

# Interest-Based Personalized Search

ZHONGMING MA, GAUTAM PANT, and OLIVIA R. LIU SHENG

The University of Utah

---

Web search engines typically provide search results without considering user interests or context. We propose a personalized search approach that can easily extend a conventional search engine on the client side. Our mapping framework automatically maps a set of known user interests onto a group of categories in the Open Directory Project (ODP) and takes advantage of manually edited data available in ODP for training text classifiers that correspond to, and therefore categorize and personalize search results according to user interests. In two sets of controlled experiments, we compare our personalized categorization system (PCAT) with a list interface system (LIST) that mimics a typical search engine and with a nonpersonalized categorization system (CAT). In both experiments, we analyze system performances on the basis of the type of task and query length. We find that PCAT is preferable to LIST for information gathering types of tasks and for searches with short queries, and PCAT outperforms CAT in both information gathering and finding types of tasks, and for searches associated with free-form queries. From the subjects' answers to a questionnaire, we find that PCAT is perceived as a system that can find relevant Web pages quicker and easier than LIST and CAT.

Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Dictionaries, Linguistic processing*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search process*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation (efficiency and effectiveness)*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Graphical user interfaces (GUI)*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Personalized search, user interest, user interface, World Wide Web, information retrieval, Open Directory

## ACM Reference Format:

Ma, Z., Pant, G., and Liu Sheng, O. R. 2007. Interest-based personalized search. *ACM Trans. Inform. Syst.* 25, 1, Article 5 (February 2007), 38 pages. DOI = 10.1145/1198296.1198301 <http://doi.acm.org/10.1145/1198296.1198301>.

---

This research was supported by Global Knowledge Management Center and the School of Accounting and Information Systems at the University of Utah, and eBusiness Research Center at Pennsylvania State University.

Authors' address: Zhongming Ma, Gautam Pant, and Olivia R. Liu Sheng, School of Accounting and Information Systems, The University of Utah, UT 84112; email: {zhongming.ma,gautam.pant,olivia.sheng}@business.utah.edu.

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., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org). © 2007 ACM 1046-8188/2007/02-ART5 \$5.00 DOI 10.1145/1198296.1198301 <http://doi.acm.org/10.1145/1198296.1198301>

## 1. INTRODUCTION

The Web provides an extremely large and dynamic source of information, and the continuous creation and updating of Web pages magnifies information overload on the Web. Both casual and noncasual users (e.g., knowledge workers) often use search engines to find a needle in this constantly growing haystack. Sellen et al. [2002], who define a knowledge worker as someone “whose paid work involves significant time spent in gathering, finding, analyzing, creating, producing or archiving information”, report that 59% of the tasks performed on the Web by a sample of knowledge workers fall into the categories of information gathering and finding, which require an active use of Web search engines.

Most existing Web search engines return a list of search results based on a user’s query but ignore the user’s specific interests and/or search context. Therefore, the identical query from different users or in different contexts will generate the same set of results displayed in the same way for all users, a so called one-size-fits-all [Lawrence 2000] approach. Furthermore, the number of search results returned by a search engine is often so large that the results must be partitioned into multiple result pages. In addition, individual differences in information needs, polysemy (multiple meanings of the same word), and synonymy (multiple words with same meaning) pose problems [Deerwester et al. 1990] in that a user may have to go through many irrelevant results or try several queries before finding the desired information. Problems encountered in searching are exaggerated further when the search engine users employ short queries [Jansen et al. 1998]. However, personalization techniques that put a search in the context of the user’s interests may alleviate some of these issues.

In this study, which focuses on knowledge workers’ search for information online in a workplace setting, we assume that some information about the knowledge workers, such as their professional interests and skills, is known to the employing organization and can be extracted automatically with an information extraction (IE) tool or with database queries. The organization can then use such information as an input to a system based on our proposed approach and provide knowledge workers with a personalized search tool that will reduce their search time and boost their productivity.

For a given query, a personalized search can provide different results for different users or organize the same results differently for each user. It can be implemented on either the server side (search engine) or the client side (organization’s intranet or user’s computer). Personalized search implemented on the server side is computationally expensive when millions of users are using the search engine, and it also raises privacy concerns when information about users is stored on the server. A personalized search on the client side can be achieved by query expansion and/or result processing [Pitkow et al. 2002]. By adding extra query terms associated with user interests or search context, the query expansion approach can retrieve different sets of results. The result processing includes result filtering, such as removal of some results, and reorganizing, such as reranking, clustering, and categorizing the results.

Our proposed approach is a form of client-side personalization based on an interest-to-taxonomy mapping framework and result categorization. It piggybacks on a standard search engine such as Google<sup>1</sup> and categorizes and displays search results on the basis of known user interests. As a novel feature of our approach, the mapping framework automatically maps the known user interests onto a set of categories in a Web directory, such as the Open Directory Project<sup>2</sup> (ODP) or Yahoo!<sup>3</sup> directory. An advantage of this mapping framework is that, after user interests have been mapped onto the categories, a large amount of manually edited data under these categories is freely available to be used to build text classifiers that correspond to these user interests. The text classifiers then can categorize search results according to the user's various interests at query time. The same text classifiers may be used to categorize emails and other digital documents which suggests that our approach may be extended to a broader domain of content management.

The main research questions that we explore are as follows: (1) What is an appropriate framework for mapping a user's professional interests and skills onto a group of concepts in a taxonomy such as a Web directory? (2) How does a personalized categorization system (PCAT) based on our proposed approach perform differently from a list interface system (LIST) similar to a conventional search engine? (3) How does PCAT perform differently from a nonpersonalized categorization system (CAT) that categorizes results without any personalization? The third question attempts to separate the effect of categorization from the effect of personalization in the proposed system. We explore the second and third questions along two dimensions, type of task and query length.

Figure 1 illustrates the input and output of these three systems. LIST requires two inputs: a search query and a search engine, and its output, similar to what a conventional search engine adopts, is a page-by-page list of search results. Using a large taxonomy (ODP Web directory), CAT classifies search results and displays them under some taxonomy categories; in other words, it uses the ODP taxonomy as an additional input. Finally, PCAT adds another input, namely, a set of user interests. The mapping framework in PCAT automatically identifies a group of categories from the ODP taxonomy as relevant to the user's interests. Using data from these relevant categories, the system generates text classifiers to categorize search results under the user's various interests at query time.

We compare PCAT with LIST and with CAT in two sets of controlled experiments. Compared with LIST, PCAT works better for searches with short queries and for information gathering tasks. In addition, PCAT outperforms CAT for both information gathering and finding tasks and for searches with free-form queries. Subjects indicate that PCAT enable them to identify relevant results and complete given tasks more quickly and easily than does LIST or CAT.

---

<sup>1</sup><http://www.google.com>.

<sup>2</sup><http://www.dmoz.com>.

<sup>3</sup><http://www.yahoo.com>.

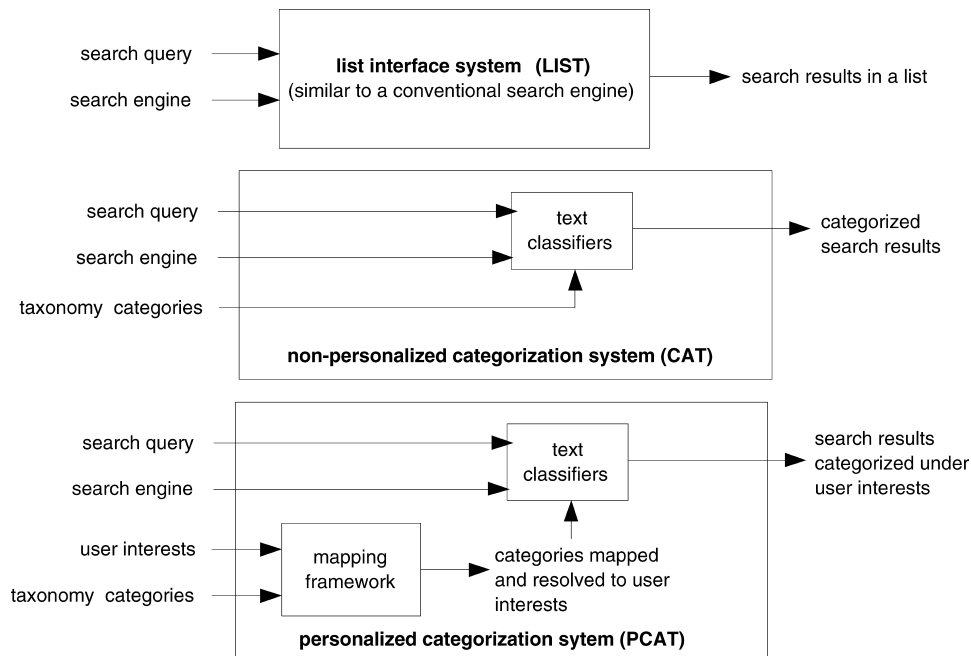


Fig. 1. Input and output of the three systems.

## 2. RELATED LITERATURE

This section reviews prior studies pertaining to personalized search. We also consider several studies using the ODP taxonomy to represent a search context, review studies on the taxonomy of Web activities, and end by briefly discussing text categorization.

According to Lawrence [2000], next-generation search engines will increasingly use context information. Pitkow et al. [2002] also suggest that a contextual computing approach that enhances user interactions through a greater understanding of the user, the context, and the applications may prove a breakthrough in personalized search efficiency. They further identify two primary ways, query expansion and result processing, to personalize search [Pitkow et al. 2002] which can complement each other.

### 2.1 Query Expansion

We use an approach similar to query expansion for finding terms related to user interests in our interest mapping framework. Query expansion refers to the process of augmenting a query from a user with other words or phrases in order to improve search effectiveness. It originally was applied in information retrieval (IR) to solve the problem of word mismatch that arises when search engine users employ different terms than those used by content authors to describe the same concept [Xu and Croft 1996]. Because the word mismatch problem can be reduced through the use of longer queries, query expansion may offer a solution [Xu and Croft 1996].

In line with query expansion, current literature provides various definitions of context. In the Inquirus 2 project [Glover et al. 1999], a user manually chooses a context in the form of a category, such as research papers or organizational homepages, before starting a search. Y!Q<sup>4</sup>, a large-scale contextual search system, allows a user to choose a context in the form of a few words or a whole article through three methods: a novel information widget executed in the user's Web browser, Yahoo! Toolbar<sup>5</sup>, or Yahoo! Messenger<sup>6</sup> [Kraft et al. 2005]. In the Watson project, Budzik and Hammond [2000] derive context information from the whole document a user views. Instead of using a whole document, Finkelstein et al. [2002] limit the context to the text surrounding a user-marked query term(s) in the document. That text is part of the whole document so their query expansion is based on a local context analysis approach [Xu and Croft 1996]. Leroy et al. [2003] define context as the combination of titles and descriptions of clicked search results after an initial query. In all these studies, queries get expanded on the basis of the context information, and results are generated according to the expanded queries.

## 2.2 Result Processing

Relatively fewer studies deal with result processing which includes result filtering and reorganizing. Domain filtering eliminates documents irrelevant to given domains from the search results [Oyama et al. 2004]. For example, Ahoy!, a homepage finder system, uses domain-specific filtering to eliminate most results returned by one or more search engines but retains the few pages that are likely to be personal homepages [Shakes et al. 1997]. Tan and Teo [1998] propose a system that filters out news items that may not be of interest to a given user according to that user's explicit (e.g., satisfaction ratings) and implicit (e.g., viewing order, duration) feedback to create personalized news.

Another approach to result processing is to reorganize, which involves reranking, clustering, and categorizing search results. For example, Teevan et al. [2005] construct a user profile (context) over time with rich resources including issued queries, visited Web pages, composed or read documents and emails. When the user sends a query, the system reranks the search results on the basis of the learned profile. Shen et al. [2005a] use previous queries and summaries of clicked results in the current session to rerank results for a given query. Similarly,UCAIR [Shen et al. 2005b], a client-side personalized search agent, employs both query expansion on the basis of the immediately preceding query and result reranking on the basis of summaries of viewed results. Other works also consider reranking according to a user profile [Gauch et al. 2003; Sugiyama et al. 2004; Speretta and Gauch 2005; Chirita et al. 2005; Kraft et al. 2005]. Gauch et al. [2003] and Sugiyama et al. [2004] learn a user's profile from his or her browsing history, whereas Speretta and Gauch [2005] build the profile on the basis of search history, and Chirita et al. [2005] require the user to specify the profile entries manually.

<sup>4</sup><http://yq.search.yahoo.com>.

<sup>5</sup><http://toolbar.yahoo.com>.

