ServiceFinder: A Method Towards Enhancing Service Portals

XIAO FANG
University of Toledo
OLIVIA R. LIU SHENG
University of Utah
and
MICHAEL CHAU
The University of Hong Kong

The rapid advancement of Internet technologies enables more and more educational institutes, companies, and government agencies to provide services, namely online services, through web portals. With hundreds of online services provided through a web portal, it is critical to design web portals, namely service portals, through which online services can be easily accessed by their consumers. This article addresses this critical issue from the perspective of service selection, that is, how to select a small number of service-links (i.e., hyperlinks pointing to online services) to be featured in the homepage of a service portal such that users can be directed to find the online services they seek most effectively. We propose a mathematically formulated metric to measure the effectiveness of the selected service-links in directing users to locate their desired online services and formally define the service selection problem. A solution method, ServiceFinder, is then proposed. Using real-world data obtained from the Utah State Government service portal, we show that ServiceFinder outperforms both the current practice of service selection and previous algorithms for adaptive website design. We also show that the performance of ServiceFinder is close to that of the optimal solution resulting from exhaustive search.

Categories and Subject Descriptors: H.3.5 [**Information Storage and Retrieval**]: Online Information Services

General Terms: Design, Experimentation

Additional Key Words and Phrases: Service portal, online service, service selection

Authors' addresses: X. Fang, Department of Information, Operations and Technology Management, University of Toledo, Toledo, OH 43606; email: xiao.fang@utoledo.edu; O. R. Liu Sheng, School of Accounting and Information Systems, University of Utah, Salt Lake City, UT 84112; email: actos@business.utah.edu; M. Chau (corresponding author), School of Business, The University of Hong Kong, Pokfulam, Hong Kong; email: mchau@business.hku.hk.

ACM Reference Format:
Fang, X., Liu Sheng, O. R., and Chau, M. 2007. ServiceFinder: A method towards enhancing service portals. ACM Trans. Inform. Syst. 25, 4, Article 17 (October 2007), 28 pages. DOI = 10.1145/1281485.1281488 http://doi.acm.org/ 10.1145/1281485.1281488

1. INTRODUCTION

More and more educational institutes, companies, and government agencies now provide both internal employees and external customers with services, namely online services, through web portals. For example, a large number of universities have implemented or are implementing web portals to provide students, faculty, and staff with online services such as Class Registration and Grant Administration. Bank or financial web portals provide online services such as Personal Account Management and Mortgage Management. Government web portals providing online services such as Renew Vehicle Registration also have been implemented all over the world. In the U.S., web portals have become the major means of government service delivery, starting in 2003 [Wood et al. 2003]. With hundreds of online services provided through a web portal, finding desired online services is not an easy task for many users. Hence, a critical research problem is how to design web portals (hereafter called *service portals*) through which online services can be easily accessed by their consumers.[1]

A common way of finding online services in a service portal is to use a site-specific search engine. As pointed out in Huberman et al. [1998], using search engines suffers from the disadvantage of returning too much irrelevant information. A wealth of research has attempted to address this disadvantage [Chakrabarti 2000]. An alternative popular way of seeking online services is to surf through hyperlinks. According to Nielsen [2000], almost half of web users seek information either by surfing through hyperlinks or by switching between search engines and hyperlink surfing. As users click through a set of hyperlinks to find their desired online services, placing appropriate hyperlinks in the right webpages is critical in improving their experience with the services.

The homepage of a service portal is the starting point for seeking online services. Usually, a small number of service-links are selected and featured in the homepage. A service-link is a hyperlink pointing to an online service (see Figure 1 for an example). As shown in Figure 1, 6 out of 145 service-links are featured in the homepage of the Utah State Government service portal (hereafter called *Utah.gov*),[2] while 6 out of 203 are highlighted in the homepage of the Texas State Government service portal. Service portals with a good selection of featured service-links guide users to locate the online services they seek easily and effectively and attract more users, while service those with a bad selection of featured service-links make online service searching difficult and lose users,

[1]In this article, an online service refers to a service provided by an organization and rendered to its consumers through a web portal. Readers should note that the online services discussed in this article are not the same as "web services", the programmatic interface to web-based applications promoted by W3C.

[2]Utah.gov was restructured after we submitted the article. The claim of featuring six service-links was based on the older website on which we conducted our research.

Fig. 1. Service-links featured in homepages of sample state government service portals: Utah and Texas.

as users may be time-pressed or impatient [Nielsen and Wagner 1996]. This research studies how to select a small number of service-links to be featured in the homepage of a service portal such that users can locate their desired online services most easily and effectively. We name this problem *service selection*.

A well-designed service portal normally features a small number of service-links in its homepage. Currently, a typical service portal provides several hundred online services. The number of online services provided in a service portal grows over time as more and more services are provided online. It is computationally too expensive to enumerate all combinations of several service-links from a pool of several hundred and find the one that is most effective in guiding users to locate the online services they seek. Current practice of service selection relies on domain experts' (e.g., webmasters) expertise, namely *expert*

selection. Domain experts usually select service-links based on their experience with users or on a survey of the requirements of a small group of users. Expert selection is often subjective. In addition, it reflects the perspectives of domain experts or a small group of users on what service-links should be selected. In this research, we formally define the service selection problem and propose an objective measurement of the effectiveness of the selected service-links in directing users to locate their desired online services. ServiceFinder, a heuristic solution based on objective user visiting patterns, both recorded in web logs and the existing structure of a service portal, is presented and evaluated using real-world data collected from Utah.gov.

The rest of the article is organized as follows. We review related work and discuss the difference between this research and prior work in Section 2. In Section 3, we formally define the service selection problem. ServiceFinder is presented in Section 4. In Section 5, we evaluate the performance of ServiceFinder using data obtained from Utah.gov. We conclude the work in Section 6.

2. RELATED WORK

Research related to this article covers web structure mining, web usage mining, browsing agents, recommender systems, and adaptive websites. In the area of web structure mining, various techniques have been proposed to use link-based metrics to measure the importance of webpages, such as the PageRank [Brin and Page 1998] and HITS algorithms [Kleinberg 1998]. However, these techniques are often good for measuring the relationships among a large number of websites rather than intrasite relationships, as evidenced by the fact that intrasite links are often discounted in these algorithms and their variations. Also, the importance measures discovered by these algorithms are not associated with visit frequencies; a page with the highest score may not be the one most requested by users. Web usage mining [Srivastava et al. 2000], the process of applying data mining techniques to discover web access patterns from a web log, is also related to our research. Detailed surveys can be found in Chakrabarti [2000] and Kosala and Blockeel [2000].

A browsing agent is a software agent that assists a user in browsing the web. WebWatcher [Armstrong et al. 1995; Joachims et al. 1997] is indicative of the earliest research on browsing agents. WebWatcher first asks a user's web visiting goal. It then predicts and highlights hyperlinks that are likely to lead to the user's target information. Another notable web browsing agent, Letizia [Lieberman 1995; Lieberman et al. 2001], infers a user's goals implicitly from the user's browsing behavior. Letizia explores the web ahead of the user and uses inferred user's goals to recommend webpages that could be visited next.

Recommender systems recommend items (e.g., articles) through content-based or collaborative filtering. Recommender systems using content-based filtering, such as NewsWeeder [Lang 1995], recommend an item to a user based on the similarity between the content of the item and the user's profile. Recommender systems employing collaborative filtering, such as Ringo [Shardanand and Maes 1995], GroupLens [Resnick et al. 1994], and associative retrieval

[Huang et al. 2004], recommend an item to a user if other users with similar tastes like this item. Fab [Babanovic and Shoham 1997] is a recommender system combining content-based and collaborative filtering. A good review on recommender systems research can be found in Adomavicius and Tuzhilin [2005].

Research on browsing agents and recommender systems, however, is not applicable to the service selection problem for several reasons. First, the objective of service selection is to select a set of service-links such that seeking online service would be easier for every visitor to a service portal. Research on browsing agents and recommender systems nevertheless focuses on personalization, that is, recommending personalized items to a specific user. Further, as pointed out in Perkowitz and Etzioni [1997], personalization can be genuinely useful for repeat visitors, but does not benefit first-time visitors, while a major objective of service portals is to encourage more and more first-time visitors to use online services.

Research on adaptive websites is the closest research to this article. Perkowitz and Etzioni [1997] introduced the research on adaptive websites: websites that automatically improve their organization and presentation by learning from visitors' access patterns. In particular, they investigated the index page synthesis problem and proposed the PageGather algorithm [Perkowitz and Etzioni 2000]. The PageGather algorithm aims at creating index pages with hyperlinks to related, but unlinked, pages. Related but unlinked pages are pages that share a common topic, but are currently unlinked at a website. A lot of research such as Anderson et al. [2001], Czyzowicz et al. [2003], and Fang and Liu [2004] has followed Perkowitz and Etzioni's call to adaptive website research. In this article, we report our work on adaptive websites in the context of service portal design. Our work differs from previous research on adaptive websites in the following ways. First, we propose a mathematically formulated metric to measure the effectiveness of service selection in terms of expected number of online services located by web surfers. By incorporating the structure of a service portal, the probability of surfing depth [Huberman et al. 1998], and the web surfing behaviors recorded in web logs into metric design, we propose a new and more advanced metric than those proposed in previous adaptive website research, such as those introduced in Fang and Liu [2004], with regard to reflecting web users' information searching behaviors. Moreover, the method introduced in this work, ServiceFinder, differs significantly from methods proposed previously such as LinkSelector [Fang and Liu 2004] and PageGather [Perkowitz and Etzioni 2000] in that: (1) ServiceFinder considers the sequential visiting patterns of online services, whereas both LinkSelector and PageGather neglected web page visiting sequences; (2) ServiceFinder assigns weights to service-links and service-link pairs based on their contribution to the proposed metric; (3) ServiceFinder employs a genetic algorithm to select service-links based on the assigned weights of service-links and service-link pairs; and (4) ServiceFinder considers not only service-links to related but unlinked online services, but also those to related and linked online services (which were neglected in PageGather).

