User Priority Based Search for Organizing and Grouping

Vallamalla Pranitha, Mrs. M. Jhansi Lakshmi
P.G. Scholar, Assoc. Prof.
Computer Science and Engineering, Computer Science and Engineering
Global Institute of Engineering and Technology, Hyderabad

Abstract **–** Most users want their search engine to incorporate three key features in query results. This paper addresses on the design of search history displays to support information seeking (IS). we try to give improved results on the search mechanism, where Information needs to be tracked in the perspective making the user flexibility to make the complex search to the extent of making the format of user-friendly, To better support users in their long-term information quests on the Web, search engines keep track of their queries and clicks while searching online. In this paper, we study the problem of organizing a user's historical queries into groups in a dynamic and automated fashion. Automatically identifying query groups is helpful for a number of different search engine components and applications, such as query suggestions, result ranking, query alterations, sessionization, and collaborative search.

Keywords **-** user history, search history, query clustering, Search engine, user profiling, task identification

1. INTRODUCTION

With the increasing number of published electronic materials, the World Wide Web (WWW) has become a vast resource for individuals to acquire knowledge, solve problems, and complete tasks that use Web information. As the size and richness of information on the Web grows, so does the variety and the complexity of tasks that users try to accomplish online. Users are no longer content with issuing simple navigational queries. We use our memory to bridge across different information sources and activities but human memory is limited and selective. Searchers create external memory aids to help keep track of progress, plan steps, and collect information. users are usually reluctant to explicitly provide their preferences due to the extra manual effort involved, recent research has focused on the automatic learning of user preferences from users' search histories or browsed documents and the development of personalized systems based on the learned user preferences.

One of information-seeking[15] tasks often performed by students is Information Gathering, which is the extracting, evaluating, and organizing relevant information for a given topic. One important step towards enabling services and features that can help users during their complex search quests online is the capability to identify and group related queries together. Recently, some of the major search engines have introduced a new "Search History" feature, which allows users to track their online searches by recording their queries and clicks. The user's past (implicit) indication of document relevance we can predict his/her reaction to the current retrieved documents. For example, if the user searched with the same query "python" before and clicked on Python language website's link, we have high confidence that the user would do it again this time, and it makes good sense to list that webpage in the top. Even when there is no exact occurrence of the current query in history, we may still find similar queries like "python doc" helpful (e.g., discovering that the user prefers results from the www.python.org site). Recommendations for search history displays and two search history based user interface tools are described here, which take advantage of automatically recorded information.

In fact, identifying groups of related queries has applications beyond helping the users to make sense and keep track of queries and clicks in their search history[5]. First and foremost, query grouping allows the search engine to better understand a user's session and potentially tailor that user's search experience according to her needs. Once query groups have been identified, search engines can have a good representation of the search context behind the current query using queries and clicks in the corresponding query group. This will help to improve the quality of key components of search engines such as query suggestions, result ranking, query alterations, sessionization, and collaborative search. For example, if a search engine knows that a current query "financial statement" belongs to a {"bank of america", "financial statement"} query group, it can boost the rank of the page that provides information about how to get a Bank of America statement instead of the Wikipedia article on "financial statement", or the pages related to financial statements from other banks.

Each query group is a collection of queries by the same user that are relevant to each other around a common informational need. These query groups are dynamically updated as the user issues new queries, and new query groups may be created

over time. Existing click through-based user profiling strategies can be categorized into document-based and concept base approaches. They both assume that user clicks can be used to infer users' interests, although their inference methods and the outcomes of the inference are different. Document-based profiling methods try to estimate users' document preferences (i.e., users are interested in some documents more than others).

Search history can be divided into short-term and long- term types. Short-term search history is limited to a single search session, which contains a (normally consecutive) sequence of searches with a coherent information need and usually spans a short period of time. Often, a user composes an initial query, views the returned documents, and if unsatisfied, modifies the query and repeats the search process. All these activities, which form the short- term search history, shed light on the current information need and make useful search context. Long-term search history[17] is, in contrast, unlimited in time scope and may include all search activities in the past. Compared with short-term search history, it has several advantages. There is no need to detect session boundaries (determining whether a previous search shares the same information need as the current one), which is often a difficult task.

