Beyond Hyperlinks: Organizing Information Footprints in Search Logs to Support Effective Browsing

Xuanhui Wang†, Bin Tan†, Azadeh Shakery‡, ChengXiang Zhai†
†Department of Computer Science
University of Illinois at Urbana-Champaign
{xwang20, bintan, czhai}@illinois.edu

‡School of Electrical and Computer Engineering
University of Tehran
shakery@ut.ac.ir

ABSTRACT

While current search engines serve known-item search such as homepage finding very well, they generally cannot support exploratory search effectively. In exploratory search, users do not know their information needs precisely and also often lack the needed knowledge to formulate effective queries, thus querying alone, as supported by the current search engines, is insufficient, and browsing into related information would be very useful. Currently, browsing is mostly done by following hyperlinks embedded on Web pages. In this paper, we propose to leverage search logs to allow a user to browse beyond hyperlinks with a multi-resolution topic map constructed based on search logs. Specifically, we treat search logs as "footprints" left by previous users in the information space and build a multi-resolution topic map to semantically capture and organize them in multiple granularities. Such a topic map can support a user to zoom in, zoom out, and navigate horizontally over the information space, and thus provide flexible and effective browsing capabilities for end users. To test the effectiveness of the proposed methods of supporting browsing, we rely on real search logs and a commercial search engine to implement our proposed methods. Our experimental results show that the proposed topic map is effective to support browsing beyond hyperlinks.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Search process, Clustering

General Terms: Algorithms

Keywords: Beyond hyperlinks, effective browsing, multi-resolution topic maps, information footprints

1. INTRODUCTION

Users' search tasks vary a lot from a simple known-item search to very complex exploratory search [48]. In known-item search, a user has a well-defined information need and can generally formulate an effective query and thus the current search engines often work very well. In exploratory search, however, the information need is often complex and vague, and the goal of search is mainly to gather and study information about some topic. Thus a user generally does not know well about the information to be found in exploratory search (which is the reason why the user needs to initiate the search in the first place). As a result, it is often difficult for a user to formulate effective queries in exploratory search, and the user has to reformulate queries many times in a trial-and-error manner. For example, when a user wants to buy a used car, what he/she needs is not just a single piece of information such as a list of used car dealers, but also opinions about the dealers by previous customers, advantages/disadvantages of different brands, and advice on car insurance, etc. Formulating effective queries to find all this information is quite challenging, especially for a user who does not know well about the domain. For these reasons, the current search engines generally do not perform well for exploratory search compared with known-item search [28]. Since exploratory search happens very often, it is very important to study how to help users to conduct effective exploratory search [28, 48].

Querying alone is often insufficient to support exploratory search well due to the difficulty in formulating good queries. When a user is unable to formulate effective queries, browsing would be intuitively very useful because it enables a user to navigate into relevant information (and explore the information space in general) *without* formulating a query. Indeed, being able to browse the Web through hyperlinks is essential to web users, and quite often, a user would find relevant information by following hyperlinks in the result pages [12]. Had all the hyperlinks been broken, the utility of a search engine would be significantly reduced.

Unfortunately, with the current search engines, browsing is mostly through following static hyperlinks. This is very restrictive and would not allow a user to go very far in the information space. A main research question we want to study in this paper is how to support browsing more effectively for ad hoc exploratory queries so that users can go beyond hyperlinks to freely navigate into remotely related topics in the entire information space.

There have been some efforts on providing more powerful navigation support, but they tend to rely on manually created meta data and usually can only support "vertical" navigation through hierarchies. For example, Web directories such as Yahoo![1] and ODP[2] (Open Directory Project) directories use manually constructed hierarchies to support drill-down and roll-up. Faceted hierarchies [20, 50] go beyond a single hierarchy to support browsing with multiple hierarchies. The multiple hierarchies are carefully designed

CIKM'09, November 2–6, 2009, Hong Kong, China.

[1] http://dir.yahoo.com/
[2] http://www.dmoz.org/

and built along different dimensions in a given domain (e.g., time or location dimensions for news articles) so that a user can flexibly choose different dimensions to narrow down their search. However, these hierarchies are mostly created manually and thus need a lot of human efforts to adapt them in a new domain. More importantly, they only allow users to move *vertically* i.e., drill-down or roll-up. These two operators are not sufficient enough for users to exploit related information. Horizontal navigation in educational hypertext is studied in [7] but it is only limited to connect basic documents. When users' information needs are complex, combining vertical and horizontal navigation is especially beneficial.

Thus it remains a significant challenge to effectively support effective browsing beyond hyperlinks for arbitrary *ad-hoc queries*. Ideally we should support both vertical and horizontal navigation. In this paper, we propose to achieve this goal through a novel navigation structure: multi-resolution topic maps. A topic map is an analogy of geographical map, but it is to capture the global structure of an information space. Technically, a topic map is an extension of hierarchy but it has two distinct features: (1) Topics in the same level of the map have similar resolution. (2) There are horizontal links between topics in the same level, besides the vertical links as in a hierarchy. In a multi-resolution topic map, topics with coarse granularities (i.e., low resolutions) subsume those with finer granularities (i.e., high resolutions). For example, "car" can subsume "car rental," "car pricing" and "car insurance." Related topics in the same granularity are connected horizontally. For example, "flight" can be connected with "hotel," "vacation" and "car." With a multi-resolution topic map, a user can easily reach topics of different granularities through *vertical navigation* (i.e., zoom in and zoom out) as well as topics with the same level of granularity through *horizontal navigation* (i.e., moving to a neighbor area), achieving flexible navigation. Just as a geographical map can help a tourist tour a city, a topic map can guide a user in an information space.

To construct a multi-resolution topic map, we rely on past search engine logs, which can be regarded as "footprints" left by previous users in the information space. Just as the footprints of previous visitors in a park can help guide future visitors, the footprints in an information space left by previous users can also help future information seekers. Our multi-resolution topic map is to capture these footprints in a semantical way so that it can guide a user to reach relevant information by following the "wisdom of crowds." Compared with past work on exploiting search logs (which is discussed in length in Section 2), turning the entire search logs into a topic map is a novel way to leverage search logs to help search. Moreover, as new users use the topic map to navigate the information space, they will add footprints to the information space, which can then be used to improve and refine the map dynamically for the benefit of future users, thus enabling users to surf the information space in a collaborative manner. The topic map essentially serves as a sustainable and continuously growing infrastructure for social surfing.

We extend an existing clustering method to build a topic map in three steps: (1) finding topic regions in different granularities with dense footprints, (2) building horizontal links to connect topics with the same granularity, and (3) labelling topic regions with meaningful keywords. Our method of constructing the map is completely unsupervised and can be easily applied to any search logs. The constructed topic map can be used to support flexible browsing in a topic space, which can be integrated with regular querying in a search system to enable users to flexibly query, browse a map node, and navigate over the topic map to find relevant information. We evaluate the potential benefit of our topic map using a sample of search logs from a commercial search engine. The experimental results show that the idea of supporting flexible browsing with a multi-resolution topic map is promising and that our search log-based topic map is effective in helping a user reach useful pages quickly through pure browsing.