Table I. Notation Summary

Notation	Description
S	a set of service-links in a service portal
s	a service-link $s \in S$
$e(s)$	an online service pointed to by s
$R(s)$	structurally related service-links of s
$d_L(s)$	to navigate from an online service pointed to by s to an online service pointed to by any service-link in $d_L(s)$, at least L service-links need to be surfed, $L \geq 1$; see Eq. (2)
$G(L)$	the probability of surfing at least$L$ hyperlinks during a website visit, $L \geq 1$; see Eq. (5)
S_f	a set of service-links selected and featured in the homepage of a service portal
E_v	a sequence of online services sought by a user during one visit to a service portal
$eff(S_f, E_v)$	the effectiveness of S_f for locating online services in E_v
$P(e(s_{v_j}) \mid \text{homepage})$	the probability of finding online service e(s_{v_j})by navigating from the homepage of a service portal, see Eq. (11)
$P(e(s_{v_j}) \mid \text{homepage}, s_{f_i})$	the probability of finding online service e(s_{v_j}) by surfing from the homepage and by clicking on service-link s_{f_i} on the homepage
$P(e(s_{v_j}) \mid e(s))$	the probability of finding online service $e(s_{v_j})$, given that the search starts from online service $e(s)$, see Eq. (16)
t	the probability of restarting a fresh search from the homepage of a service portal
$eff(S_f, \log)$	the effectiveness of S_f for locating online services in a web log
$< e_i, e_t >$	a 2-consecutive-sequence
$\sup(< e_i, e_t >)$	support of $< e_i, e_t >$
$C_i(s), i = 1, 2, 3, 4$	category i service-links of s; see Table II
$w(s)$	weight of a service-link s; see Eq. (25)
$v(e(s))$	visiting rate of an online service $e(s)$
$w(\{s_i, s_j\})$	weight of a service-link pair $\{s_i, s_j\}$

3. THE SERVICE SELECTION PROBLEM

The purpose of service selection is to assist users to locate as many desired online services as possible. First of all, metrics need to be developed to measure the number of user-desired online services found. In this article, we propose a metric that can be evaluated using web logs, which record what online services are visited in sessions and their visiting sequences. The design of the proposed metric considers both the existing structure of a service portal and the probability of user surfing depth. For the convenience of readers, the notation used in this work is summarized in Table I.

3.1 Structure Relationships between Service-Links

Service-links are structurally related, that is, some service-links are placed in the online service pointed to by some other service-link. For example, as shown in Figure 2, service-links pointing to online services Network Registration, "Build a Customized Business List and Principal Search by Name are placed in the online service Business Entity Search. Structure relationships between service-links are important in determining which service-links are preferable of selection for the homepage. Let us consider the scenario that a

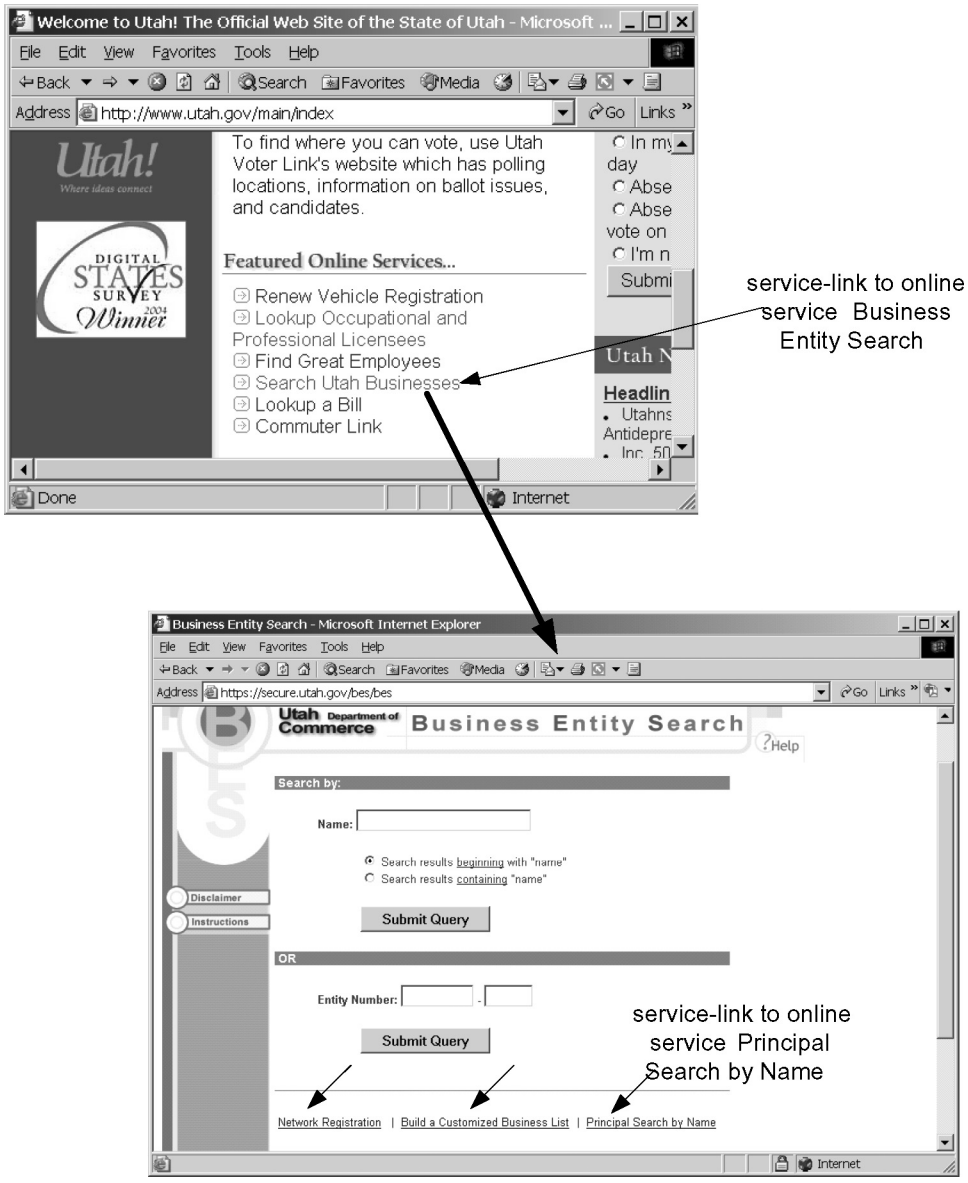

Fig. 2. Structure relationships between service-links.

large number of surfers visit online services Business Entity Search and Principal Search by Name sequentially, starting from the homepage of Utah.gov. Placing a service-link to Business Entity Search in the homepage of Utah.gov makes the online service Business Entity Search easily located. Further, after visiting online service Business Entity Search, online service Principal Search by Name is at surfers' fingertips (see Figure 2), as there is a structure relationship between the service-link to Business Entity Search and that to Principal Search by Name.

We introduce $R(s)$ to describe structure relationship between service-links. Let S be a set of service-links pointing to all online services provided in a service portal.

Definition 1. For a service-link s, $s \in S$, we define $R(s)$as a set of service-links placed in the online service pointed to by s, where $R(s) \subseteq S$. We name service-links in $R(s)$ as structurally related service-links of s.

In the example shown in Figure 2, we have the following.

R(service-link to online service "Business Entity Search") =
{service-link to online service "Network Registration",
service-link to online service "Build a Customized Business List",
service-link to online service "Principal Search by Name"}

Based on $R(s)$, we next measure the number of service-links that have to be clicked to navigate from one online service to another. For a service-link s, $s \in S$, we define

$$d_0(s) = \{s\}, \tag{1}$$

and

$$d_L(s) = \bigcup_{\forall k \in d_{L-1}(s)} R(k) - \bigcup_{i=0}^{L-1} d_i(s) \quad L \geq 1. \tag{2}$$

To navigate from an online service pointed to by s to one pointed to by any service-link in $d_L(s)$, at least L service-links need to be traversed.

Example 1. As shown in Figure 3, online services A to H are pointed to by service-links s_1 to s_8, respectively, and service-links s_1, s_2, and s_3 are service-links featured in the portal homepage. In this example, $d_0(s_1) = \{s_1\}$. According to Eq. (2), $d_1(s_1) = R(s_1) - d_0(s_1) = \{s_4, s_5\} - \{s_1\} = \{s_4, s_5\}$, and $d_2(s_1) = R(s_4) \cup R(s_5) - [d_0(s_1) \cup d_1(s_1)] = \{s_5, s_6, s_8\} \cup \{s_7\} - [\{s_1\} \cup \{s_4, s_5\}] = \{s_6, s_7, s_8\}$.

According to the preceding calculations, to navigate from online service A (pointed to by service-link s_1) to online service G (pointed to by service-link s_7), at least two service-links need to be clicked (i.e., clicking on service-link s_5 and then s_7), since $s_7 \in d_2(s_1)$. Note that there are other paths to navigate from online service A to online service G which take more than two clicks (e.g., clicking on service-link s_4, then s_5, and thens_7).

3.2 Probability of Surfing Depth

Past research has revealed some regularities from the fast-growing web, either using empirical data (e.g., Huberman et al. [1998], Pitkow [1998], and Levene et al. [2001]) or through simulation (e.g., Liu et al. [2004]). In Huberman et al. [1998], the law of surfing was proposed: the probability $p(L)$ of surfing L hyperlinks during a website visit is modeled by

$$p(L) = \sqrt{\frac{\lambda}{2\pi L^3}} \exp\left[\frac{-\lambda(L-\mu)^2}{2\mu^2 L}\right], \quad L \geq 1. \tag{3}$$