<sup>6</sup><http://beta.messenger.yahoo.com>.

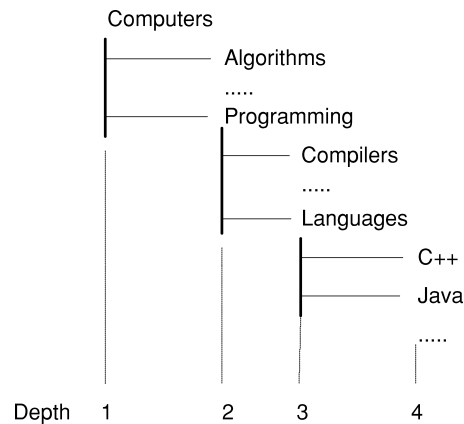


Fig. 2. ODP taxonomy.

Scatter/Gather [Cutting et al. 1992] is one of the first systems to present documents in clusters. Another system, Grouper [Zamir and Etzioni 1999], uses snippets of search engine results to cluster the results. Tan [2002] presents a user-configurable clustering approach that clusters search results using titles and snippets of search results and the user can manually modify these clusters.

Finally, in comparing seven interfaces that display search results, Dumais and Chen [2001] report that all interfaces that group results into categories are more effective than conventional interfaces that display results as a list. They also conclude that the best performance occurs when both category names and individual page titles and summaries are presented. We closely follow these recommendations for the two categorization systems we study (PCAT and CAT). In recent work, Kaki [2005] also finds that result categorization is helpful when the search engine fails to provide relevant results at the top of the list.

### 2.3 Representing Context Using Taxonomy

In our approach, we map user interests to categories in the ODP taxonomy. Figure 2 shows a portion of the ODP taxonomy in which Computers is a depth-one category, and C++ and Java are categories at depth four. We refer to Computers/Programming/Languages as the parent category of category C++ or Java. Hence various concepts (categories) are related through a hierarchy in the taxonomy. Currently, the ODP is a manually edited directory of 4.6 million URLs that have been categorized into 787,774 categories by 68,983 human editors. The ODP taxonomy has been applied to personalization of Web search in some prior studies [Pitkow et al. 2002; Gauch et al. 2003; Liu et al. 2004; and Chirita et al. 2005].

For example, the Outride personalized search system (acquired by Google) performs both query modification and result processing. It builds a user profile (context) on the basis of a set of personal favorite links, the user's last 1000 unique clicks, and the ODP taxonomy, then modifies queries according to that profile. It also reranks search results on the basis of usage and the user profile. The main focus of the Outride system is capturing a user's profile through

his or her search and browsing behavior [Pitkow et al. 2002]. The OBIWAN system [Gauch et al. 2003] automatically learns a user's interest profile from his or her browsing history and represents those interests with concepts in Magellan taxonomy. It maps each visited Web page into five taxonomy concepts with the highest similarities; thus, the user profile consists of accumulated categories generated over a collection of visited pages. Liu et al. [2004] also build a user profile that consists of previous search query terms and five words that surround each query term in each Web page clicked after the query is issued. The user profile then is used to map the user's search query onto three depth-two ODP categories. In contrast, Chirita et al. [2005] use a system in which a user manually selects ODP categories as entries in his or her profile. When reranking search results, they measure the similarity between a search result and the user profile using the node distance in a taxonomy concept tree, which means the search result must associate with an ODP category. A difficulty in their study is that many parameters' values have been set without explanations. The current Google personalized search<sup>7</sup> also explicitly asks users to specify their interests through the Google directory.

Similar to Gauch et al. [2003], we represent user interests with taxonomy concepts, but we do not need to collect browsing history. Unlike Liu et al. [2004], we do not need to gather previous search history, such as search queries and clicked pages, or know the ODP categories corresponding to the clicked pages. Whereas Gauch et al. [2003] map a visited page onto five ODP categories and Liu et al. [2004] map a search query onto three categories, we automatically map a user interest onto an ODP category. A difference between Chirita et al. [2005] and our approach is that when mapping a user's interest onto a taxonomy concept, we employ text, that is, page titles and summaries associated with the concept in taxonomy, while they use the taxonomy category title and its position in the concept tree when computing the tree-node distance. Also, in contrast to UCAIR [Shen et al. 2005b] that uses contextual information in the current session (short-term context) to personalize search, our approach personalizes search according to a user's long-term interests which may be extracted from his or her resume.

Haveliwala [2002] and Jeh and Widom [2003] extend the PageRank algorithm [Brin and Page 1998] to generate personalized ranks. Using 16 depth-one categories in ODP, Haveliwala [2002] computes a set of topic-sensitive PageRank scores. The original PageRank is a global measure of the query- or topic-insensitive popularity of Web pages measured solely by a linkage graph derived from a large part of the Web. Haveliwala's experiments indicate that, compared with the original PageRank, a topic-sensitive PageRank achieves greater precision in top-ten search results. Topic-sensitive PageRank also can be used for personalization after a user's interests have been mapped onto appropriate depth-one categories of the ODP which can be achieved through our proposed mapping framework. Jeh and Widom [2003] present a scalable personalized PageRank method in which they identify a linear relationship between basis vectors and the corresponding personalized PageRank vectors. At query time,

<sup>7</sup><http://labs.google.com/personalized>.

their method constructs an approximation to the personalized PageRank vector from the precomputed basis vectors.

## 2.4 Taxonomy of Web Activities

We study the performance of the three systems (described in Section 1) by considering different types of Web activities. Sellen et al. [2002] categorize Web activities into six categories: finding (locate something specific), information gathering (answer a set of questions; less specific than finding), browsing (visit sites without explicit goals), transacting (execute a transaction), communicating (participate in chat rooms or discussion groups), and housekeeping (check the accuracy and functionality of Web resources). As Craswell et al. [2001] define a site finding task specifically as “one where the user wants to find a particular site, and their query names the site,” we consider it a type of finding task. It should be noted that some Web activities, especially information gathering, can involve several searches. On the basis of the intent behind Web queries, Broder [2002] classifies Web searches into three classes: navigational (reach a particular site), informational (acquire information from one or more Web pages), and transactional (perform some Web-mediated activities). As the taxonomy of search activities suggested by Sellen et al. [2002] is broader than that by Broder [2002], in this article we choose to study the two major types of activities studied in Sellen et al. [2002].

## 2.5 Text Categorization

In our study, CAT and PCAT systems employ text classifiers to categorize search results. Text categorization (TC) is a supervised learning task that classifies new documents into a set of predefined categories [Yang and Liu 1999]. As a joint discipline of machine learning and IR, TC has been studied extensively, and many different classification algorithms (classifiers) have been introduced and tested, including the Rocchio method, naïve Bayes, decision tree, neural networks, and support vector machines [Sebastiani 2002]. A standard information retrieval metric, cosine similarity [Salton and McGill 1986], computes the cosine angle between vector representations of two text fragments or documents. In TC, a document can be assigned to the category with the highest similarity score. Due to its simplicity and effectiveness, cosine similarity has been used by many studies for TC [e.g., Yang and Liu 1999; Sugiyama et al. 2004; Liu et al. 2004].

In summary, to generate user profiles for personalized search, previous studies have asked users for explicit feedback, such as ratings and preferences, or collected implicit feedback, such as search and browsing history. However, users are unwilling to provide explicit feedback even when they anticipate a long-run benefit [Carroll and Rosson 1987]. Implicit feedback has shown promising results for personalizing search using short-term context [Leroy et al. 2003; Shen et al. 2005b]. However, generating user profiles for long-term context through implicit feedback will take time and may raise privacy concerns. In addition, a user profile generated from implicit feedback may contain noise because the user preferences have been estimated from behaviors and not explicitly specified.

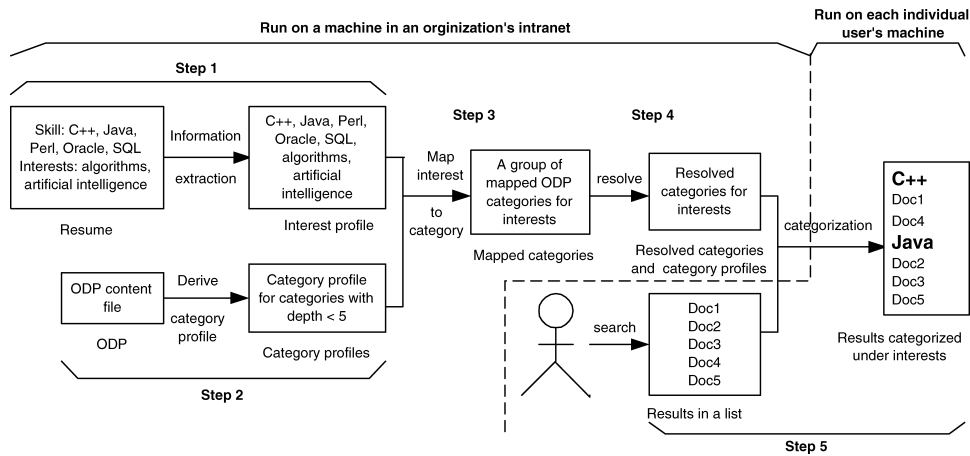


Fig. 3. Process view of proposed approach.

In our approach two user-related inputs, a search query and the user's professional interests and skills, are explicitly given to a system so some prior work [Leroy et al. 2003; Gauch et al. 2003; Liu et al. 2004; Sugiyama et al. 2004; Kraft et al. 2005] that relies on modeling user interests through searching or browsing behavior is not readily applicable.

### 3. OUR APPROACH

Our approach begins with the assumption that some user interests are known and therefore is well suited for a workplace setting in which employees' resumes often are maintained in a digital form or information about users' professional interests and skills is stored in a database. An IE tool or database queries can extract such information as input to complement the search query, search engine, and contents of the ODP taxonomy. However, we do not include such an IE program in this study and assume instead that the interests have been already given. Our interest-category mapping framework tries to automatically identify an ODP category associated with each of the given user interests. Then our system uses URLs organized under those categories as training examples to classify search results into various user interests at query time. We expect the result categorization to help the user quickly focus on results of interest and decrease total time spent in searching. The result categorization may also lead to the discovery of serendipitous connections between the concepts being searched and the user's other interests. This form of personalization therefore should reduce search effort and possibly provide interesting and useful resources the user would not notice otherwise. We focus on work-related search performance, but our approach could be easily extended to include personal interests as well. We illustrate a process view of our proposed approach in Figure 3 and present our approach in five steps. Steps 3 and 4 cover the mapping framework.

### 3.1 Step 1: Obtaining an Interest Profile

Step 1 (Figure 3) pertains to how user interests can be extracted from a resume. Our study assumes that user interests are available to our personalized search system in the form of a set of words and phrases which we call a user's *interest profile*.

### 3.2 Step 2: Generating Category Profiles

As we explained previously, ODP is a manually edited Web directory with millions of URLs placed under different categories. Each ODP category contains URLs that point to external Web pages that human editors consider relevant to the category. Those URLs are accompanied by manually composed titles and summaries that we believe accurately represent the corresponding Web page content. The *category profile* of an ODP category thus is built by concatenating the titles and summaries of the URLs listed under the category. The constructed category profiles provide a solution to the cold-start problem, which arises from the difficulty of creating a profile for a new user from scratch [Maltz and Ehrlich 1995], and they later serve to categorize the search results. Gauch et al. [2003], Menczer et al. [2004], and Srinivasan et al. [2005] use similar concatenation to build topic profiles. In our study, we combine up to 30 pairs of manually composed titles and summaries of URL links under an ODP category as the category profile.<sup>8</sup> In support of this approach, Shen et al. [2004] report that classification using manually composed summarization in the LookSmart Web directory achieves higher accuracy than the use of the content of Web pages. For building the category profile, we pick the first 30 URLs based on the sequence in which they are provided by ODP. We note that ODP can have more than 30 URLs listed under a category. In order to use similar amounts of information for creating profiles for different ODP categories, we only use the titles and summaries of the first 30 URLs. When generating profiles for categories in Magellan taxonomy, Gauch et al. [2003] show that a number of documents between 5 and 60 provide reasonably accurate classification.

At depth-one, ODP contains 17 categories (for a depth-one category, Computers, see Figure 2). We select five of these (Business, Computers, Games, Reference, and Science) that are likely to be relevant to our subjects and their interests. These five broad categories comprise a total of 8,257 categories between depths one and four. We generate category profiles by removing stop words and applying Porter stemming<sup>9</sup> [Porter 1980]. We also filter out any terms that appear only once in a profile to avoid noise and remove any profiles that contain fewer than two terms. Finally, the category profile is represented as a term vector [Salton and McGill 1986] with term frequencies (*tf*) as weights. Shen et al. [2004] also use *tf*-based weighting scheme to represent manually composed summaries in the LookSmart Web directory to represent a Web page.

