Exploiting Navigational Queries for Result Presentation and Caching in Web Search Engines

Rifat Ozcan, Ismail Sengor Altingovde, and Özgür Ulusoy
Computer Engineering, Bilkent University, 06800, Ankara, Turkey.
E-mail: {rozcan, ismaila, oulusoy}@cs.bilkent.edu.tr

Caching of query results is an important mechanism for efficiency and scalability of web search engines. Query results are cached and presented in terms of pages, which typically include 10 results each. In navigational queries, users seek a particular website, which would be typically listed at the top ranks (maybe, first or second) by the search engine, if found. For this type of query, caching and presenting results in the 10-per-page manner may waste cache space and network bandwidth. In this article, we propose nonuniform result page models with varying numbers of results for navigational queries. The experimental results show that our approach reduces the cache miss count by up to 9.17% (because of better utilization of cache space). Furthermore, bandwidth usage, which is measured in terms of number of snippets sent, is also reduced by 71% for navigational queries. This means a considerable reduction in the number of transmitted network packets, i.e., a crucial gain especially for mobile-search scenarios. A user study reveals that users easily adapt to the proposed result page model and that the efficiency gains observed in the experiments can be carried over to real-life situations.

Introduction

Web search engines answer millions of queries per day to satisfy the information needs of their users. In the literature (Broder, 2002; Rose & Levinson, 2004), two essential goals for web searches are identified as "informational" or "navigational." Additionally, some other classifications of user search intents including transactional (Broder, 2002) and resource queries (Rose & Levinson, 2004) are also proposed. In the context of this study, we focus on the navigational queries and broadly refer to all queries that are not navigational as informational. To illustrate the difference between these categories, consider the following two queries as examples of informational and navigational types, respectively: "caching in web search engines" and "united airlines." The users asking the first query are trying to find information on caching in web search engines and they do not have a specific website in mind. Therefore, there may browse several result pages,[1] read the snippets, and decide to visit many different web pages. On the other hand, the users asking the second query is most probably looking for United Airlines' company website. In this case, if the search engine finds the target page, then it can be expected to be among the top-ranked results, because Liu, Zhang, Ru, and Ma (2006) have noted that information retrieval systems have much better effectiveness for navigational queries than informational queries. Thus, the user would typically glance over only a few of the top-ranked URLs and not attempt to visit alternative pages, other than the target page in his or her mind. The difference between user intention and behavior is clear: In the navigational case, users are only interested in accessing the particular webpage that they know or expect to exist (e.g., www.united.com for this example), and not any other results related to the topic of the query (e.g., financial news about United Airlines, etc.).

Web search engines, regardless of the user's search goal, present and cache the results in terms of pages, each of which includes a fixed number (typically 10) of results. By result page, we mean the URLs and snippets of the documents in the query result (Fagni, Perego, Silvestri, & Orlando, 2006) that is necessary to construct the final HTML output. The visual content of the page (e.g., with advertisements, etc.) are not considered. This uniform result page model may be appropriate for informational queries, but for navigational queries, caching and presenting results in this 10-per-page manner may waste cache space and network bandwidth. In other words, for navigational queries, it may be possible to provide, say, fewer than 10 results on the first result page

Received June 30, 2010; revised December 17, 2010; accepted December 20, 2010

 • Published online 9 February 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/asi.21496

[1] Jansen, Booth, and Spink (2008) report the viewing of multiple result pages as a characteristic of informational query.

and still satisfy the user, while saving the valuable resources mentioned above.

In this article, we propose to use nonuniform result page models, i.e., pages with varying number of results, for navigational queries. By doing so, our goals are to reduce network bandwidth usage during the result page presentation and to best utilize the cache space at search engine side. To our knowledge, this is the first study that explores the display and caching of query result pages with varying granularities (number of results).

More specifically, the contributions of this study are as follows:

- We define a cost model for the result page models and investigate the bandwidth utilization of nonuniform result page models for presenting up to the top 20 results. We restrict ourselves to the top 20 results, because for the vast majority of web queries, at most this many results are inspected by users (Pass, Chowdhury, & Torgeson, 2006). Using the cost model and a large query log we experimentally show that nonuniform result page models considerably reduce the number of snippets transmitted to the user and, hence, improve bandwidth usage for navigational queries. This reduction in bandwidth usage is more crucial for mobile search scenarios with low bandwidth capacities. Considering that a significant portion of mobile queries is navigational (Church, Smyth, Bradley, & Cotter, 2008), this improvement will be an important contribution for this field. In particular, our findings reveal that presenting top 20 results in three pages containing two, eight, and 10 results, respectively, would be the most efficient model for bandwidth consumption.
- We investigate and evaluate the gains of using nonuniform result page models in web search engine caching. We adapt a realistic framework that involves a hybrid result cache (comprising static and dynamic parts) along with an adaptive prefetching mechanism (Fagni et al., 2006). Our experimental findings show that the proposed model improves cache space utilization and subsequently reduces the number of cache misses.
- For our proposal to make sense in practical settings, it is crucial to observe the reaction of users for the nonuniform result page models because they are accustomed to seeing a constant (e.g., 10) number of results per page. That is, in a navigational search scenario, if the users keep on browsing successive result pages even after they find the target page, then this would diminish the efficiency promises of our proposal. We explore whether the proposed model fits the real-life browsing behavior of the users by conducting a user study. The study reveals that users easily adapt to nonuniform result page models and interact with the web search engine as expected, i.e., do not look further result pages if they are satisfied with the current result page. This means that the efficiency gains observed in the experiments are realistic and can be carried into real-life situations.

The rest of the article is organized as follows. The next section presents related work on web search engine caching and identifying user search goals, with special emphasis on the works about navigational queries. We then define and discuss alternative result page models for navigational queries and evaluate their presentation costs. The effects of nonuniform page models on caching are evaluated in the subsequent section. We then present the results of the user study, in which user browsing behavior is observed regarding nonuniform result pages. Next, we discuss the experimental results. Finally, we conclude the paper and give future research directions.

Related Work

Caching in Web Search Engines

Caching is one of the key techniques search engines used to cope with high query loads. Typically, search engines cache query results (Markatos, 2001; Fagni et al., 2006; Lempel & Moran, 2003; Ozcan, Altingovde, & Ulusoy, 2008c) or posting lists for query terms, or both (Baeza-Yates, & Saint-Jean, 2003; Long & Suel, 2005; Baeza-Yates et al., 2007). Caching the posting lists may lead to higher cache hit ratios simply because the same query terms may appear in several different queries (e.g., see Baeza-Yates et al., 2007). On the other hand, caching query results would provide more gains in terms of efficiency, especially when network communication dominates query-processing costs. Furthermore, in this case, there is no need for query processing, and it is adequate to simply send the result page(s) to the user.

Two types of caching frameworks are possible, namely, static and dynamic. In static caching, the cache is typically filled with the results of the most frequent queries in a query log. More recently, alternative strategies to determine the static cache content have been proposed (Ozcan, Altingovde, & Ulusoy, 2008a). The cache is static in the sense that it is populated by the selected queries in batch mode and it is not modified until the next batch update (i.e. it is read-only). In dynamic caching, the cache is not read-only and its entries change based on the query stream. It starts with an empty cache and fills its entries as new queries are submitted. When its capacity is reached, a victim cache entry must be chosen to make room for a new query result, based on the underlying cache replacement policy. Several cache replacement policies are proposed in the context of web proxy caching (Podlipnig & Böszörmenyi, 2003) and memory management in operating systems. One of the most widely used approaches is the least recently used (LRU) policy, which orders cache items based on their last usage time and evicts the one that has been requested least recently.

Fagni et al. (2006) propose a static-dynamic caching policy, in which the cache is divided into two parts: one devoted to static caching and the other to dynamic caching of the result pages. In this article, we follow this framework to evaluate the effects of using nonuniform result pages on the caching mechanism in search engines.

