Site-Based Dynamic Pruning for Query Processing in Search Engines

Ismail Sengor Altingovde, Engin Demir, Fazli Can, Özgür Ulusoy
Computer Engineering Department, Bilkent University, Ankara 06800, Turkey
{ismaila, endemir, canf, oulusoy}@cs.bilkent.edu.tr

ABSTRACT

Web search engines typically index and retrieve at the page level. In this study, we investigate a dynamic pruning strategy that allows the query processor to first determine the most promising websites and then proceed with the similarity computations for those pages only within these sites.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *clustering, search process, query formulation.*

General Terms
Performance, Experimentation.

Keywords
Cluster-skipping, dynamic query pruning, inverted index.

1. INTRODUCTION

In most of the search engines, a typical unit of the indexing and retrieval is a single Web page. That is, each page is considered as a separate entity (sometimes associated with the anchor text of the referring pages), which is indexed off-line and compared to a query on-line. On the other hand, Web pages are usually hosted by a particular organization, person, etc. and pages at the same site may form a more coherent set in terms of the content, with respect to the pages that reside in other sites. In this study, we propose a (conceptually) two-stage query processing strategy, in which first the websites that are most similar to a query are determined, and then pages within these sites and most similar to the query are returned as the final result. Our goal is to reduce the query processing time while maintaining the quality of the top-K results (where K is a small number, typically less than 30, since very few Web users look at more than the first 30 results). The contributions of this work are as follows: we *(i)* investigate the quality of results for a site-based pruning approach, and *(ii)* adapt a specifically tailored inverted index [1, 3] that allows the efficient computation of the proposed strategy.

2. SITE-BASED DYNAMIC PRUNING

For a given query, we first determine the top-N sites, *best-sites*, that are most similar to the query, and then we obtain the top-K Web pages, *best-pages*, within these sites. Notice that, this is similar to the cluster-based retrieval as described in [1, 3]. That is, it can be considered as if each Web page belongs to the "cluster" identified by its website (i.e., the hostname part of its URL). Thus, we employ the cluster-skipping inverted index structure (CS-IIS) of [1] to create an index which interleaves the site and page information and allows efficient query processing.

In CS-IIS, the <*document, term frequency*> pairs in a posting list are reorganized such that all documents from the same site are grouped together, and at the beginning of each such group two extra elements are stored in the form of <*site id, next site address*> and <*no of documents, avg. term frequency*> [1]. The former element allows query processor to jump to the next site without processing the postings of a particular site. In the latter additional element, we store adequate information to compute the (partial) similarity of a site S to a query including a term t: the first value is the *number of documents* including t at S, and the second one is the *average term frequency* of t at S.

During query processing, there are two possible strategies. In the *typical* strategy (i.e., similar to CBR) there are two stages [3]: in the first stage, best-sites are determined by using the additional elements in the posting lists of the each query term; i.e., posting list for each query term is retrieved and processed, to obtain the top-N best-sites. Notice that, during this stage, only the additional elements are accessed. In the second stage, for each posting list, only those portions of the list that are from the best-sites are accessed: if a site is not in best-sites, it is simply skipped by jumping to the next site pointed by the "next site address".

An alternative *incremental* strategy is proposed in [1]. In this case, each posting list is fetched only once, but two passes are made on each list. In the first pass, the *current* best-sites are determined, and in the second pass, partial similarities for only those pages that are from these current best-sites are computed. In [1], it is stated that both strategies provide comparable effectiveness, whereas the latter avoids fetching each list twice. This is important in case that the lists are stored on disk.

In this study, without loss of generality, we assume that all index structures are stored in the main memory (as in [8]) and compare two strategies: *(i)* the baseline (no-pruning) strategy which involves a traditional inverted index and processes the entire posting list for each query term, and *(ii)* the site-based pruning approach with typical and incremental strategies, both of which employ a CS-IIS.

3. EXPERIMENTS

Dataset. In this study, we use a collection of about 4.3 million Web pages obtained from Stanford University's WebBase Project repository. This dataset involves pages crawled from the US government domain during the first quarter of 2007. The pages in the dataset are from 1,103 websites.

Indexing. We eliminated HTML tags, scripts, etc. and English stopwords. No stemming is applied. Next, the typical inverted index and CS-IIS are constructed. Both files are compressed using the same procedures as described in [1]. The resulting typical and cluster-skipping inverted files take 6.3 GB and 6.6 GB (uncompressed) and 767 MB and 785 MB (compressed), respectively. Note that, the increase in the CS-ISS file size is only

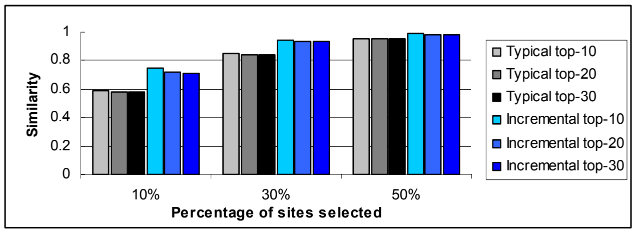

Figure 1. Similarity of pruned results to the baseline results.