Organizing the query groups within a user's history is challenging for a number of reasons. First, related queries may not appear close to one another, as a search task may span days or even weeks. This is further complicated by the interleaving of queries and clicks from different search tasks due to users' multitasking [3], opening multiple browser tabs, and frequently changing search topics. We then evaluate the methods on a test set of Web search histories collected from some real users. We also find that although recent history tends to be much more useful than remote history (especially for fresh queries), all of the entire history is helpful for improving the search accuracy of recurring queries.

The rest of the paper is organized as follows: Section 2 discusses the related works. We classify the existing user profiling strategies into two categories and review methods among the categories. In Section 3, we review our personalized concept-based clustering strategy to exploit the relationship among ambiguous queries according to the user conceptual preferences recorded in the concept-based user profiles. In Section 4, we present the proposed concept-based user profiling strategies. Experimental results comparing our user profiling strategies are presented in Section 5. Section 6 concludes the paper.

2. RELATED WORKS

A user needs assessment is the first step in designing usable interfaces. The task of users in this research is information seeking. Our goal is to automatically organize a user's search history into query groups, each containing one or more related queries and their corresponding clicks. Each query group corresponds to an atomic information need that may require a small number of queries and clicks related to the same search goal. For example, in the case of navigational queries, a query group may involve as few as one query and one click. They highlight the importance of external problem representation, and planning, and evaluation in problem solving, which can be supported by search histories. History displays have to incorporate both analytical searches and hypertext browsing in full-text systems. Explicit representation of searchers' path through a hypertext system can alleviate disorientation.Users' document preferences are first extracted from the click through data, and then, used to learn the user behavior model which is usually represented as a set of weighted features. On the other hand, concept-based user profiling methods aim at capturing users' conceptual needs. Users' browsed documents and search histories are automatically mapped into a set of topical categories. User profiles are created based on the users' preferences on the extracted topical categories.

Information Gathering [15] is a knowledge construction process. Web learners begin this process with recognizing an anomalous state of knowledge related to a topic (Cole, Leide, Behesht, Large, & Brooks, 2005). This state is the interest or concern mental state that triggers the information gathering process. Thus, they make an initial search plan based on their prior knowledge. With each piece of new and useful information encountered giving them new ideas on their topic, they thus extend or evolve their plan to other relevant topics/subtopics (Lin & Belkin, 2005) or associate the piece of information with their knowledge structure. Finally, the process is ended up with resolving the anomalous state.

Information Gathering is a very complex information- seeking task. It can be completed not by a specific answer but by a series of extractions, comparisons, and syntheses of a broad range of information related to these topics/subtopics (Morrison, Pirolli, & Card, 2001; Sellen, Murphy, & Shaw, 2002). Learners are frequently required to maintain many extracted results for later use and reference. However, to keep a huge amount of information in a human's mind is difficult because the limitation of working memory (Anderson, 2004). To support the limitation of memory capacity, learners have to employ external memory aids.

Even the earliest information retrieval systems provided some kind of history mechanism. These usually involved the display of "query–result set" pairs. As an example, Back (1976) integrated search

review features in his TIRES system, a management information retrieval system, based on the findings of four previous studies and systems. Many early commercial systems had a history feature that allowed users to recall past search commands and reuse them. The importance of search histories in user interfaces has remained clear in the decades that passed. Hearst (1999) discussed information-seeking behaviors and strategies in her chapter on information retrieval user interfaces and visualizations [7]. She highlighted the need for search system user interfaces to show what steps had been taken in the past and what short- and long-term strategies had been followed. She also called for annotation tools for users to comment on the actions and information found. She concluded that user observations suggest the need for search histories in the user interface of information retrieval and visualization systems, and she pointed out that these functions are not well supported in current systems. Although the need for search histories in search interfaces is clear, not many innovative solutions are available to present and manipulate them. One exception is the Ariadne tool developed by Twidale and Nichols (1998). The Ariadne system was proposed to support collaboration among users by visualizing search session histories. The system captures "query–result set pairs and displays them to the user as thumbnails of screen shots. Searchers can annotate and share these graphical histories with others. This article reports on the results of a thorough examination of the use of interaction histories in one specific application domain area, legal information seeking, and propose search history tools for user support. The problem is related to coordination of information. To coordinate information kept in the three kinds of memory aids, students have to frequently change attention among them. The frequently changed on attention make students easily disoriented. In addition, the structures of information organized in the three memory aids are inconsistent. For example, students organize bookmarks in a hierarchical structure but keep open Web pages in a sequential order. To find and recall a piece of information that is previously kept in these memory aids becomes difficult.