The previous works in this area (Fagni et al., 2006; Lempel & Moran, 2003) assume that query results are cached as typical pages that include 10 results for all queries. In this study, we claim that it is possible to obtain a better usage of

DOI: 10.1002/asi

cache space by proposed result presentation models for navigational queries, as users will usually click on only one or two results.

Identifying User Search Goals

User search goal identification is well-explored in the literature. Early works (Broder, 2002; Rose & Levinson, 2004) analyze query logs and classify different user search goals manually. Later works (Kang & Kim, 2003; Fujii, 2008; Lee, Liu, & Cho, 2005; Baeza-Yates, Calderon-Benavides, & Gonzalez-Caro, 2006; Jansen, Booth, & Spink, 2008; Lu, Peng, Li, & Ahmed, 2006; Liu et al, 2006; Mendoza & Baeza-Yates, 2008; Kofler & Lux, 2009; Jansen & Booth, 2010; Ashkan, Clarke, Agichtein, & Guo, 2009; Mendoza & Zamora, 2009) in this area focus on the automatic identification of user search goals by several machine learning techniques that exploit various features. The majority of these works essentially evaluates the accuracy of determining the user search's intent and proposes some future directions for the utilization of this knowledge. That is, there are relatively few works as ours that build upon these automatic identification methods. In particular, only two of the above works (Kang & Kim, 2003; Fujii, 2008) devise different ranking algorithms based on the user's search intent. As an alternative use case, Rose (2006) suggests that different user interfaces (or different forms of interaction with the users) should be provided to match users' different search goals. An illustrative study in this direction is by Kofler and Lux. In this work, once users' search goals are predicted, the results (digital photos in their case) are displayed accordingly, e.g., in either thumbnail or list view.

In this study, our aim is not to propose another identification method. Therefore, we adopt a simple and effective approach from Liu et al. (2006)'s work. They propose two features extracted from click-through data for user intent classification. These features are N clicks satisfied (nCS) and Top-n results satisfied (nRS). nCS measures the frequency of query submissions such that fewer than n clicks are performed in those cases. Similarly, nRS measures the frequency of query submissions such that all clicks in these submissions are within Top-n results. A decision tree classification exploiting these features and Lee et al.'s (2005) click distribution feature achieve 85% F-measure for navigational queries. Brenes, Gayo-Avello, and Perez-Gonzalez (2009) survey several methods for user intent classification and evaluate their accuracy using a set of 6,624 manually labeled queries. It is found that Liu et al.'s nRS feature gives the best accuracy for navigational queries. As noted in this survey, a combination of various features increases the accuracy. Therefore, in this article, we also decide to use a complementary method proposed by Jansen et al. (2008). In this work, navigational queries are identified by a set of heuristics, such as whether it contains person/organization/company names and query length is less than three, etc.

Lu et al. (2006) propose to use thousands of features for navigational query identification. They claim that for navigational queries, "presentation of the search results or the user perceived relevance can be improved by only showing the top results and reserving the rest of [the] space for other purposes." In this article, we explore different result page models for navigational searches to improve system efficiency, i.e., to best utilize the bandwidth and cache space.

Piwowarski and Zaragoza (2007) try to predict the possibility of the user clicking on a particular result for a query. They achieve over 90% accuracy for navigational queries. They suggest that if it can be determined that it is highly likely that a user will click on a certain result, the snippet for that result could be highlighted or the user could be directly sent to the page for that result. Our approach is in the middle of these extremes. Showing the top results on the first result page is a more conservative approach than directly forwarding the user to the top result page, but it is less conservative than just highlighting the snippet.

Teevan, Adar, Jones, and Potts (2007) examine the re-finding behavior of users by considering repeat queries in Yahoo!'s logs. Their results show that navigational queries constitute a significant portion of repeat queries. The authors suggest that the web search engine designers should take this re-finding behavior into account when designing user interfaces. They propose that a history of a user's past queries would be helpful. They also propose providing direct links, labeled with the most frequent query term, to websites for users who issue a high number of navigational queries. Their findings confirm our motivation for this study, because we also propose a special treatment for navigational query types.

Finally, it is possible that some commercial search engines may already be treating different query types in different manners. For instance, Google's Browse by Name facility (Google_toolbar; http://www.google.com/support/toolbar/bin/answer.py?hl=en&answer=9267) allows users to type queries into the address bar and if the system determines the site that the user wants to go to, it directly forwards him/her to that site; otherwise it gives the usual result page for the query. A recent work by Tann and Sanderson (2009) shows that some informational queries become partly navigational nowadays. For example, many users who issue an informational query about a film actor want to see the page in Internet movie database (IMDB) website or Wikipedia website. This work also shows that web search engines take this user expectation into account and shows the results from IMDB or Wikipedia at high ranks. However, the details of such technologies are not publicly available.

Our preliminary work (Ozcan, Altingovde, & Ulusoy, 2008b) gives the first results for using nonuniform result page models for navigational queries. However, that work employs a relatively small dataset and a static cache only. In this article, we extend the previous work in several ways. First, a more realistic caching experiment setup is constructed in this study. This setup involves a state-of-the-art caching mechanism (Fagni et al.'s static-dynamic caching approach) with an adaptive prefetching policy. In addition, experiments are performed with a remarkably larger query log. Second,

DOI: 10.1002/asi

the precision of the navigational query identification process is evaluated by a manual analysis of 500 queries. Finally, and most importantly, a user study is conducted to observe the effect of using nonuniform result pages for navigational queries on users' browsing behavior.

Result Page Models for Navigational Queries: Cost Analysis and Evaluation

Navigational queries constitute a nontrivial portion of web search queries (Jansen et al., 2008). It is hard to develop a result page model tailored to informational queries, as user behavior is unpredictable. Even if the search is successful, the user may click on several results or even request another result page to learn different aspects of the topic in question. On the other hand, those who submit navigational queries mostly aim to obtain the address of a target site. For such a query, a successful search would return the target result at the highest ranks, because Liu et al. (2006) note that information retrieval systems have much better effectiveness for navigational queries than informational queries. Subsequently, the user would be expected to click on only the top few results, and no more. Therefore, it might be beneficial to show the top few results on the first result page for navigational queries. Note that this choice of result presentation for navigational queries is also confirmed by previous works on user browsing behaviors by Joachims, Granka, Pan, Hembrooke, and Gay (2005) and Dupret and Piwowarski (2008). In these works it is observed that in almost all cases users check the first two query results right after the result page is displayed. If these results are not satisfying, then those at the lower ranks are examined. These earlier findings imply that it may be possible to present and cache the results of navigational queries in a more efficient manner, i.e., by presenting only a few results on the first result page.

In this section, we first develop a model for evaluating the presentation cost of query results in terms of the network usage and user browsing process. This cost model is used to determine whether there exists a better way of presenting the results of a navigational query to the user than the typical 10-results-per-page approach. We define some of the basic notions and introduce measures used in our cost model as follows:

- **Query Instance:** A query submission by a user to the search engine at a specific time. This also includes activities after query submission such as browsing the result pages. It ends when the user submits another query or the query session expires, i.e., the user does not perform any activity for 30 minutes.
- **Click Requests:** The set of clicks performed by the user after query submission. The users browse the returned results and click on the ones that seem to be relevant to their information need.
- **Result Page Model:** A result page is an atomic item for internal (e.g., result caching) and external (e.g., result display) purposes of the web search engine. A result page model describes how the query results are placed into the pages, each of which may include a fixed (uniform model) or variable number of results (nonuniform model).
- **numSnippetsSent:** A measure of the number of snippets sent by the web search engine to the users. It shows the network bandwidth cost incurred by the query result display. The ideal result page model must minimize this quantity.
- **numResultPagesBrowsed:** Indicates the total number of result pages that the users browse to reach the target document(s) for their query. The ideal result page model should minimize the number of result pages browsed. This quantity also corresponds to the number of requests that must be handled by the web search engine.