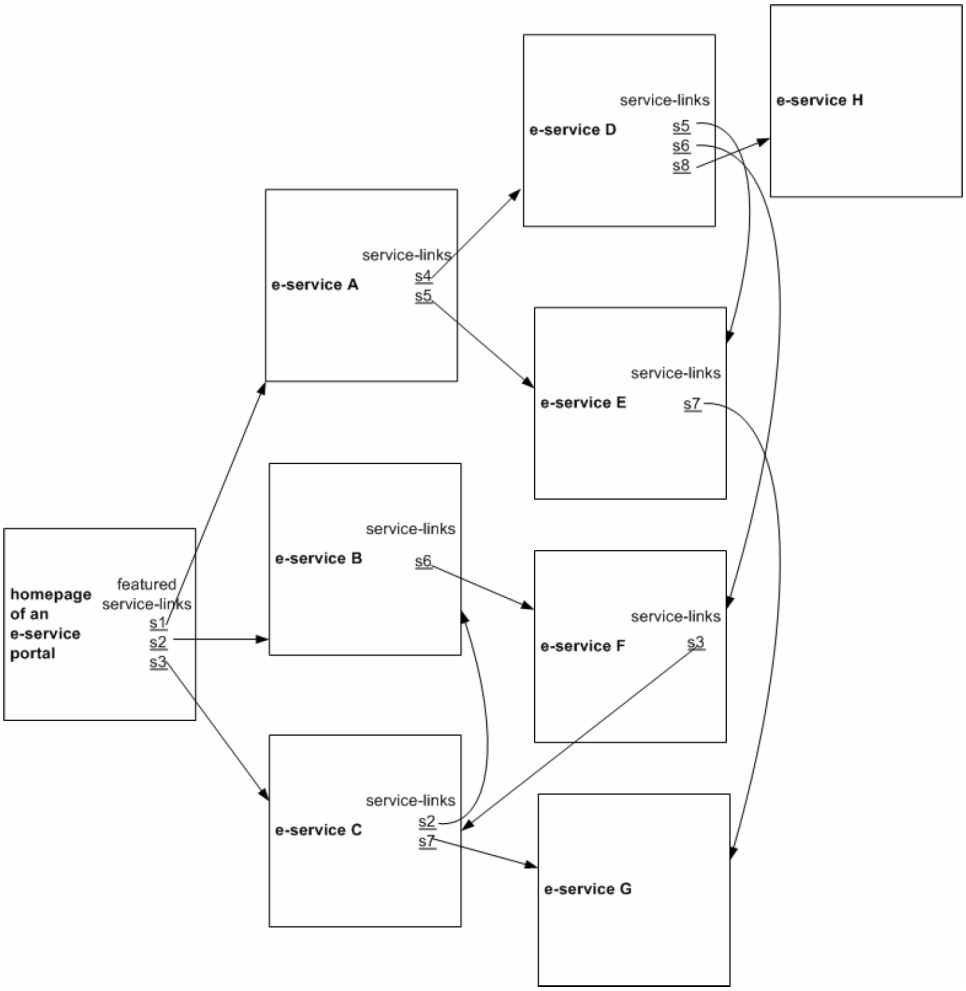

Fig. 3. An example service portal

with mean $E(L) = \mu$ and variance $Var(L) = \frac{\mu^3}{\lambda}$. Huberman et al. [1998] also show that $p(L)$ fits well with real-world data, with a mean of 3.86 and variance of 6.08. Extending the work in Huberman et al. [1998], we define$G(L)$as the probability of surfing at least$L$ hyperlinks during a website visit, and

$$G(L) = \sum_{\forall\, x \geq L} p(x), \qquad L \geq 1. \tag{4}$$

It is easy to derive Eq. (5) from (4)

$$G(L) = \begin{cases} 1 \text{ if } L = 1 \\ G(L-1) - p(L-1) \text{ if } L > 1 \end{cases} \tag{5}$$

The metric $G(L)$ is useful in measuring the least number links a user would click to search for an online service before giving up.

3.3 The Service Selection Problem

In this section, we first introduce a metric to the service selection problem $eff(S_f, E_v)$, then formally define the problem. Let S_f be a set of service-links selected and featured in the homepage of a service portal, $S_f = \{s_{f_i}\}$, where $s_{f_i} \in S$ for $i = 1, 2, \cdots, |S_f|$ and $S_f \subset S$. In the example shown in Figure 3, $S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$ and $S_f = \{s_1, s_2, s_3\}$. We denote $e(s)$ as an online service pointed to by a service-link s, $s \in S$. Let $E_v =< e(s_{v_1}), e(s_{v_2}), \cdots, e(s_{v_k}) >$ be a sequence of k online services sought by a user during one visit to a service portal, where $k \geq 1$,$s_{v_j} \in S$ for $j = 1, 2, \cdots, k$.

Definition 2. We define $eff(S_f, E_v)$, namely the effectiveness of S_f for locating online services in E_v, as the expected number of online services in E_v located by navigating from the homepage featuring S_f.

An online service with its service-link featured in the homepage can be easily located by web surfers. Hence, we assume that the probability of finding an online service with its service-link in S_f is 1 (Assumption 1). In the following subsections, we describe how to calculate $eff(S_f, E_v)$ in detail.

3.3.1 $P(e(s_{v_j}) \mid homepage)$. Let $P(e(s_{v_j}) \mid \text{homepage})$ be the probability of finding a user-sought online service $e(s_{v_j})$ by navigating from the homepage, where $e(s_{v_j}) \in E_v$ for $j = 1, 2, \cdots k$. If $s_{v_j} \in S_f$, then according to Assumption 1,

$$P(e(s_{v_j}) \mid \text{homepage}) = 1 \qquad \text{if} \quad s_{v_j} \in S_f. \tag{6}$$

Example 2. In Figure 3, the probability of finding online service A (i.e., $e(s_1)$) from the homepage is

$$P(e(s_1) \mid \text{homepage}) = 1 \qquad \text{since } s_1 \in S_f.$$

If $s_{v_j} \notin S_f$, a surfer could navigate from any service-link in S_f to search for $e(s_{v_j})$. We denote $P(e(s_{v_j}) \mid \text{homepage}, s_{f_i})$ as the probability of finding online service $e(s_{v_j})$ by surfing from service-link s_{f_i} in S_f. Assuming that a surfer has equal probability of choosing any service-link in S_f to search for $e(s_{v_j})$,[3] we have

$$P(e(s_{v_j}) \mid \text{homepage}) = \frac{\sum_{\forall s_{f_i} \in S_f} P(e(s_{v_j}) \mid \text{homepage}, s_{f_i})}{|S_f|} \qquad \text{if } s_{v_j} \notin S_f. \tag{7}$$

Example 3. In Figure 3, the probability of finding online service G (i.e., $e(s_7)$) from the homepage is

$$P(e(s_7) \mid \text{homepage}) = \frac{P(e(s_7) \mid \text{homepage}, s_1) + P(e(s_7) \mid \text{homepage}, s_2) + P(e(s_7) \mid \text{homepage}, s_3)}{3}.$$

Next, we need to solve $P(e(s_{v_j}) \mid \text{homepage}, s_{f_i})$ in Eq. (7). If $s_{v_j} \in d_L(s_{f_i})$, $L \geq 1$, according to Eq. (2), it takes at least L clicks to navigate from online service $e(s_{f_i})$ to online service $e(s_{v_j})$. Since there is still one click from the homepage to

[3]In real-world situations, a surfer can often choose the appropriate link based on information such as anchor text or prior experience. Here we only focus on the expected probability without relying on any extra assumption.

online service $e(s_{f_i})$, it takes at least $L+1$ clicks to navigate from s_{f_i} featured in the homepage to online service $e(s_{v_j})$. Let r be a path from s_{f_i} to $e(s_{v_j})$, p_r be the probability of taking path r, and z_r be the distance of path r, $z_r \geq L+1$. If path r is taken, users who are willing to surf z_r or more hyperlinks can locate online service $e(s_{v_j})$. Hence,

$$P(e(s_{v_j}) \mid \text{homepage}, s_{f_i}) = \sum_{\forall \mathbf{r}} p_r G(z_r) \quad \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \in d_L(s_{f_i}), L \geq 1, \tag{8}$$

where $\sum_{\forall \mathbf{r}} p_r = 1$, and $z_r \geq L+1$.

According to Eq. (5), $G(.)$ is a decreasing function. Hence,

$$G(z_r) \leq G(L+1),$$

and

$$P(e(s_{v_j}) \mid \text{homepage}, s_{f_i}) \leq \sum_{\forall \mathbf{r}} p_r G(L+1) = G(L+1).$$

It is too complicated to enumerate all paths from s_{f_i} to $e(s_{v_j})$, given that a surfer could backtrack at any time during their navigation. In this article, we approximate $P(e(s_{v_j}) \mid \text{homepage}, s_{f_i})$ using its upper bound $G(L+1)$.

$$P(e(s_{v_j}) \mid \text{homepage}, s_{f_i}) \approx G(L+1) \quad \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \in d_L(s_{f_i}), L \geq 1 \tag{9}$$

Example 4. Given the service portal shown in Figure 3, since $s_7 \in d_2(s_1)$, it takes at least two clicks to navigate from online service A (i.e., $e(s_1)$) to online service G (i.e., $e(s_7)$). There is still one click from the homepage to online service A. Hence, it takes at least three clicks to navigate from service-link s_1 featured in the homepage to online service G. According to Eq. (9), $P(e(s_7) \mid \text{homepage}, s_1) \approx G(3)$, in which $G(3)$ can be calculated using Eqs. (5) and (3).

If $s_{v_j} \notin d_L(s_{f_i})$, for all $L \geq 1$, $e(s_{v_j})$ is not navigable from $e(s_{f_i})$. Hence

$$P(e(s_{v_j}) \mid \text{homepage}, s_{f_i}) = 0 \qquad \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \notin d_L(s_{f_i}), \text{ for all } L \geq 1. \tag{10}$$

Example 5. In Figure 3, $s_8 \notin d_L(s_2)$, for all $L \geq 1$. Hence $e(s_8)$ (i.e., online service H) is not navigable from $e(s_2)$ (i.e., online service B) and $P(e(s_8) \mid \text{homepage}, s_2) = 0$.

In summary,

$$P(e(s_{v_j}) \mid \text{homepage}) = \begin{cases} 1 & \text{if } s_{v_j} \in S_f \\ \frac{\sum_{\forall s_{f_i} \in S_f} P(e(s_{v_j}) \mid \text{homepage}, s_{f_i})}{|S_f|} & \text{if } s_{v_j} \notin S_f, \end{cases} \tag{11}$$

and

$$P(e(s_{v_j}) \mid \text{homepage}, s_{f_i}) \begin{cases} = 0 & \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \notin d_L(s_{f_i}) \text{ for all } L \geq 1 \\ \approx G(L+1) & \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \in d_L(s_{f_i}),\ L \geq 1. \end{cases} \tag{12}$$

3.3.2 $P(e(s_{v_j}) \mid e(s))$. We denote $P(e(s_{v_j}) \mid e(s))$ as the probability of finding online service $e(s_{v_j})$, given that the search starts from online service $e(s)$, where $e(s_{v_j}) \in E_v$ for $j = 1, 2, \cdots k$, $s_{v_j} \in S$, $s \in S$ and $s_{v_j} \neq s$. If $s_{v_j} \in S_f$, according to

Assumption 1,

$$P(e(s_{v_j}) \mid e(s)) = 1 \qquad \text{if } s_{v_j} \in S_f. \tag{13}$$

If $s_{v_j} \notin S_f$, and $s_{v_j} \in d_L(s), L \geq 1$, according to Eq. (2), it takes at least L clicks to navigate from online service $e(s)$ to online service $e(s_{v_j})$. Similar to the discussion leading to Eq. (9), we approximate $P(e(s_{v_j}) \mid e(s))$ using its upper bound $G(L)$.

$$P(e(s_{v_j}) \mid e(s)) \approx G(L) \quad \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \in d_L(s), L \geq 1 \tag{14}$$

Example 6. For the service portal shown in Figure 3, the probability of finding online service G, given that the search starts from online service B, is $P(e(s_7) \mid e(s_2)) \approx G(3)$ because $s_7 \in d_3(s_2)$.