<sup>8</sup>A category profile does not include titles or summaries of its child (subcategory) URLs.

<sup>9</sup>[http://www.dcs.gla.ac.uk/idom/ir\\_resources/linguistic\\_utils/porter.java](http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/porter.java).

### 3.3 Step 3: Mapping Interests to ODP Categories

Next, we need a framework to map a user's interests onto appropriate ODP categories. The framework then can identify category profiles for building text classifiers that correspond to the user's interests. Some prior studies [Pitkow et al. 2002; Liu et al. 2004] and the existing Google personalized search use ODP categories with a few hundred categories up to depth two, but for our study, categories up to depth two may lack sufficient specificity. For example, Programming, a depth-two category, is too broad to map a user interest in specific programming languages such as C++, Java, or Perl. Therefore, we map user interests to ODP categories up to depth four. As we mentioned in Step 2, a total of 8,257 such categories can be used for interest mapping. We employ four different mapping methods to evaluate the mapping performance by testing and comparing them individually as well as in different combinations. When generating an output category, a mapping method includes the parent category of the mapped category; for example, if the mapped category is C++, the output will be Computers/Programming/Languages/C++.

**3.3.1 Mapping Method 1 (m1-category-label): Simple Term Match.** The first method uses a string comparison to find a match between an interest and the label of the category in ODP. If an interest is the same as a category label, the category is considered a match to the interest. Plural forms of terms are transformed to their singular forms by a software tool from the National Library of Medicine.<sup>10</sup> Therefore, the interest of search engine is matched with the ODP category Search Engines, and the output category is Computers/Internet/Searching/Search Engines.

**3.3.2 Mapping Method 2 (m2-category-profile): Most Similar Category Profile.** The cosine similarities between an interest and each of the category profiles are computed in which case the ODP category with the highest similarity is selected as the output.

**3.3.3 Mapping Method 3 (m3-category-profile-noun): Most Similar Category Profile While Augmenting Interest With Potentially Related Nouns.** The m1-category-label and m2-category-profile will fail if the category labels and profiles do not contain any of the words that form a given interest so it may be worthwhile to augment the interest concept by adding a few semantically similar or related terms. According to Harris [1985], terms in a language do not occur arbitrarily but appear at a certain position relative to other terms. On the basis of the concept of cooccurrence, Riloff and Shepherd [1997] present a corpus-based bootstrapping algorithm that starts with a few given seed words that belong to a specific domain and discovers more domain-specific semantically-related lexicons from a corpus. Similar to query expansion, it is desirable to augment the original interest with a few semantically similar or related terms.

For m3-category-profile-noun, one of our programs conducts a search on Google using an interest as a search query and finds the N nouns that most

<sup>10</sup><http://umlslex.nlm.nih.gov/nlsRepository/nlp/doc/userDoc/index.html>.

Table I. Frequently Cooccurring Nouns and NPs

Domain	Interest	Two Cooccurring nouns	Cooccurring NP
Computer	C++	programme, resource	general c
	IBM DB2	database, software	database
	Java	tutorial, sun	sun
	Machine Learning	information, game	ai topic
	Natural Language Processing	intelligence, speech	intelligence
	Object Oriented Programming	concept, link	data
	Text Mining	information, data	text mine tool
	UML	model tool	acceptance*
Finance	Web Site Design	html, development	library resource web development
	Bonds	saving, rate	saving bond
	Day Trading	resource, article	book
	Derivatives	trade, international	gold
	Mutual Funds	news, stock	account
	Offshore Banking	company, formation	bank account
	Risk Management	open source*	software risk evaluation*
	Stocks Exchange	trade, information	official site
Technical Analysis	market, chart	market pullback	
	Trading Cost	service, cap	product

\*Some cooccurring nouns or NPs may be not semantically similar or related.

frequently cooccur in the top ten search results (page titles and snippets). We find cooccurring nouns because most terms in interest profiles are nouns (for terms from some sample user interests, see Table I). Terms semantically similar or related to those of the original interest thus can be obtained without having to ask a user for input such as feedback or a corpus. A noun is identified by looking up the word in a lexical reference system,<sup>11</sup> WordNet [Miller et al. 1990], to determine whether the word has the part-of-speech tag of noun. The similarities between a concatenated text (a combination of the interest and N most frequently cooccurring nouns) and each of the category profiles then are computed to determine the category with the highest similarity as the output of this method.

*3.3.4 Mapping Method 4 (m4-category-profile-np): Most Similar Category Profile While Augmenting Interest With Potentially Related Noun Phrases.* Although similar to m3-category-profile-noun, this method finds the M most frequently cooccurring noun phrases on the first result page from up to ten search results. We developed a shallow parser program to parse sentences in the search results into NPs (noun phrases), VPs (verb phrases), and PPs (prepositional phrases), where a NP can appear in different forms, such as a single noun, a concatenation of multiple nouns, an article followed by a noun, or any number of adjectives followed by a noun.

Table I lists some examples of frequently cooccurring nouns and NPs identified by m3-category-profile-noun and m4-category-profile-np. Certain single-noun NPs generated by m4-category-profile-np differ from individual nouns identified by m3-category-profile-noun because a noun identified by this method

<sup>11</sup><http://wordnet.princeton.edu/>.

Table II. Individual Mapping Method Comparison (Based on 56 Computer Interests)

Mapping Method	m1	m2	m3	m4
Number of correctly mapped interests	27	29	25	19
Number of incorrectly mapped interests	2	25	30	36
Number of total mapped interests	29	54	55	55
Precision ( $\frac{\text{Number of correct mapped interests}}{\text{Number of total mapped interests}}$ )	93.0%	53.7%	45.5%	34.5%
Recall ( $\frac{\text{Number of correct mapped interests}}{56}$ )	48.2%	51.8%	44.6%	33.9%
F1	63.5%	52.7%	45.0%	34.2%

may combine with other terms to form a phrase in m4-category-profile-np and therefore not be present in the result generated by m4-category-profile-np.

### 3.4 Step 4: Resolving Mapped Categories

For a given interest, each mapping method in Step 3 may generate a different mapped ODP category, and m1-category-label may generate multiple ODP categories for the same interest because the same category label sometimes is repeated in the ODP taxonomy. For example, the category Databases appears in several different places in the hierarchy of the taxonomy, such as Computers/Programming/Databases and Computers/Programming/Internet/Databases.

Using 56 professional interests in the computer domain which were manually extracted from several resumes of professionals collected from ODP (eight interests are shown in the first column of Table I), Table II compares the performances of each individual mapping method. After verification by a domain expert, m1-category-label generated mapped categories for 29 of 56 interests, and only two did not contain the right category. We note that m1-category-label has much higher precision than the other three methods, but it generates the fewest mapped interests. Machine learning research [e.g., Dietterich 1997] has shown that an ensemble of classifiers can outperform each classifier in that ensemble. Since the mapping methods can be viewed as classification techniques that classify interests into ODP categories, a combination of the mapping methods may outperform any one method.

Figure 4 lists the detailed pseudocode of the procedure used to automatically resolve a final set of categories for an interest profile with the four mapping methods. M1 represents a set of mapped category/categories generated by m1-category-label as do M2, M3, and M4. Because of its high precision, we prioritize the category/categories generated by m1-category-label as shown in Step (2); if a category generated by m1-category-label is the same as, or a parent category of, a category generated by any other method, we include the category generated by m1-category-label in the list of final resolved categories. Because m1-category-label uses an exact match strategy, it does not always generate a category for a given interest. In Step (3), if methods m2-category-profile, m3-category-profile-noun, and m4-category-profile-np generate the same mapped category, we select that category, irrespective of whether m1-category-label generates one. Steps (2) and (3) attempt to produce a category for an interest by considering overlapping categories from different methods. If no such overlap is found, we look for overlapping categories generated for different interests in

```

(1) For each interest i in interest profile
    Given i, the four mapping methods generate M1, M2, M3, and M4
(2)  For each category c in M1
    If c is the same as, or a parent of, a category in M2, M3, or M4, add c to a list
of final categories, then go to Step (1)
    End For
(3)  If M2, M3, and M4 contain the same category c, add c into the list of final
categories, then go to Step (1)
(4)  Put any category c in M1, M2, M3, and M4 into a list of candidate categories12
    End For
(5) For each category c in candidate categories
    Count the frequency for c
    End For
(6) For each depth-four category c in candidate categories
    If frequency of c >= threshold, add c into final categories. (We chose the
threshold equal to the number of mapping methods - 1. The threshold was three in our
tests because we used four mapping methods. The number of three or larger means
there is an overlap of candidate category between at least two different interests. Then
we choose the overlapped candidate category to represent these interests.)
    End For
(7) Removing all candidate categories for the mapped interests in Step (6)
(8) Resolving all remaining categories of depth four into depth three by truncating the
category at depth four. For example, after truncating to depth three from depth four,
reference/knowledge management/publications/articles is resolved as reference/
knowledge management/publications
(9) For each category c in candidate categories
    Count the frequency for c
    End For
(10) For each depth-three category c in candidate categories
    If frequency of c >= threshold, add c into final categories
    End For

```

Fig. 4. Category resolving procedures.

Step (6) because if more than one interest is mapped to the same category, it is likely to be of interest. In Step (8), we try to represent all remaining categories at a depth of three or less by truncating the category at depth four and thereby hope to find overlapped categories through the parent categories. Step (9) is similar to Step (5) except that all remaining categories are at the depth of three or less.

To determine appropriate values for N (number of nouns) and M (number of NPs) for m3-category-profile-noun and m4-category-profile-np, we tested

<sup>12</sup>Candidate categories cannot be used as final resolved categories unless the frequency of a candidate category is greater than or equal to the threshold in Step (6).

Table III. Comparison of Combined Mapping Methods

Combination of Mapping Methods	m1+m2+m3	m1+m2+m4	m1+m3+m4	m1+m2+m3+m4
Number of correctly mapped interests	34	35	32	39
Precision ( $\frac{\text{Number of correct mapped interest}}{56}$ )*	60.7%	62.5%	57.1%	69.6%

\*Recall and F1 were same as precision because the number of mapped interests was 56.

Table IV. Resolved Categories

Domain	Interest	ODP Category
Computer	C++	computers/programming/languages/c++
	IBM DB2	computers/software/databases/ibm db2
	Java	computers/programming/languages/java
	Machine Learning	computers/artificial intelligence/machine learning
	Natural Language Processing	computers/artificial intelligence/natural language
	Object Oriented Programming	computers/software/object-oriented
	Text Mining	reference/knowledge management/ knowledge discovery/text mining
	UML	computers/software/data administration*
	Web Site Design	computers/internet/web design and development
Finance	Bonds	business/investing/stocks and bonds/bonds
	Day Trading	business/investing/day trading
	Derivatives	business/investing/derivatives
	Mutual Funds	business/investing/mutual funds
	Offshore Banking	business/financial services/offshore services
	Risk Management	business/management/software*
	Stocks Exchange	business/investing/stocks and bonds/exchanges
	Technical Analysis	business/investing/research and analysis/ technical analysis
	Trading Cost	business/investing/derivatives/brokerages

\*Because the mapping and resolving steps are automatic, some resolved categories are erroneous.

different combinations of values ranging from 1 to 3 with the 56 computer interests. According to the number of correctly mapped interests, choosing the two most frequently cooccurring nouns and one most frequently cooccurring NP offers the best mapping result (see Table I for some examples of identified nouns and NPs.) With the 56 interests, Table III compares the number of correctly mapped interests when different mapping methods are combined. Using all four mapping methods provides the best results; 39 of the 56 interests were correctly mapped onto ODP categories. The resolving procedures in Figure 4 thus are based on four mapping methods. When using three methods, we adjusted the procedures accordingly, such as setting the thresholds in Steps (6) and (10) to two instead of three.

Table IV lists mapped and resolved categories for some interests in computer and finance domains.

After the automatic resolving procedures, mapped categories for some interests may not be resolved because different mapping methods generate different categories. Unresolved interests can be handled by having the user manually map them onto the ODP taxonomy. An alternative approach could use a unresolved user interest as a query to a search engine (in a manner similar to m3-category-profile-noun and m4-category-profile-np), then combine the search

results, such as page titles and snippets, to compose an ad hoc category profile for the interest. Such a profile could flexibly represent any interest and avoid the limitation of taxonomy in that it contains a finite set of categories. It would be worthwhile to examine the effectiveness of such ad hoc category profiles in a future study. In this article, user interests are fully mapped and resolved to ODP categories.