Note that the underlying objectives of the last two measures conflict with each other. In an extreme case, the result page model could show one result per result page. This model would guarantee that the number of snippets sent would be minimal but it would also maximize the number of result pages browsed. At another extreme, the search engine may show all the results (e.g., top 100 or top 1000) in one result page. While minimizing the number of the result pages browsed to only one, this model maximizes the number of snippets sent to the user. In what follows, we introduce a cost model based on these measures to find the optimal result page model for navigational queries.

Cost Analysis of the Result Page Models

In this section, we present a practical cost analysis model for the result page models with varying granularities. Various previous studies (Beitzel, Jensen, Chowdhury, Grossman, & Frieder, 2004; Jansen & Spink, 2005; Jansen, Spink, Blakely, & Koshman, 2006) agree on the finding that users rarely click on more than the top 20 results; therefore, without loss of generality we restrict our analysis to the result page models for this most common case. We explore how the top 20 results can be "paged/organized" to optimize the measures (*numSnippetsSent* and *numResultPagesBrowsed*) introduced above.

Consider a query instance Q and the click requests $C = \{c_1, c_2, \ldots, c_k\}$ of k clicks for this query. Assume that click requests are at the ranks $R = \{r_1, r_2, \ldots, r_k\}$, where $r_i \leq r_{i+1}$ and $r_k \leq 20$. Then, the result at rank r_k is defined as the *lowest-ranked clicked document* for this query instance. For simplicity, we assume that the user requested all the result pages up to the result page containing the result at rank r_k and that the user will not request more results after this rank. This assumption agrees with the "cascade model" proposed by Craswell, Zoeter, Taylor, and Ramsey (2008) for user click behavior. The cascade model assumes that users view each query result in a linear order, from top to bottom, and decide to click on it or not for each result. The users do not examine documents below a clicked result according to this model. Based on these assumptions, we collect all $<Q, r_k>$ pairs for the top 20 clicks. Then, we obtain the list $A = \{A_1, A_2, \ldots, A_{10}, A_{11}, A_{12}, \ldots, A_{20}\}$, where A_i denotes the number of query instances for which the lowest-ranked clicked document is at rank i (i.e., $r_k = i$).

DOI: 10.1002/asi

Next, we can derive formulas for *numSnippetsSent* and *numResultPagesBrowsed* measures using this list. Assume we have a two-page result model for top 20 as X_Y, which denotes that the first result page contains X results and the second result page contains Y (or 20-X) results. For this case, the formulas for *numSnippetsSent* and *numResultPagesBrowsed* are presented below:

$$numSnippetsSent_{X\_Y} = X \times \sum_{i=1}^{20} A_i + Y \times \sum_{i=X+1}^{20} A_i \quad (1)$$

$$numResultPagesBrowsed_{X\_Y} = \sum_{i=1}^{20} A_i + \sum_{i=X+1}^{20} A_i \quad (2)$$

Equation 1 expresses that we have to send the first result page that contains X number of snippets for all query submissions in the list A (captured with the first summation), but the second result page that contains Y snippets will be sent only to query submissions that also request results at the rank $X + 1$ or more (expressed with the second summation). Similarly, in Equation 2, the first summation accounts for the first result page that is clearly browsed for all queries and the second summation counts the number of queries that ask for the second page of results.

The objective of the result page model is to minimize these two quantities. We have to normalize these expressions to find a result page model that minimizes the overall cost. To this end, we use the conventional model of 10-results-per-page schema (10_10) as our baseline model and normalize the above expressions as follows:

$$numSnippetsSent\_norm_{X\_Y} = \frac{numSnippetsSent_{X\_Y}}{numSnippetsSent_{10\_10}} \quad (3)$$

$$numResultPagesBrowsed\_norm_{X\_Y} = \frac{numResultPagesBrowsed_{X\_Y}}{numResultPagesBrowsed_{10\_10}} \quad (4)$$

Note that we do not expect any two-page result model to achieve better figures than the baseline model of 10_10 for both of the above measures. That is, a model X_Y will decrease one quantity at the cost of an increase in the other quantity. However, the question is whether it is possible to find a model better than the baseline for the overall case. For this purpose, the summation of the normalized values in Equations 3 and 4 can be used as an overall measure for the result presentation models (see Equation 5). Note that, it is possible to have different weightings for the components in Equation 5 for different objectives. For example, in a mobile search scenario, the bandwidth savings might be more important and, therefore, the weight of *numSnippetsSent* can be higher. Because the value of such a sum for the baseline model is 2, we can use the following formulas to calculate the overall improvement percentage of the result page models over the baseline model in the experiments:

$$overall_{X\_Y} = numSnippetsSent\_norm_{X\_Y} + numResultPagesBrowsed\_norm_{X\_Y} \quad (5)$$

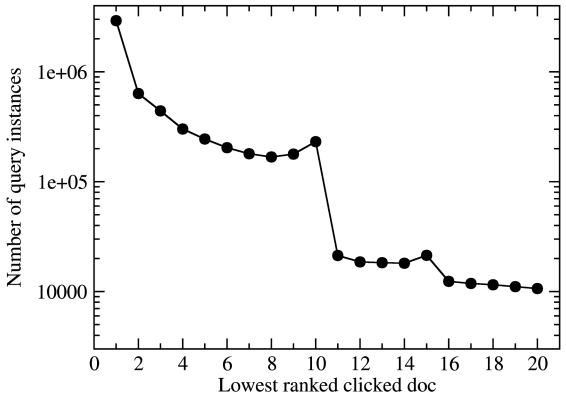

FIG. 1. The log graph shows the number of query instances, of which the last click is at a given rank (for training log).

$$\%improvement = \frac{2 - overall_{X\_Y}}{2} \times 100 \quad (6)$$

Note that although we restrict the discussion in this section to the top 20 results and result page models with two pages, the cost formulations can be generalized for top-K results and M pages. In the experimental evaluations, we consider models involving two and three result pages for the top 20 results.

Evaluation of the Result Page Models

Experimental setup. We use the AOL Query Log (Pass et al., 2006) that contains around 20 million queries of about 650K people for a period of 3 months. We exclude query instances that do not have any clicks (In AOL log, it is not possible to determine exactly which result pages are viewed by the users without any knowledge of clicks). Our subset contains 10,733,457 query instances that have at least one click. Among those queries, 5,853,929 are submitted in the first 6 weeks of 3 months and are reserved as the training set. This set is used to determine the navigational queries and to fill the cache (discussed in the next section). The remaining 4,879,528 queries constitute the test set, which will be used to evaluate the cost of the result page models.

We first present the characteristics of the training query log. Figure 1 shows the number of queries that have the lowest-ranked clicks at a given rank, i.e., it illustrates the list A described in the previous subsection. Note that in our query log, 93.7% of all clicks are for results in the top 20, justifying our use of those results in this study. The cut between ranks 10 and 11 is clearly visible, which indicates that the number of requests for results on the second page is an order of magnitude less than the number of requests for the results on the first page.

Navigational Query Identification. As mentioned in the related work section, there are several methods proposed for automatic navigational query identification in the literature. Because the proposal of another method for this purpose is not among the objectives of this study, we adopt a simple and

DOI: 10.1002/asi

effective approach from Liu et al.'s (2006) work and slightly extend it for more flexibility. We also combine Jansen et al.'s (2008) method into our approach for higher accuracy. The first stage of our method identifies navigational queries by using the click distribution, such that if users clicked up to rank two for that query 90% of the time,[2] it is classified as a navigational query. However, as we will see later, this definition is too restrictive; so we have an additional heuristic as follows. For a query, we define the notion of confidence in navigational query identification as in Equation 7. In this formula, $f_{Q,Top2}$ denotes the frequency of query instances of query Q, in which users clicked up to rank two; the right side of the multiplication represents the log normalized frequency of query Q. Here, f_Q is the frequency of query Q in the training set and f_{MAX} is the maximum frequency value in the training set.

$$Confidence = f_{Q,Top2} \times \frac{\log(f_Q + 1)}{\log(f_{MAX} + 1)} \quad (7)$$

We also use the above confidence measure for identifying navigational queries. In particular, we call those queries with $f_{Q,Top2}$ greater than 80% and a confidence score greater than 0.2 navigational queries, as well. The intuition underlying the components of this confidence score, namely, $f_{Q,Top2}$ and query frequency, is as follows. In navigational queries, users click on only one or two result document most of the time. Therefore, $f_{Q,Top2}$ is an important indicator in determining whether a query is navigational or not. Second, we use query frequency because the general rule in machine learning's is that as training size increases, classification accuracy also increases. Therefore, the queries with a high occurrence frequency and high $f_{Q,Top2}$ are identified as navigational queries with a higher confidence score. Finally, we exclude the queries that occur only once in the training log regardless of the above considerations, except those that include domain suffixes (www, com, edu, org, etc.), because Jansen et al. (2008) report that the existence of such suffixes in the query is an important characteristic of navigational queries. In a recent study by Lee and Sanderson (2010), it is found that 86% of URL queries (i.e., those that comprise full or partial URLs) are navigational.