A query group is an ordered list of queries, qi, together with the corresponding set of clicked URLs, clki of qi. A query group is denoted as s = h{q1, clk1}, . . . , {qk, clkk}i. The specific formulation of our problem is as follows:

Given: a set of existing query groups of a user, S = {s1, s2, . . . , sn}, and her current query and clicks, {qc, clkc}, Find: the query group for {qc, clkc}, which is either one of the existing query groups in S that is most related to, or a new query group sc = {qc, clkc} if there does not exist a query group in S that is not sufficiently related to {qc, clkc}. Below, we will motivate the dynamic nature of this formulation, and give an overview of the solution. The core of the solution is a measure of relevance between two queries (or query groups). We will further motivate the need to go beyond baseline relevance measures that rely on time or text, and instead propose a relevance measure based on signals from search logs.

One approach to the identification of query groups is to first treat every query in a user's history as a singleton query group, and then merge these singleton query groups in an iterative fashion (in a k-means or agglomerative way [5]).

However, this is impractical in our scenario for two reasons. First, it may have the undesirable effect of changing a user's existing query groups, potentially undoing the user's own manual efforts in organizing her history. Second, it involves a high computational cost, since we would have to repeat a large number of query group similarity computations for every new query.

3.QUERY RELEVANCE USING SEARCH LOGS

We now develop the machinery to define the query relevance based on Web search logs [2]. Our measure of relevance is aimed at capturing two important properties of relevant queries, namely: (1) queries that frequently appear together as reformulations and (2) queries that have induced the users to click on similar sets of pages. We start our discussion by introducing three search behavior graphs that capture the aforementioned properties. Following that, we show how we can use these graphs to compute query relevance and how we can incorporate the clicks following a user's query in order to enhance our relevance metric.

One way to identify relevant queries is to consider query reformulations that are typically found within the query logs of a search engine. If two queries that are issued consecutively by many users occur frequently enough, they are likely to be reformulations of each other. To measure the relevance between two queries issued by a user, the time-based metric, sometime, makes use of the interval between the timestamps of the queries within the user's search history. In contrast, our approach is defined by the statistical frequency with which two queries appear next to each other in the entire query log, over all of the users of the system.

A different way to capture relevant queries from the search logs is to consider queries that are likely to induce users to click frequently on the same set of URLs. For example, although the queries "ipod" and "apple store" do not share any text or appear temporally close in a user's search history, they are relevant because they are likely to have resulted in clicks about the ipod product. In

order to capture such property of relevant queries, we construct a graph called the query click graph, QCG. The query reformulation graph, QRG, and the query click graph, QCG, capture two important properties of relevant queries respectively. In order to make more effective use of both properties, we combine the query reformulation information within QRG and the query click information within QCG into a single graph, QFG = (VQ, EQF), that we refer to as the query fusion graph. At a high level, EQF contains the set of edges that exist in either EQR or EQC. The weight of edge (qi, qj) in QFG, wf (qi, qj), is taken to be a linear sum of the edge's weights, wr (qi, qj) in EQR and wc(qi, qj) in EQC,as follows:

wf (qi, qj) = _ × wr(qi, qj) + (1 − α) × wc (qi, qj) Algorithm [4] for calculating the query relevance by simulating random walks over the query fusion graph.

