Recommending Anchor Points in Structure-Preserving Hypertext Document Retrieval

Ben C.M. Kao, Joseph K.W. Lee, David W.L. Cheung, C.Y. Ng
Department of Computer Science,
The University of Hong Kong,
Pokfulam, Hong Kong
Email: {kao | jkwlee | dcheung | cyng}@cs.hku.hk

Abstract

Traditional WWW search engines index and recommend individual Web pages to assist users in locating relevant documents. Users are often overwhelmed by the large answer set recommended by the search engines. The logical starting point of the hyper-document is thus hidden among the large basket of matching pages. Users need to spend a lot of effort browsing through the pages to locate the starting point, a very time consuming process. This paper studies the anchor point indexing problem. The anchor points of a given user query is a small set of key pages from which the larger set of documents that are relevant to the query can be easily reached. The use of anchor points help solve the problems of huge answer set and low precision suffered by most search engines by considering the hyper-link structures of the relevant documents, and by providing a summary view of the result set.

1. Introduction

Although search engines have been proven in practical use as indispensable tools for Web information retrieval, they suffer from a number of drawbacks. The simple keyword matching criteria very often do not provide queries with the expressive power to distinguish the target documents from the millions of pages available on the Web. This results in large answer sets and thus low precision. Users are often overloaded by the myriad pages returned, seriously weakening the usability and the effectiveness of the search engines. Also, if the target document is a hypertext that consists of a number of Web pages being connected by hyper-links, the document structure is destroyed. Individual pages that compose the document are scrambled and are returned out of order. The logical starting page of the hyper-document is thus hidden among the large basket of matching pages. The purpose of this paper is to study the low precision problem of traditional search engines and suggest solutions to relief users from being overloaded by the large numbers of recommended pages. In particular, we consider the hyper-link structures of the Web pages when processing queries to achieve the following goals:

- to improve the ranking of matching documents,
- to identify good starting points (anchors) that allow efficient and orderly accesses to pages that belong to the same logical hypertext documents, and
- to reduce the size of the answer set by recommending only those pages that are representative of the ones which match the query and which belong to the same logical clusters. (For example, if three pages that match a query are inter-connected by hyper-links, only one of the three is recommended to the user.)

The rest of this paper is organized as follows. In Section 2 we discuss some common deficiencies of traditional search engines. In particular, we study the large-size, low-precision answer set problem, and the aftermath of disregarding the hyper-link structures of documents when making a recommendation. In Section 3 we mention some related works. In Section 4 we propose the idea of recommending *anchor points* instead of individual matching pages. Loosely speaking, given a query, the anchor points constitute a *small set* of *key pages* from which the set of matching pages can be accessed easily and in a logical order. The idea is that by presenting to the users a restricted set of "representative" pages, users are able to perform a fast first-level screening of the pages to single out a selected set of good candidates before examining each one of them. Also, if some of the matching pages belong to the same logical hypertext document, the anchor point that connects to them provide a good logical starting point for browsing. We propose a model for matching a query to a set of anchor points. To demonstrate the effectiveness of our approach,

Goal	Query	Search Engine	no. of hits	no. of relevant pages in the first 30 hits	Rank of the 1st relevant hit	no. of logical clusters in the first 30 hits
Site of Microsoft Windows	microsoft windows	Alta Vista	1,675,241	3	10th	4
RIFF specification	riff specification	Lycos	unknown	1	12th	26
Find NBA scoreboard	nba score	Excite	71,118	27	3rd	2
General info about cricket	cricket	Infoseek	40,990	1	22th	20

Table 1. Example queries and results.

we implemented a prototype system for indexing anchor points. In Section 5 we briefly describe our prototype. Finally, we conclude our study in Section 6.

2. Shortcomings of Search Engines

In this section we identify and discuss four sources of ineffectiveness of traditional search engines:

- large answer set,
- low precision,
- unable to preserve the hypertext structures of matching hyper-documents, and
- ineffective for general-concept queries.