If $s_{v_j} \notin S_f$ and $s_{v_j} \notin d_L(s)$, for all $L \geq 1$, $e(s_{v_j})$ is not navigable from $e(s)$. Hence

$$P(e(s_{v_j}) \mid e(s)) = 0 \quad \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \notin d_L(s), \text{ for all } L \geq 1. \tag{15}$$

In summary,

$$P(e(s_{v_j}) \mid e(s)) \begin{cases} = 1 & \text{if } s_{v_j} \in S_f \\ \approx G(L) & \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \in d_L(s), \quad L \geq 1 \\ = 0 & \text{if } s_{v_j} \notin S_f \text{ and } s_{v_j} \notin d_L(s) \text{ for all } L \geq 1 \end{cases}. \tag{16}$$

3.3.3 $eff(S_f, E_v)$. For an online service $e(s_{v_j})$ in E_v, we denote $I(e(s_{v_j}))$ as an indicator variable for the event of finding $e(s_{v_j})$ during the course of searching for E_v from the homepage of a service portal.

$$I(e(s_{v_j})) = \begin{cases} 1 & \text{if finding } e(s_{v_j}) \\ 0 & \text{if not finding } e(s_{v_j}) \end{cases} \qquad e(s_{v_j}) \in E_v, \text{ for } j = 1, 2, \cdots k, k \geq 1 \tag{17}$$

Moreover, $eff(S_f, E_v)$, namely the expected number of online services in E_v located by navigating from the homepage featuring S_f, is

$$eff(S_f, E_v) = \sum_{j=1}^{k} E[I(e(s_{v_j}))]. \tag{18}$$

Let $p(e(s_{v_j}))$ be the probability of finding $e(s_{v_j})$ during the course of searching for E_v. We have

$$eff(S_f, E_v) = \sum_{j=1}^{k} p(e(s_{v_j})). \tag{19}$$

We start from calculating $p(e(s_{v_1}))$. The search for online services in E_v starts from the homepage and $e(s_{v_1})$ is the first online service sought. Hence, the search for $e(s_{v_1})$ starts from the homepage and

$$p(e(s_{v_1})) = P(\text{e}(s_{v_1}) \mid \text{homepage}). \tag{20}$$

In Eq. (20), $P(\text{ e}(s_{v_1}) \mid \text{homepage})$ can be calculated using Eqs. (11) and (12). Next, we consider calculating $p(e(s_{v_2}))$, namely the probability of locating the second

online service in E_v. If $e(s_{v_1})$has been located, the search for $e(s_{v_2})$ starts from $e(s_{v_1})$. Hence, the probability of finding $e(s_{v_2})$, given that $e(s_{v_1})$ has been located, is $P(e(s_{v_2}) \mid e(s_{v_1}))$. If $e(s_{v_1})$ is not found, there is a probability t of restarting a fresh search for $e(s_{v_2})$ from the homepage. With the remaining probability of $1 - t$, a surfer could become frustrated and give up searching or try to locate online services using search engines, etc.[4] We have

$$p(e(s_{v_2})) = p(e(s_{v_1}))P(e(s_{v_2}) \mid e(s_{v_1})) + [1 - p(e(s_{v_1}))]tP(e(s_{v_2}) \mid \text{homepage}). \quad (21)$$

In Eq. (21), $P(e(s_{v_2}) \mid e(s_{v_1}))$ can be calculated using (16) and$P(e(s_{v_2}) \mid \text{homepage})$ can be calculated using (11) and (12).

Generalizing Eq. (21), the probability $p(e(s_{v_j}))$ of finding $e(s_{v_j})$, $j \geq 2$, is

$$p(e(s_{v_j})) = p(e(s_{v_{j-1}}))P(e(s_{v_j}) \mid e(s_{v_{j-1}})) + [1 - p(e(s_{v_{j-1}}))]tP(e(s_{v_j}) \mid \text{homepage}). \quad (22)$$

In Eq. (22), $P(\text{e}(s_{v_j}) \mid e(s_{v_{j-1}}))$ can be calculated using (16) and$P(\text{e}(s_{v_j}) \mid \text{homepage})$ can be calculated using (11) and (12). Applying (20), (22), and (19), we get $eff(S_f, E_v)$.

Example 7. For the service portal shown in Figure 3, given E_v = {online service F, online service G} = $\{e(s_6), e(s_7)\}$, according to Eq. (20), the probability of finding online service F is $p(e(s_6)) = P(\text{e}(s_6) \mid \text{homepage})$. Applying (11) and (12),

$$P(\text{e}(s_6) \mid \text{homepage}) = \frac{P(e(s_6) \mid \text{homepage}, s_1) + P(e(s_6) \mid \text{homepage}, s_2) + P(e(s_6) \mid \text{homepage}, s_3)}{3} \approx \frac{G(3) + G(2) + G(3)}{3}.$$

According to (22), the probability of finding online service G is

$$p(e(s_7)) = p(e(s_6))P(e(s_7) \mid e(s_6)) + [1 - p(e(s_6))]tP(e(s_7) \mid \text{homepage}).$$

Applying (16), $P(e(s_7) \mid e(s_6)) \approx G(2)$. Applying (11) and (12), we get

$$P(e(s_7) \mid \text{homepage}) = \frac{P(e(s_7) \mid \text{homepage}, s_1) + P(e(s_7) \mid \text{homepage}, s_2) + P(e(s_7) \mid \text{homepage}, s_3)}{3} \approx \frac{G(3) + G(4) + G(2)}{3}.$$

According to (19), $eff(S_f, E_v) = p(e(s_6)) + p(e(s_7))$.

Here $eff(S_f, E_v)$measures the effectiveness of S_f for locating online services in E_v, where E_v is a sequence of online services sought by a user during one visit to a service portal. For a web log recording a large number of visits, log,

[4]Specifically, $eff(S_f, E_v)$ measures the effectiveness of service links featured in the homepage in helping users find online services. Hence, only the probability of restarting a fresh search from the homepage t is considered in Eq. (21). Although determining the value of t is out of the scope of the article, we show in Section 5 that our method outperforms expert selection under a series of different values of t.

Input: S, the set of service-links provided in a service portal
N, the number of service-links to be featured in the homepage
log, a training web log,
Output: S_f, a set of selected service-links

step 1: extracting structure relationships between service-links
step 2: preprocessing *log*
step 3: discovering service visiting patterns
step 4: determining weights of service-links
step 5: determining weights of service-link pairs
step 6: selecting service-links

Fig. 4. ServiceFinder.

$eff(S_f, \log)$ measures the effectiveness of S_f for locating online services in a web log, and

$$eff(S_f, \log) = \sum_{\text{for all } E_v \text{ inlog}} eff(S_f, E_v). \tag{23}$$

Definition (*The Service Selection Problem*). Given a set of service-links provided in a service portal S, the number of service-links to be featured in the homepage of the portal N, and a web log *log*, the service selection problem is to select N service-links from S such that the result of the selection S_f maximizes $eff(S_f, log)$.

4. SERVICEFINDER

In this section, we present a heuristic solution, namely ServiceFinder, to the service selection problem. Figure 4 illustrates the sketch of ServiceFinder. In the following sections, we first describe steps 1, 2, and 3, which serve as the basis of ServiceFinder. We then discuss the rationale behind the design of ServiceFinder, as well as steps 4, 5, and 6.

4.1 Extracting Structure Relationships Between Service-Links

For each service-link s,$s \in S$, its structurally related service-links $R(s)$ are discovered by parsing the webpages in a service portal. Then $d_L(s)$ is derived using Eqs. (1) and (2).

4.2 Preprocessing Web Log

In this step, a training web log is converted into a set of E'_vs, where each E_v is a sequence of online services sought by a user during one visit to a service portal. A training web log is first cleaned by removing error logs and accessory logs. Error logs record failed web accesses and usually have status code ≥ 400. Accessory logs record the requests associated with a webpage request, such as a request for a picture in a webpage. Next, the cleaned web log is divided into sessions, where each session represents a visit to a service portal, using the widely applied rule of thumb stipulating the maximal session length cannot exceed 30 minutes [Cooley et al. 1999; Spiliopoulou et al. 2003]. Finally, $E_v$is extracted from each session by filtering out: (1) visits other than online service

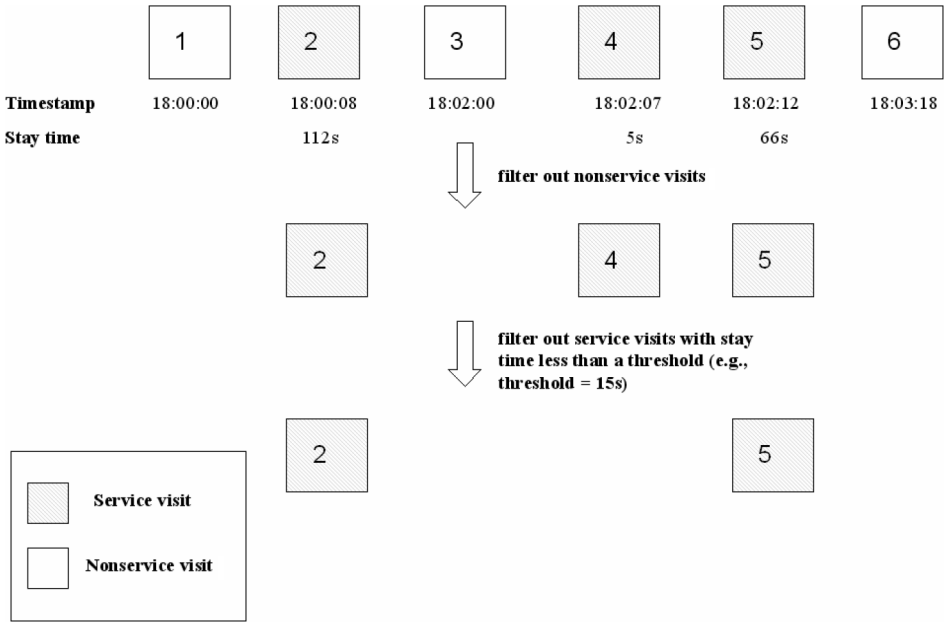

Fig. 5. The process of extracting E_v from a session: an illustrative example.

visits; and (2) online service visits with stay time less than a certain threshold.[5] Figure 5 illustrates the process of extracting E_v from a session. The pages visited in a session and their timestamps are shown in Figure 5. The stay time of a page is calculated as the difference between the timestamp of the page and the timestamp of the page visited next.