The main contributions of this paper are:

- We propose a novel structure (i.e., multi-resolution topic map) to enable a user to go beyond hyperlink-following to flexibly navigate in the information space.
- We propose a novel way to exploit search logs for improving search by treating search logs as information footprints and organizing all the search logs into a topic map, which can potentially provide sustainable social surfing in the information space.
- We propose a general method for turning any search logs into a multi-resolution topic map based on the star-clustering algorithm.
- We evaluate the effectiveness of the novel topic map in supporting browsing and show promising results.

The rest of the paper is organized as follows. We first review related work in Section 2. We then define a multi-resolution topic map formally in Section 3 and describe our algorithms of building a topic map in Section 4. After that, we will discuss how to use a topic map to support flexible browsing in Section 5, and present our experiment results in Section 6. We discuss how we can leverage a topic map to unify querying and browsing in a single navigation framework in Section 7 and conclude our paper in Section 8.

2. RELATED WORK

Exploratory Search. Exploratory search has attracted much attention recently [28, 25, 16, 48]. Different from the previous works in the HCI community which mostly focus on interface design [20, 10, 18], our emphasis is on turning search logs into a topic map to enable users to navigate flexibly in the whole information space. Querying and browsing are two common information search paradigms [14, 26, 27, 51, 17]. A recent study [41] has shown that orienteering behaviors are common in search and can not be well supported by direct querying. Thus supporting browsing is indeed important, especially for exploratory search needs. Most existing work has relied on catalog, meta-data, or other domain-specific knowledge to support browsing [3, 29, 5]. Our work focuses on the general Web domain and leverages naturally-growing unstructured search logs to support browsing for ad-hoc topics.

Hierarchy and Faceted Hierarchies. Traditional hierarchies are Web directories such as Yahoo! and ODP directories. Both are built manually and only focus on *vertical* relations. As a result, current hierarchies do not have a clear notion of resolution and thus topics in the same level of a hierarchy are not always comparable with respect to granularities. Our multi-resolution topic map is an extension of traditional hierarchy with distinct features: topics in the same level have similar granularities and horizontal links are constructed to connect related topics. Hierarchical clustering has been studied extensively in information retrieval [49, 36] and data mining [19]. Our method of building multi-resolution topic maps can be regarded as an extension of hierarchical clustering to make the topics in the same level have similar granularities and create meaningful horizontal links.

Faceted hierarchies [16, 50] is an extension of traditional hierarchies to support browsing. They contain multiple hierarchies along different dimensions and have been used in commercial Websites and digital libraries [38, 25]. Most of the current facets are built manually and designed specifically for a well-understood domain. Automatically constructing faceted hierarchies is admitted to be a very challenging task [16]. Some work such as [36, 11] tries to automatically extract facet terms in a text database with certain progress. With the same goal of supporting browsing, our topic maps are constructed automatically based on search logs which are naturally available and can sustain continuous revision and improvement of the topic map.

While hierarchy and faceted hierarchies are generally used to support a user to explore *inside* the search results. Our topic maps can enable a user to *horizontally* navigate to neighbor areas which is *outside* of the current search results.

Search Log Mining. Search logs have been exploited for several different purposes such as query clustering [45, 4, 37], personalized search [39], learning to rank [32, 1], and query suggestion [23, 34, 43]. Post-search browsing logs are studied in [47, 6] to identify relevant pages for queries. Community-based search logs are used in [40, 17] for social search and navigation. The difference of our work from the previous work mainly lies in our attempt to characterize the *global* structure of users' information footprints and use the topic map structure to better support browsing so that users can navigate into remotely related topics.

Search Result Organization. Our work is also related to search result organization, which includes clustering based methods [53, 52, 42, 21], categorization based methods [8, 13], and faceted hierarchy based methods [20, 16]. All the work in search result organization is to help users navigate *inside* the current search results. A major difference of our work is that we help users navigate *outside* the search results to explore remotely related topic regions, which is more important for exploratory search when a user's information need is not well-defined.

Others. The notion of information footprints has been used in some previous work such as [46], where footprints were also used to build maps, trials, and annotations to help a new user for information exploration. Our novelty lies in that we treat search logs as footprints and propose algorithms to turn search logs into a multi-resolution topic map to support flexible browsing in the entire information space. Some other works such as [31, 30] do not create new ways to support browsing, but try to make the existing hyperlinks easier for users to browse. Our work is to break the limitation of hyperlink following through more flexible browsing with a topic map. Horizontal links has been studied for navigation in works such as [7, 9]. They mostly rely on the contents of document collections and try to build horizontal links between basic objects such as Web pages, while we rely on users' information footprints, i.e., search logs, and furthermore, we automatically construct both vertical and horizontal links in *multiple* granularities.

3. MULTI-RESOLUTION TOPIC MAP

In this section, we formally define multi-resolution topic maps. Suppose we have an information space consisting of a collection of documents C. We first define *Topic Region* and *Topic Region Space*:

DEFINITION 1 (TOPIC REGION). *A topic region $T \subset C$ is a subset of documents that are about a topic. For example, all the documents matching a phrase can form a topic region characterized by the phrase.*

DEFINITION 2 (TOPIC REGION SPACE). *The topic region space S is the set of all possible topic regions defined on C. That is, $S = 2^C$.*

Note that for generality, we allow the topic region space to contain potentially non-coherent topic regions. We now define *Topic Map*.

DEFINITION 3 (TOPIC MAP). *A topic map $M = (V, E)$ is a graph with regions as vertices (i.e., $V \subset S$). An edge between two topic regions means that the user can navigate from one topic region into the other. That is, if $(v_i, v_j) \in E$, then a user would be able to navigate between v_i and v_j.*

A topic map is to guide a user navigating in the information space just as a geographic map can guide a traveller touring a city. As a geographic map would show roads to connect different regions to enable transportation, our topic map would also have semantic connections between topic regions to enable browsing.

In any interesting application, especially an unrestricted domain such as Web, the topic map can be quite large. How to facilitate a user in navigating on this map would be itself a challenge. We solve this problem by constructing a topic map with multiple resolutions. The idea of multiple resolutions is again analogous to the idea of displaying a geographic map in multiple resolutions, and it would allow a user to get to one region from another easily on the map. Specifically, if the user wants to visit a topic region far away on the map, he/she can simply "zoom out" to a high-level general topic region (e.g., sports) and quickly navigate into a quite different (general) topic region (e.g., economy); similarly, if the user is interested in a region and wants to explore more in the region, he/she can "zoom in" and get a detailed view of the region (e.g., from "sports" to a set of regions such as "baseball," "basketball," and "football").

We now define the multi-resolution topic map formally.