These four steps are performed just once for each user, possibly during a software installation phase, unless the user's interest profile changes. To reflect such a change in interests, our system can automatically update the mapping periodically or allow a user to request an update from the system. As shown in Figure 3, the first four steps can be performed in a client-side server, such as a machine on the organization's intranet, and the category profiles can be shared by each user's machine.

Finally, user interests, even long-term professional ones, are dynamic in nature. In the future, we will explore more techniques to learn about and finetune interest mapping and handle the dynamics of user interests.

### 3.5 Step 5: Categorizing Search Results

When a user submits a query, our system obtains search results from Google and downloads the content of up to the top-50 results which correspond to the first five result pages. The average number of result pages viewed by a typical user for a query is 2.35 [Jansen et al. 2000], and a more recent study [Jansen et al. 2005] reports that about 85–92% of users view no more than two result pages. Hence, our system covers approximately double the number of results normally viewed by a search engine user. On the basis of page content, the system categorizes the results into various user interests. In PCAT, we employ a user's original interests as class labels rather than the ODP category labels because the mapped and resolved ODP categories are associated with user interests. Therefore, the use of ODP (or any other Web directory) is transparent to the user. A Web page that corresponds to a search result is categorized by (1) computing the cosine similarity between the page content and each of the category profiles of the mapped and resolved ODP categories that correspond to user interests and (2) assigning the page to the category with the maximum similarity if the similarity is greater than a threshold. If a search result does not fall into any of the resolved user interests, it is assigned to the Other category.

The focus of our study is to explore the use of PCAT, an implementation based on the proposed approach, and compare it with LIST and CAT. With regard to interest mapping and result categorization (classification problems), we choose the simple and effective cosine similarity instead of comparing different classification algorithms and selecting the best one.

## 4. IMPLEMENTATION

We developed three search systems<sup>13</sup> with different interfaces to display search results, and the online searching portion was implemented as a wrapper on

<sup>13</sup>In experiments, we named the systems A, B, or C; in this article, we call them PCAT, LIST, or CAT, respectively.

Google search engine using the Google Web API.<sup>14</sup> Although the current implementation of our approach uses a single search engine (Google), following the metasearch approach [Dreilinger and Howe 1997], it can be extended to handle results from multiple engines.

Because Google has become the most popular search engine<sup>15</sup>, we use Google's search results to feed the three systems. That is, the systems have the same set of search results for the same query; recall that LIST can be considered very similar to Google. For simplicity, we limit the search results in each system to Web pages in HTML format. In addition, for a given query, each of the systems retrieves up to 50 search results.

PCAT and CAT download the contents of Web pages that correspond to search results and categorize them according to user interests and ODP categories, respectively. For faster processing, the systems use multithreading for simultaneous HTTP connections and download up to 10KB of text for each page. It took our program about five seconds to fetch 50 pages. We note that our page-fetching program is not an industry strength module and much better concurrent download speeds have been reported by other works [Hafri and Djeraba 2004; Najork and Heydon 2001]. Hence, we feel that our page-fetching time can be greatly reduced in a production implementation. After fetching the pages, the systems remove stop words and perform word stemming before computing the cosine similarity between each page content and a category profile. Each Web page is assigned to the category (and its associated interest for PCAT) with the greatest cosine similarity. However, if the similarity is not greater than a similarity threshold, the page is assigned to the Other category. We determined the similarity threshold by testing query terms from *irrelevant* domains (not relevant to any of the user's interests). For example, given that our user interests are related to computer and finance, we tested ten irrelevant queries, such as NFL, Seinfeld, allergy, and golden retriever. For these irrelevant queries, when we set the threshold at 0.1, at least 90% (often 96% or higher) of retrieved results were categorized under the Other category. Thus we chose 0.1 as our similarity threshold. The time for classifying results according to user interests in PCAT is negligible (tens of milliseconds). However, the time for CAT is three magnitudes greater than that for PCAT because the number of potential categories for CAT is 8,547, whereas the number of interests is less than 8 in PCAT.

Figure 5 displays a sample output from PCAT for the query regular expression. Once a user logs in with his or her unique identification, PCAT displays a list of the user's interests on top of the GUI. After a query is issued, search results are categorized into various interests and displayed in the result area as shown in Figure 5. A number next to the interest indicates how many search results are classified under that interest; if there is no classified search result, the interest will not be displayed in the result area. Under each interest (category), PCAT (CAT) shows no more than three results on the main page. If more than three results occur under an interest or category, a More link appears next to the number of results. (In Figure 5, there is a More link for the interest of

<sup>14</sup><http://www.google.com/apis/>.

<sup>15</sup><http://www.comscore.com/press/release.asp?press=873>.

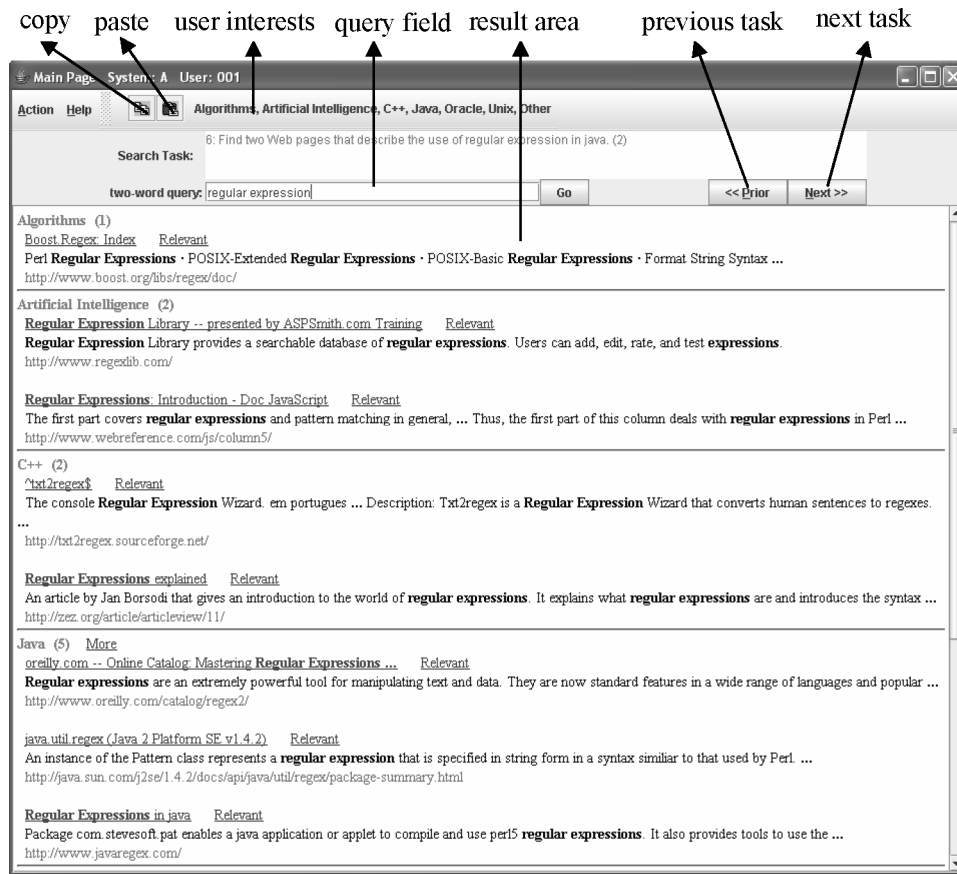


Fig. 5. Sample output of PCAT. Category titles are user interests mapped and resolved to ODP categories.

Java.) Upon clicking this link, the user sees all of the results under that interest in a new window as shown in Figure 6.

Figure 7 displays a sample output of LIST for the same query (regular expression) and shows all search results in the result area as a page-by-page list. Clicking a page number causes a result page with up to ten results to appear in the result area of the same window. For the search task in Figure 7, the first relevant document is shown as the sixth result on page 2 in LIST.

Figure 8 displays a sample output for CAT in which the category labels in the result area are ODP category names sorted alphabetically such that output categories under business are displayed before those under computers.

We now describe some of the features of the implemented systems that would not appear in a production system but are meant only for experimental use. We predefined a set of search tasks the subjects used to conduct searches during the experiments that specified what information and how many Web pages needed to be found (Section 5.2.2 describes the search tasks in more detail.) Each search result consists of a page title, snippet, URL, and a link called



Fig. 6. More window to show all of the results under the interest Java.

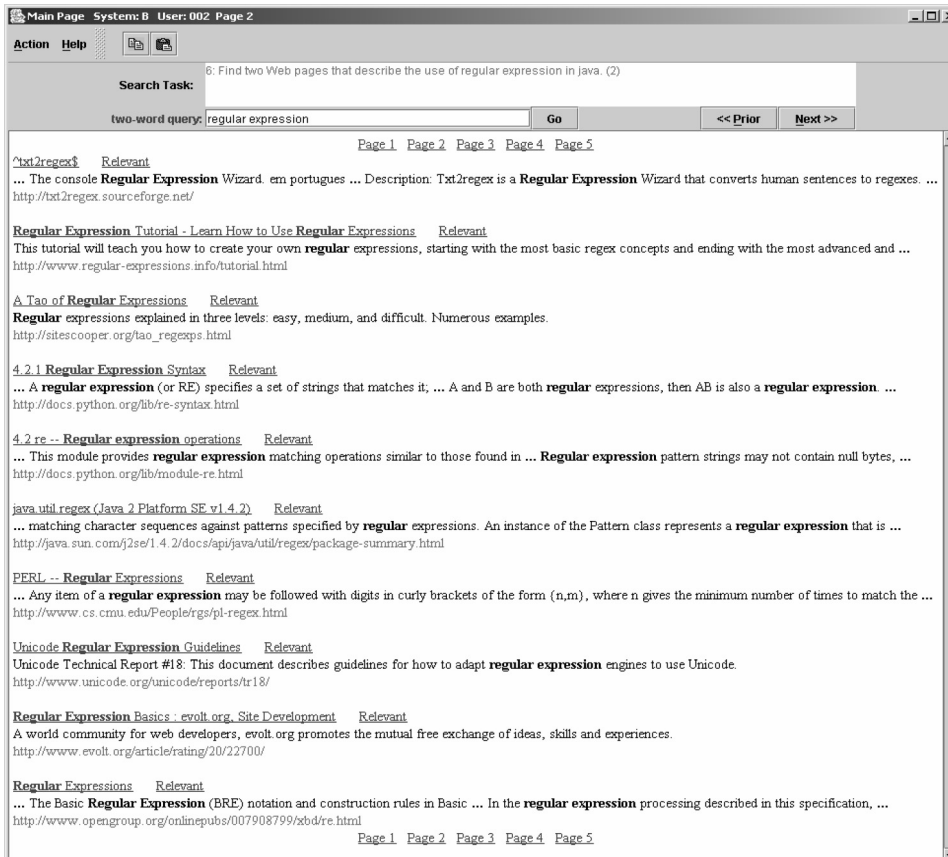


Fig. 7. Sample output of LIST.

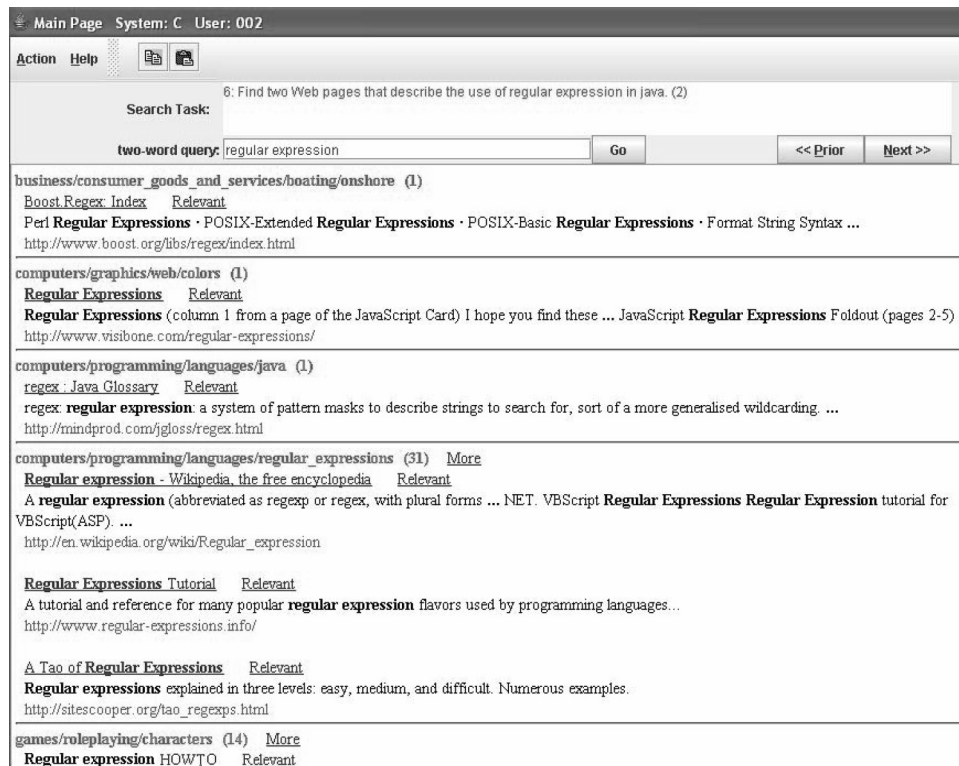


Fig. 8. Sample output of CAT. Category labels are ODP category titles.