4.3 Discovering Service Visiting Patterns

In this step, the sequential visiting patterns of online services are discovered from the set of E'_vs resulting from step 2. Particularly, we mine large 2-consecutive-sequences. A 2-consecutive-sequence $< e_i, e_t >$ is a sequence of 2 online services visited in consecutive order, where e_i is called the initial online service and e_t the terminal online service. A 2-consecutive-sequence $< e_i, e_t >$ is said to be contained in an $E_v =< e(s_{v_1}), e(s_{v_2}), \ldots, e(s_{v_k}) >$ if there exists m, $1 \leq m \leq k-1$, such that $e_i = e(s_{v_m})$ and $e_t = e(s_{v_{m+1}})$. The support of a 2-consecutive-sequence sup$(< e_i, e_t >)$ is the fraction of the number of E'_vs that contain $< e_i, e_t >$ out of the total number of E'_vs. A large 2-consecutive-sequence is a 2-consecutive-sequence which satisfies a minimum support constraint.

Agrawal and Srikant [1995] proposed efficient algorithms for discovering large sequences from a database of customer transactions. Chen et al. [1998] introduced algorithms to mine path-traversal patterns from a web log. This research adapts the AprioriAll algorithm proposed in Agrawal and Srikant

[5]Some online services may be visited because of their location, rather than the service they provide. For example, an online service may be visited simply because it is on the path to some other online service. Condition (2) is used to filter out this kind of online service visit.

Table II. Four Categories of Service-Links

Condition (1)	Condition (2)	Category
satisfied	satisfied	category-1 service-links of s, $C_1(s)$
not satisfied	satisfied	category-2 service-links of s, $C_2(s)$
satisfied	not satisfied	category-3 service-links of s, $C_3(s)$
not satisfied	not satisfied	category-4 service-links of s, $C_4(s)$

[1995] to discover large 2-consecutive-sequences. The AprioriAll algorithm is modified to increment the count of a 2-consecutive-sequence only if the online services in the sequence appear in an E_v in consecutive order. Readers are referred to Agrawal and Srikant [1995] for the AprioriAll algorithm.

4.4 Categories of Service-Links

Based on the results from the first three steps, for a service-link s in S, any service-link s_o in $S - \{s\}$ can be classified into one of the four categories shown in Table II, according to the following conditions: (1) whether s_o is an element of $R(s)$; and (2) whether there exists a large 2-consecutive-sequence with $e(s)$ as the initial online service and $e(s_o)$ as the terminal online service. For example, if both conditions (1) and (2) are satisfied, s_o belongs to $C_1(s)$.

The category-1 service-links of s, denoted by $C_1(s)$, are service-links structurally related to s. Further, there exists a pattern of visiting an online service pointed to by a service-link s_o in $C_1(s)$ right after visiting $e(s)$ with a probability of $\sup(< e(s), e(s_o) >)$. The relationship between a service-link in $C_1(s)$ and s, which was considered uninteresting and overlooked in Perkowitz and Etzioni [2000], is employed in this research as a factor in determining the weights of service-links (see Section 4.5). Category-2 service-links of s, denoted $C_2(s)$, are service-links not structurally related to s. However, there exists a pattern of visiting an online service pointed to by a service-link s_o in $C_2(s)$right after visiting $e(s)$ with a probability of $\sup(< e(s), e(s_o) >)$. In this research, the relationship between a service-link in $C_2(s)$ and s is used in determining the weights of service-link pairs (see Section 4.6). Category-3 service-links of s, denoted $C_3(s)$, are service-links structurally related to s, but with no pattern of visiting an online service pointed to by a service-link in $C_3(s)$ right after visiting $e(s)$. The relationship between a service-link in $C_3(s)$ and s reveals a possible design problem associated with the webpage of $e(s)$, that is, there exist infrequently visited service-links (i.e., service-links in $C_3(s)$) in that webpage. Since ServiceFinder focuses on selecting service-links for the homepage of a service portal, we leave this finding as a future research topic. Category-4 service-links of s, denoted $C_4(s)$, do not reveal any patterns, and hence are not discussed.

Example 8. Given the service portal shown in Figure 3, and large 2-consecutive-sequences $< e(s_3), e(s_7) >$ and $< e(s_3), e(s_8) >$, we have $R(s_3) = \{s_2, s_7\}$. According to Table II, $s_7 \in C_1(s_3)$, $s_8 \in C_2(s_3)$,$s_2 \in C_3(s_3)$, and $s_4 \in C_4(s_3)$.

4.5 Determining Weights of Service-Links

In this step, we determine the weight $w(s)$ of a service-link s in S. The higher the weight of a service-link, the higher the priority of the service-link being

selected for the homepage. Selecting s for the homepage makes $e(s)$ easily located. Obviously, a service-link pointing to a frequently sought online service is favored for inclusion in the homepage, as the purpose of service selection is to enable as many user-sought online services to be easily located as possible. We denote $v(e(s))$ as the visiting rate of $e(s)$, which is the fraction of the number of $e(s)$ visits out of the total number of online service visits. Moreover, $v(e(s))$ can be easily derived from a preprocessed web log and serves as one factor in determining $w(s)$.

Selecting a service-link s for the homepage should not only benefit the search for $e(s)$, but also the search for online services other than $e(s)$. Therefore, another factor determining $w(s)$ is how easily online services other than $e(s)$ could be located if s were selected. Putting $s$in the homepage makes the online services pointed to by service-links in $\cup_{L\geq 1} d_L(s)$navigable from the homepage. Since the online services pointed to by service-links featured in the homepage (i.e., service-links in S_f) can be easily located, we are specifically interested in how the selection of s benefits the search for online services pointed to by service-links in $\cup_{L\geq 1} d_L(s) - S_f$. The greater the number of service-links in $\cup_{L\geq 1} d_L(s) - S_f$, the greater also the number of online services navigable from the homepage if s were selected, hence, the higher the weight $w(s)$ of s. The quality of the online services pointed to by service-links in $\cup_{L\geq 1} d_L(s) - S_f$, such as the visiting rates of these online services and their distances from the homepage, also affects $w(s)$. The higher the visiting rates of these online services, which indicates that selection of s enables more frequently sought online services navigable from the homepage, the higher the weight $w(s)$ of s. The shorter the distances from the homepage to these online services, the easier and more likely of being located these online services, hence the higher the weight $w(s)$ of s.

Based on the preceding discussion, we define the weight $w(s_{f_i})$ of a selected service-link s_{f_i}, where $s_{f_i} \in S_f$, for $i = 1, 2, \cdots N$ as

$$w(s_{f_i}) = v(e(s_{f_i})) + \sum_{\forall k \in (\bigcup\limits_{L\geq 1} d_L(s_{f_i}) - S_f)} G(X(k)+1)v(e(k)). \tag{24}$$

In Eq. (24), $X(k)$ is the distance from $e(s_{f_i})$ to$e(k)$, where $k \in (\bigcup\limits_{L\geq 1} d_L(s_{f_i}) - S_f)$, and $X(k) = L$ if $k \in d_L(s_{f_i}), L \geq 1$. Here $X(k)+1$ is the distance from the homepage to $e(k)$, and $G(X(k)+1)$ and $v(e(k))$ represent the quality of $e(k)$. According to (24), a service-link s_{f_i} has higher weight $w(s_{f_i})$, if visiting rate is higher (i.e., higher $v(e(s_{f_i}))$), if selection of s_{f_i} enables more online services navigable from the homepage (i.e., more service-links in $\cup_{L\geq 1} d_L(s_{f_i}) - S_f$), and if the quality of these navigable online services is higher (i.e., smaller $X(k)$ and higher $v(e(k))$).

Example 9. Given the service portal shown in Figure 3, if $S_f = \{s_1, s_2, s_3\}$, we have

$$\bigcup_{L\geq 1} d_L(s_1) - S_f = \{s_4, s_5, s_6, s_7, s_8\}.$$

Given $v(e(s_i)) = 0.1$ for $i = 1, 2, 3, 6, 7, 8$, and $v(e(s_j)) = 0.2$ for$j = 4, 5$ and

applying Eq. (24), we have

$$w(s_1) = v(e(s_1)) + G(2)[v(e(s_4)) + v(e(s_5))] + G(3)[v(e(s_6)) + v(e(s_7)) + v(e(s_8))] = 0.66.^6$$

Similarly, we get $w(s_2) = 0.23$ and $w(s_3) = 0.26$. If $S_f = \{s_6, s_7, s_8\}$, applying (24), we have $w(s_6) = 0.26$, $w(s_7) = 0.1$, and $w(s_8) = 0.1$. Based on calculated weights, $\{s_1, s_2, s_3\}$ is preferable for selection for the homepage over $\{s_6, s_7, s_8\}$, although the visiting rates of service-links in both sets are the same. The reason is that selecting $\{s_1, s_2, s_3\}$ enables more online services that are navigable from and closer to the homepage than selecting $\{s_6, s_7, s_8\}$.