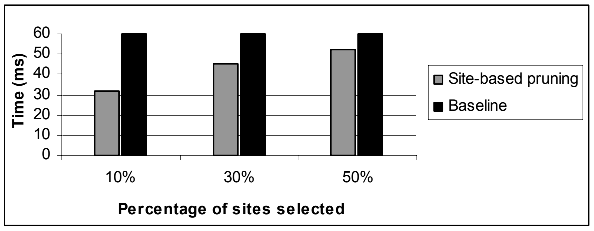

Figure 2. Average times for query processing strategies.

2%, an affordable overhead. During the experiments, only the compressed files are used.

Query processing. We use the efficiency task topics of TREC 2005 terabyte track, including 50K queries and 2.3 terms per query, on the average. The similarity computations between queries and sites/documents use TF-IDF and the cosine metric [1].

Effectiveness Experiments. We first compare the typical and incremental strategies for site-based pruning to the baseline strategy. Since there are no relevance judgments for our collection and query set, the top-*K* results obtained from each pruning strategy is compared to those results from the baseline. A measure based on the symmetric difference is used for comparing two lists [4]. In Figure 1, we plot the similarity between the top-*K* ($K \in \{10, 20, 30\}$) results of the baseline approach and site-based pruning approaches, namely typical and incremental strategies, versus the pruning level. The pruning level is simply controlled by the parameter *N*, which denotes that the top-*N*% of the sites is selected as the best-sites.

Our findings reveal that *(i)* the pruned results reveal a high similarity to the non-pruned results (e.g., incremental strategy achieves 74% similarity for top-10 results using 10% of the sites only), *(ii)* the incremental strategy for site-based pruning does not degrade result quality with respect to the typical strategy (as also observed in [1]) and may even improve the latter, and *(iii)* as the number of the selected sites increase, the results converge.

Efficiency Experiments. The performance of the baseline strategy and site-based pruning strategy are compared with respect to pruning level. Note that, since the inverted index is assumed to be in-memory, both typical and incremental pruning strategies work in almost the same time and thus a single figure is presented. Figure 2 reveals that the site-based pruning strategy provides significant efficiency improvements over the baseline, reaching up to 46% when the top-10% of the sites is selected.

4. COMPARISON WITH RELATED WORK

Li et al. [5] defines the notion of Web information unit (WIU) as logical Web document including several real Web pages. During query processing, they retrieve such WIUs, which are computed by graph traversals starting from the pages that include the query terms (i.e., initial pages). Our work is significantly different from that in two aspects: First, they essentially aim to improve search effectiveness by retrieving semantically coherent pages as a logical unit, whereas our goal is improving efficiency at the first place. Second, their logical units are constructed in a bottom-up manner. In contrast, we apply a simple top-down approach in which the relatively non-promising sites, a rather coarse unit of content, are eliminated so that the resources for the query processing can only be devoted to these pages within promising sites. Notice that, websites may not always include semantically coherent pages. Still, we envision that for most of the sites, the overlap in terms of the content is higher for pages in a particular site than those pages that are not within this site. For instance, the findings in [7] imply that as the degree of overlap among the URLs increase, the coherency in the content (i.e., terms) in the corresponding pages also increase.

There are efficient dynamic pruning techniques such as those based on the quit-continue approach [6] and impact-sorted lists [2]. Both the baseline and the proposed approaches can benefit from such techniques. For instance, the impact-based pruning may be coupled with our site-based pruning, for further improvements in efficiency (i.e., postings for each site in CS-IIS can be sorted with respect to impacts).

5. CONCLUSION

We present a dynamic query pruning technique that eliminates relatively less promising sites (and Web pages) during retrieval. The preliminary results are encouraging in that the top-*K* results returned by the site-based pruning strategy exhibit strong similarity to those of the no-pruning case, while the proposed strategy achieves significant reductions in processing times. The future work involves exploiting URL hierarchy to obtain more coherent groups of Web pages and comparison and combination of the proposed approach with other dynamic pruning approaches.

6. ACKNOWLEDGEMENTS

This work is partially supported by The Scientific and Technical Research Council of Turkey (TÜBİTAK) by the grant # 108E008.

7. REFERENCES

[1] Altingovde, I. S., Demir, E., Can, F., Ulusoy, Ö. Incremental cluster-based retrieval using compressed cluster-skipping inverted files. *ACM TOIS*, in press. Available at http://139.179.21.106/tois.pdf

[2] Anh, V.N., Moffat, A. Simplified similarity scoring using term ranks. In *SIGIR'05*, 226–233, 2005.

[3] Can, F., Altingovde, I.S., Demir, E. Efficiency and effectiveness of query processing in cluster-based retrieval. *Information Systems 29, 8,* 697-71, 2004.

[4] Carmel, D., Cohen, D., Fagin, R., Farchi, E., Herscovici, M., Maarek, Y. S., Soffer, A. Static index pruning for information retrieval systems. In *SIGIR'01*, 43-50, 2001.

[5] Li, W.-S., Candan, K. S., Vu, Q., Agrawal, D. Retrieving and organizing Web pages by "information unit". In *WWW'01*, 230–244, 2001.

[6] Moffat, A., Zobel, J. Self-indexing inverted files for fast text retrieval. *ACM TOIS 14, 4*, 349-379, 1996.

[7] Silvestri, F. Sorting out the document identifier assignment problem. In *ECIR'07*, 101-112, 2007.

[8] Strohman, T., Croft, W. B. Efficient document retrieval in main memory. In *SIGIR'07*, 175-182, 2007.