Relevance(q)
Input:
1) the query fusion graph, QFG
2) the jump vector, g
3) the damping factor, d
4) the total number of random walks, numRWs
5) the size of neighborhood, maxHops
6) the given query, q

Out
put:

the fusion relevance vector for q, relF q

(0) Initialize relF q = 0
(1) numWalks = 0; numVisits = 0 (2) while numWalks < numRWs
(3) numHops = 0; v = q
(4) while v 6= NULL ^ numHops < maxHops
(5) numHops++
(6) relF q (v)++; numVisits++
(7) v = SelectNextNodeToVisit (v) (8) numWalks++
(9) For each v, normalize relF q (v) = relF , q (v)/numVisits

we use the jump vector gq to pick the random walk starting
point. At each node v, for a given damping factor d, the random walk either continues by following one of the outgoing edges of v with a probability of d, or stops and re-starts at one of the starting points in gq with a probability of (1−d). Then, each outgoing edge, (v, qi), is selected with probability wf (v, qi), and the random walk always re-starts if v has no outgoing edge. The selection of the next node to visit based on the outgoing edges of the current node v in QFG and the damping factor d is performed by the SelectNextNodeToVisit process in Step (7) of the algorithm. In addition to query reformulations, user activities also include clicks on the URLs following each query submission.

The clicks of a user may further help us infer her search interests behind a query q and thus identify queries and query groups relevant to q more effectively. We give a motivating example that illustrates why it may be helpful to take into account clicked URLs of q to compute the query relevance. Let us consider that a user submitted a query "jaguar". If we compute the relevance scores of each query in VQ with respect to the given query only, both the queries related to the car "jaguar" and those related to the animal "jaguar" get high fusion relevance scores. This happens because we do not know the actual search interest of the current user when she issues the query "jaguar". However, if we know the URLs clicked by the current user following the query "jaguar" (e.g. the Wikipedia article on animal "jaguar"), we can infer the search interest behind the current query and assign query relevance scores to queries in VQ accordingly. In this way, by making use of the clicks, we can give much higher query relevance scores to queries related to "animal jaguar" than those related to "car jaguar".

4. QUERY GROUPING USING THE QFG

In this section, we outline our proposed similarity function simrel to be used in the online query grouping process outline. For each query, we maintain a query image, which represents the relevance of other queries to this query. For each query group, we maintain a context vector, which aggregates the images of its member queries to form an overall representation. We then propose a similarity function simrel for two query groups based on these concepts of context vectors and query images. Note that our proposed definitions of query reformulation graph, query images, and context vectors are crucial ingredients, which lend significant novelty to the Markov chain process for determining relevance between queries and query groups[4].

Context Vector. For each query group, we maintain a context vector which is used to compute the similarity between the query group and the user's latest singleton query group. The context vector for a query group s, denoted cxts, contains the relevance scores of each query in VQ to the query group s, and is obtained by aggregating the fusion relevance vectors of the queries and clicks in s. If s is a singleton query group containing only {qs1 , clks1}, it is defined as the fusion relevance vector rel(qs1,clks1). For a query group s = h{qs1 , clks1}, . . . , {qsk , clksk}i with k > 1, there are a number of different ways to define cxts. For instance, we can define it as the fusion relevance vector of the most recently added query and clicks, rel(qsk ,clksk). Other possibilities include the average or the weighted sum of all the fusion relevance vectors of the queries and clicks in the query group.

Query Image**,** The fusion relevance vector of a given query q, relq, captures the degree of relevance of each query q0 2 VQ to q. However, we observed that it is not effective or robust to use relq itself as a relevance measure for our online query grouping. We may use the relevance value in the fusion relevance vectors, rel"fs00 ("boa00) or rel"boa00 ("fs00). Usually, however, it is a very tiny number that does not comprehensively express the relevance of the search tasks of the queries, thus is not an adequate relevance measure for an effective and robust online query grouping. Instead, we want to capture the fact that both queries highly pertain to financials.

Online Query Grouping. The similarity metric that we described in Definition 4.1 operates on the images of a query and a query group. Some applications such as query suggestion may be facilitated by fast on-the fly grouping of user queries. For such applications, we can avoid performing the random walk computation of fusion relevance vector for every new query in real-time, and instead pre-compute and cache these vectors for some queries in our graph. This works especially well for the popular queries. In this case, we are essentially trading off disk storage for run-time performance. This additional storage space is insignificant relative to the overall storage requirement of a search engine. Meanwhile, retrieval of fusion relevance vectors from the cache can be done in milliseconds. Hence, for the remainder of this paper, we will focus on evaluating the effectiveness of the proposed algorithms in capturing query relevance.