DEFINITION 4 (MULTI-RESOLUTION TOPIC MAP). *A k-level multi-resolution topic map M consists of k topic maps ordered by resolution decreasingly, $M = (M_1, ..., M_k)$, such that for any two adjacent maps $M_i = (V_i, E_i)$ and $M_{i+1} = (V_{i+1}, E_{i+1})$, we have a zooming relation $Z \subset V_i \times V_{i+1}$. A zooming edge $(v_i, v_j) \in Z$ means v_i is subsumed by v_j or v_j subsumes v_i.*

In a multi-resolution topic map, we can refine browsing into *vertical browsing* and *horizontal browsing*. The zooming relation tells us how to refocus on a map with a new resolution if the user zooms in/out on a current map. Specifically, suppose the user is currently visiting region v_i on map M_i. If the user zooms in, he/she will see a set of "children" topic regions $\{v_{i-1}|(v_{i-1}, v_i) \in Z\}$ on map M_{i-1}. Similarly, if the user zooms out, he/she will see a set of "parent" topic regions $\{v_{i+1}|(v_i, v_{i+1}) \in Z\}$ on map M_{i+1}. Thus, with the zooming relation and maps of multiple resolutions, a user can potentially navigate into remotely related topics quickly.

4. SEARCH LOG BASED TOPIC MAP

While a multi-resolution topic map can be constructed in many ways, in this paper, we focus on studying how to construct such a map based on search logs. Turning search logs into a topic map to support browsing has two attractive benefits: First, since queries and click-throughs in search logs can both be regarded as "information footprints" left by previous users in the information space, thus constructing such a map would enable the current users to follow these footprints and leverage the "wisdom of crowds." Second, as new users use the map to navigate and leave more footprints, we will be able to use the new footprints to dynamically update and refine the topic map for the benefit of future users, thus achieving a powerful naturally sustainable model of social surfing.

We now present a general method for constructing a topic map to organize the information footprints in search logs. Our approach is based on an extension of the star clustering algorithm [2], which has a parameter to naturally control the granularity of the obtained topic regions, thus helping attain the goal of multiple resolutions.

4.1 Representing Footprints

We use both queries and click-throughs to represent information footprints. Our method is to generate a pseudo-document for each query. We utilize the click-through information in search logs for this purpose. For each query in the logs, we have all the clicked URLs by all past users. However, only URL information would not give meaningful representations since URLs alone are not informative enough to capture the footprints accurately. To gather rich information, we enrich each URL with additional text contents. Specifically, given any query, we can obtain its top-ranked results using the same search engine as the one from which we obtained our log data, and extract the search engine snippets of the clicked results, according to the log data. Given a query, all the snippets of its clicked URLs are used to generate a pseudo-document. Thus, each pseudo-document corresponds to a unique query and the keywords contained in the query itself can be regarded as a brief summary of the corresponding pseudo-documents. Intuitively, all these pseudo-documents and their associated queries capture the footprints in the information space and we use them to build our topic regions through clustering techniques.

4.2 Forming Topic Regions

Let $Q = \{q_1, ..., q_n\}$ be all the queries in the search logs and $L_0 = \{d_1, ..., d_n\}$ their corresponding pseudo-documents. We use the star clustering algorithm [2] to discover coherent topic regions.

Given L_0, star clustering starts with constructing a pairwise similarity graph on this collection based on the vector space model in information retrieval [35]. Then the clusters are formed by dense subgraphs that are star-shaped. These clusters form a cover of the similarity graph. Formally, for each of the n pseudo-documents $\{d_1, ..., d_n\}$ in the collection L_0, we compute a TF-IDF vector. Then, for each pair of documents d_i and d_j ($i \neq j$), their similarity is computed as the cosine score of their corresponding vectors. A similarity graph G_σ can then be constructed using a similarity threshold parameter σ as follows. Each document d_i is a vertex of G_σ. If $sim(d_i, d_j) > \sigma$, there would be an edge connecting the corresponding two vertices. After the similarity graph G_σ is built, the star clustering algorithm clusters the documents using a greedy algorithm. We outline the star clustering algorithm in Algorithm 1.

Algorithm 1 Star clustering algorithm

1: Given a parameter $\sigma (0 \leq \sigma \leq 1)$, generate a similarity graph $G_\sigma = (V, E)$.
2: Associate a flag $I(v) = unmarked$ for $\forall v \in V$.
3: **repeat**
4: Let $u = \arg\max_{I(v)=unmarked} degree(v)$, i.e., u is the *unmarked* vertex with the largest degree.
5: Mark $I(u) = center$.
6: Form a cluster $C_u : \{u\} \bigcup \{v : (u, v) \in E\}$ where u is the center of the cluster.
7: Mark $I(v) = satellite$ if $(u, v) \in E$.
8: **until** $I(v) \neq unmarked$ for $\forall v \in V$.

In star clustering, each obtained cluster is *star-shaped*, which consists of a single *center* and several *satellites*. There is only one parameter σ in the star clustering algorithm. A big σ enforces that the connected documents have high similarities, and thus the clusters tend to be small. Such a small cluster corresponds a topic region with finer granularity. On the other hand, a small σ will make the clusters big and such a cluster corresponds to a topic region with coarse granularity.

4.3 Building a Multi-Resolution Topic Map

For a multi-resolution topic map, we can build it in either a top-down or a bottom-up manner. In this section, we adopt a bottom-up hierarchical clustering method.

4.3.1 Generating Multi-Resolution Map Nodes

We use hierarchical star clustering to build map nodes and their zooming relations. Let L_0 be the set of individual queries. We apply our star clustering algorithm on L_0 with a high σ_1 values so that we can find small but very coherent topic regions. Each region/cluster provides a *center* query and all these center queries form a set L_1. Recursively, we can apply star clustering on L_1 with a medium threshold σ_2 to generate another set of center queries L_2. L_2 can then be used to generate L_3 with a small threshold σ_3 and etc. In our experiments, we generate a three-level topic map by setting $\sigma_1 = 0.7$, $\sigma_2 = 0.5$, and $\sigma_3 = 0.3$. Recursive clustering gives us clusters in different granularities. Since we have the same threshold σ for each level, we loosely ensure that all the topics in the same level have similar granularities. Each cluster is a node in our map and all clusters in L_i form the set of nodes in i-th level of our map.

4.3.2 Connecting Topic Regions for Browsing

The procedure above generates a k-level hierarchy which can support vertical zoom in/out naturally: A cluster in a coarse granularity subsumes several clusters in a finer granularity. Thus in our map, we have vertical or zooming relations among the corresponding nodes. Each cluster in different levels is a topic region which contains a set of pseudo-documents in L_0 and a set of queries.

Here we describe our methods to connect nodes/clusters in the same level to support horizontal navigation. In the same level, each cluster has a set of queries in Q and all these queries in the set can be used as the content of the cluster. Intuitively, semantically closely related clusters would have high similarities in their contents. Therefore, we can build a vector representation for each cluster and use cosine similarity score to measure the closeness of two clusters. In this paper, we propose a random walk based similarity measure which can be used to incorporate other useful information in logs such as query sequences in user sessions.