Next, we refine (24) using the category-1 service-links discussed in Section 4.4. If $s_{f_i} \in C_1(s_{f_j})$, where $s_{f_i}, s_{f_j} \in S_f$ and $s_{f_i} \neq s_{f_j}$, according to Table II, $s_{f_i} \in R(s_{f_j})$ and there exists a large 2-consecutive-sequence $< e(s_{f_j}), e(s_{f_i}) >$ with support $\sup(< e(s_{f_j}), e(s_{f_i}) >)$. Since $s_{f_i} \in R(s_{f_j})$, after visiting $e(s_{f_j})$, s_{f_i} is at a web surfer's fingertips and $e(s_{f_i})$ can be easily located. In this regard, given that s_{f_j} is selected for the homepage, it is unnecessary to place s_{f_i} in the homepage for visits containing the sequence $< e(s_{f_j}), e(s_{f_i}) >$. Hence, $\sup(< e(s_{f_j}), e(s_{f_i}) >)$ can be deducted from weight $w(s_{f_i})$, if $s_{f_i} \in C_1(s_{f_j})$. We refine Eq. (24) as

$$w(s_{f_i}) = v(e(s_{f_i})) + \sum_{\forall k \in (\bigcup_{L \geq 1} d_L(s_{f_i}) - S_f)} G(X(k) + 1) v(e(k)) - \sum_{\forall s_{f_j} \text{ such that } s_{f_i} \in C_1(s_{f_j})} \sup(e(s_{f_j}), e(s_{f_i})). \tag{25}$$

Example 10. Let us refine $w(s_2)$ calculated in Example 9. Given $s_2 \in C_1(s_3)$ and $\sup(< e(s_3), e(s_2) >) = 0.03$, according to (25), we have $w(s_2) = 0.23 - 0.03 = 0.2$.

4.6 Determining Weights of Service-Link Pairs

In this step, we determine the weight $w(\{s_i, s_j\})$ of a service-link pair $\{s_i, s_j\}$. The higher the weight of a service-link pair, the higher the priority of the service-link pair being selected for the homepage. For a service-link pair $\{s_i, s_j\}$, if $s_j \in C_2(s_i)$, then according to Table II, there exists a large 2-consecutive-sequence $< e(s_i), e(s_j) >$ with support $\sup(< e(s_i), e(s_j) >)$ and $s_j \notin R(s_i)$. Since $s_j \notin R(s_i)$, in order to navigate from $e(s_i)$ to $e(s_j)$, visitors have to explore a service portal to find the path. This creates inconvenience for web surfing and the situation becomes worse, since $e(s_i)$ and $e(s_j)$ are frequently visited in sequence. In contrast, if s_i and s_j were placed in the homepage, both $e(s_i)$ and $e(s_j)$ could easily be found from the homepage. Therefore, we set the weight $w(\{s_i, s_j\})$ of a service-link pair $\{s_i, s_j\}$ as

$$w(\{s_i, s_j\}) = \sup(< e(s_i), e(s_j) >) \quad \text{if } s_j \in C_2(s_i), \tag{26}$$

and we set the weights of all other service-link pairs to be 0.

[6]We calculate $G(.)$ using the mean and variance provided in Section 3.2. In this example, $G(2) = 0.91$ and $G(3) = 0.66$.

4.7 Selecting Service-Links

The weight of a service-link and that of a service-link pair, defined in previous sections, reflect their contributions to facilitating users' search of online services. On the other hand, the objective of the service selection problem described in Definition 3 is to maximize the expected number of online services located by users. Hence, the service selection problem can be approximated as follows.

Given a set of service-links provided in a service portal S, and the number of service-links to be featured in the homepage of the portal N, select N service-links from S such that the result of the selection S_f maximizes $w(S_f)$, where $w(S_f)$ is

$$w(S_f) = \sum_{\forall s_{f_i} \in S_f} w(s_{f_i}) + \sum_{\forall \{s_{f_i}, s_{f_j}\} \subset S_f} w(\{s_{f_i}, s_{f_j}\}). \tag{27}$$

It is computationally too expensive to enumerate all combinations of several service-links from a pool of several hundred and to find the one that maximizes $w(S_f)$. We apply a genetic algorithm to search for suboptimal $w(S_f)$ efficiently. Genetic algorithms, a type of evolutionary computing, were developed based on Darwinian survival-of-the-fittest theory and the principle of genetics [Chen et al. 1998a, 1998b; Goldberg 1989; Holland 1976]. A population of in which each individual represents a potential solution is first initiated. A fitness function is defined to evaluate the adequacy of each potential solution. This population undergoes a set of genetic operations known as crossover and mutation. Crossover is a high-level process that aims at exploitation, while mutation is a unary process that aims at exploration. Individuals strive for survival based on a selection scheme that is biased toward selecting fitter individuals (i.e., individuals that represent better solutions). The selected individuals form the next generation and the process continues. After some number of generations, the program converges and the best individual represents the best solution (according to the fitness function) found by the program.

In our implementation, each individual (chromosome) was represented as a potential S_f. Each chromosome consists of a number of integers which correspond to the IDs of the selected service-links. The length of the chromosome is equal to the number of service-links to be selected. The fitness function is defined as the function we want to maximize: $w(S_f)$. Crossover is implemented as in other genetic algorithms, in which chromosomes will be paired up and a crossover point randomly chosen. The two chromosomes will exchange their genes (i.e., the integers) to the right of the crossover point. In our implementation, each chromosome will also be checked for validity after crossover. If a chromosome which has performed crossover contains two duplicate integers (i.e., a set contains two duplicate service-links), the crossover process will be reversed and both chromosomes involved in the crossover will be reverted to their original values. In mutation, an integer randomly chosen will be mutated to a random integer. We also perform checking to make sure that the mutated gene will not be the same as any existing integer in the chromosome.

5. PERFORMANCE EVALUATION

ServiceFinder was tested using data collected from Utah.gov. Utah.gov, which was named the best state government service portal in the U.S. in 2003 [Center for Digital Government 2003], currently provides 145 online services ranging from citizen (e.g., Renew Vehicle Registration) to business (e.g., Businesses Entity Search) and government-to-government (e.g., Federal Surplus Property Search) services. To ensure the validity and reliability of the experiment, we ran it using web logs collected in two separate months (i.e., November 2003 and May 2004, respectively). Web logs collected in November 2003, which recorded visits to Utah.gov from November 1st, 2003 through November 30th, 2003, were transformed into a set of 30,691 E'_vs according to the procedure described in Section 4.2. Each E_v represented a sequence of online services visited in a session. Web logs collected in May 2004, which recorded visits to Utah.gov from May 12th, 2004 through May 31st, 2004,[7] were transformed into a set of 26,722 E'_vs using the same procedure.

To ensure the reliability of the experimental results, we applied K-fold cross-validation in the experiment. The set of E'_vs was divided into K equally-sized segments. Each validation test used each of the K segments in turn as the test data and used the remaining (K-1) segments as the training data. Specifically, we applied 30-fold cross-validation to the web logs collected in November 2003 and 20-fold cross-validation to those collected in May 2004. Each segment consisted of records of one day of online service visits. The purpose of the experiment is to find:

(1) whether ServiceFinder outperforms the current practice of service selection, namely expert selection, and if so, how much better ServiceFinder is;
(2) whether ServiceFinder outperforms LinkSelector [Fang and Liu 2004] and PageGather [Perkowitz and Etzioni 2000], and if so, how much better ServiceFinder is; and
(3) how close the performance of ServiceFinder is to the optimal solution to the service selection problem.

5.1 Performance Comparison with Expert Selection

Current practice of service selection, namely expert selection, selects and features six service-links in the homepage of Utah.gov. Among the six featured service-links, three are fixed and the other three randomly selected from the pool of all the service-links provided in Utah.gov (except for the three fixed ones) whenever Utah.gov is accessed. The three fixed service-links were determined through surveys and experimental studies of several small groups of users. Table III lists the service-links selected by expert selection.

In each validation test, we applied ServiceFinder to the training data to select six service-links. Table III lists the service-links selected by ServiceFinder in one validation test. For the genetic algorithm, the same parameters were

[7]Due to technical problems at the web server of Utah.gov, only 20 days of the May 2004 web logs were kept at the web server. However, the data is still large enough for our experiment.

Table III. Service-Links Selected by Expert Selection and ServiceFinder

Expert Selection	ServiceFinder
Renew Vehicle Registration	Tax Commission
Lookup Occupational and Professional Licensees	Business Name Availability
Find Great Employees	Annual Business Renewal
Randomly Selected	Titles, Liens, and Registration Search
Randomly Selected	Statewide Calendar
Randomly Selected	Business Entity Search

Table IV. Parameters Used in the Genetic Algorithm

Crossover rate	0.25
Mutation rate	0.02
Number of chromosomes	1000
Number of generations	1000

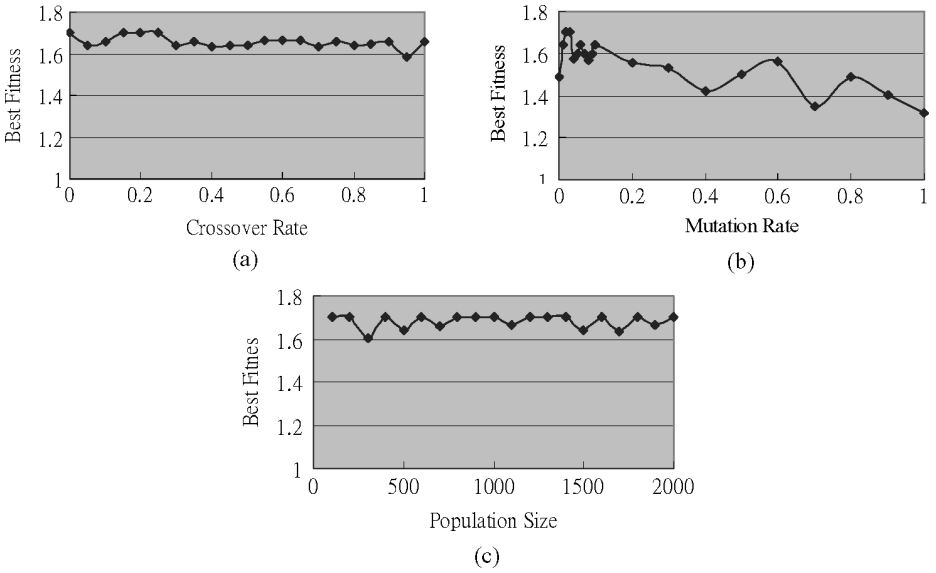

Fig. 6. The best fitness value obtained by the genetic algorithm versus: (a) crossover rate; (b) mutation rate; and (c) population size.

used for every validation test. Table IV lists the parameters used. These parameters were chosen by the following procedure. First, one set of training data was randomly chosen. Preliminary testing of the genetic algorithm was then performed on this data. Different values of population size, crossover rate, and mutation rate were used and the best fitness value in each setting was observed. We observed that the best fitness score was found in the first 1,000 generations in most cases, so the number of generations was fixed to be 1,000. We identified the values of the parameters in the setting which performed best for this test set, and these values were used for all the validation tests. This procedure can be automated when the algorithm is applied to other sites.