Relevant<sup>16</sup> next to the title. Except for the relevant link, the items are the same as those found in typical search engines. A subject can click the hyperlinked page title to open the page in a regular Web browser such as Internet Explorer. The subject determines whether a result is relevant to a search task by looking at the page title, snippet, URL, and/or the content of the page.

Many of our search tasks require subjects to find one relevant Web page for a task but some require two. In Figure 5, the task requires finding two Web pages which is also indicated by the number 2 at the end of the task description. Once the user finds enough relevant pages, he or she can click the Next button to proceed to the next task; clicking on Next button before enough relevant page(s) have been found prompts a warning message, which allows the user to either give up or continue the current search task.

We record search time, or the time spent on a task, as the difference between the time that the search results appear in the result area and the time that the user finds the required number of relevant result(s).

<sup>16</sup>When a user clicks on the relevant link, the corresponding search result is treated as the answer or solution for the current search task. This clicked result is considered as relevant and is not necessarily the most relevant among all search results.

## 5. EXPERIMENTS

We conducted two sets of controlled experiments to examine the effects of personalization and categorization. In experiment I, we compare PCAT with LIST, that is, a personalized system that uses categorization versus a system similar to a typical search engine. Experiment II compares PCAT with CAT in order to study the difference between personalization and nonpersonalization, given that categorization is common to both systems. These experiments were designed to examine whether subjects' mean log search time<sup>17</sup> for different types of search tasks and query lengths varied between the compared systems. The metric evaluates the efficiency of each system because all three systems return the same set of search results for the same query. Before experiment I, we conducted a preliminary experiment comparing PCAT and LIST with several subjects who later did not participate in either the experiment I or II. The preliminary experiment helped us make decisions relating to experiment and system design. Next we introduce our experiments I and II in detail.

### 5.1 Studied Domains and Domain Experts

Because we were interested in personalizing search according to a user's professional interests, we chose two representative professional domains, computer and finance, that appear largely disjointed.

For the computer domain, two of the authors, who are researchers in the area of information systems, served as the domain experts. Both experts also have industrial experiences related to computer science. For the finance domain, one expert has a doctoral degree and the other has a master's degree in finance.

### 5.2 Professional Interests, Search Tasks, and Query Length

**5.2.1 Professional Interests (Interest Profiles).** For each domain, the two domain experts manually chose several interests and skills that could be considered fundamental, which enables us to form a generic interest profile that would be shared by all subjects within the domain. Moreover, the fundamental nature of these interests allows us to recruit more subjects, leading to greater statistical significance in our results. By defining some fundamental skills in the computer domain, such as programming language, operating system, database, and applications, the two computer domain experts identified six professional interests: algorithms, artificial intelligence, C++, Java, Oracle, and Unix. Similarly, the two finance experts provided seven fundamental professional interests: bonds, corporate finance, day trading, derivatives, investment banking, mutual funds, and stock exchange.

**5.2.2 Search Tasks.** The domain experts generated search tasks on the basis of the chosen interest areas but also considered different types of tasks, that is, finding and information gathering. The content of those search tasks include

---

<sup>17</sup>Mean log search time is the average log-transformed search time for a task across a group of subjects using the same system. We transformed the original search times (measured in seconds) with base 2 log to make the log search times closer to a normal distribution. In addition, taking the average makes the mean log search times more normally distributed.

Table V. Examples of Search Tasks, Types of Tasks, and Query Lengths

Domain	Search Task	Type of Search Task	Query Length
Computer	You need an open source IDE (Integrated Development Environment) for C++. Find a page that provides any details about such an IDE.	Finding	one-word
Computer	You need to provide a Web service to your clients. Find two pages that describe Web services support using Java technology.	Information Gathering	two-word
Finance	Find a portfolio management spreadsheet program.	Finding	three-word
Finance	Find the homepage of New York Stock Exchange.	Site Finding	free-form

finding a software tool, locating a person’s or organization’s homepage, finding pages to learn about a certain concept or technique, collecting information from multiple pages, and so forth. Our domain experts predefined 26 nondemo search tasks for each domain as well as 8 and 6 demo tasks for the computer and finance domains, respectively. The demo tasks were similar to, but not identical to the nondemo tasks, and therefore offer subjects some familiarity with both systems before they started to work on the nondemo tasks. Nondemo tasks are used in postexperiment analysis, while demo tasks are not. All demo and nondemo search tasks belong to the categories of finding and information gathering [Sellen et al. 2002] as discussed in Section 2.4, and within the finding tasks, we included some site finding tasks [Craswell et al. 2001].

**5.2.3 Query Length.** Using different query lengths, we specified four types of queries for search tasks in each domain:

- (1) One-word query (e.g., jsp, underinvestment)
- (2) Two-word query (e.g., neural network, security line)
- (3) Three-word query (e.g., social network analysis)
- (4) Free-form query, which had no limitations on the number of words used

For a given task a user was free to enter any query word(s) of his or her own choice that conformed to the associated query-length requirement, and the user could issue multiple queries for the same task. For example, Table V shows some sample search tasks, types of search tasks, and their associated query lengths.

Table VI lists the distributions of search tasks and their associated query lengths. For each domain, we divided the 26 nondemo search tasks and demo tasks into two groups such that the two groups have the same number of tasks and distribution of query lengths. During each experiment, subjects searched for the first group of tasks using one system, and the second group of tasks using the other.

We chose these different query lengths for several reasons. First, numerous studies show that users tend to submit short Web queries with an average

Table VI. Distribution of Search Tasks and Their Associated Query Lengths

Experiment	Domain\Query Length	One-word	Two-word	Three-word	Free-form	Total Tasks
I & II	Computer	6	6	4	10	26
	Finance	8	6	6	6	26

length of two words. A survey by the NEC Research Institute in Princeton reports that up to 70% of users typically issue a query with one word in Web searches, and nearly half of the Institute’s staff—who should be Web-savvy (knowledge workers and researchers)—fail to define their searches precisely with query terms [Butler 2000]. By collecting search histories for a two-month period from 16 faculty members across various disciplines at a university, Käki [2005] found that the average query length was 2.1 words. Similarly, Jansen et al. [1998] find through their analysis of transaction logs on Excite that, on average, a query contains 2.35 words. In yet another study, Jansen et al. [2000] report that the average length of a search query is 2.21 words. From their analysis of users’ logs in the Encarta encyclopedia, Wen et al. [2002] report that the average length of Web queries is less than 2 words.

Second, we chose different query lengths to simulate different types of Web queries and examine how these different types affect system performance. A prior study follows a similar approach; in comparing the IntelliZap system with four popular search engines, Finkelstein et al. [2002] set the length of queries to one, two, and three words and allow users to type in their own query terms.

Third, in practice, queries are often incomplete or may not incorporate enough contextual information which leads to many irrelevant results and/or relevant results that do not appear at the top of the list. A user then has two obvious options: enter a different query to start a new search session or go through the long result list page-by-page, both of which consume time and effort. From a study with 33,000 respondents, Sullivan [2000] finds that 76% of users employ the same search engine and engage in multiple search sessions on the same topic. To investigate this problem of incomplete or vague queries, we associate search tasks with different query lengths to simulate the real-world problem of incomplete or vague queries. We believe that categorization will present results in such a way to help disambiguate such queries. Unlike Leroy et al. [2003], who extract extra query terms from users’ behaviors during consecutive searches, we do not modify users’ queries but rather observe how a result-processing approach (personalized categorization of search results) can improve search performance.

### 5.3 Subjects

Prior to the experiments, we sent emails to students in the business school and the computer science department of our university, as well as to some professionals in the computer industry, to solicit their participation. In these emails, we explicitly listed the predefined interests and skills we expected potential subjects to have. We also asked several questions, including the following two self-reported ones.

Table VII-1. Educational Status of Subjects

Experiment	Domain\Status	Undergraduate	Graduate	Professional	Total
I	Computer	3	7	4	14
	Finance	4	16	0	20
II	Computer	3	11	2	16
	Finance	0	20	0	20

Table VII-2. Self-Reported Performance on Search Within a Domain

Experiment	Domain\Performance	Slow	Normal	Fast
I	Computer	0	8	6
	Finance	2	15	3
II	Computer	1	8	7
	Finance	2	11	7

Table VII-3. Self-Reported Time (Hours) Spent Searching and Browsing Per Week

Experiment	Domain\Time (hours)	[0, 7)	[7, 14)	[14+)
I	Computer	1	9	4
	Finance	5	10	5
II	Computer	2	7	7
	Finance	2	11	7

- (1) When searching online for topics in the computer or finance domain, what do you think of your search performance (with a search engine) in general?  
(a) slow (b) normal (c) fast
- (2) How many hours do you spend online browsing and searching per week (not limited to your major)?  
(a) [0, 7) (b) [7+, 14) (c) [14+)

We verified their responses to ensure each subject possessed the predefined skills and interests. After the experiments, we did not manually verify the correctness of subject-selected relevant documents. However, in our preliminary experiment with different subjects, we manually examined all of the relevant documents chosen by subjects and we confirmed that, on an average, nearly 90% of their choices were correct. We assume that, with the sufficient background, the subjects were capable of identifying the relevant pages. Because we used PCAT in both experiments, no subject from experiment I participated in experiment II. We summarize some demographic characteristics of the subjects in Tables VII-1 through VII-3.

To compare the two studied systems for each domain, we divided the subjects into two groups such that subjects in one group were as closely equivalent to the subjects in the other as possible with respect to their self-reported search performance, weekly browsing and searching time, and educational status. We computed the mean log search time for a task by averaging the log search times for each group.

Table VIII. Distribution of System Uses by Tasks and User Groups

Task Group	First Half Demo Tasks	Second Half Demo Tasks	Non-demo Tasks 1–13	Non-demo Tasks 14–26
Group one	PCAT	LIST	PCAT	LIST
Group two	LIST	PCAT	LIST	PCAT

Table IX-1. Average Mean Log Search Time Across Tasks Associated with Four Types of Queries (PCAT vs. LIST)

Experiment	Query Length	One-word	Two-word	Three-word	Free-form	Total
	Domain-System					
I (PCAT vs. LIST)	Computer-PCAT	4.03 ± 0.38	4.00 ± 0.36	4.16 ± 0.54	3.79 ± 0.32	3.95 ± 0.18
	Computer-LIST	5.16 ± 0.37	4.67 ± 0.20	4.69 ± 0.56	3.67 ± 0.36	4.40 ± 0.22
	Finance-PCAT	<b>3.97 ± 0.34</b>	4.81 ± 0.39	5.22 ± 0.31	4.06 ± 0.35	4.47 ± 0.19
	Finance-LIST	<b>5.10 ± 0.26</b>	5.31 ± 0.37	5.08 ± 0.38	4.21 ± 0.59	4.93 ± 0.20

Table IX-2. Average Mean Log Search Time Across Tasks Associated with Four Types of Queries (PCAT vs. CAT)

Experiment	Query Length	One-word	Two-word	Three-word	Free-form	Total
	Domain-System					
II (PCAT vs. CAT)	Computer-PCAT	4.71 ± 0.32	<b>4.14 ± 0.26</b>	5.01 ± 0.37	<b>3.88 ± 0.19</b>	<b>4.30 ± 0.15</b>
	Computer-CAT	5.43 ± 0.32	<b>4.96 ± 0.26</b>	5.65 ± 0.44	<b>4.94 ± 0.34</b>	<b>5.17 ± 0.17</b>
	Finance-PCAT	4.28 ± 0.26	4.80 ± 0.35	4.70 ± 0.09	<b>4.10 ± 0.35</b>	<b>4.46 ± 0.14</b>
	Finance-CAT	4.72 ± 0.19	5.67 ± 0.44	5.10 ± 0.31	<b>5.10 ± 0.25</b>	<b>5.11 ± 0.16</b>

## 5.4 Experiment Process

In experiment I, all subjects used both PCAT and LIST and searched for the same demo and nondemo tasks. As we show in Table VIII, the program automatically switched between PCAT and LIST according to the task numbers, and the group identified by user id so users in different groups always used different systems for the same task. The same system-switching mechanism was adopted in experiment II to switch between PCAT and CAT.