Specifically, given two clusters C_i and C_j, we would calculate a probability $P(C_j|C_i)$ to measure the probability of arriving at cluster C_j if we start a random walk from C_i. The general random walk works as follows: From C_i, we randomly walk to a query $Q_b \in C_i$. Then we randomly walk to another query Q_a from Q_b. The last step is another random walk from Q_a to a cluster C_j which contains Q_a. Therefore

$$P(C_j|C_i) = \sum_{Q_a, Q_b} P(C_j|Q_a)P(Q_a|Q_b)P(Q_b|C_i). \quad (1)$$

All those probabilities can be modelled flexibly. For example, $P(Q_a|Q_b)$ can be modelled as the probability of a user reformulates queries from Q_b to Q_a. Another version of random walk is to change Q_a and Q_b to two terms w_a and w_b respectively. Then we have a similar formula

$$P(C_j|C_i) = \sum_{w_a, w_b} P(C_j|w_a)P(w_a|w_b)P(w_b|C_i). \quad (2)$$

where $P(w_a|w_b)$ can be modelled as the probability of seeing w_a in a subsequent query given its previous query containing w_b in user sessions. Without using any additional information, we can assume $P(w_a|w_b) = 1$ if $w_a = w_b$ and 0 otherwise. Then Equation (2) can be simplified as

$$P(C_j|C_i) = \sum_w P(C_j|w)P(w|C_i). \quad (3)$$

In our experiments, we use Equation 3 and estimation $P(w|C_i) = \frac{c(w,C_i)}{\sum_w c(w,C_i)}$ and $P(C_j|w) = \frac{c(w,C_j)}{\sum_C c(w,C)}$ where $c(w,C)$ is the count of w appearing as a content word in cluster C.

4.3.3 Labelling Map Nodes

Each cluster generated above corresponds to a node in our topic map. To provide effective guidance when end users navigate in our topic map, we need to associate a meaningful label with each node. A label should be informative enough to represent the node's content in the corresponding cluster. Similar to [42], We use query words to generate labels for each node in our map since query words are more meaningful from a user's viewpoint. In this paper, we use a variant of frequent pattern algorithm to generate the labels in a top-down manner. We start from the nodes in the highest level (Level 3) of our map. For each node, we take every query in the corresponding cluster as a word sequence and find the most frequent one (unigram) or two words (bigram) in the corresponding query set as its label. For example, we can get a label "car" for a node in Level 3. After generating labels for Level 3, we apply the similar procedure to Level 2, but with a constraint that a word will not be selected if it has been used by its parent node. After we get the frequent word(s) for a node in Level 2, we *append* the label of the node's parent node in Level 3 as prefix to label the node. For example, if we get the most frequent word of a node in Level 2 as "rental" and the node's parent's label is "car", then we label the node by "car::rental". For a node in Level 1, we use the *center* queries output by the star clustering algorithm as labels.

5. BROWSING WITH TOPIC MAPS

Once a topic map is built, we can integrate it into a regular search engine to enhance browsing. We developed a prototype system based on a map constructed using a sample of search logs and a commercial search engine. Two snapshots of our system interface are shown in Figure 1. In our system, a user has access to three operators all the time: querying, viewing a map node, and navigating in the map.

Querying. When a user submits a new query through the search box (see Figure 1(a)), the search results from a search engine will be shown in the right pane. At the same time, we build a "query-extended" map by connecting the query defined topic region with its closest map nodes in Level 1. The closeness is computed as follows: given the query, we first retrieve the top m pseudo-documents using the standard Okapi method [33]. Each pseudo-document corresponds to a past query. For nodes/clusters in Level 1, we count how many of the retrieved pseudo-documents each contains and use these counts as the closeness measure. The closest map nodes are then ordered accordingly and shown in the left pane of Figure 1.

Viewing a map node. When a user *double* clicks on a map node, we would display the topic region corresponding to the current node on the right pane (see Figure 1(b)). In this paper, the topic region consists of two parts: (1) the click-throughs of all the past queries in the current map node, and (2) the returned search results of using the label of the current map node as a query. The content in the right pane shows a user the most frequently visited pages for the current node (i.e., footprints) and also the search results. The user can thus follow the footprints of previous users or leave his/her own footprints by examining new search results.

Navigating in the map. The left pane in our interface is to let a user navigate in the map. When a user clicks on a map node, this pane will be refreshed and a local view around the clicked node will be displayed. Specifically, we show the parents, the children, and the horizontal neighbors of the current node in focus (labelled as "center"). A user can thus zoom into a child node, zoom out to a parent node, or navigate into a horizontal neighbor node. In our current implementation, the children and neighbor nodes are ordered by Equation 3 and the parent nodes are ordered by their size, i.e., the number of children they contain.

The three different operators provide flexibility for users to conduct either querying or browsing interchangeably. Navigating in the map helps a user reach related topic regions through "semantic roads" without needing to formulate a query.

6. EXPERIMENTS

6.1 Data Set

Our data set is a sample of search log data from Microsoft Live Labs. In total, this log data spans 31 days from 05/01/2006 to 05/31/2006; there are 8,144,000 queries, 3,441,000 distinct queries, and 4,649,000 distinct URLs in the raw data.

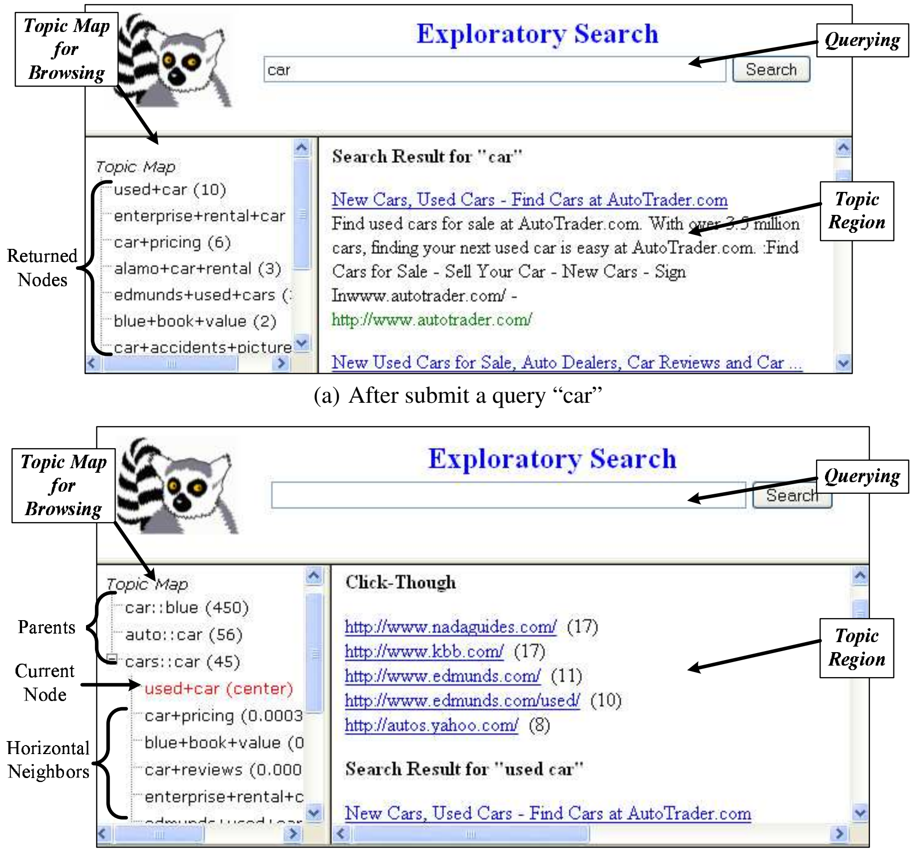
To test our system, we separate the whole data set into two parts according to the time: the first 2/3 data is used to simulate the historical data that a search engine accumulated. We treat this log data as footprints and build our topic map. The last 1/3 data is held out to serve as our test cases, which will be described in detail in a later section. In the history collection, we clean the logs by only keeping those frequent, well-formatted, English queries (queries which only contain characters 'a', 'b', ..., 'z', and space, and appear more than 5 times). After cleaning, we get 169,057 unique queries in our history collection in total. On average, each query has 3.5 distinct clicks. For each query, we build a "pseudo-document" based on its clicked snippets. The average length of these pseudo-documents is 68 words and the total data size of our history collection is 129MB.

