing behaviors on the world wide web. *Comput. Netw. ISDN Syst. 27*, 6, 1065–1073.

CHAKRABARTI, S., DOM, B. E., KUMAR, S. R., RAGHAVAN, P., RAJAGOPALAN, S., TOMKINS, A., GIBSON D., AND KLEINBIRG, J. 1999. Mining the Web's link structure. *IEEE Comput. 32*, 8, 60–67.

CHAKRABARTI, S. 2000. Data mining for hypertext: a tutorial survey. *ACM SIGKDD Explorations 1*, 2, 1–11.

CHEN, M., PARK, J., AND YU, P. 1996. Data mining for path traversal patterns in a Web environment. In *Proceedings of the 16th International Conference on Distributed Computing Systems*, Hong Kong, China, May.

COOLEY, R., MOBASHER, B., AND SRIVASTAVA, J. 1997. Web mining: information and pattern discovery on the World Wide Web. In *Proceedings of IEEE International Conference on Tools with AI*, Newport Beach, CA, Nov.

COOLEY, R., TAN, P., AND SRIVASTAVA, J. 1999a. WebSIFT: the Web site information filter system. In *Proceedings of the Web Usage Analysis and User Profiling Workshop*, San Diego, CA, August.

COOLEY, R., MOBASHER, B., AND SRIVASTAVA, J. 1999b. Data preparation for mining World Wide Web browsing patterns. *Knowl. Info. Syst. 1*, 1, 1–27.

KLEINBERG J. 1998. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, CA, Jan.

KOSALA, R. AND BLOCKEEL, H. 2000. Web mining research: a survey. *ACM SIGKDD Explorations 2*, 1, 1–15.

LANG, K. 1995. NewsWeeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, Lake Tahoe, CA.

LAWRENCE, S. AND GILES, C. L. 1998. Searching the World Wide Web. *Science 280*, 98–100.

LAWRENCE, S. AND GILES, C. L. 1999. Accessibility of information on the Web. *Nature 400*, 107–109.

LEE, J. AND PODLASECK, M. 2001. Visualization and analysis of clickstream data of online stores for understanding Web merchandising. *Data Mining Knowl. Disc. 5*, 1–2, 59–84.

LI, Y. 1998. Towards a qualitative search engine. *IEEE Internet Comput. 2*, 4, 24–29.

LIEBERMAN, H. 1995. Letizia: An agent that assists web browsing. In *Proceedings of the 1995 International Joint Conference on Artificial Intelligence*, Quebec, Canada, August.

MANBER, U., PATEL, A., AND ROBISON, J. 2000. Experience with personalization on Yahoo!. *Comm. ACM 43*, 8, 35–39.

MLADENIC, D. 1999. Machine learning used by personal WebWatcher. In *Proceedingsof ACAI-99 Workshop on Machine Learning and Intelligent Agents*, Chania, Greece, July.

MOBASHER, B., COOLEY, R., AND SRIVASTAVA J. 2001. Automatic personalization based on web usage mining. *Comm. ACM 43*, 8, 142–151.

NIELSON, J. 1999. User interface directions for the Web.*Comm. ACM 42*, 1, 65–72.

NIELSON, J. AND WAGNER, A. 1996. User interface design for the WWW. In *Proceedings of ACM CHI 96*, British Columbia, Canada, April, 330–331.

PERKOWITZ, M. AND ETZIONI, O. 2000. Towards adaptive web sites: conceptual framework and case study. *Artif. Intell. 118*, 1–2, 245–275.

RESNICK, P. ET AL. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedingsof the ACM Conference on CSCW*, Chapel Hill, NC, October, 175–186.

SALTON, G. 1968. Automatic Information Organization and Retrieval. *McGraw-Hill Inc.*

SHARDANAND, U. AND MAES, P. 1995. Social information filtering: algorithms for automating "word of mouth". In *Proceedings of ACM CHI'95*, Denver, CO, May, 210–217.

SPILIOPOULOU, M. AND POHLE, C. 2001. Data mining to measure and improve the success of web sites. *Data Mining Knowl. Disc. 5*, 1–2, 85–114.

SRIVASTAVA, J., COOLEY, R., DESHPANDE, M., AND TAN, P. 2000. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explorations 1*, 2, 1–12.

YAN, T., JACOBSEN, M., GARCIA-MOLINA, H., AND DAYAL, U. 1996. From user access patterns to dynamic hypertext linking. In *Proceedings of the 5th International World Wide Web Conference*, Paris, France, May.

Received May 2002; revised December 2002; accepted February 2003