## 6. EVALUATIONS

In this section, we compare two pairs of systems (PCAT vs. LIST, PCAT vs. CAT) on the basis of the mean log search time along two dimensions: query length and type of task. We also test five hypotheses using the responses to a postexperiment questionnaire provided to the subjects. Finally, we demonstrate the differences of the indices of the relevant results across all tasks for the two pairs of systems.

### 6.1 Comparing Mean Log Search Time by Query Length

We first compared the two systems by different query lengths. Tables IX-1 and IX-2 contain the average mean log search times across tasks with the same query length and  $\pm 1$  standard error for different systems in the two experiments (lower values are better). The last column of each table provides the average mean log search time across all 26 search tasks and  $\pm 1$  standard error. For most of the comparisons between PCAT vs. LIST (Table IX-1) or PCAT vs. CAT (Table IX-2), for a given domain and query length, PCAT has lower average

Table X. The  $t$ -Test Comparisons (Degrees of Freedom,  $p$ -Values)

Experiment	Domain	One-word	Two-word	Three-word	Free-form	Total
I (PCAT vs. LIST)	Computer	10, 0.058	10, 0.137	6, 0.517	18, 0.796	50, 0.116
	Finance	<b>14, 0.015</b>	10, 0.370	10, 0.752	10, 0.829	50, 0.096
II (PCAT vs. CAT)	Computer	10, 0.147	<b>10, 0.050</b>	6, 0.309	<b>18, 0.013</b>	<b>50, 0.0005</b>
	Finance	14, 0.193	10, 0.152	10, 0.237	<b>10, 0.041</b>	<b>50, 0.003</b>

mean log search times. We conducted two-tailed  $t$ -tests to determine whether PCAT was significantly faster than LIST or CAT for different domains and query lengths. Table X shows the degrees of freedom and  $p$ -values for the  $t$ -tests. The numbers in bold in the Tables IX and X highlight the systems with statistically significant differences ( $p < 0.05$ ) in average mean log search times.

In Table X, for both computer and finance domains, PCAT has a lower mean log search time than LIST for one-word query tasks with greater than 90% statistical significance. The two systems are not statistically significantly different for tasks associated with two-word, three-word, or free-form queries. Compared with a long query, a one-word query may be more vague or incomplete so a search engine may not provide relevant pages in its top results, whereas PCAT may show the relevant result at the top of a user interest. The user therefore could directly jump to the right category in PCAT and locate the relevant document quickly.

Compared with CAT, PCAT has a significantly lower mean log search time for free-form queries ( $p < 0.05$ ). The better performance of PCAT can be attributed to two main factors. First, the number of categories in the result area for CAT is often large (about 20) so even if the categorization is accurate, the user must still commit additional search effort to sift through the various categories. Second, the categorization of CAT might not be as accurate as that of PCAT because of the much larger number (8,547) of potential categories which can be expected to be less helpful in disambiguating a vague or incomplete query. The fact that category labels in CAT are longer than those in PCAT may also have a marginal effect on the time needed for scanning them.

For all 26 search tasks, PCAT has a lower mean log search time than LIST or CAT with 90% or higher statistical significance except for the computer domain in experiment I that indicates a  $p$ -value of 0.116. When computing the  $p$ -values across all tasks, we notice that the result depends on the distribution of different query lengths and types of tasks. Therefore, it is important to drill down the systems' performance for each type of task.

For reference, Table XI illustrates the systems' performance in terms of the number of tasks that had a lower mean log search time for each type of query length. For example, the table entry 4 vs. 2 for one-word query in the computer domain of experiment I indicates that four out of the six one-word query tasks had lower mean log search time with PCAT, whereas two had a lower mean log search time with LIST.

## 6.2 Comparing Mean Log Search Time for Information Gathering Tasks

According to Sellen et al. [2002], during information gathering, a user finds multiple pages to answer a set of questions. Figure 9 compares the mean log

Table XI. Numbers of Tasks with a Lower Mean Log Search Time

Experiment	Domain \ Query length	One-word	Two-word	Three-word	Free-form	Total
I (PCAT vs. LIST)	Computer	4 vs. 2	6 vs. 0	3 vs. 1	6 vs. 4	19 vs. 7
	Finance	6 vs. 2	5 vs. 1	3 vs. 3	3 vs. 3	17 vs. 9
II (PCAT vs. CAT)	Computer	4 vs. 2	5 vs. 1	3 vs. 1	10 vs. 0	22 vs. 4
	Finance	6 vs. 2	6 vs. 0	5 vs. 1	6 vs. 0	23 vs. 3

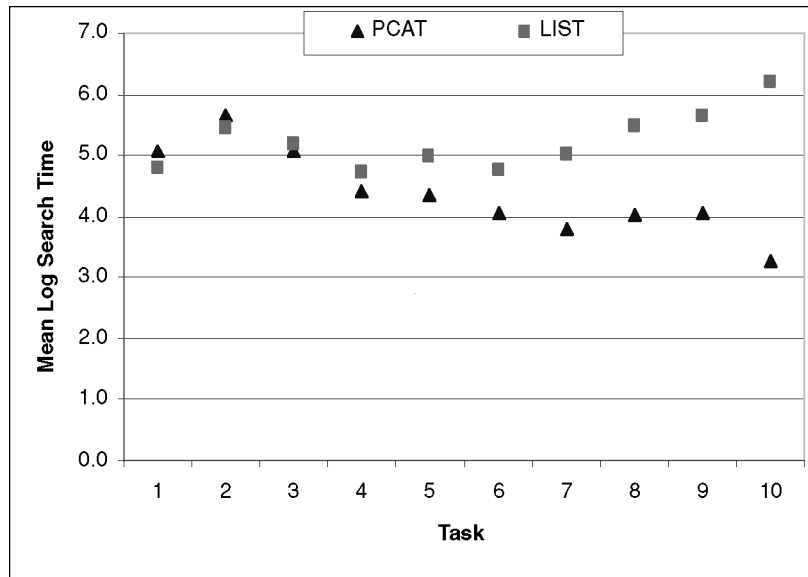


Fig. 9. Mean log search times for information gathering tasks (computer domain).

search times of the ten search tasks in the computer domain in experiment I that required the user to find two relevant results for each task. We sorted the tasks by the differences in their mean log search times between PCAT and LIST. On average, PCAT allowed the users to finish eight of ten information gathering tasks more quickly than LIST ( $t(18)$ ,  $p = 0.005$ ), possibly because PCAT already groups the similar results into a given category. Therefore, if in a category one page is relevant, the other results in that category are likely to be relevant as well. This spatial localization of relevant results enables PCAT to perform this type of task faster than LIST. For the computer domain, experiment II has a similar result in that PCAT is faster than CAT ( $t(18)$ ,  $p = 0.007$ ). Since the finance domain contains only two information gathering tasks (too few to make a statistically robust argument), we only report the mean log search times for the tasks in Table XII. We observe that the general trend of the results for the finance domain is the same as for the computer domain (i.e., PCAT has lower search time than LIST or CAT).

Table XII. Mean Log Search Times for Information Gathering Tasks (Finance Domain)

	Experiment I		Experiment II	
	PCAT	LIST	PCAT	CAT
Information Gathering task 1	6.33	6.96	6.23	7.64
Information Gathering task 2	4.62	5.13	4.72	5.61

Table XIII. Average Mean Log Search Times for Six Site Finding Tasks in Computer Domain

Experiment	System	Average Mean Log Search Time
I (PCAT vs. LIST)	PCAT	3.33 ± 0.31
	LIST	3.01 ± 0.33
II (PCAT vs. CAT)	PCAT	<b>3.51 ± 0.12</b>
	CAT	<b>4.46 ± 0.32</b>

Table XIV. The *t*-Tests for Finding Tasks

Experiment	Domain	Type of Task	Degrees, <i>p</i> -value
I (PCAT vs. LIST)	Computer	Site Finding	10, 0.508
	Computer	Finding (including Site Finding)	30, 0.592
	Finance	Finding (including Site Finding)	46, 0.101
II (PCAT vs. CAT)	Computer	Site Finding	<b>10, 0.019</b>
	Computer	Finding (including Site Finding)	<b>30, 0.013</b>
	Finance	Finding (including Site Finding)	<b>46, 0.002</b>

### 6.3 Comparing Mean Log Search Time for Site Finding Tasks

In the computer domain, there were six tasks related to finding particular sites, such as “Find the home page for the University of Arizona AI Lab.” All six tasks were associated with free-form queries, and we note that the queries from all subjects contained site names. Therefore, according to Craswell et al. [2001], those tasks were site finding tasks. Table XIII shows the average mean log search times for the site finding tasks and  $\pm 1$  standard error. There is no significant difference ( $t(10)$ ,  $p = 0.508$ ) between PCAT and LIST, as shown in Table XIV. This result seems reasonable because for this type of search task, LIST normally shows the desired result at the top of the first result page when the site name is in the query. Even if PCAT tended to rank it at the top of a certain category, users often found the relevant result faster with the LIST layout, possibly because with PCAT the users had to move to a proper category first and then look for the relevant result. However, there is a significant difference between PCAT and CAT ( $t(10)$ ,  $p = 0.019$ ); again, the larger number of output categories in CAT may have required more time for a user to find the relevant site, given that both CAT and PCAT arrange the output categories alphabetically.

### 6.4 Comparing Mean Log Search Time for Finding Tasks

As Table XIV shows, for 16 finding tasks in the computer domain, we do not observe a statistically significant difference in the mean log search time between PCAT and LIST ( $t(30)$ ,  $p = 0.592$ ), but the difference between PCAT and CAT is significant ( $t(30)$ ,  $p = 0.013$ ). However, PCAT has lower average mean

log search time than both LIST and CAT. Similarly, for 24 finding tasks in the finance domain, PCAT achieves a lower mean log search time than both LIST ( $t(46)$ ,  $p = 0.101$ ) and CAT ( $t(46)$ ,  $p = 0.002$ ). The computer domain includes 6 site finding tasks of 16 finding tasks, whereas the finance domain has only 2 (of 24). To a certain extent, this situation confirms our observations about finding tasks in the computer domain. We conclude that PCAT had a lower mean log search time for finding tasks than CAT but not LIST.

## 6.5 Questionnaire and Hypotheses

After a subject finished the search tasks with the two systems, he or she filled out a questionnaire with five multiple-choice questions designed to compare the two systems in terms of their usefulness and ease of use. We use their answers to test several hypotheses relating to the two systems.

**6.5.1 Questionnaire.** Subjects completed a five-item, seven-point questionnaire in which their responses could range from (1) strongly disagree to (7) strongly agree. (The phrase system B was replaced by system C in experiment II. As explained in footnote 13, systems A, B, and C refer to PCAT, LIST, and CAT, respectively.)

- Q1. System A allows me to identify relevant documents more easily than system B.
- Q2. System B allows me to identify relevant documents more quickly than system A.
- Q3. I can finish search tasks faster with system A than with system B.
- Q4. It's easier to identify one relevant document with system B than with system A.
- Q5. Overall I prefer to use system A over system B.

**6.5.2 Hypotheses.** We developed five hypotheses corresponding to these five questions. (The phrase system B was replaced by system C for experiment II.)

- H1. System A allows users to identify relevant documents more easily than system B.
- H2. System B allows users to identify relevant documents more quickly than system A.
- H3. Users can finish search tasks more quickly with system A than with system B.
- H4. It is easier to identify one relevant document with system B than with system A.
- H5. Overall, users prefer to use system A over system B.

## 6.6 Hypothesis Test Based on Questionnaire

Table XV shows the means for the choice responses to each of the questions in the questionnaire. Based on seven scale options described in Section 6.5, we

Table XV. Mean Responses to Questionnaire Items. Degrees of Freedom: 13 for Computer and 19 for Finance in Experiment I; 15 for Computer and 19 for Finance in Experiment II

Experiment	Domain	Q1	Q2	Q3	Q4	Q5
I (PCAT vs. LIST)	Computer	6.21***	2.36***	5.43*	2.71*	5.57**
	Finance	5.25	3.65*	5.45***	3.65**	5.40**
II (PCAT vs. CAT)	Computer	6.25***	2.00***	6.06***	2.50***	6.31***
	Finance	6.20***	1.90***	6.20***	2.65*	6.50***

\*\*\*  $p < 0.001$ , \*\*  $p < 0.01$ , \*  $p < 0.05$ .

compute numbers in this table by replacing strongly disagree with 1, strongly agree by 7, and so on.