Large answer set. Most search engines handle user queries with reasonable response times. The quality of the responses, however, are sometimes questionable. The large numbers of pages indexed plus relatively loose matching criteria often result in large answer sets (sets of matching pages). Users are overloaded with the vast amount of information returned from the search engines. Even though search engines rank the pages in the answer set by guessing how relevant they are with respect to the queries, the ranking systems are far from perfect given the limited expressive power of the keyword-based query interfaces. Browsing through the numerous pages returned is a tiring and time consuming process. Users don't usually have the patience to toil through more than the first thirty hits returned by a search engine. Table 1 illustrates this problem by showing the results obtained from querying three popular search engines with some sample queries.

We observed that it is not unusual that a number of the pages returned are in fact parts of a logical cluster or of a hypertext document. For example, if one submits the query "nba scoreboard november 19 1997" to *Alta Vista*, one would get a whole bunch of matching pages from a sports site, one for each baseketball game that occured on a November day in some year. The last column of Table 1 refers to the number of clusters that the first thirty hits can be grouped into.

From the table we see that the answer sets are huge. If it takes a person 5 seconds to decide whether a page is relevant or not, screening through the 100,000 recommendations takes about 6 man-days. Of course, one would argue that the screening would stop as soon as one good recommendation is found. Still, as suggested by Table 1, the first relevant page may not be found until a couple dozens pages have been examined, many more if one is unlucky. Also, the first relevant page may not be the best page that can be found in the answer set. More screening is required if one would like to compare relevant hits looking for a better match.

The last column of the table suggests that the large numbers of pages can be grouped into a small number of logical clusters. Now, if the search engine could be smart enough to identify the clusters and recommend to the users only one *representative* page per cluster, the users would be able to screen through the suggested list much more efficiently. If these are not the pages the user is looking for, all of them could be skipped by one inspection of a page.

Low precision. In addition to "information overload", low precision of the answer sets is sometimes another concern of the effectiveness of search engines. Previous studies have conducted experiments showing that relevant pages are often interspersed with irrelevant ones in the ranked query outputs [5]. The implication is that users cannot afford to examine only the first few, or any small subset, of the answer set. Table 1 illustrates this problem by showing the number of pages among the first 30 hits that are relevant to a search goal. We see that, for some queries, the numbers are less than honorable.

The problem of low precision has been documented in a number of previous studies. We will mention some of the references in Section 3. While there are many factors that lead to low precision results, we remark that under the simple model of matching Web pages against user queries

based solely on word statistics, the current approach taken by most search engines might already be representing the best effort. We believe that working on a better user interface that assists the users to better express their search goals should be a more fruitful option in precision improvement. One example system developed by The University of Arizona using the idea of concept space [1, 2] has demonstrated that user queries can be semi-automatically enhanced to improve the precision of the answer sets. (*Excite* also takes a similar approach.)

In this paper we do not attempt to *solve* the low precision problem. However, we identify and study one important source of the problem and discuss how the concept of anchor points could help tackle it.

Destroying the hypertext structures of matching hyper-documents. Another inadequacy of traditional search engines again results from not preserving the hypertext structures of matching hyper-documents when making recommendations. Even if an engine could match a query with the pages of a hyper-document, the hypertext structure is flattened and the individual pages are returned out of order. As an example, we submitted the query "C80 faq" to *Alta Vista* looking for information about a DSP chip named C80. *Alta Vista* successfully located a forum with hundreds of postings about the chip. The postings were organized as a hypertext with an index page pointing to the numerous postings, ordered by date. Each posting was contained in a separate Web page. Although the pages returned by the search engine matched the query goal, the postings listed in the answer set were totally out of order, and the index page was not listed in the first 100 hits returned. A user thus needs to reconstruct the logical reading path from the disorganized answer set. A better approach would be to recognize that the pages belong to the same cluster and to return a logical starting point (such as the index page). The concept of anchor point again applies here. We will demonstrate how anchor points help ameliorate the problem in Section 5.