In Figure 6, we show the performance of the algorithm by varying one parameter at-a-time while keeping the other parameters constant. In the charts, we

can see that the algorithm was relatively less sensitive to changes in crossover rate or population size. From Figure 6(a), we can see that the fitness score was highest when the crossover rate was in the range 0.15 to 0.25. The performance of the algorithm slightly worsened when crossover rate increased. Figure 6(b) shows that the best fitness score obtained by the algorithm peaked when the mutation rate was 0.02 and 0.03. Performance deteriorated when the mutation rate increased. This may be explained by the fact that a high mutation rate resulted in chromosomes that were more random, and the algorithm thus became more like a random search. Figure 6(c) shows that the performance was less sensitive to changes in population size (i.e., number of chromosomes). The small fluctuations in the curve were mostly due to the random nature of genetic algorithms.

We first report an effectiveness comparison between ServiceFinder and expert selection across 30 validation tests using web logs collected in November 2003, where the probability t of restarting a fresh search is set at 0.5. Compared with expert selection, ServiceFinder increases effectiveness by an average of 612 and relatively improves effectiveness by an average of 295%. Each test data in the 30 validation tests consists of the records of one day of online service visits. Hence, the enhancement of ServiceFinder indicates that around 612 more user-sought online services per day could be located if service-links featured in the homepage of Utah.gov were recommended by ServiceFinder, rather than expert selection. Dividing effectiveness by the number of user-sought online services per day, we normalize effectiveness onto the range of [0,1]. The average normalized effectiveness of ServiceFinder is 0.73, which indicates that an average of 73% of user-sought online services could be located by clicking through hyperlinks if the service-links featured in the homepage of Utah.gov were recommended by ServiceFinder, whereas the average normalized effectiveness of expert selection is 0.19.

Increasing t from 0.0 to 1.0 with a step of 0.1, we compared performance between ServiceFinder and expert selection. As shown in Figure 7, ServiceFinder consistently outperforms expert selection when t is increased from 0.0 to 1.0. The relative improvements of effectiveness range from 284% to 306%. The improvements are statistically significant at the 0.001 level using the paired-t test.

Similar performance comparison results were observed when using the May 2004 web log. Compared with expert selection, ServiceFinder relatively improves effectiveness by an average of 266% when the probability t of restarting a fresh search is set at 0.5. ServiceFinder consistently outperforms expert selection when $t$is increased from 0.0 to 1.0. The relative improvements of effectiveness range from 252% to 268%. The improvements are statistically significant at the 0.001 level using the paired-t test.

The enhancement of ServiceFinder is not surprising. First, the improvement is attributed to the design of ServiceFinder, which integrates the structure of a service portal and previous service visiting patterns in selecting service-links. For example, the online services Business Name Availability and Annual Business Renewal are frequently visited in sequence. However, service-links to these online services are not structurally related. Placing both service-links

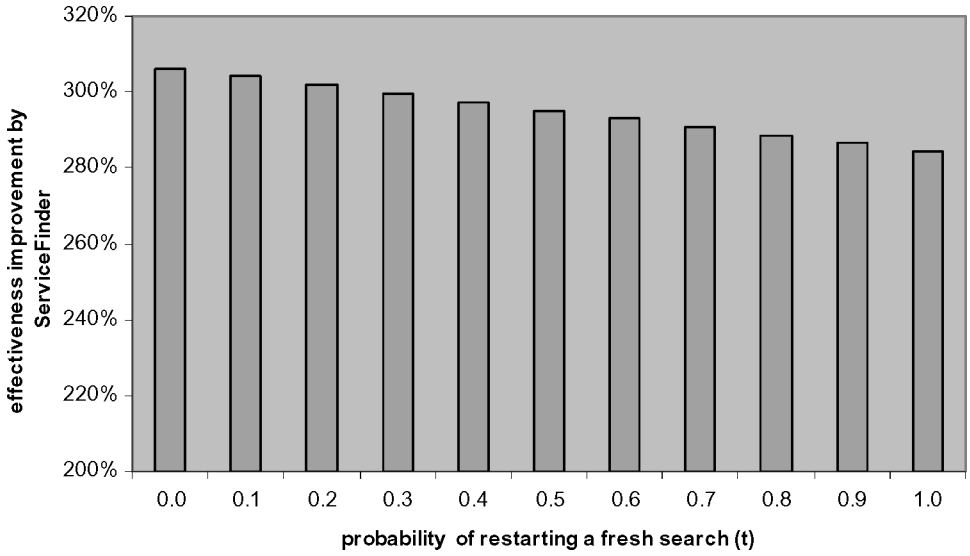

Fig. 7. Percentage effectiveness improvement due to ServiceFinder over expert selection.

in the homepage would save web surfers' efforts in finding a path from one online service to the other. Applying Eq. (28), ServiceFinder assigns higher weights to both service-links and selects them. Second, according to domain experts from Utah.gov, the three fixed service-links recommended by expert selection had not been changed for six months due to the cost and inflexibility of user survey and experimental study. On the other hand, it is easy and flexible to apply ServiceFinder to recommend service-links based on most recent user visiting records and most recent website structure. Further, the other three service-links recommended by expert selection are chosen randomly without considering either user visiting patterns or website structure.

5.2 Performance Comparison with PageGather and LinkSelector

We compared the performance of ServiceFinder with that of PageGather [Perkowitz and Etzioni 2000], a classical algorithm for adaptive website design. In particular, we compared ServiceFinder with PageGather-CC, a variant of PageGather with the best performance among all the variants of PageGather [Perkowitz and Etzioni 2000]. Interested readers are directed to Perkowitz and Etzioni [2000] for the details of PageGather-CC. We first report an effectiveness comparison between ServiceFinder and PageGather-CC across 30 validation tests using web logs collected in November 2003, where t is set to be 0.5. On average, ServiceFinder relatively improves effectiveness by 10.3%. Normalizing effectiveness onto the range of [0,1], the average normalized effectiveness of ServiceFinder is 0.73, while the average normalized effectiveness of PageGather-CC is 0.66.

Increasing t from 0.0 to 1.0 with a step of 0.1, we compared performance between ServiceFinder and PageGather-CC. As shown in Figure 8, ServiceFinder consistently outperforms PageGather-CC when t is increased from 0.0 to 1.0.

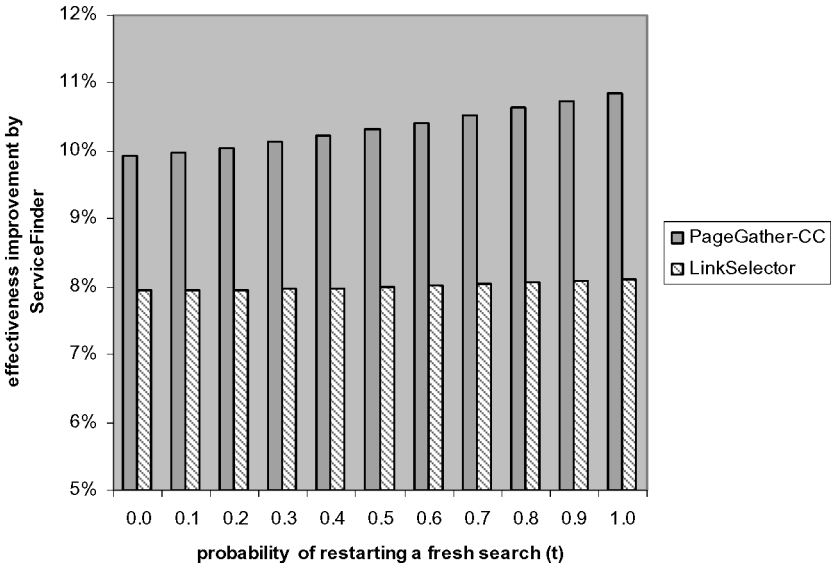

Fig. 8. Percentage effectiveness improvement due to ServiceFinder over PageGather-CC and LinkSelector.

The relative improvements of effectiveness range from 9.9% to 10.8%. These improvements are statistically significant at the 0.001 level using the pairwise t-test. Similar performance comparison results were observed when using the May 2004 web log. Detailed results of the experiment are omitted for the sake of brevity.

We also compared the performance of ServiceFinder to that of LinkSelector [Fang and Liu 2004]. Setting the probability t of restarting a fresh search at 0.5, we compared effectiveness between ServiceFinder and LinkSelector across 30 validation tests using web logs collected in November 2003. Compared with LinkSelector, ServiceFinder relatively improves effectiveness by an average of 8.0%. Normalizing effectiveness onto the range of [0,1], the average normalized effectiveness of ServiceFinder is 0.73, which indicates that on average, 73% of user-sought online services could be located by clicking through hyperlinks if the service-links featured in the homepage of Utah.gov were recommended by ServiceFinder, while the average normalized effectiveness of LinkSelector is 0.67. Given the large number of user-sought online services at a service portal (e.g., 33,725 user-sought online services at Utah.gov in November 2003), the performance enhancement of ServiceFinder over LinkSelector is significant in facilitating users locating their desired online services.

Increasing t from 0.0 to 1.0 with a step of 0.1, we compared performance between ServiceFinder and LinkSelector. As shown in Figure 8, ServiceFinder consistently outperforms LinkSelector when t is increased from 0.0 to 1.0. The relative improvements of effectiveness range from 7.9% to 8.1%. The improvements are statistically significant at the 0.001 level using the paired-t test. Similar performance comparison results were obtained when using the May 2004 web log. Detailed results of the experiment are omitted for the sake of brevity.

The performance improvement of ServiceFinder over PageGather and LinkSelector is attributed to the difference in how service-links are weighted in these methods. ServiceFinder weights service-links based on the following factors: (1) visiting rate of a service-link; (2) how much a selected service-link would benefit the search for other online services; (3) related but unlinked service-links; (4) related and linked service-links; (5) the probability of surfing depth; and (6) sequential information among online service visits. On the other hand, PageGather [Perkowitz and Etzioni 2000] assigs high weights to related but unlinked service-links and prefering them pick while LinkSelector [Fang and Liu 2004] neglects the probability of surfing depth and sequential information among online service visits.