6.2 Three-Level Topic Map

Based on the history collection we described above, we built a three-level topic map according to the method we described in Section 4. The first level has the finest granularity and the third level has the most coarse granularity. We show a part of the nodes/clusters in different levels in Figure 2 where each node is represented by its label words and arrows and lines denote vertical and horizontal relations, respectively. We use past queries as labels for the first-level nodes, but we use query words instead of entire queries to label the nodes/clusters in the second and third levels. A major reason for doing this is that user queries tend to be very specific, so they may not always be suitable to label clusters in a coarse granularity, and using query words may be better. From this figure, we can see that the nodes/clusters in the first level are relatively narrow topics such as "alamo car rental." On the other hand, the second and third level clusters represent more general concepts such as "car." On the same level, we have horizontal neighbors whose closeness is calculated by random walk based similarity in Equation 3. We can see that the closest neighbors are indeed related. For example, we can go from "car" to "auto," to "loan," or to "insurance." All these neighbors provide useful guidance/choices for users to navigate into related topic regions.

6.3 Effectiveness of Map-Based Browsing

It is a challenging task to evaluate our method since by nature, browsing is interactive [48]. Following some previous works such as [24], we evaluate our system by simulations.



(a) After submit a query "car"

(b) After view a map node "used car"

Figure 1: Interface snapshots of our topic map-based browsing.

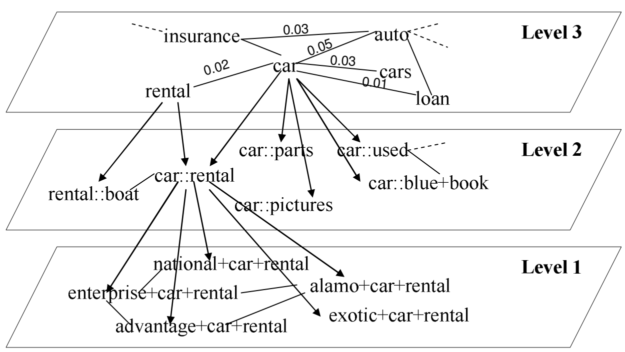


Figure 2: Examples of map nodes in the three-level topic map.

6.3.1 Experiment Design

We construct our test cases using the sessions in our held-out test logs. The test logs consist of user sessions and in each session, a user submitted several queries sequentially and clicked on certain documents for each of the submitted queries. A typical scenario is that a user first tried an initial query and clicked on certain documents. If the current results were poor or the user wanted to find more relevant information, the user would reformulate queries several times and click on more documents. In our experiments, we use each session as a test case and use the clicked documents in a session to *approximate* the relevant documents [22]. In a search session recorded in the search logs, the user tried to find additional relevant documents through repeatedly reformulating queries. Our idea is to do a simulation evaluation and see whether a user would have been able to find such additional relevant documents more effectively only through browsing the topic map.

Formally, let $\{Q_1, Q_2, ..., Q_k\}$ be a sequence of queries that a user tried in a session and let R be all the clicked documents for queries $\{Q_2, ..., Q_k\}$. R is regarded as the *additional* relevant documents to the user's information need. Note that we do not include the clicked documents of Q_1 in R because we want to simulate the scenario of a user browsing a map to find more relevant documents *after* finishing the search with the first query instead of formulating additional queries to find more relevant documents (the clicked documents of Q_1 would presumably have already been seen by the user by this point). Our experiments are designed to test how effective our guided navigation is in enabling a user to reach the documents in R *only* through browsing.

As expected, most sessions in our logs are not exploratory search. Since our goal is to support exploratory search, we use several heuristics to filter those sessions in our test cases. Each session is required to have at least 2 different queries and at least 10 clicked documents (including the clicks for Q_1). To ensure that queries in a session are about a coherent information need, we further require that two adjacent queries in a session should share at least one word. After applying the above heuristics to our test data, we obtain 76 sessions as our test cases. On average, each session has 2.22 queries and the size of R is 7.74.

To evaluate our methods, we conduct experiments to simulate a user's actions when the user uses our system. In particular, we simulate a one-step action where a user views one node in our map after submitting the very first query Q_1. We will compare the benefit of this navigation action in our system with a query reformulation action of submitting a second query Q_2.

We compare our methods with two baselines. Our first baseline method (BL1) is to use Q_1 to retrieve a ranked list from a search engine. Our second baseline (BL2) is to use Q_2 to retrieve documents from the same search engine. We use R to evaluate the accuracy of these two baselines. For our method, we use Q_1 as input to return a list of map nodes in Level 1 to a user. Then the user can first *examine* several nodes and finally decide to *view* a returned node. After the user views a node in our map, a ranked list of URLs of previously clicked documents in the node/cluster will be presented, as well as a list of organic search results from the search engine (refer to Section 5). For simplicity, we rank all the clicked URLs on the top of the search results and their rankings are decided by the historical click frequencies (see "Click-Through" in Figure 1). We then use R as relevant set to evaluate the returned URL lists after the user views a map node. To simulate which node a user will view in our map, we use 4 variants as follows.

Simu0Default: This variant is the most naive method which assumes that the user will view 1st ranked map node.

Simu0Best: This variant is to assume that the user will view the "best node" after examining the top 10 map nodes returned for Q_1. We will describe what is the best node soon.

Simu1Default: This variant is an extension of Simu0Default. In this variant, a user single-clicks on the 1st ranked node and our system will display a local view of the current node. The user then examines both the 1st ranked node and its top 10 horizontal neighbor nodes. The best node of these 11 nodes is finally "viewed" by the user.

Simu1Best: This variant is an extension of Simu0Best. In this variant, a user single-clicks on the node selected in Simu0Best and our system will display a local view of the current node. The user then examines both this clicked node and its top 10 horizontal neighbors. Finally the user decides to "view" the best node among all these 11 nodes.

Simu0Best, Simu1Default, and Simu1Best assume a user would optimally choose the best node to view, where the best node is the one whose ranked list of URLs (in its defined topic region) have the best P@10, evaluated based on R. These are optimal simulations which are to show the performance upper-bound of our system. However, given informative and accessible labels in our map, users can probably choose the best or nearly best node to view in reality. Simu1Default and Simu1Best are extensions of Simu0Default and Simu0Best, and are to test whether a user can get even more useful information after more exploration.

Treating R as the relevant documents, we use P@5 (Precision at 5 documents) and P@10 (Precision at 10 documents) to evaluate different methods. Note that P@5 and P@10 are very meaningful measures since users usually only look at and selectively click the top ranked results. Thus P@5 and P@10 measure the perceived ranking accuracy.