General concept queries. Finally, while traditional search engines perform quite well for "specific" and "precise" queries (e.g., "find me the solution for solving the Rubik's cube), they are not particularly effective in serving general concept queries. As an example, someone may want to know about the sport cricket. Ideally, a site such as *CricInfo* that is dedicated to the sport would be a perfect match. Unfortunately, the query "cricket" is too general for traditional search engines with their simple keyword-matching systems. The result is that any Web page that contains the keyword "cricket" matches the query, be it the start page of *CricInfo*, a page written by a fifth grader on his favorite sports, or a page reporting the scores of an international cricket tournament.

As the Web develops, we see more and more information sources that are dedicated to specific topics of interests. There are sites for *tennis*, sites for *salmon*, and even sites for *Big Foot*. If a user is looking for some general information about a topic, chances are that a site exists on the Web that is specialized on that topic. Recommending Web sites instead of individual Web pages becomes more meaningful to this type of general concept queries. Currently, the best way to look for specialized information sources is to use a directory service, such as *Yahoo!*. The down sides of this approach are that subject categorization is done by hand and that the sites need to be suggested.

As an alternative, one could imagine that the Web pages of a specialized site circle around a major subject, and thus could be considered as parts of a very big cluster or a hypertext document rooted at the site's home page. Also, as we have discussed, an anchor point is a representative page of a cluster and that it is associated with the keywords found in the cluster. It is reasonable to argue that a specialized site's home page is a good anchor point that matches the general concept query (on the site's specialized subject). The concept of anchor points is thus useful in recommending Web sites and answering general concept queries.

3. Related Works

The goal of our study is to improve the effectiveness of information retrieval on the WWW. In particular, we focus on improving the ranking system of search engines dealing with hypertext documents, cutting down on the size of answer sets, supporting general concept queries, and identifying good starting points for efficient and orderly accesses to hypertext documents.

Some research studies take another approach to matching users to information on the Web. Instead of indexing the whole Web like traditional search engines do (a *server-based* approach), these studies work on the design of intelligent *Client-based* Web tools that "learn" about a user's interests and guide the user in traversing the Web, zeroing on the target documents. Interested readers are referred to [3, 10] for more details.

In [6], an intelligent system is designed which tracks users' browsing behavior to deduce sets of keywords (called term vectors). These term vectors are used to describe the information that the users are interested. The paper proposes an architecture of an intelligent system that integrates various tools to analyze the users' accessing behavior and automatically brings in relevant documents for the users. The system consists of two learning agents, one for discovering users' topics of interest, and another for discovering the topics covered by information sources. The system then matches users to Web sites based on the topics accordingly.

The expressive power of conventional search engine query interfaces is relatively weak when restricted to keyword-based search. The deficiencies are documented

in [4]. One of the many problems is that a concept under interest may be described by many terms. For example, someone looking for information about "AIDS" would miss documents that mention only "HIV". One approach to this problem is to construct a *concept space graph* representing the important terms and their weighted relationships. This concept space graph is then used to augment keyword queries by complementing user supplied keywords with their strongly associated terms [1]. Concept space construction involves *frequency analysis* and *cluster analysis* [2]. However, these analyses are very computational expensive and thus are not suitable for a highly dynamic Web environment.

4. Anchor Points

In Section 2 we explained how recommending anchor points can improve the effectiveness of search engines, particularly in dealing with hypertext documents. To reiterate, given a user query and subsequently a set of matching Web pages which form a number of clusters, we propose that the system recommend, instead of all the matching pages, a set of anchors (possibly one for each cluster) such that

1. an anchor is a representative page of a cluster (i.e., by inspecting the anchor, a user can easily deduce what the pages in the corresponding cluster are about); and
2. an anchor provides efficient and orderly accesses to the pages in the corresponding cluster.

In our discussion so far, we have only presented the idea of anchor points fairly informally. For example, we have not defined what we mean by a cluster, what we mean by a page being a representative of a cluster, or what we mean by efficient accesses to a cluster of pages. The reason is that there are, in fact, many different ways one could interpret the terms. In order to avoid restricting ourselves to one interpretation, we focused our previous discussion on the general concept and the motivation of the problem.