In the second stage of our approach, we further apply Jansen et al.'s (2008) method for those queries that cannot be identified as navigational by the first stage. In this method, navigational queries are identified based on a set of rules derived from characteristics of those queries. Jansen et al. list the characteristics of navigational queries, as follows: (a) containing company, organization, and people names; (b) having less than three terms; (c) including domain suffixes (com, edu, etc.); and (d) viewing only the first result page. We used the freebase database (http://www.freebase.com/) that was also applied in Brenes et al. (2009) to get the list of company, organization, and people names. We obtained 1,579K people names, 492K organization names, and 89K company names. We check whether the query contains any of these names and satisfies the characteristics listed above to classify it as navigational.

In our training log, we discover that among the 2,778,591 distinct queries, 446,026 of them are navigational queries. When we consider the occurrence frequency of navigational queries, they constitute 32.5% of the query log. Three annotators manually inspected 500 randomly chosen queries that are identified as "navigational" to measure the precision. It is found that 416 are correctly identified, corresponding to 83.2% precision. There was a disagreement among three annotators for only 72 queries, and so they agree in 85.6% of the queries. When we consider the query occurrence frequency, the precision increases to 87.9%, such that out of 1,418 occurrences of 500 distinct queries, 416 correctly identified navigational queries constitute 1,246 submissions. This shows that identification is highly accurate for frequent queries but less accurate for rare queries. We further observe that there are some informational queries for which users mostly click on the top two results. We envision that the nonuniform result page models that we propose in this paper can also serve well for such queries.

Result Page Model Experiments. In these set of experiments, our aim is to compare alternative result page models for navigational queries based on numSnippetsSent and numResultPagesBrowsed measures defined previously. We use the training set to find the navigational queries as mentioned in the previous subsection. We process the test set, and for those queries that are identified as navigational in the training set, we apply the result presentation model X_Y (the first page contains X results/snippets and the second page contains Y, or 20-X, results). We use the uniform result presentation model (i.e., 10_10) for informational queries.

As we mentioned in the Cost Analysis of the Result Page Models subsection, we perform experiments for two-page and three-page result models for the top 20 clicks. Figure 2 shows the graph for the normalized cost of two-page models (X_Y), ranging from 1_19 to 19_1 for navigational query types (those queries in the test set that are identified as navigational query in the training set). As expected, models left to the 10_10 baseline approach achieve lower numSnippetsSent but higher numResultPagesBrowsed values than the baseline. On the other hand, models to the right of the baseline provide improvements in terms of the browsed result pages but increase the number of snippets sent.

The best result presentation model is 3_17, which has the normalized numSnippetsSent value as 0.43 and the normalized numResultPagesBrowsed as 1.08. The overall improvement for this model is 24.5% (excluding the cost of the informational query type). If we consider the improvement in the existence of informational queries such that we use the 3_17 model for navigational queries and the baseline 10_10 model for the remaining queries, then the overall improvement becomes 8.8% (numSnippetsSent = 0.80 and numResultPagesBrowsed = 1.025, such that a 20% decrease in the number of snippets sent occurs at the cost of a 2.5% increase in the number of result pages browsed.).

[2]Note that this corresponds to the nRS feature for $n = 2$ in Liu et al. (2006).

DOI: 10.1002/asi

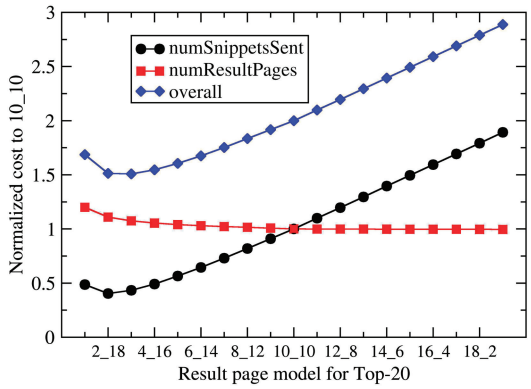

FIG. 2. The graph shows the costs of two-page result presentation models for only navigational queries.

Experiments with three-page result presentation models are also conducted, but the results for all 171 combinations are not reported because of space considerations. Here, we mention only the best models achieved. In this case, the best model is 2_8_10, which achieves a normalized numSnippetsSent value of 0.29 and a normalized numResultPagesBrowsed of 1.11. The overall improvement for this model is 29.5% (excluding the cost of the informational query type). If we consider the improvement in the existence of informational queries such that we use the 2_8_10 model for navigational queries and the baseline 10_10 model for the remaining queries, then the overall improvement becomes 10.1% (numSnippetsSent = 0.76 and numResultPagesBrowsed = 1.039, such that a 24% decrease in the number of snippets sent occurs at the cost of a 3.9% increase in the number of result pages browsed). In the rest of this article, we use the 2_8_10 model for navigational queries unless stated otherwise.

Caching with the Result Page Models and Experimental Evaluation

Employing Result Page Models for Caching

In this section, we examine the effect of using a nonuniform result page model for navigational queries on the caching mechanism in web search engines. We adapt a hybrid-caching framework proposed by Fagni et al. (2006), in which some part of the cache is reserved for static caching and the remaining part is reserved for dynamic caching. Their work shows that such an organization of the cache outperforms its purely dynamic and purely static variants.

The static cache is populated with the most frequent query results. In the traditional case, query results comprise result pages of 10 results. The static cache is filled with the most frequent <*query, result_page_no*> pairs in the training query log. Because the size of the result pages is the same in this case, considering frequency would be enough. However, as in our case, we have pages of 2, 8, and 10 results. The size of the cached items must be taken into account in addition to the frequency. Because snippets are large enough to dominate the size, we assume that the page sizes are directly proportional to the number of results, i.e., the size of the first page of a navigational query is only 20% of the size of a 10-results page, and the second page is 80%. Based on these considerations, the query result pages are ordered by the following score formula (adapted from Baeza-Yates et al., 2007) and the cache is populated with the items having the highest scores.

$$Score_{<query,result\_page\_no>} = \frac{FREQ_{<query,result\_page\_no>}}{SIZE_{<query,result\_page\_no>}} \quad (8)$$

In the dynamic part of the cache, we employ the LRU replacement policy that orders cache items based on their last usage time, and when the cache is full, it evicts the one that has been the least recently requested. Note that we do not attempt to identify navigational queries dynamically, because our identification method essentially requires the click frequencies of queries, which are obtained from previous query logs in an offline manner.[3] Therefore, for all query types, the result pages are stored in the uniform 10-per-page manner in the dynamic cache. The only exception is for those navigational queries of which only the top 2 results are cached in the static cache but not the rest. For these queries, if the user requests the second result page, i.e., results between ranks three to 10, then this second page (of 8 results) should also be inserted into the dynamic cache. Thus, the dynamic cache would include pages of size 10 or 8 in our setup.

Experiments

Experimental setup. We perform our experiments in the static-dynamic caching environment proposed by Fagni et al. (2006). It is important to decide what portions of the cache are reserved for static or dynamic caching for the best-hit ratio. As noted in Fagni et al., the best fraction value is based on the query log. The fraction of cache space reserved for the static cache is denoted as f_s; the cache with $f_s = 0$ corresponds to a purely dynamic cache and the cache with $f_s = 1.0$ corresponds to a purely static cache.