5.3 Performance Comparison with the Optimal Solution

We define the service selection problem as an optimization problem in Definition 3. It is interesting to find out how close the performance of ServiceFinder is to that of the optimal solution. We used exhaustive search to find the optimal solution. For the case of Utah.gov, the search space consisted of 11.6 billion different combinations of service-links (i.e., $\binom{145}{6}$), which was too computationally expensive to be implemented. Hence, we reduced the service-link pool to 20 randomly selected service-links. The reduction made exhaustive search for the optimal solution implementable, while still keeping a large search space. For the reduced service-link pool, it required to enumerate 38,760 (i.e., $\binom{20}{6}$) different combinations of service-links to find the optimal solution. We also applied ServiceFinder to the reduced service-link pool. We first report an effectiveness comparison between ServiceFinder and the optimal solution across 30 validation tests using web logs collected in November 2003, where t is set to be 0.5. On average, the optimal solution outperforms ServiceFinder only by 4.7%. Increasing t from 0.0 to 1.0 with a step of 0.1, we compared the performance between ServiceFinder and the optimal solution. As shown in Figure 9, the improvement in effectiveness by the optimal solution ranges from 4.5% to 5.1%.

Setting t at 0.5 and increasing the number of selected links N from 6 to 10 with a step of 1, we compared performance between ServiceFinder and the optimal solution. As N is increases from 6 to 10, the search space for the optimal solution increased from 38,760 (i.e.,$\binom{20}{6}$) different combinations of service-links to 184,756 (i.e.,$\binom{20}{10}$). The improvement in effectiveness by the optimal solution ranges from 3.7% to 6.1% as N is increased from 6 to 10. Similar performance comparison results were obtained when using the May 2004 web log.

Extensive experimental results have shown that the performance of ServiceFinder is close to that of the optimal solution when applied to a fairly large search space (in this case, a search space with up to 184,756 possible solutions). However, it took only several minutes for ServiceFinder to recommend service-links, while it took several hours to find the optimal solution through exhaustive search.

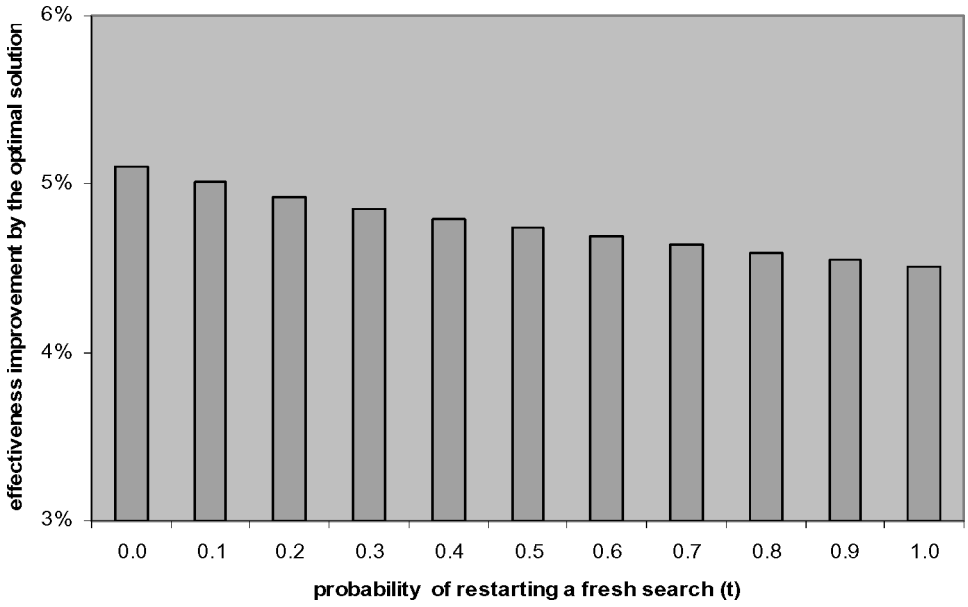

Fig. 9. Percentage effectiveness improvement due to the optimal solution over ServiceFinder.

6. CONCLUSION

In this article, we introduced an important research problem in the area of service portal design, namely that of service selection. To address this problem, we first proposed a mathematically formulated metric to measure the effectiveness of selected service-links in directing users to locate their desired online services, and formally defined the service selection problem. A solution method, ServiceFinder, was then proposed. Using real-world data obtained from Utah.gov, we showed that ServiceFinder outperforms both the current practice of service selection and previous algorithms for adaptive website design. We also showed that the performance of ServiceFinder is close to that of the optimal solution. The study has the following contributions: (1) We have formally defined an important research problem—service selection—and proposed the ServiceFinder approach to address this problem with superior performance; and (2) the metric $eff(S_f, E_v)$, uniquely integrates the structure of a service portal, the probability of user surfing depth, and user surfing behaviors recorded in web logs for measuring the effectiveness of service portal design.

Future studies are needed in the following areas: (1) User-involved experiments are needed to empirically study: (a) how much the proposed metric reflects user satisfaction with a service portal; and (b) to what extent the proposed method improves user satisfaction with a service portal. (2) Further testing of the effect of different crossover and mutation rates, as well as different crossover and mutation operations, would be useful. (3) This article studies the service selection problem from the perspective of facilitating service consumers in locating online services easily and effectively. Yet another perspective on service selection could come from service providers. For example, a government agency

might be more interested in making sure that people find tax forms, rather than forms where complaints can be submitted. Finally, yet another area worthy of exploration is to find a way to incorporate and balance both perspectives in service selection.

ACKNOWLEDGMENTS

We gratefully thank the Utah State Government for kindly providing us with the data and information used in this study. We also thank the Associate Editor and the four anonymous reviewers for their insightful suggestions.

REFERENCES

ADOMAVICIUS, G. AND TUZHILIN, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng. 17*, 6, 734–749.

AGRAWAL, R. AND SRIKANT, R. 1995. Mining sequential patterns. In *Proceedings of the 11th International Conference on Data Engineering* (Taipei, China), 3–14.

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), 6–13.

BABANOVIC, M. AND SHOHAM, Y. 1997. Content-Based collaborative recommendation. *Commun. ACM 40*, 3, 66–727.

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).

CATLEDGE, L. AND PITKOW, J. 1995. Characterizing browsing behaviors on the World Wide Web. *Comput. Netw. ISDN Syst. 27*, 6, 1065–1073.

CENTER FOR DIGITAL GOVERNMENT. 2003. Utah state portal ranks no. 1. http://www.centerdigitalgov.com/center/highlightstory.phtml?docid=69811.

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

CHAKRABARTI, S., VAN DEN BERG, M., AND DOM, B. 1999. Focused crawling: A new approach to topic-specific web resource discovery. In *Proceedings of the 8th International World Wide Web Conference* (Toronto, Canada, May).

CHEN, H., CHUNG, Y., RAMSEY, M., AND YANG, C. 1998. A smart itsy bitsy spider for the web. *J. Amer. Soc. Inf. Sci. 49*, 7, 604–618.

CHEN, H., CHUNG, Y., RAMSEY, M., AND YANG, C. 1998. An intelligent personal spider (agent) for dynamic Internet/intranet searching. *Decision Supp. Syst. 23*, 41–58.

CHEN, M., PARK J., AND YU P. 1998. Efficient data mining for path traversal patterns. *IEEE Trans. Knowl. Data Eng. 10*, 2, 209–221.

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

CZYZOWICZ, J., KRANAKIS E., KRIZANC D., PELC A., AND MARTIN M. V. 2003. Enhancing hyperlink structure for improving web performance. *J. Web Eng. 1*, 2, 93–127.

FANG, X. AND LIU SHENG, O. R. 2004. LinkSelector: A web mining approach to hyperlink selection for web portals. *ACM Trans. Internet Technol. 4*, 2, 209–237.

GOLDBERG, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley.

HOLLAND, J. H. 1976. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

HUANG, Z., CHEN, H., AND ZENG, D. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst. 22*, 1, 116–142.

HUBERMAN, B. A., PIROLLI, P. L. T., PITKOW, J. E., AND LUKOSE, R. M. 1998. Strong regularities in World Wide Web surfing. *Science 280*, 3, 95–97.

JOACHIMS, T., FREITAG, D., AND MITCHELL, T. 1997. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the International Joint Conference on Artificial Intelligence* (Nagoya, Japan), 770–775.

KLEINBERG, J. 1998. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms* (San Francisco, CA), 668–677.

KOSALA, R. AND BLOCKEEL, H. 2000. Web mining research: A survey. *ACM SIGKDD Explor. 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), 331–339.

LEVENE, M., BORGES, J., AND LOIZOU, G. 2001. Zipf's law for web surfers. *Knowl. Inf. Syst. 3*, 120–129.

LIEBERMAN, H. 1995. Letizia: An agent that assists web browsing. In *Proceedings of the International Joint Conference on Artificial Intelligence* (Quebec, Canada), 924–929.

LIEBERMAN, H., FRY, C., AND WEITZMAN, L. 2001. Exploring the web with reconnaissance agents. *Commun. ACM 44*, 8, 69–75.

LIU, J., ZHANG, S., AND YANG, J. 2004. Characterizing web usage regularities with information foraging agents. *IEEE Trans. Knowl. Data Eng. 16*, 5, 566–584.

NIELSEN, J. 2000. *Designing Web Usability*. New Riders Publishing.

NIELSEN, J. AND WAGNER, A. 1996. User interface design for the WWW. In *Proceedings of ACM Conference on Computer-Human Interaction* (British Columbia, Canada), 330–331.

PERKOWITZ, M. AND ETZIONI, O. 1997. Adaptive web sites: An AI challenge. In *Proceedings of the International Joint Conference on Artificial Intelligence* (Nogoya, Japan).

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

PITKOW, J. E. 1998. Summary of WWW characterizations. *Comput. Netw. ISDN Syst. 30*, 1-7, 551–558.

RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM P., AND RIEDL J. 1994. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW)* (Chapel Hill, NC), 175–186.

SHARDANAND, U. AND MAES, P. 1995. Social information filtering: Algorithms for automating word-of-mouth. In *Proceedings of the ACM Conference on Computer-Human Interaction (CHI)* (Denver, CO), 210–217.

SPILIOPOULOU, M., MOBASHER, B., BERENDT, B., AND NAKAGAWA, M. 2003. A framework for the evaluation of session reconstruction heuristics in web-usage analysis. *INFORMS J. Comput. 15*, 2, 171–190.

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

WOOD F. B., SIEGEL E. R., LACROIX, E.-M., LYON, B. J., BENSON, D. A., CID, V., AND FARISS, S. 2003. A practical approach to online service web evaluation. *IEEE IT Professional 5*, 3, 22–28.

Received January 2006; revised November 2006; accepted February 2007