In this section we present one formal definition of anchor points. We show how to process queries in anchor point recommedation. We will demonstrate the effectiveness of our approach in solving the commonly occuring problems in traditional search engines (Section 2) via experiments in Section 5.

4.1. Definitions

We define the *distance* from a Web page A to a Web page B, denoted by $D(A, B)$, to be the minimum number of hyper-links that need to be traversed to reach page B starting from page A. If page B is unreachable from A, we have $D(A, B) = \infty$. We define the *k-neighborhood* of a Web page A, denoted by $N_k(A)$, to be the set of all the pages that are within distance k or less from A.

We assume a *scoring function* f exists such that given a keyword a and a Web page X, $f(X, a)$ measures the extent that page X matches the keyword a. We call the value of this function the *score* of page X with respect to a. There are many choices for such a function. One simple example would be:

$$f(X, a) = \begin{cases} 0 & \text{if } X \text{ does not contain } a \\ 1 & \text{if } X \text{ contains } a \end{cases}$$

Alternatively, we could have $f(X, a)$ return the normalized occurrence frequency of a in X, as in done in TFIDF [11]. A third example would be the the probabilistic model as suggested in [11] in which the authors proposed a function that estimates the *probability* that a document X is relevant to a keyword a.

Recall that our goal is to recommend a page to a user starting from which he is likely to find relevant pages via *simple navigation*. Here, let us assume that by simple navigation, we mean that the user does not need to traverse more than k links away from the suggested starting point. In order to measure how well potentially a page X can lead a user to pages that match a keyword a, we define a *potential function* $P_k(X, a)$ as follows:

$$P_k(X, a) = \sum_{Y \in N_k(X)} f(Y, a) \times \alpha^{D(X,Y)}$$

where α is a constant parameter between 0 and 1. (We take $0^0 = 1$.) In words, the potential function of a page X with respect to a keyword a gives the sum of all the scores of the pages in X's k-neighborhood, with each page's contribution to the score scaled down exponentially with respect to its distance from page X. The constant α controls the pace of the scale-down (the smaller the value, the faster the pace).

Depending on the semantics of the function f, different quantitative interpretations can be associated with the potential function. For example, if $f(Y, a)$ represents the *amount* of information that a page Y contains about the keyword a, and α is the probability that a user follows a hyperlink, then $\alpha^{D(X,Y)}$ gives the probability that page Y is visited if a user starts at page X, and $P_k(X, a)$ gives the *expected* amount of information that a user would learn about keyword a if he starts from page X (assuming that the user does not take more than k hops away from X). As another example, if $f(Y, a)$ measures the *probability* that page Y is relevant to keyword a, then $f(Y, a)\alpha^{D(X,Y)}$ is the probability that a user starting from page X would get a relevant page in Y about the keyword a. Hence, $P_k(X, a)$ is equal to the *expected* number of pages that are relevant to a that a user would visit given that he starts at page X. In any case, intuitively, $P_k(X, a)$ is an indicator of how much information about a one can get starting from page X. For the

purpose of discussion, we will use the second interpretation of f (i.e., $f(Y,a)$ measures the probability that page Y is relevant to a) for the rest of this section.

4.2. Queries

Given a query Q and a page X, our approach to evaluate whether X is a good anchor point for Q is by estimating the number of relevant documents (w.r.t. Q) that are within distance k from X and that a user will visit if he starts from X. We call this estimate the *potentials* of X with respect to Q (Potential(X,Q)). Anchor points are ranked based on their potentials. In this subsection we give a formal definition of anchor point and show how their potentials are computed.

Given a query Q, a page X is an anchor point of Q if it satisfies the following conditions:

1. Potential$(X,Q) > 0$,
2. $\nexists Y$ s.t. Potential$(Y,Q) >$ Potential(X,Q) and $X \in N_k(Y)$.

That is, page X has a positive potential with respect to the query and that X is not in any other page's k-neighborhood which has a better potential.