Figure 3 presents hit ratios of static-dynamic caches with different f_s values, ranging from purely dynamic to purely static for small (50K), medium (500K), and large (2500K) cache sizes. In the experiments reported throughout this section, the cache size is expressed in terms of the number of typical result pages in the cache. For small or medium cache sizes, most of the cache space must be reserved for the static cache, but for large cache sizes the dynamic cache portion must dominate the cache. For our query log, a static-dynamic cache with $f_s = 0.8$ gives the best-hit ratio for most of the cache sizes (It has the best hit ratio for small and medium cache sizes and very close to the best ratio for the large cache size, as shown in Figure 3). We, therefore, perform caching experiments with this cache configuration.

[3]It may be possible to accumulate query click statistics during the query evaluation for the purposes of online navigational query identification, but this direction is not further explored in this study and left as a future work.

DOI: 10.1002/asi

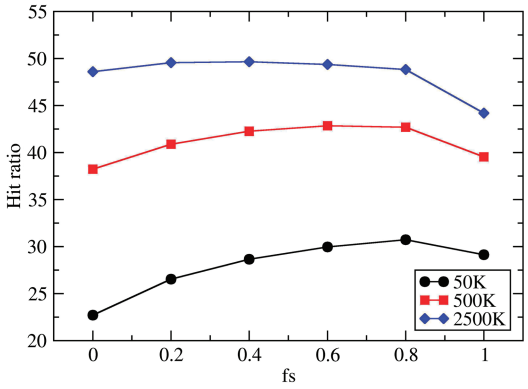

FIG. 3. Hit ratios versus f_s (fraction of the cache reserved for static cache); $f_s = 0$: purely dynamic, $f_s = 1$: purely static cache.

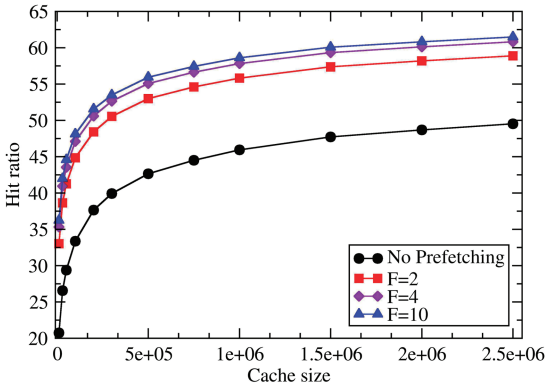

FIG. 4. Caching performances due to various levels of prefetching (F as the prefetching factor) with $f_s = 0.8$ and the cache size as the number of cached result page entries.

Second, prefetching query results is an important mechanism that increases the cache-hit ratio, as noted in Fagni et al. (2006). Normally, when a user requests a page of results and it is not in the cache, only the requested page of results is brought into the (dynamic) cache. If the user asks for the next page of results and it is not in the cache, then the same process is repeated. In the case of prefetching, if the requested page is not in the cache, instead of just one page, successive F (prefetching factor) result pages (including the requested page) are brought into the cache. Fagni et al. (2006) proposed an adaptive prefetching policy that remarkably increases the prefetching performance. This method prefetches F successive result pages only if the miss occurs for a page other than the first result page. When a miss occurs for the first result page, only the second result page is prefetched (for details, see Table V in Fagni et al.).

We applied the Fagni et al.'s (2006) adaptive prefetching policy in our experiments. Figure 4 presents the results of the experiments with different prefetching factors for a cache with $f_s = 0.8$. Note that minimum prefetching with $F = 2$ improves the "No prefetching" case considerably. The hit ratios increase as the prefetching factor increases but the improvements are marginal after $F = 4$. It is important to note that in the case of prefetching 20 successive pages (not shown in the figure), the hit ratio starts to decrease, because after this point, the cache is filled with too many useless prefetched pages that results in eviction of pages that will likely be requested in the near future.

In the following experiments, we compare the cache performance in the case of using nonuniform result pages (2_8_10) to that of using the baseline (10_10) approach. In the light of the above discussions, we use $f_s = 0.8$ as it gives the best performance for almost all cache sizes that are experimented. For the prefetching strategy, we experiment with the prefetching factor $F = 4$, because after this point improvements in the hit ratio are marginal. The static part of the cache is filled with the queries in the training set as described in the previous subsection and the remaining queries are submitted to dynamic part of the cache. Then, the overall cache performance is evaluated by using the disjoint test set.

In the literature, the hit ratio is widely used as a measure for comparing the different caching approaches. However, in our case, because the size of the cached items is not the same, using the hit ratio as an evaluation measure is not possible. For instance, assume that we have cached the first (including results at ranks one and two) and the second result pages (including results at ranks three to 10) of a navigational query in our caching scheme, and then let the conventional caching scheme cache the first result page (results of ranks one to 10) of the same query. Assume that clicks for this query in the test log are at the ranks $R = \{1, 2, 5, 12\}$. For our system, we would have hits for the first and second result pages, but we would have a miss for the third result page, which results in a hit ratio of two thirds. On the other hand, the conventional system would have one hit for the first result page and one miss for the second result page, giving a hit ratio of one half. As can be seen, even though these two caching strategies cached the same amount of results for the same query, the hit ratio measure artificially differs, and thus is not appropriate to use here. Therefore, we use another approach to measure the effectiveness of the two strategies. Instead of hit ratio, we compute the absolute number of cache misses for each strategy. For instance, in the above example, there is only one cache miss for both caches, which is a fair evaluation.

Experimental Results. Our caching experiment results are shown in Table 1. The cache size is given as the number of 10-results-per-page cache size entries. The reduction percentages in the total miss counts are shown in a separate column. As the cache size increases, improvements decrease, because this causes many of the second result pages of navigational queries to be cached also. Therefore, our method is more effective for small-sized and medium-sized caches.

For large cache sizes, our approach experiences higher miss counts than the baseline, which seems surprising at first, but has a sound explanation. For some of the queries that are identified as navigational in the training log and that have all their clicks in the top two results, our static caching scheme would never cache the second results page (including results three to 10), regardless of the cache size. That is, recall that we

DOI: 10.1002/asi

TABLE 1. Caching performances with $f_s = 0.8$ and prefetching $F = 4$.

Cache size	Baseline cache Cache miss counts	Our cache(with 2_8_10) Cache miss counts	Our cache(with 2_8_10) % reduction
5K	4,072K	3,851K	5.42
10K	3,824K	3,604K	5.75
30K	3,437K	3,264K	5.04
50K	3,269K	3,124K	4.43
100K	3,059K	2,952K	3.49
200K	2,866K	2,806K	2.11
300K	2,765K	2,702K	2.27
500K	2,657K	2,613K	1.67
750K	2,546K	2,538K	0.30
1500K	2,448K	2,453K	−0.21
2500K	2,350K	2,358K	−0.33

TABLE 2. Caching performances with $f_s = 0.8$ and prefetching $F = 4$ with smoothing.

Cache size	Baseline cache Cache miss counts	Our cache(with 2_8_10) Cache miss counts	Our cache(with 2_8_10) % reduction
5K	4,072K	3,868K	5.00
10K	3,824K	3,622K	5.28
30K	3,437K	3,283K	4.47
50K	3,269K	3,145K	3.80
100K	3,059K	2,974K	2.78
200K	2,866K	2,805K	2.14
300K	2,765K	2,702K	2.28
500K	2,657K	2,604K	2.01
750K	2,546K	2,524K	0.87
1500K	2,448K	2,434K	0.59
2500K	2,350K	2,339K	0.46

compute the frequency of the *<query, result page>* pairs in the training set, and thus if all clicks are for the top two results, then the frequency of the second result page would be 0, and can never be cached in the static portion of the cache. The second result page for such a query can placed in the dynamic cache only when a user requests it for the first time, which would also cause a cache miss. To remedy this problem, we use the confidence score used during the identification of navigational queries for a smoothing operation. That is, for each query identified as a navigational query, we multiply its (first result page) frequency with $(1 - confidence)/c$, where c is an experimental constant (which is found as 2), and add this score to the frequency of the second result page. This creates a smoothing effect and allows us to cache the second result page of a navigational query with low confidence, even if results between ranks three and 10 are never clicked on the training log. In Table 2, we report the results with smoothing. Notice that our model does not create an increase in miss counts for any cache size, in turn of a slight decrease in the reductions for smaller cache sizes (compare Tables 1 and 2). The proposed result page model provides reductions of 4% to 5% in absolute miss counts.