5. EXPERIMENTS

we study the behavior and performance of our algorithms on partitioning a user's query history into one or more groups of related queries. For example, for the sequence of queries "caribbean cruise";"bank of america"; "expedia"; "financial statement", we would expect two output partitions: first, {"Caribbean cruise", "expedia"} pertaining to travel-related queries, and, second, {"bank of america", "financial statement"} pertaining to money- related queries.

The empirical findings on the role of search histories formed the basis for designing search history interfaces. Providing a continuously growing history record in the user interface is the most common use of search histories. Interface design recommendations for displaying search history data are presented to feed the recorded information back to the user. Initial user interface prototypes are included and described to illustrate some of the design recommendations. In addition to direct search history displays, tools building on search history data can help searchers in search-related tasks.

Search-history-based user interface functions are described organized around a scratchpad and a results collection tool. our query grouping algorithm relies heavily on the use of search logs in two ways: first, to construct the query fusion graph used in computing query relevance, and, second, to expand the set of queries considered when computing query relevance. We start our experimental evaluation, by investigating how we can make the most out of the search logs.

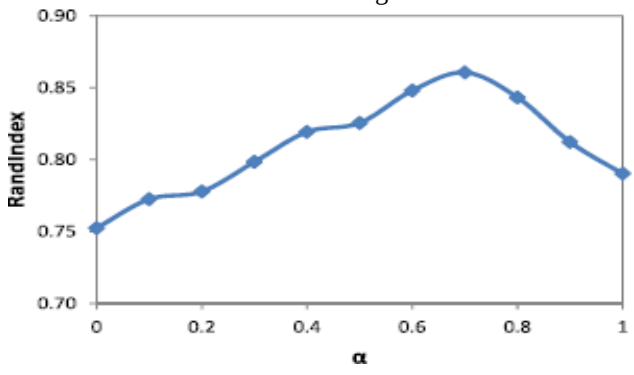
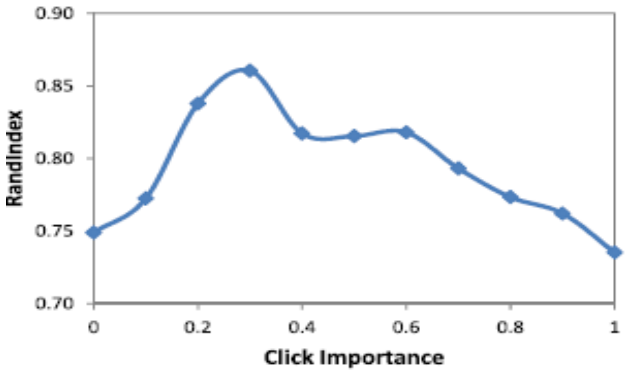


Fig.1 Varying mix of query and click graphs



we evaluated our algorithm over the graphs that we constructed for increasing values of α. The result is shown in Figure 1. To this end, we evaluated the performance of our algorithm for increasing values of click importance ws and we show the result in Figure 2.

6. PERFORMANCE COMPARISION

We now compare the performance of our proposed methods against five different baselines. For these baselines, we use the same SelectBestQueryGroup as in Figure 3 with varying relevance metrics. As the first baseline, we use a time-based method (henceforth referred to as *Time*) that groups queries based on whether the time difference between a query and the most recent previous query is above a threshold. It is essentially the same as the *Time* metric introduced in Section, except that instead of measuring similarity as the inverse of the time interval, we measure the distance in terms of the time interval (in seconds). In particular, since our QFG method relies on the accurate estimation of a query image within the query fusion graph, it is expected to perform better when the estimation was based on more information and is therefore more accurate. On the other hand, if there are queries that are rare in the search logs or do not have many outgoing edges in our graph to facilitate the random walk, the graph-based techniques may perform worse due to the lack of edges.