As each question in Section 6.5.1 corresponds to a hypothesis in Section 6.5.2, we conducted a two-tailed  $t$ -test based on subjects' responses to each question to test the hypotheses. We calculated  $p$  values by comparing the subjects' responses with the mean, neither agree nor disagree that had a value of 4. The table shows that for both computer and finance domains, H1, H3, and H5 are supported with at least 95% significance, and H2 and H4 are not supported.<sup>18</sup> The only exception to these results is that we find only 90% significance ( $p = 0.083$ ) for H1 in the finance domain of experiment I. According to these responses on the questionnaire, we conclude that users perceive PCAT as a system that allows them to identify relevant documents more easily and quickly than LIST or CAT.

Several results reported in a recent work [Käki 2005] are similar to our findings. In particular,

- categories are helpful when document ranking in a list interface fails, which fits with our explanation of why PCAT is faster than LIST for short queries;
- when desired results are found at the top of the list, the list interface is faster, in line with our result and analysis pertaining to site finding tasks;
- Categories make it easier to access multiple results, consistent with our report for the information gathering tasks.

However, the categorization employed in Käki [2005] does not use examples to build a classifier. The author simply identifies some frequent words and phrases in search result summaries and uses them as category labels. Hence, each frequent word or phrase becomes a category (label). A search result is assigned to a category if the result's summary contains the category label. Käki [2005] also does not analyze or compare the two interfaces according to different types of tasks. Moreover, Käki [2005: Figure 3] shows, though without explicit explanations, that categorization is always slower than a list. This result contradicts our findings and several prior studies [e.g., Dumais and Chen 2001]. We notice that the system described by Käki [2005] uses a list interface to show the search results by default so a user may always look for a desired page from

<sup>18</sup>For example, the mean choice in the computer domain for H2 was 2.36 with  $p < 0.001$ . According to our scale, 2 means disagree and 3 means mildly disagree, so a score of 2.36 indicates subjects did not quite agree with H2. Hence, we claim that H2 is not supported. The same is true for H4.

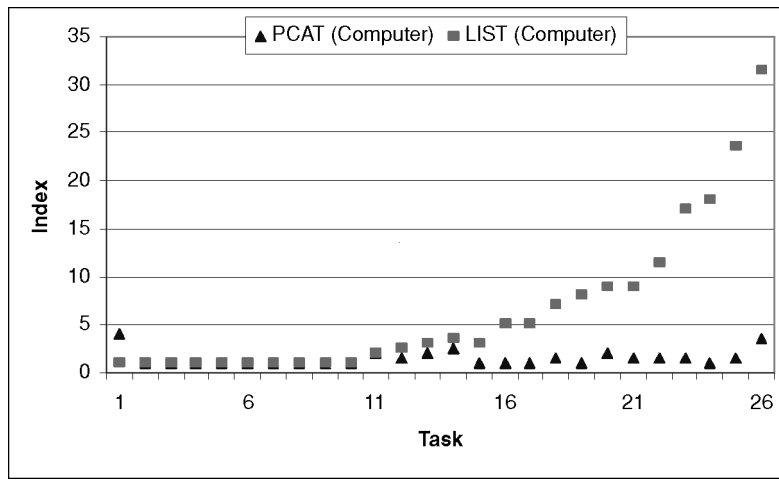


Fig. 10-1. Indices of relevant results in PCAT and LIST (computer domain).

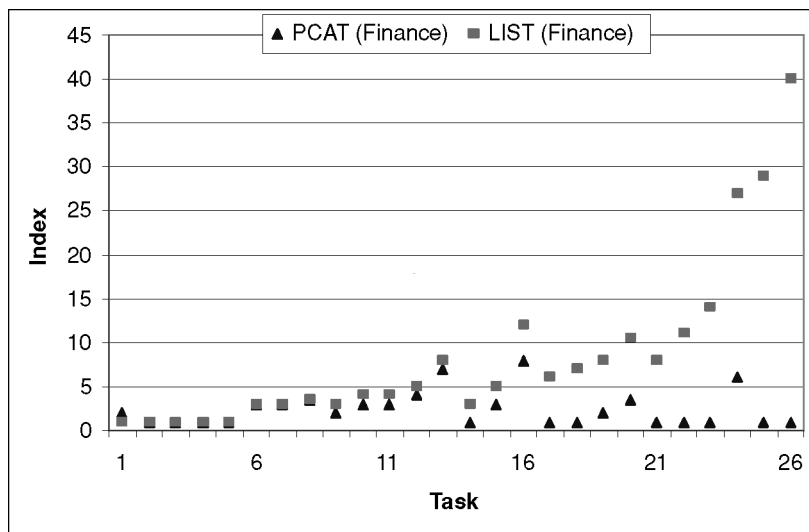


Fig. 10-2. Indices of relevant results in PCAT and LIST (finance domain).

the list interface first and switch to the category interface only if he or she does not find it within a reasonable time.

### 6.7 Comparing Indices of Relevant Results

To better understand why PCAT was perceived as faster and easier to use by the subjects as compared with LIST or CAT, we looked at the indices of relevant results in the different systems. An expert from each domain completed all search tasks using PCAT and LIST. Using the relevant results identified by them, we compare the indices of the relevant search results for the two systems, as we show in Figures 10-1 and 10-2.

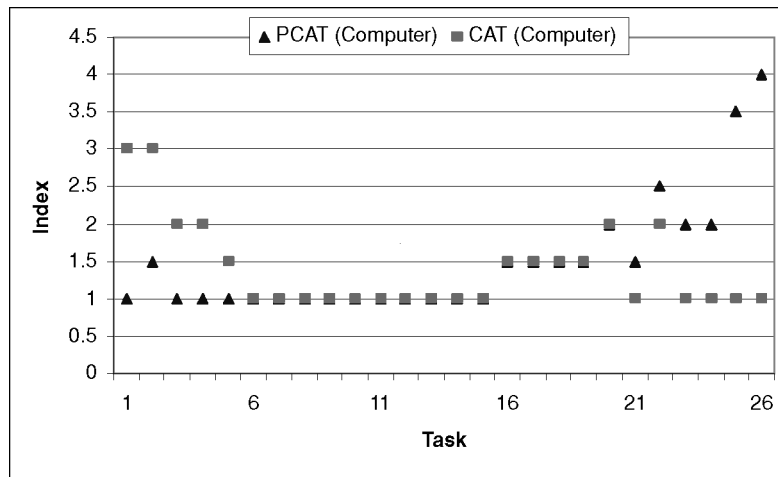


Fig. 11-1. Indices of relevant results in PCAT and CAT (computer domain).

We sort the tasks by the index differences between LIST and PCAT in ascending order. Thus, the task numbers on the x-axis are not necessarily the original task numbers in our experiments. Because PCAT organizes the search results into different categories (interests), the index of a result reflects the relative position of that result under a category. In LIST, a relevant result's index number equals its relative position on the particular page on which it appears plus ten (i.e., the number of results per page) times the number of preceding pages. Thus, a result that appears in the fourth position on the third page would have an index number of 24 ( $4 + 10 \times 2$ ). If users had to find two relevant results for a task, we took the average of the indices. In Figure 10-1, PCAT and LIST share the same indices in 10 of 26 tasks, and PCAT has lower indices than LIST in 15 tasks. In Figure 10-2, PCAT and LIST share the same indices in 7 of 26 tasks, and PCAT has smaller indices than LIST in 18 tasks.

Similarly, Figures 11-1 and 11-2 show indices of the relevant search results of PCAT and CAT in experiment II. The data for PCAT in Figures 11-1 and 11-2 are same as those in Figures 10-1 and 10-2, and we show tasks by the index differences between PCAT and CAT in ascending order. In Figure 11-1 for the computer domain, PCAT and CAT share same indices in 15 of 26 tasks, and CAT has lower indices in 6 tasks. In Figure 11-2 for the finance domain, the two systems share same indices in 10 of 26 tasks, and CAT has lower indices in 14 of 26 tasks.

The indices for PCAT in Figures 10 and 11, and CAT in Figures 11-1 and 11-2 reflect an assumption that a user first jumps to the right category and then finds a relevant page by looking through the results under that category. This assumption may not always hold, so Figures 10-1 and 10-2 may be optimistic in favor of PCAT. However, if the time taken to locate the right category is not large (as probably in the case of PCAT), the figures provide a possible explanation for some of the results we observe such as the lower search times for PCAT with one-word query and information gathering tasks in experiment I. However,

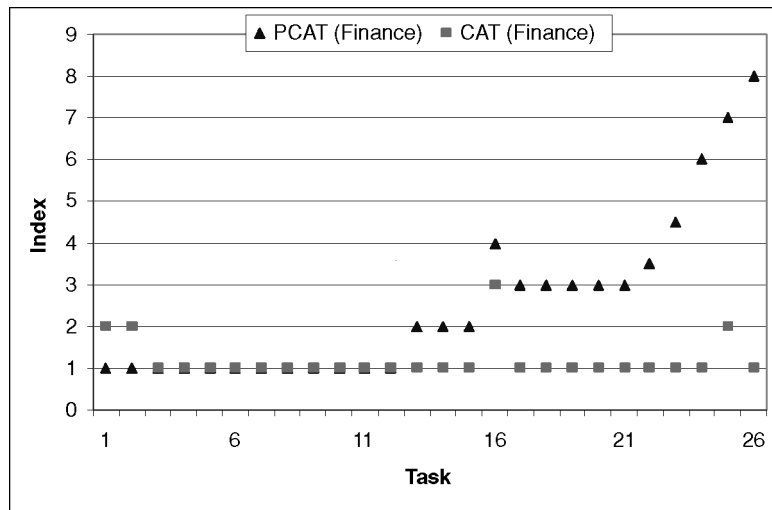


Fig. 11-2. Indices of relevant results in PCAT and CAT (finance domain).

CAT has smaller index numbers for relevant results than PCAT which may seem to contradict the better performance (lower search time) for PCAT in experiment II. We note that, due to its nonpersonalized nature, CAT has a much larger number of potential categories as compared to PCAT. Therefore, a user can be expected to take a longer time to locate the right category (before jumping to the relevant result in it) as compared to PCAT.

## 7. CONCLUSIONS

This article presents an automatic approach to personalizing Web searches given a set of user interests. The approach is well suited for a workplace setting where information about professional interests and skills can be obtained automatically from an employee's resume or a database using an IE tool or database queries. We present a variety of mapping methods which we combine into an interest-to-taxonomy mapping framework. The mapping framework automatically maps and resolves a set of user interests with a group of categories in the ODP taxonomy. Our approach then uses data from ODP to build text classifiers to automatically categorize search results according to various user interests. This approach has several advantages, in that it does not (1) collect a user's browsing or search history, (2) ask a user to provide explicit or implicit feedback about the search results, or (3) require a user to manually specify the mappings between his or her interests and taxonomy categories. In addition to mapping interests into categories in a Web directory, our mapping framework can be applied to other types of data, such as queries, documents, and emails. Moreover, the use of taxonomy is transparent to the user.

We implemented three search systems: A (personalized categorization system, PCAT), B (list interface system, LIST,) and C (nonpersonalized categorization system, CAT). PCAT followed our proposed approach and categorized search results according to a user's interests, whereas LIST simply displayed

search results in a page-by-page list, similar to conventional search engines, and CAT categorized search results using a large number of ODP categories without personalization. We experimentally compared two pairs of systems with different interfaces (PCAT vs. LIST and PCAT vs. CAT) in two domains, computer and finance. We recruited 14 subjects for the computer domain and 20 subjects for the finance domain to compare PCAT with LIST in experiment I, and 16 in the computer domain and 20 in finance to compare PCAT with CAT in experiment II. There was no common subject across the experiments. Based on the mean log search times obtained from our experiments, we examined search tasks associated with four types of queries. We also considered different types of search tasks to tease out the relative performances of the compared systems as the nature of task varied.

We find that PCAT outperforms LIST for searches with short queries (especially one-word queries) and for information gathering tasks; by providing personalized categorization results, PCAT also is better than CAT for searches with free-form queries and for both information gathering and finding tasks. From subjects' responses to five questionnaire items, we conclude that, overall, users identify PCAT as a system that allows them to find relevant pages more easily and quickly than LIST or CAT. Considering the fact that most users (even noncasual users) often cannot issue appropriate queries or provide query terms to fully disambiguate what they are looking for, a PCAT approach could help users find relevant pages with less time and effort. In comparing two pairs of search systems with different presentation interfaces, we realize that no system with a particular interface is universally more efficient than the other, and the performance of a search system depends on parameters such as the type of search task and the query length.