As we have discussed in Section 4.1, if $f(Y,a)$ is a measure of the probability that page Y is relevant to a keyword a, then $P_k(X,a)$ is the expected number of relevant pages (w.r.t. a) in X's k-neighborhood that a user would visit if he starts from X. So, for a single-keyword query $Q = a$, Potential(X,Q) is simply $P_k(X,a)$.

For a multiple-keyword query Q, Potential(X,Q) can also be estimated based on $P_k(X,a_i)$ where a_i's are the keywords in Q. Here, we distinguish two cases: conjunctive queries and disjunctive queries.

To simplify our discussion, let us define $Pr[Q]$ to be that, given a page in X's k-neighborhood that a user visits if he starts from X, the probability that the page is relevant to Q. Since the expected number of pages ($n_k(X)$) in X's k-neighborhood that a user would visit if he starts from page X is simply:

$$n_k(X) = \sum_{Y \in N_k(X)} \alpha^{D(X,Y)},$$

we have $Pr[a] = P_k(X,a)/n_k(X)$ for any keyword a.

Assuming that the occurrences of keywords in a document are independent from each other, we have, given a conjunctive query $Q = a_1 \wedge a_2 \wedge ... \wedge a_m$,

$$\begin{aligned} Pr[Q] &= Pr[a_1 \wedge a_2 \wedge ... \wedge a_m] \\ &= Pr[a_1] \cdot Pr[a_2] ... \cdot Pr[a_m] \\ &= \frac{P_k(X,a_1)}{n_k(X)} \cdot \frac{P_k(X,a_2)}{n_k(X)} ... \cdot \frac{P_k(X,a_m)}{n_k(X)} \\ &= \frac{(\prod_{i=1}^{m} P_k(X,a_i))}{(n_k(X))^m}. \end{aligned}$$

Thus,

$$\begin{aligned} Potential(X,Q) &= Pr[Q] \cdot n_k(X) \\ &= (\prod_{i=1}^{m} P_k(X,a_i))/(n_k(X))^{m-1}. \end{aligned}$$

We remark that in practice the independent assumption may not hold. However, we would like to point out that the Potential function as defined above is used only to rank anchor points. It is not used to compute an accurate estimate of the number of relevant pages. In fact, this independent assumption is used in other information retrieval techniques, such as *Gloss* [7, 8, 9]. A *Gloss* server is one that maintains certain statistics about a number of *information sources* such as document libraries. Given a user query (conjunctive or disjunctive), the *Gloss* server estimates, based on the statistics, which information source is most likely to contain the largest numbers of matching documents. Basically, the server remembers, for each information source and each keyword a, the number of documents in the information source that contain a. *Gloss* uses this keyword statistics and the independent assumption to estimate the expected number of documents that each information source contains that match a query. It is shown that, in practice, such estimation is extremely effective in *ranking* information sources on how likely they contain relevant documents to queries.

The potential of a page with respect to a disjunctive query can be similarly estimated. Given a disjunctive query $Q = a_1 \vee a_2 \vee ... \vee a_m$, by the principal of inclusion and exclusion, we have,

$$\begin{aligned} Pr[Q] &= Pr[a_1 \vee a_2 \vee ... \vee a_m] \\ &= \sum_i Pr[a_i] - \sum_{i_1 \neq i_2} Pr[a_{i_1} \wedge a_{i_2}] + \\ &\quad ... + (-1)^{m-1} \sum_{i_1 \neq i_2 \neq ... \neq i_m} Pr[a_{i_1} \wedge ... \wedge a_{i_m}] \\ &= \sum_i \frac{P_k(X,a_i)}{n_k(X)} - \sum_{i_1 \neq i_2} \frac{P_k(X,a_{i_1})}{n_k(X)} \cdot \frac{P_k(X,a_{i_2})}{n_k(X)} \\ &\quad + ... + (-1)^{m-1} \sum_{i_1 \neq i_2 \neq ... \neq i_m} (\prod_{j=1}^{m} \frac{P_k(X,a_{i_j})}{n_k(X)}). \end{aligned}$$

And Potential(X,Q) is simply equal to the above value multiplied by $n_k(X)$. Therefore, Potential(X,Q) can be estimated by the $P_k(X,a_i)$'s.