6.3.2 *Result Comparison and Analysis*

In Figure 3, we compare different methods using the two primary measures. We compare the two baseline methods BL1 and BL2 with four variants of our method. In this figure, we can see that BL1 is very poor as expected and it means that the first query is ineffective to retrieve additional documents. Simu0Default is a naive method which assumes the user would view the first node.

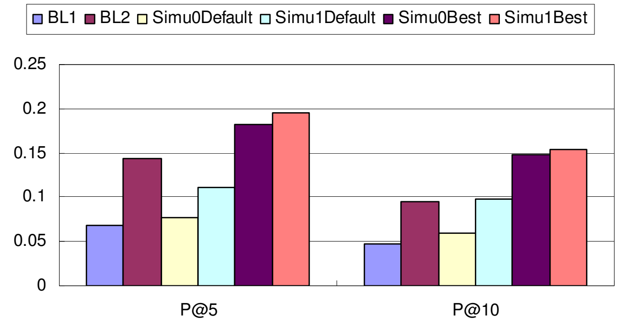


Figure 3: Comparison of different methods

	#Sessions	BL1	Simu0Best	Impr.
Part I (P@10 = 0)	50	0	0.114	0.114
Part II (P@10 > 0)	26	0.138	0.173	0.035

Table 1: Improvement over difficult queries with respect to average P@10. Part I corresponds to those more difficult queries.

Since the first node is the most similar to the current query Q_1, it is not surprising that its result is also poor. BL2 uses the second query Q_2 and the result is much better. Intuitively, this means that reformulating queries can get more clicked documents. Compared with BL2 based on P@10, our variant Simu0Best achieves a relative improvement of 57% and Simu1Best achieves a relative improvement of 63%. Both improvements are statistically significant according to Wilcoxon test: p-values are 0.003 and 0.002 respectively. This means that selectively viewing a node in our map can reach more relevant documents than reformulating a query. This benefit mostly comes from collaborative surfing since viewing a node would bring the user to a topic region with all the clicked documents by previous users when searching for similar information.

From this figure, we can also see that Simu1Default and Simu1Best achieve better accuracy than Simu0Default and Simu0Best respectively. The Wilcoxon tests show that the improvements are also significant: p-values are 0.01 and 0.02 respectively. This means that more relevant documents can be reached through navigating to and viewing a neighbor node. All these confirm the benefit of browsing with a topic map.

Difficult Query Analysis. We show the effectiveness of our method for difficult queries. In this experiment, we use BL1 to assess the difficulty of queries. For all the test cases, we separate them into two parts according to their P@10 in BL1. The first part (Part I) corresponds to the cases with P@10 = 0, which means Q_1 can not retrieve any additional documents to top 10. The second part (Part II) corresponds to the cases with P@10 $>$ 0. This means that we can retrieve at least 1 document using the original query Q_1. We compare the improvement of our Simu0Best over BL1 for these two sets of test cases using P@10. Table 1 summarizes the results. In this table, we can see that 50 test cases fall into Part I and 26 test cases fall into Part II. For Part I, we can improve P@10 by 0.114 from 0 to 0.114 on average. For Part II, the improvement is only 0.035 from 0.138 to 0.173 on average. Since the cases in Part I are more difficult than the cases in Part II, this means that navigation based on our topic map can help more for more difficult queries.

History Richness. Our topic map is based on search logs. Different test cases have different amount of similar history information in our logs. Our hypothesis is that a test case with richer history information in our logs will benefit more from our topic map. To

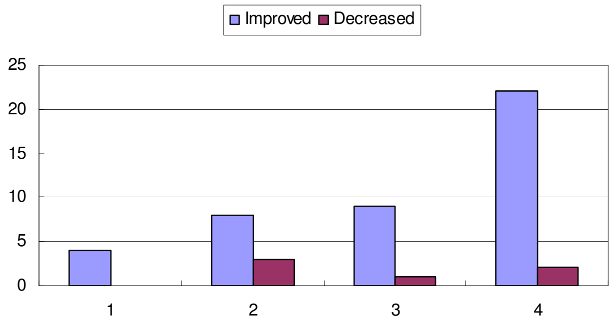


Figure 4: The impact of history richness.

verify this, we use Q_2 to retrieve our history collection and use the number of returned pseudo-documents as the indicator of the history richness for the test case. According to the number of returned pseudo-documents, we separate the test cases into 4 bins. Bin 1 has 0∼40 , Bin 2 has 40∼80, Bin 3 has 80∼120, and Bin 4 has more than 120 returned pseudo-documents. Bin 1 corresponds to those cases without much history while Bin 4 corresponds to those cases having rich history. For each bin, we show the number of test cases whose P@10's are improved versus decreased, by comparing Simu0Best with BL1. The result is shown in Figure 4. From this figure, we can see that the percentage of improved test cases increases along with the increase of the history richness. For example, in Bin 4, we improve 22 and decrease 2 cases. But in Bin 2, we increase 8 and decrease 3. This confirms that the more history we have, the better we can help users for browsing. More importantly, as time goes, more and more queries will have sufficient history, so we can improve more and more exploratory searches, resulting in a sustainable model for effective social surfing.

7. DISCUSSIONS

A main point made in this paper is that the current search engines would be more powerful, especially for supporting exploratory search and helping users find hard-to-find information, if they can offer better support for browsing through a multi-resolution topic map, such as the one constructed based on search logs. Although our topic map is built upon a relatively small sample of search logs, our experiment results have clearly demonstrated the feasibility of supporting browsing for ad-hoc queries in a general way. We also demonstrated that queries with more history information can benefit more from topic map-based browsing. This is very encouraging since there are more search logs in the commercial search engines that can be leveraged, and as a system is being used more, more search logs would be naturally accumulated.

Our work raises some interesting new possibilities in advancing the search engine technologies, which we will briefly discuss in this section.

7.1 Multi-Faceted Browsing for Ad Hoc Queries

Although we only experimented with a topic map built based on search logs in this paper, one can easily imagine that we can also build a topic map in many other ways based on various data sources. Indeed, even with search logs, we have multiple ways to build a topic map. For example, instead of using semantic similarity of queries to construct a topic map as we have done in our experiments, we can also leverage the co-occurrence relation of queries in a user session to build an alternative topic map where related queries to the same task may be connected together (e.g., queries about "flight ticket" may be connected with those about "car rental" even though the two sets of queries may not be similar by contents). Yet another way to construct a multi-resolution topic map is based on query word editing patterns [44]. In such a map, a node corresponds to a query. All queries with the same number of keywords belong to the same level. The children of a map node are obtained by adding a keyword into the current query and the neighbors of the query are obtained by substituting a keyword in the current query.

A topic map can also be constructed by other data sources. For example, any domain-specific ontology can also be extended to a topic map by adding horizontal relations. We can also build a topic map solely based on a plain document collection itself.

With multiple topic maps constructed using different criteria, a search engine system would be able to potentially support multi-faceted browsing for ad hoc topics, i.e., a user would be allowed to switch between from one facet (map) to another to explore information in an extremely flexible and powerful way.