## 8. LIMITATIONS AND FUTURE DIRECTIONS

Our search tasks were generated on the basis of user interests. We realize some limitations of this experimentation setup in adequately capturing the workplace scenario. The first limitation is that some of the user interests may not be known in a real-world application, and hence some search tasks may not reflect the known user interests. Secondly, a worker may search for information that is unrelated with his or her job. In both of these cases, tasks may not match up with any of the known interests. However, these limitations reflect a general fact that personalization can only benefit based on what is known about the user. A future direction of research is to model the dynamics of user interests over time.

For the purposes of a comparative study, we carefully separated the personalized system (PCAT) from the nonpersonalized (CAT) one by maintaining a low overlap between the two systems. This allows us to understand the merits of personalization alone. However, we can envision a new system that is a combination of the current CAT and PCAT systems. In particular, the new system replaces the Other category in PCAT by adding categories of ODP that match the results that are currently placed in the Other category. A study of such a PCAT+CAT system could be a future direction for this research. An interesting

and related direction is a smart system that can automatically choose a proper interface (e.g., categorization, clustering, list) to display search results on the basis of the nature of the query, the search results, and the user interest profile (context).

As shown in Figures 5 and 8, for PCAT in experiments I and II and CAT in experiment II, we rank the categories alphabetically but always leave the Other category at the end.<sup>19</sup> There are various alternatives for the order in which categories are displayed such as by the number of (relevant) results in each category or by the total relevance of results under each category. We recognize that choosing different methods may provide different individual and relative performances. Also, CAT tends to show more categories on the main page than PCAT. On one hand, more categories on a page may be a negative factor for locating a relevant result. On the other hand, more categories provide more results in the same page which may speed up the discovery of a relevant result as compared to clicking a More link to open another window. We think that the issues of category ordering and number of categories on a page deserve further examination.

From the subjects' log files, we observed that when some of the subjects could not find a relevant document under a relevant category due to result misclassification, they moved to another category or tried a new query. Such a situation can be expected to increase the search time for categorization-based systems. Thus, another direction of future research is to compare different result classification techniques based on their effect on mean log search time.

It would be worthwhile to study the performance of result categorization using other types of data such as title and snippets (from search engine results) instead of page content which would save the time on fetching Web pages. In addition, it may be interesting to examine how a user could improve his or her performance in Internet searches in a collaborative (e.g., intranet) environment. In particular, we would like to measure the benefit the user can derive from the search experiences of other people with similar interests and skills in a workplace setting.

#### ACKNOWLEDGMENTS

During the program development, in addition to the software tools mentioned in prior sections of the article, we employed BrowserLauncher<sup>20</sup> by Eric Albert and XOM<sup>21</sup> (XML API) by Elliotte Rusty Harold. We thank them for their work. We would also like to thank the Associate Editor and anonymous reviewers for their helpful suggestions.

#### REFERENCES

BRIN, S. AND PAGE, L. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Sys.* 30, 1-7, 107-117.

<sup>19</sup>For the computer domain in experiment I, PCAT shows C++ and Java before other alphabetically ordered interests, and the Other category is at the end.

<sup>20</sup><http://browserlauncher.sourceforge.net/>.

<sup>21</sup><http://www.cafeconleche.org/XOM/>.

- BRODER, A. 2002. A taxonomy of Web search. *ACM SIGIR Forum* 36, 2, 3–10.
- BUDZIK, J. AND HAMMOND, K. 2000. User interactions with everyday applications as context for just-in-time information access. In *Proceedings of the 5th International Conference on Intelligent User Interfaces*. New Orleans, LA, 44–51.
- BUTLER, D. 2000. Souped-up search engines. *Nature* 405, 112–115.
- CARROLL, J. AND ROSSON, M. B. 1987. The paradox of the active user. In *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, J. M. Carroll, Ed. MIT Press, Cambridge, MA.
- CHIRITA, P. A., NEJDL, W., PAIU, R., AND KOHLSCHUTTER, C. 2005. Using ODP metadata to personalize search. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Salvador, Brazil, 178–185.
- CRASWELL, N., HAWKING, D., AND ROBERTSON, S. 2001. Effective site finding using link information. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New Orleans, LA, 250–257.
- CUTTING, D. R., KARGER, D. R., PEDERSEN, J. O., AND TUKEY, J. W. 1992. Scatter/Gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Copenhagen, Denmark, 318–329.
- DEERWESTER, S., DUMAIS, S. T., FURNAS, G. W., LANDAUER, T. K., AND HARSHMAN, R. 1990. Indexing by latent semantic analysis. *J. Amer. Soc. Inform. Sci.* 41, 6, 391–407.
- DIETTERICH, T. G. 1997. Machine learning research: Four current directions. *AI Magazine* 18, 4, 97–136.
- DREILINGER, D. AND HOWE, A. E. 1997. Experiences with selecting search engines using metasearch. *ACM Inform. Sys.* 15, 3, 195–222.
- DUMAIS S. AND CHEN, H. 2001. Optimizing search by showing results in context. In *Proceedings of Computer-Human Interaction*. Seattle, WA, 277–284.
- FINKELSTEIN, L., GABRILOVICH, E., MATIAS, Y., RIVLIN, E., SOLAN, Z., WOLFMAN, G., AND RUPPIN, E. 2002. Placing search in context: The concept revisited. *ACM Trans. Inform. Syst.* 20, 1, 116–131.
- GAUCH, S., CHAFFEE, J., AND PRETSCHNER, A. 2003. Ontology-based personalized search and browsing. *Web Intell. Agent Syst.* 1, 3/4, 219–234.
- GLOVER, E., LAWRENCE, S., BRIMINGHAM, W., AND GILES, C. L. 1999. Architecture of a metasearch engine that supports user information needs. In *Proceedings of the 8th International Conference on Information Knowledge Management*. Kansas City, MO, 210–216.
- HAFRI, Y. AND DJERABA, C. 2004. Dominos: A new Web crawler’s design. In *Proceedings of the 4th International Web Archiving Workshop (IWA)*. Beth, UK.
- HARRIS, Z. 1985. Distributional structure. In *The Philosophy of Linguistics*. Katz, J. J., Ed. Oxford University Press, Oxford, UK. 26–47.
- HAVELIWALA, T. H. 2003. Topic-Sensitive PageRank. *IEEE Trans. Knowl. Data Engin.* 15, 4, 784–796.
- JANSEN, B. J., SPINK, A., BATEMAN, J., AND SARACEVIC, T. 1998. Real life information retrieval: A study of user queries on the Web. *ACM SIGIR Forum* 32, 1, 5–17.
- JANSEN, B. J., SPINK, A., AND SARACEVIC, T. 2000. Real life, real users, and real needs: A study and analysis of user queries on the Web. *Inform. Process. Manag.* 36, 2, 207–227.
- JANSEN, B. J., SPINK, A., AND PEDERSON, J. 2005. A temporal comparison of AltaVista Web searching. *J. Amer. Soc. Inform. Sci. Techno.* 56, 6, 559–570.
- JANSEN, B. J. AND SPINK, A. 2005. An analysis of Web searching by european AlltheWeb.com users. *Inform. Process. Manage.* 41, 361–381.
- JEH, G. AND WIDOM, J. 2003. Scaling personalized Web search. In *Proceedings of the 12th International Conference on World Wide Web*. Budapest, Hungary, 271–279.
- KÄKI, M. 2005. Findex: Search result categories help users when document ranking Fails. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Portland, OR, 131–140.
- KRAFT, R., MAGHOUL, F., AND CHANG, C. C. 2005. Y!Q: Contextual search at the point of inspiration. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*. Bremen, Germany, 816–823.
- LAWRENCE, S. 2000. Context in Web search. *IEEE Data Engi. Bull.* 23, 3, 25–32.

- LEORY, G., LALLY, A. M., AND CHEN, H. 2003. The use of dynamic contexts to improve casual internet searching. *ACM Trans. Inform. Syst.* 21, 3, 229–253.
- LIU, F., YU, C., AND MENG, W. 2004. Personalized Web search for improving retrieval effectiveness. *IEEE Trans. Knowl. Data Engin.* 16, 1, 28–40.
- MALTZ, D. AND EHRLICH, K. 1995. Pointing the way: Active collaborative filtering. In *Proceedings of the Conference on Computer-Human Interaction*. Denver, CO, 202–209.
- MENCZER, F., PANT, G., AND SRINIVASAN, P. 2004. Topical Web crawlers: Evaluating adaptive algorithms. *ACM Trans. Internet Techn.* 4, 4, 378–419.
- MILLER, G., BECKWITH, R., FELLBAUM, C., GROSS, D., AND MILLER, K. 1990. Introduction to WORDNET: An online lexical database. *Int. J. Lexico.* 3, 4, 235–244.
- NAJORK, M. AND HEYDON, A. 2001. High-performance Web crawling. In *Handbook of Massive Data Sets*, J. ABELLO, P. PARDALOS, AND M. RESENDE, Eds. Kluwer Academic Publishers, 25–45.
- OYAMA, S., KOKUBO, T., AND ISHIDA, T. 2004. Domain-specific Web search with keyword spices. *IEEE Trans. Knowl. Data Engin.* 16, 1, 17–27.
- PITKOW, J., SCHUTZE, H., CASS, T., COOLEY, R., TURNBULL, D., EDMONDS, A., ADAR, E., AND BREUEL, T. 2002. Personalized search. *Commun. ACM* 45, 9, 50–55.
- PORTER, M. 1980. An Algorithm for suffix stripping. *Program* 14, 3, 130–137.
- RILOFF, E. AND SHEPHERD, J. A Corpus-based approach for building semantic lexicons. In *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing*. Providence, RI, 117–124.
- SALTON, G. AND MCGILL, M. J. 1986. *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, NY.
- SEBASTIANI, F. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.* 34, 1, 1–47.
- SELLEN, A. J., MURPHY, R., AND SHAW, K. L. 2002. How knowledge workers use the Web. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing our World, Changing Ourselves*. Minneapolis, MN, 227–234.
- SHAKES, J., LANGHEINRICH, M., AND ETZIONI, O. 1997. Dynamic reference sifting: A Case study in the homepage domain. In *Proceedings of the 6th International World Wide Web Conference*. Santa Clara, CA, 189–200.
- SHEN, D., CHEN, Z., YANG, Q., ZENG, H., ZHANG, B., LU, Y., AND MA, W. 2004. Web-page classification through summarization. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Sheffield, South Yorkshire, UK, 242–249.
- SHEN, X., TAN, B., AND ZHAI, C. X. 2005a. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Salvador, Brazil, 43–50.
- SHEN, X., TAN, B., AND ZHAI, C. X. 2005b. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*. Bremen, Germany, 824–831.
- SPERETTA, M. AND GAUCH, S. 2005. Personalizing search based on user search histories. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence*. Compiegne University of Technology, France, 622–628.
- SRINIVASAN, P., MENCZER, F., AND PANT, G. 2005. A general evaluation framework for topical crawlers. *Inform. Retrieval* 8, 3, 417–447.
- SUGIYAMA, K., HATANO, K., AND YOSHIKAWA, M. 2004. Adaptive Web search based on user profile constructed without any effort from users. In *Proceedings of the 13th International Conference on World Wide Web*. New York, NY, 675–684.
- SULLIVAN, D. 2000. NPD Search and portal site study. Search engine watch. <http://searchenginewatch.com/sereport/article.php/2162791>.
- TAN, A. H. 2002. Personalized information management for Web intelligence. In *Proceedings of World Congress on Computational Intelligence*. Honolulu, HI, 1045–1050.
- TAN, A. H. AND TEO, C. 1998. Learning user profiles for personalized information dissemination. In *Proceedings of International Joint Conference on Neural Network*. Anchorage, AK, 183–188.
- TEEVAN, J., DUMAIS, S. T., AND HORVITZ, E. 2005. Personalizing search via automated analysis of interests and activities. In *Proceedings of 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Salvador, Brazil, 449–456.

- WEN, J. R., NIE, J. Y., AND ZHANG, H. J. 2002. Query clustering using user logs. *ACM Trans. Inform. Syst.* 20, 1, 59–81.
- XU, J. AND CROFT W. B. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Zurich, Switzerland, 4–11.
- YANG, Y. AND LIU, X. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Berkeley, CA, 42–49.
- ZAMIR, O. AND ETZIONI, O. 1999. Grouper: A dynamic clustering interface to Web search results. *Compu. Netw.: Int. J. Comput. Telecomm. Netw.* 31, 11–16, 1361–1374.

Received October 2005; revised March and August 2006; accepted September 2006