TABLE 3. Caching performances by excluding costs for singleton query misses ($f_s = 0.8$ and prefetching $F = 4$).

Cache size	Baseline cache Cache miss counts	Our cache(with 2_8_10) Cache miss counts	Our cache(with 2_8_10) % reduction
5K	2,445K	2,241K	8.35
10K	2,196K	1,995K	9.17
30K	1,810K	1,656K	8.53
50K	1,642K	1,517K	7.60
100K	1,431K	1,347K	5.90
200K	1,239K	1,177K	4.98
300K	1,137K	1,074K	5.50
500K	1,029K	976K	5.15
750K	919K	896K	2.48
1500K	820K	806K	1.70
2500K	722K	712K	1.44

Finally, we realized that reporting our gains in terms of miss counts disfavors us in that the majority of misses in our system is for singleton queries (queries that occur only once in the test log and do not appear at all in the training log), which can never be resolved in a caching mechanism. For comparison purposes, we also report the experimental results for the test set queries excluding the singleton query miss counts in Table 3 (again with smoothing). For this case, reductions are more emphasized, reaching up to 9.17%.

User Browsing Behavior with the Nonuniform Result Page Model

Search engine users' browsing behavior is well-studied in the literature (Lorigo et al., 2006; Craswell et al., 2008; Dupret & Piwowarski, 2008) for purposes such as providing better search interfaces for users and better exploitation of query logs as implicit relevance judgments. All these studies are based on the uniform result presentation model that shows 10 results per page. In this article, we propose a nonuniform result page model for search engines using navigational query identification; thus, it is important to investigate the effect of such a result presentation model on user browsing behavior. To this end, we conduct a user study. In this section, we first describe the experimental setup and then present our findings.

User Study Setup

Ten search tasks, presented in the Appendix, are prepared for the user study (http://139.179.11.31/rifat/Search/). Five of these tasks are navigational and the remaining five have informational intent. Navigational tasks are chosen so that the target result could be found among the top results in one of the major search engines.

The user study is designed as follows. When a subject logs in to the system, at the top of the usual search interface (i.e., a textbox in which to type the query), the system randomly displays a task from among the search

DOI: 10.1002/asi

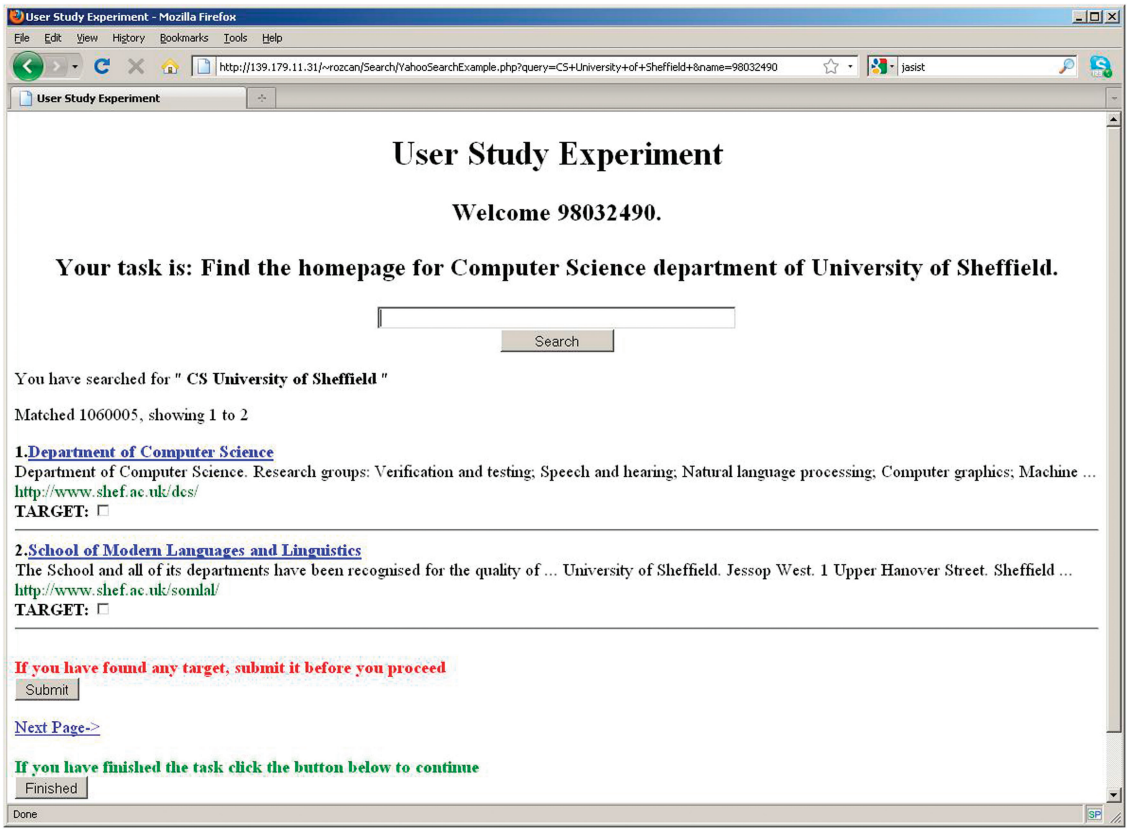

FIG. 5. The search interface of the user study.

tasks shown in the Appendix and that has never been completed by that user (see Figure 5). The user is asked to accomplish the given task by submitting queries and viewing the results, including URLs and snippets, as usual. We used Yahoo!'s search engine "web search" web service (http://developer.yahoo.com/search/web/V1/webSearch.html) to get the results of the user queries. For the navigational tasks,[4] only the top two results are shown based on the nonuniform result page model (2_8_10) discussed in the previous sections. If the user clicks the "next page" link, then the top 3 to top 10 results are shown in the second result page. On the other hand, for informational queries, the baseline result presentation is used and 10 results are shown on each page. In the tutorial stage of the user study, we requested that subjects mark the search result(s) that answer(s) the given task by clicking the "TARGET" checkboxes provided at the end of each result snippet. The subjects are allowed to click on the result URL to further see the content of the result document. Note that in this study, subjects do not perform an evaluation of the information retrieval system; if that were the case, then they would be requested to scan Top100 or Top1000 results and decide whether each result is relevant or irrelevant. In this case, subjects are instructed to stop when they think they have accomplished the given task, i.e., found the web pages that answer the question in the corresponding search task.

[4]Note that the user is not informed about navigational or informational query types and they do not know what we are measuring.

Fifty-four subjects comprising graduate and undergraduate students from the computer engineering and industrial engineering departments in our university perform the user study. There are 14 female subjects and 40 male student subjects. Each subject is given a brief introduction to the experiment. Our aim is to measure the browsing behavior of subjects to the nonuniform result page model for navigational queries. Specifically, we will measure the users' tendency to request the next result page even if they can identify the target answer between top two results in the first page. In this sense, we investigate whether the users can easily accommodate to a nonuniform result page model, or they insist on seeing all top 10 results even when they find the answer in top two.

User Study Results

The result of the experiment reveals that in only 24 of the 237 navigational tasks (ignoring the cases where the subjects did not enter any target result for the query), the subjects wanted to look for the results beyond the top two. In other words, subjects were satisfied with the top two results 89.9% of the time.