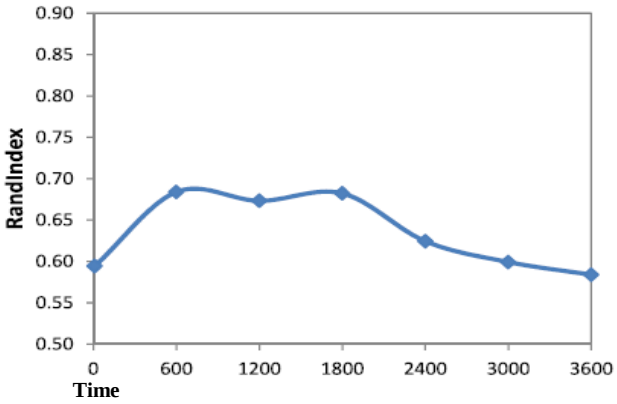


Fig.3 Varying the time

Fig.2 Varying the click importance wclick

7. CONCLUSIONS

The query reformulation and click graphs contain useful information on user behavior when searching online. we systematically explored how to exploit long term search history, which consists of past queries, result documents and click through, as useful search context that can improve retrieval performance. In this paper, we show how such information can be used effectively for the task of

organizing user search histories into query groups. We also want to conduct a more in-depth testing that is performed with a wide range of material, task, and target groups. we would like to combine the user profiles with the document selection process, not just the document re-ranking, to provide a wider set of relevant results to the user rather than just reorganizing the existing results. As future work, we intend to investigate the usefulness of the knowledge gained from these query groups in various applications such as providing query suggestions and biasing the ranking of search results.

REFERENCES

[1] J. Han and M. Kamber, "Data Mining: Concepts and Techniques",Morgan Kaufmann, 2000.

[2] A. Broder, "A taxonomy of web search," SIGIR Forum, 2002.

[3] A. Spink, M. Park, B. J. Jansen, and J. Pedersen, "Multitasking during Web search sessions", Information Processing and Management, 2006.

[4] Heasoo Hwang, Hady W. Lauw, Lise Getoor and Alexandros Ntoulas," Organizing User Search Histories",in conf. IEEE Transactions on Knowledge and Data Engineering,2012.

[5] D. Beeferman and A. Berger, "Agglomerative clustering of a search engine query log," in KDD,
2000.

[6] J. Teevan, E. Adar, R. Jones, and M. A. S. Potts, "Information reretrieval: repeat queries in yahoo's logs," in SIGIR. New York, 2007.

[7] Komlodi, A," Search history for user support in information seeking interfaces, University of Maryland"; College Park -2002.

[8] T. Joachims," Optimizing search engines using click through data", In Proceedings of SIGKDD
2002, [9] D. Kelly and J. Teevan," Implicit feedback for inferring user preference: A Bibliography", SIGIR Forum, 2003.

[10] M. Speretta," Personalizing Search Based on User Search Histories", Master's thesis, The University of Kansas, 2004.

[11]E. Agichtein, E. Brill, and S. Dumais, "Improving Web Search Ranking by Incorporating User Behavior Information," Proc. ACMSIGIR, 2006.

[12] E. Agichtein, E. Brill, S. Dumais, and R. Ragno, "Learning User Interaction Models for Predicting Web Search Result Preferences," Proc. ACM SIGIR,
2006.

[13]Aula, A., Jhaveri, N., & Käki, M," Information search and re-access strategies of experienced Web users.", Proceedings of the 14th international conference on World Wide Web, New York:2005.

[14] Lee, Y. J.," Concept mapping your Web searches: a design rationale and Web-enabled application", Journal of Computer Assisted Learning, 2004

[15] Spink, A., Wilson, T. D., Ford, N., Foster, A., & Ellis, D. ," Information seeking and mediated searching study. part 3.successive searching. Journal of the American Society for Information Science and Technology", 2002.

[16] R. Jones and K. L. Klinkner, "Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs", in CIKM, 2008.

[17] Bin Tan, Xuehua Shen, ChengXiang Zhai," Mining Long-Term Search History to Improve Search Accuracy", in KDD,2006.

[18] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna, "The query-flow graph: Model and applications," in *CIKM*, 2008.