7.2 Unify Querying and Browsing

Querying and browsing are the two most important information seeking strategies. They are complementary, and both are needed in a search task [51]. An important research question is thus: Can we integrate querying and browsing in a *unified formal framework*? Interestingly, as we will further discuss below, it is possible to view querying as a special way of navigating in the information space, thus we can integrate querying and browsing within a single unified navigation framework.

A Formal Navigation Framework. In Section 3, we have defined *topic region* and *topic region space*. Under these definitions, we can view querying as navigation in this space. More specifically, after querying, a user would end up viewing a subset of documents (i.e., search results), thus we can view this process as helping the user navigate into the topic region consisting of the search results. When a user repeatedly submits a query, the user would be essentially visiting different topic regions defined by the queries.

When a user follows a path on a topic map, the user would also be moving from one topic region to another, just like submitting reformulated queries. Thus both querying and browsing can be formalized as navigation operators defined below:

DEFINITION 5 (NAVIGATION OPERATOR). *A navigation operator is a function that maps one topic region to another. We use N as the set of all navigation operators. That is, $N = \{f : S \rightarrow S\}$, where S is topic region space.*

DEFINITION 6 (QUERY NAVIGATION OPERATOR). *A query navigation operator $Q(q)$ is defined as $Q(q)(T) = T_q$, where q is a query and T_q is the topic region corresponding to the search results of using the query q. For any $T_1 \neq T_2$, we have $Q(q)(T_1) = Q(q)(T_2)$. Therefore, such a definition assumes that a query navigation operator returns a topic region regardless the current region. It is thus a "memoryless" navigator and we can use $Q(q)$ to represent T_q without incurring confusion.*

DEFINITION 7 (BROWSING NAVIGATION OPERATOR). *A browsing navigation operator $B(v_1, v_2)$ is defined as $B(v_1, v_2)(v_1) = v_2$, where $(v_1, v_2) \in E$ is an edge on the topic map. $B(v_1, v_2)(v)$ is undefined if $v \neq v_1$. Intuitively, a browsing navigation operator $B(v_1, v_2)$ brings a user from topic region v_1 to v_2.*

DEFINITION 8 (COMPATIBILITY). *Two navigation operators N_i and N_j are compatible if and only if one of the following three conditions holds: (1) N_j is a query navigation operator; (2) $N_i = B(v_1, v_2)$ and $N_j = B(v_2, v_3)$; (3) $N_i = Q(q)$ and $N_j = B(Q(q), v)$.*

DEFINITION 9 (NAVIGATION TRACE). *A navigation trace is a sequence of navigation operators $N_1, N_2, ..., N_k$ such that N_i and N_{i+1} are compatible.*

With these definitions, we can describe any user's information seeking process as a navigation trace. For example, if the user submits a query q_1, navigates into a region T_1 from the search result region, navigates further from T_1 to T_2, and finally submits another query q_2, then the process can be formally described by the navigation trace $Q(q_1), B(Q(q_1), T_1), B(T_1, T_2), Q(q_2)$. The flexibility of combining multiple operators formally to describe an arbitrary information seeking process shows the expressiveness of our framework. Indeed, it provides a solid theoretical basis for studying many different ways to combine querying and browsing as well as developing systems to integrate querying and browsing.

Viewing existing search engines in our navigation framework, we see that they mostly only support query navigation operators. A main contribution of our work is to study how to effectively support browsing navigation operators by a good topic map.

Ranking in the navigation framework. While not explored in this paper, ranking is another important component in our framework. It is thus worth some discussion.

Ranking is important for three reasons. First, the ranking function is critical for supporting the query navigation operator as we generally define the target topic region of a query navigation operator as the top-ranked documents using the query. Second, even when a user reaches a region through browsing, it is still desirable to rank the documents in the region. As the user navigates from document to document within a region, the order of unseen documents can also be dynamically ordered as in the case of implicit feedback [39]. Third, when a user is landing on a region that is not exactly a region on our map, we will need to leverage the ranking function to find the closest regions on the map. A similar need also arises when the user takes a zoom-in or zoom-out action to change the resolution of the map, in which case a user may end up having multiple regions to choose.

While ranking of documents has been the central research topic in information retrieval and Web search, the navigation framework raises some new interesting research questions related to ranking: (1) Ranking documents within a topic region. In our framework, a user would leave a richer interaction history which would include not only queries, click-throughs, but also browsing actions such as zoom in/out operations and neighborhood explorations. Existing work in personalized search and implicit feedback has already shown the usefulness of the existing query-based history information [39]. It would be very interesting to study how we can incorporate all the navigation information to further improve a ranking function and personalize search results. (2) Ranking topic regions. While traditionally, ranking is mainly to order documents, in the navigation framework, we also need to rank the topic regions of a map. How to generalize the current document ranking functions or design new ranking functions to perform region ranking is another very interesting research question. Some recent work on blog feeds has shown the promise of this research direction [15].

As a first step in studying the navigation framework, in this paper, we simply reused the ranking function provided by an existing search engine, leaving all these questions for future work.

8. CONCLUSIONS

In this paper, we study how to support flexible browsing for exploratory search. We define a novel multi-resolution topic map to extend a hierarchy to support more flexible browsing. We propose a novel way of exploiting and organizing search logs to enable users to follow information footprints left by other users in the process of information seeking, which can potentially lead to an interesting sustainable model for social surfing. We also propose a general computational method based on the star-clustering algorithm to generate a multi-resolution topic map based on search logs. Experimental results using a sample of search logs from a commercial search engine show that browsing through such a search-log-based topic map is effective for supporting exploratory search.

Our work opens up many interesting new research directions as we have already discussed in Section 7. The preliminary simulation based experimental results are very encouraging. However, such simulation based approaches are simplified and do not take users' cognitive overheads into account. It would be especially interesting to use a much larger data set of search logs to build a larger-scale topic map and evaluate its effectiveness with a system by real user traffic and user feedback. It is also very interesting to study how to learn effectively from the rich interaction traces that a user leaves when interacting with a system that supports browsing with a topic map. For example, users' browsing logs on the topic maps can be further utilized to adaptively rank topic regions for a future user. Clearly more effective implicit feedback techniques can be developed by leveraging such interaction traces. Finally, topic maps can be constructed in multiple ways. How to design effective evaluation methodology to compare different topic map construction methods is an important research question.

9. ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable suggestions and Qihong Shao for proofreading. This work is in part supported by the National Science Foundation under award numbers IIS-0347933 and IIS-0713581, the Microsoft "Beyond Search" Grant, a Sloan Research Fellowship, and a Yahoo! Ph.D. Fellowship.

10. REFERENCES

[1] E. Agichtein, E. Brill, and S. T. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR*, pages 19–26, 2006.

[2] J. A. Aslam, E. Pelekov, and D. Rus. The star clustering algorithm for static and dynamic information organization. *Journal of Graph Algorithms and Applicatins*, 8(1):95–129, 2004.

[3] M. J. Bates. The design of browsing and berrypicking techniques for the online search interface. *Online Review*, 13:407–424, 1989.

[4] D. Beeferman and A. L. Berger. Agglomerative clustering of a search engine query log. In *KDD*, pages 407–416, 2000.