Table 4 presents additional statistics obtained in this user study. Each statistic is given as an average for navigational and informational queries. As expected, the average informational task duration is much longer than the average navigational task duration. Subjects type more queries for informational tasks. The number of target sites selected

DOI: 10.1002/asi

TABLE 4. User study experiment statistics.

Statistic	Navigational	Informational
Average task completion time (sec)	38.33	83.94
Average number of queries submitted for a task	1.09	1.27
Average number of results selected as TARGET for a task	1.25	4.09
Average number of URLs clicked for a task	0.33	1.06
Average percentage of TARGET results selected at ranks Top1-2 for a task	96%	34%

for informational queries is much higher than the number of targets selected for navigational queries. We observe that the actual URLs and/or snippets are much more helpful for navigational queries than informational queries because users click on fewer numbers of URLs in the former case. Note that this number should be close to one in a real-life case; in that case, users are expected to click on the target website because the sole purpose of a navigational query is to reach that website. However, in this experimental setting, subjects are instructed to just select the correct sites; most of the time they understand the target website from snippets and do not need to actually visit that website.

The last statistic evaluates the ranks of the target sites selected for each type of task. We observe that 96% of the target sites selected for navigational queries reside in the top one or two ranks. However, this ratio decreases to 34% for informational tasks.

To justify our approach, it is important to analyze the cases when subjects want to see beyond the top two, even though they found the target site(s) in the top two. We will refer to such cases as "Beyond Top 2" hereafter. On average, subjects also want to see beyond the top two in only 9.3% of total navigational query instances, in which users selected target sites in the top two. When we focus on Beyond Top 2 cases, we see a pattern, shown in Figure 6, between the occurrence of such cases and the viewing order of the navigational task. (Recall that search tasks are assigned to subjects in random order.) It is observed that many of the Beyond Top 2 cases occur when a subject meets our result presentation model of 2_8_10 for navigational queries for the first time (i.e., 10 of 22 such cases occur when a user evaluates his/her first navigational task, as shown in Figure 6). As users see more examples of the nonuniform result presentation model, they get used to it; the number of Beyond Top 2 cases drops dramatically after performing a few navigational tasks. Therefore, although the percentage of Beyond Top 2 cases is 9.3%, users stop asking for the next result pages after they experience just a few navigational queries with our method.

The overall result of the user study can be summarized as follows. Users easily adapt to nonuniform result page models and interact with the web search engine as expected, i.e., they do not look for further result pages if they are satisfied with the current result page. This conclusion justifies our proposal of a nonuniform result model and implies that the efficiency gains obtained in the experimental setup of the previous sections could be transferred to real-life search systems.

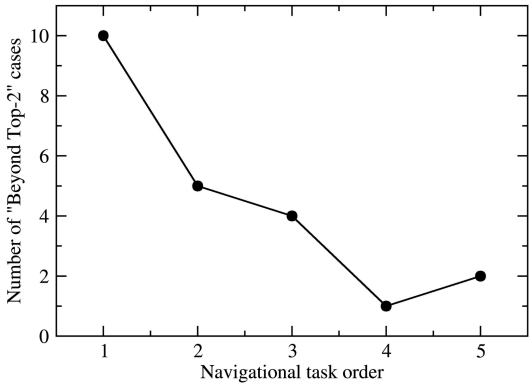

FIG. 6. The pattern between Beyond Top 2 cases and navigational task order.

Summary and Discussion of the Results

In this study, we exploit the navigational query type for result presentation and caching in web search engines. To this end, a cost analysis model is introduced to evaluate nonuniform result presentation approaches. Experiments reveal that the 2_8_10 result page model is better than the uniform model for navigational queries. The number of snippets sent for this type of query can be reduced by 71% at the cost of an 11% increase in the number of result pages browsed. In other words, assuming that an HTML result page (of top 10 results) takes 4 KB space (Fagni et al., 2006) and a TCP packet stores at most 1,460 bytes of data, our approach would require only one TCP packet for top two results, instead of three packets needed for transmitting top 10 results. This reduction in the number of packets sent will be more important in mobile search scenarios because of low bandwidth capacity and high packet loss probability.

The nonuniform result page model further improves cache performance at the search engine side as it causes fewer misses (up to 9.17%) than its uniform counterpart. However, we obtain these gains in a controlled experiment environment, and thus it is important to observe real user browsing behavior for the nonuniform query result pages. A user study conducted for this purpose justifies our motivation and findings. That is, for navigational search tasks, users quickly adapt to seeing only the top two results on the first page, and they do not request other pages if their target is already there. Given that search engines are usually more successful at returning the result of such queries in the top ranks (Liu et al.,

DOI: 10.1002/asi

2006), we expect that the gains obtained in our experimental setup can also be obtained in real-life systems. Indeed, we envision that real-life user practices identified by earlier works can give way to even higher gains than expected. For instance, the work of Teevan et al. (2007) examines the users' re-finding behavior and shows that navigational queries constitute a significant portion of repeat queries. This means that a user tends to use the same navigational query to locate a particular page in his or her mind and would be familiar with the result page. Thus, in such cases, users are even less likely to browse result pages beyond the first page. However, in the user study described above, the users are probably not very familiar with the navigational tasks assigned to them. That is, our user study serves as a worst-case scenario for the number of Beyond Top 2 cases, because, in contrast to real-life, almost no task is a repeat query.

Evidence in addition to user click behavior supports our choice of the 2_8_10 result page model for navigational queries. The eye-tracking study in Joachims et al. (2005) revealed that users almost always check the first two results immediately after a result page is displayed. Then, a few seconds later, other results are examined if the first two are not satisfying. Therefore, our nonuniform result model is confirmed by this type of user browsing behavior.

Finally, the 2_8_10 result presentation model for navigational queries enables an effective use of web browser space (Lu, 2006). Given that only two results are shown on the first (and most important) page, now it is possible to reserve the remaining page space for purposes such as advertisements, result visualization, or related query suggestions. This may have a positive impact on the number of clicks received by ads, which is an important source of revenue for commercial search engines. As another use case, search engines may want to provide more information in the result snippets, especially for news site navigational queries (e.g., "New York Times"), by giving direct links to several important headlines.

Conclusion

In this study, we propose and evaluate nonuniform result page models to display and cache the results of navigational queries. It is shown that for these types of queries, it is possible to reduce the number of snippets sent by 71%, at the cost of an 11% increase in the number of result pages browsed. The proposed result presentation model for navigational queries is also shown to be valuable for caching mechanism, especially for small-sized and medium-sized caches. It is possible to reduce the number of miss counts up to 9.17%. Our findings are encouraging and justify the need for the special treatment of different query types, especially navigational queries, submitted to a web search engine. Our user study with 54 subjects shows that users easily adapt to nonuniform result page models and interact with the search engine as expected, i.e., they do not look for further result pages if they are satisfied with the current result page. Therefore, the aforementioned efficiency gains observed in the experiments can be carried over to real-life situations.

In our future work, we first plan to investigate the use of nonuniform result page models for query types other than navigational. Furthermore, transactional query identification might be exploited for more relevant sponsored search links. Indeed, it is even possible to have a separate cache specialized for each query type, i.e., each cache employs not only different result page models but also different caching policies. This is an interesting direction, as the caching policies applicable to each query type may also differ. For example, we envision that the results of the informational queries need to be refreshed more often than those of the navigational queries, which implies a need for different cache refreshment policies for different query types. Finally, we plan to exploit navigational queries to reduce the cost of some other major tasks, such as the actual query processing, in a web search engine.

Acknowledgments

This work is supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) by the grant numbers 108E008 and 110E135. We thank A. Deniz Güven and the user study subjects for their contributions to the experiment. We are grateful to Rana Nelson for proofreading.

References

Ashkan, A., Clarke, C., Agichtein, E., & Guo, Q. (2009). Classifying and characterizing query intent. In Proceedings of the 31th European Conference on IR Research on Advances in information Retrieval. Lecture Notes in Computer Science, 2478, 578–586.

Baeza-Yates, R., & Saint-Jean, F. (2003). A three level search engine index based in query log distribution. In Proceedings of 10th International Symposium on String Processing and Information Retrieval. Lecture Notes in Computer Science, 2857, 56–65.

Baeza-Yates, R., Gionis, A., Junqueira, F., Murdock, V., Plachouras, V., & Silvestri, F. (2007). The impact of caching on search engines. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 183–190). New York: ACM Press.

Baeza-Yates, R., Calderon-Benavides, L., & Gonzalez-Caro, C. (2006). The intention behind web queries. In Proceedings of String Processing and Information Retrieval. Lecture Notes in Computer Science, 2857, 98–109.

Beitzel, S.M., Jensen, E.C., Chowdhury, A., Grossman, D., & Frieder, O. (2004). Hourly analysis of a very large topically categorized web query log. In Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval (pp. 321–328). New York: ACM Press.

Brenes, D.J., Gayo-Avello, D., & Perez-Gonzalez, K. (2009). Survey and evaluation of query intent detection methods. In Proceedings of the 2009 Workshop on Web Search Click Data (pp. 1–7). New York: ACM Press.

Broder, A. (2002). A taxonomy of web search. SIGIR Forum, 36(2), 3–10.

Church, K., Smyth, B., Bradley, K., & Cotter, P. (2008). A large scale study of European mobile search behaviour. In Proceedings of the Tenth International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '08) (pp. 13–22). New York: ACM Press.

Craswell, N., Zoeter, O., Taylor, M., & Ramsey, B. (2008). An experimental comparison of click position-bias models. In Proceedings of the International Conference on Web Search and Web Data Mining (pp. 87–94). New York: ACM Press.

Dupret, G.E., & Piwowarski, B. (2008). A user browsing model to predict search engine click data from past observations. In Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 331–338). New York: ACM Press.

DOI: 10.1002/asi

Fagni, T., Perego, R., Silvestri, F., & Orlando, S. (2006). Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data. ACM Transactions on Information Systems, 24(1), 51–78.

Fujii, A. (2008). Modeling anchor text and classifying queries to enhance web document retrieval. In Proceedings of the 17th International Conference on World Wide web (pp. 337–346). New York: ACM Press.

Jansen, B.J., & Booth, D. (2010). Classifying web queries by topic and user intent. In Proceedings of the 28th of the International Conference Extended Abstracts on Human Factors in Computing Systems 2010 (pp. 4285–4290). New York: ACM Press.

Jansen, B.J., Booth, D., & Spink, A. (2008). Determining the informational, navigational, and transactional intent of web queries. Information Processing & Management, 44(3), 1251–1266.

Jansen, B.J., & Spink, A. (2005). How are we searching the World Wide web? A comparison of nine search engine transaction logs. Information Processing & Management, 42(1), 248–263.

Jansen, B.J., Spink, A., Blakely, C., & Koshman, S. (2006). Web searcher interactions with the dogpile.com meta-search engine. Journal of the American Society for Information Science and Technology, 58(4), 1875–1887.

Joachims, T., Granka, L., Pan, B., Hembrooke, H., & Gay, G. (2005). Accurately interpreting clickthrough data as implicit feedback. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 154–161). New York: ACM Press.

Kang, I., & Kim, G. (2003). Query type classification for web document retrieval. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 64–71). New York: ACM Press.

Kofler, C., & Lux, M. (2009). Dynamic presentation adaptation based on user intent classification. In Proceedings of the 17th ACM international Conference on Multimedia (pp. 1117–1118). New York: ACM Press.

Lee, U., Liu, Z., & Cho, J. (2005). Automatic identification of user goals in web search. In Proceedings of the 14th International Conference on World Wide Web (pp. 391–400). New York: ACM Press.

Lee, W.M., & Sanderson, M. (2010). Analyzing URL queries. Journal of the American Society for Information Science and Technology, 61(11), 2300–2310.

Lempel, R., & Moran, S. (2003). Predictive caching and prefetching of query results in search engines. In Proceedings of the 12th International Conference on World Wide Web (pp. 19–28). New York: ACM Press.

Liu, Y., Zhang, M., Ru, L., & Ma, S. (2006). Automatic query type identification based on click through information. Proceedings of the Asia Information Retrieval Symposium. Lecture Notes in Computer Science, 4182, 593–600.

Long, X., & Suel, T. (2005). Three-level caching for efficient query processing in large web search engines. In Proceedings of the 14th International Conference on World Wide Web (pp. 257–266). New York: ACM Press.

Lorigo, L., Pan, B., Hembrooke, H., Joachims, T., Granka, L., & Gay, G. (2006). The influence of task and gender on search and evaluation behavior using Google. Information Processing & Management, 42(4), 1123–1131.

Lu, Y., Peng, F., Li, X., & Ahmed, N. (2006). Coupling feature selection and machine learning methods for navigational query identification. In Proceedings of the 15th ACM International Conference on Information and Knowledge Management (pp. 682–689). New York: ACM Press.

Markatos, E.P. (2001). On caching search engine query results. Computing Communications, 24(2), 137–143.

Mendoza, M., & Baeza-Yates, R. (2008). A web search analysis considering the intention behind queries. In Proceedings of the Latin American Web Conference (pp. 66–74). Washington, DC: IEEE Press.

Mendoza, M., & Zamora, J. (2009). Identifying the intent of a user query using support vector machines. In Proceedings of the 16th International Symposium on String Processing and Information Retrieval (pp. 131–142), Saariselkä, Finland.

Ozcan, R., Altingovde, I.S., & Ulusoy, Ö. (2008a). Static query result caching revisited. In Proceeding of the 17th International Conference on World Wide Web (pp. 1169–1170). New York: ACM Press.

Ozcan, R., Altingovde, I.S., & Ulusoy, Ö. (2008b). Utilization of navigational queries for result presentation and caching in search engines. In Proceedings of the 17th ACM Conference on Information and Knowledge Management (pp. 1499–1500). New York: ACM Press.

Ozcan, R., Altingovde, I.S., & Ulusoy, Ö. (2008c). Space efficient caching of query results in search engines. In Proceedings of the 23rd International Symposium on Computer and Information Sciences (pp. 1–6). Washington, DC: IEEE Press.

Pass, G., Chowdhury, A., & Torgeson, C. (2006). A picture of search. In Proceedings of the First International Conference on Scalable Information Systems, Hong Kong, (Vol. 152). New York: ACM Press.

Piwowarski, B., & Zaragoza, H. (2007). Predictive user click models based on click-through history. In Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management (pp. 175–182). New York: ACM Press.

Podlipnig, S., & Böszörmenyi, L. (2003). A survey of web cache replacement strategies. ACM Computing Surveys, 35(4), 374–398.

Rose, D.E. (2006). Reconciling information-seeking behavior with search user interfaces for the web. Journal of the American Society for Information Science and Technology, 57(6), 797–799.

Rose, D.E., & Levinson, D. (2004). Understanding user goals in web search. In Proceedings of the 13th International Conference on World Wide Web (pp. 13–19). New York: ACM Press.

Tann, C., & Sanderson, M. (2009). Are web based informational queries changing? Journal of the American Society for Information Science and Technology, 60(6), 1290–1293.

Teevan, J., Adar, E., Jones, R., & Potts, M.A. (2007). Information re-retrieval: Repeat queries in Yahoo!'s logs. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 151–158). New York: ACM Press.

Appendix

Navigational and Informational Tasks Used in the User Study

Navigational tasks

Find the official homepage for the Beijing 2008 Summer Olympics
Find the homepage for the United Airlines company
Find the homepage for the *New York Times* newspaper
Find the homepage for the computer science department of the University of Sheffield
Find the homepage for the United States Consulate in Istanbul, Turkey

Informational tasks

Find where and when the first Olympics Games were organized
Find when NASA sent a spacecraft for the first time to explore Mars
Find information on the effects of global warming on polar bears
Find information on how to quit smoking
Find information on tourist places in Turkey

DOI: 10.1002/asi