[5] N. J. Belkin. Interaction with texts: Information retrieval as information-seeking behavior. In *Information Retrieval*, pages 55–66, 1993.

[6] M. Bilenko and R. W. White. Mining the search trails of surfing crowds: identifying relevant websites from user activity. In *WWW*, pages 51–60, 2008.

[7] P. Brusilovsky and R. Rizzo. Map-based horizontal navigation in educational hypertext. In *HYPERTEXT*, pages 1–10, 2002.

[8] H. Chen and S. T. Dumais. Bringing order to the web: automatically categorizing search results. In *CHI*, pages 145–152, 2000.

[9] H. Chen, A. L. Houston, R. R. Sewell, and B. R. Schatz. Internet browsing and searching: user evaluations of category map and concept space techniques. *JASIS*, 49(7):582–603, 1998.

[10] E. Cutrell, D. Robbins, S. Dumais, and R. Sarin. Fast, flexible filtering with phlat. In *CHI*, pages 261–270, 2006.
[11] W. Dakka and P. Ipeirotis. Automatic extraction of useful facet terms from text documents. In *ICDE*, 2008.
[12] D. Downey, S. Dumais, D. Liebling, and E. Horvitz. Understanding the relationship between searchers' queries and information goals. In *CIKM*, pages 449–458, 2008.
[13] S. T. Dumais, E. Cutrell, and H. Chen. Optimizing search by showing results in context. In *CHI*, pages 277–284, 2001.
[14] E. Duval and H. Olivié. Towards the integration of a query mechanism and navigation for retrieval of data on multimedia documents. *SIGIR Forum*, 26(2):8–25, 1992.
[15] J. L. Elsas, J. Arguello, J. Callan, and J. G. Carbonell. Retrieval and feedback models for blog feed search. In *SIGIR*, pages 347–354, 2008.
[16] J. English, M. A. Hearst, R. R. Sinha, K. Swearingen, and K.-P. Yee. Hierarchical faceted metadata in site search interfaces. In *CHI Extended Abstracts*, pages 628–639, 2002.
[17] J. Freyne, R. Farzan, P. Brusilovsky, B. Smyth, and M. Coyle. Collecting community wisdom: integrating social search & social navigation. In *IUI*, pages 52–61, 2007.
[18] G. W. Furnas. Effective view navigation. In *CHI*, pages 367–374, 1997.
[19] J. Han and M. Kamber. *Data Mining: Concepts and Techniques, 2nd Ed.* Morgan Kaufmann, 2006.
[20] M. Hearst. Clustering versus faceted categories for information exploration. *CACM*, 49(4):59–61, 2006.
[21] M. Hearst and J. Pedersen. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *SIGIR*, pages 76–84, 1996.
[22] T. Joachims. *Evaluating Retrieval Performance Using Clickthrough Data.*, pages 79–96. Physica/Springer Verlag, 2003. in J. Franke and G. Nakhaeizadeh and I. Renz, "Text Mining".
[23] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *WWW*, pages 387–396, 2006.
[24] J. Lin and M. D. Smucker. How do users find things with pubmed?: towards automatic utility evaluation with user simulations. In *SIGIR*, pages 19–26, 2008.
[25] m. c. schraefel, D. A. Smith, A. Owens, A. Russell, C. Harris, and M. Wilson. The evolving mspace platform: leveraging the semantic web on the trail of the memex. In *HYPERTEXT*, pages 174–183, 2005.
[26] Y. S. Maarek. Organizing documents to support browsing in digital libraries. *SIGOIS Bull.*, 16(2):36–37, 1995.
[27] J. D. Mackinlay and P. T. Zellweger. Browsing vs. search: can we find a synergy? (panel session). In *CHI*, pages 179–180, 1995.
[28] G. Marchionini. Exploratory search: from finding to understanding. *CACM*, 49(4):41–46, 2006.
[29] V. L. O'Day and R. Jeffries. Orienteering in an information landscape: how information seekers get from here to there. In *INTERCHI*, pages 438–445, 1993.
[30] C. Olston and E. H. Chi. Scenttrails: Integrating browsing and searching on the web. *ACM Trans. Comput.-Hum. Interact.*, 10(3):177–197, 2003.
[31] S. Pandit and C. Olston. Navigation-aided retrieval. In *WWW*, pages 391–400, 2007.
[32] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *KDD*, pages 239–248, 2005.
[33] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR*, pages 232–241, 1994.
[34] M. Sahami and T. D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *WWW*, pages 377–386, 2006.
[35] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, 1975.
[36] M. Sanderson and B. Croft. Deriving concept hierarchies from text. In *SIGIR*, pages 206–213, New York, NY, USA, 1999. ACM.
[37] D. Shen, M. Qin, W. Chen, Q. Yang, and Z. Chen. Mining web query hierarchies from clickthrough data. In *AAAI*, pages 341–346, 2007.
[38] R. Shen, N. S. Vemuri, W. Fan, R. da S. Torres, and E. A. Fox. Exploring digital libraries: integrating browsing, searching, and visualization. In *JCDL*, pages 1–10, 2006.
[39] X. Shen, B. Tan, and C. Zhai. Context-sensitive information retrieval using implicit feedback. In *SIGIR*, pages 43–50, 2005.
[40] B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, and O. Boydell. Exploiting query repetition and regularity in an adaptive community-based web search engine. *User Modeling and User-Adapted Interaction*, 14(5):383–423, 2005.
[41] J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger. The perfect search engine is not enough: a study of orienteering behavior in directed search. In *CHI*, pages 415–422, 2004.
[42] X. Wang and C. Zhai. Learn from web search logs to organize search results. In *SIGIR*, pages 87–94, 2007.
[43] X. Wang and C. Zhai. Mining term association patterns from search logs for effective query reformulation. In *CIKM*, pages 479–488, 2008.
[44] X. Wang and C. Zhai. Massive implicit feedback: Organizing search logs into topic maps for collaborative surfing. In *ACM SIGIR workshop on Understanding the Users (Demo Description)*, 2009.
[45] J.-R. Wen, J.-Y. Nie, and H. Zhang. Clustering user queries of a search engine. In *WWW*, pages 162–168, 2001.
[46] A. Wexelblat and P. Maes. Footprints: history-rich tools for information foraging. In *CHI*, pages 270–277, 1999.
[47] R. W. White, M. Bilenko, and S. Cucerzan. Studying the use of popular destinations to enhance web search interaction. In *SIGIR*, pages 159–166, 2007.
[48] R. W. White and R. A. Roth. *Exploratory Search: Beyond the Query-Response Paradigm*. Morgan and Claypool, 2009.
[49] P. Willett. Recent trends in hierarchic document clustering: a critical review. *Inf. Process. Manage.*, 24(5):577–597, 1988.
[50] K.-P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *CHI*, pages 401–408, 2003.
[51] X. Yuan and N. J. Belkin. Supporting multiple information-seeking strategies in a single system framework. In *SIGIR*, pages 247–254, 2007.
[52] O. Zamir and O. Etzioni. Web document clustering: A feasibility demonstration. In *SIGIR*, pages 46–54, 1998.
[53] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma. Learning to cluster web search results. In *SIGIR*, pages 210–217, 2004.