5. Prototype

5.1. Building the prototype

We have implemented a prototype to demonstrate the idea of anchor points as discussed in Section 4. The pro-

totype consists of a query processor and a searchable index. The query processor accepts keyword-based queries and returns a set of ranked anchor points for each query. We chose the website of ESPN SportZone for building the index since it divides concepts logically into sub-concepts in a hierarchical manner (e.g. Sport $\rightarrow$ NBA $\rightarrow$ NBA Scoreboard). Traditional search engines ignore this structure and treat each document individually when creating their indexes.

A snapshot of the website was taken on 26 November 1997. Each document is represented by a vector of terms with associated weights. Terms were taken from the documents, while a stop-list [11] is used to eliminate trivial terms. The weight of a term in a document is measured by the term's normalized frequency in the document, and this is chosen as the scoring function f as discussed in Section 4.1. In order to determine the k-neighborhood of every document, we extracted external links in all documents to build an adjacency matrix of the whole site. The k-neighborhood of a document can be derived from this adjacency matrix. The constants k and α were set to 3 and 0.8 respectively in the prototype.

5.2. Experiment Results

We compared the prototype to a traditional search engine — *Alta Vista*. It was chosen because it allowed users to limit their searches over a specific host. We submitted the query "NBA" to our prototype and it returned the index page of the NBA Section under ESPN as the second best anchor point, i.e., http://ESPN.SportsZone.com/nba/index.html. However, when we submitted the same query to *Alta Vista* to search the ESPN network, there were a lot of pages which described NBA game scores on a specific date. The NBA Section index was not even in the first 50 hits. In addition to the NBA Section index, the first nine anchor points returned by our prototype are other NBA related section indexes (e.g., Editor's weekly review). Documents for individual events about NBA were ranked much lower. We have performed a number of similar experiments testing the effectiveness of our prototype. From the results, we conclude that: First, our prototype is successful in recommending good anchor points (starting pages) from which matching documents can be easily accessed. This is in sharp contrast to a traditional search engine which tends to flatten the structures of hypertext documents. Second, since an anchor point gives the essence of a hyper-document, recommending anchor points instead of individual pages vastly cuts down on the size of the answer set without hurting the quality of the result. Consequently, users can screen through the list of matching pages much faster than the traditional approach.

6. Conclusion

We identified four sources of ineffectiveness of traditional search engines and introduced the concept and use of anchor points. Given a user query, the set of anchor points is a set of key pages from which the larger set of documents that are relevant to the query can be easily reached. The use of anchor points help solve the problems of huge answer set and low precision suffered by most search engines. The major improvement is achieved by considering the hyperlink structures of the relevant documents, and by providing a summary view of the result set. We have implemented a prototype based on the concept of anchor point. Comparisons were made to traditional search engines. We found that our approach gave higher ranks to pages (such as indices) that provided better starting points for accessing relevant pages. On the other hand, traditional search engines tend to ignore the logical structure of hyper-documents, and relevant pages are distributed unpredictably in the answer set.

References

[1] H. Chen. Knowledge-based document retrieval: Framework and design. *Journal of Information Science*, 18:293–314, 1992.

[2] H. Chen and K. J. Lynch. Automatic construction of networks of concepts characterizing document databases. *IEEE Transactions on Systems, Man and Cybernetics*, 22(5):885–902, 1992.

[3] R. D. J. De Bra and Post. Information retrieval in the World-Wide Web: Making client-based searching feasible. In *Proceedings of the First International World-Wide Web conference*, Geneva, 1994.

[4] S. Feldman. Just the answers, please: Choosing a Web search service. *The Magazine for Database Professionals*, May 1997.

[5] V. N. Gudivada. Information retrieval on the World-Wide Web. *IEEE Internet Computing*, 1(5):58–68, 1997.

[6] J. K. W. Lee et. al. Intelligent agents for matching information providers and consumers on the World-Wide Web. In *Proceedings of the Thirtieth Annual Hawaii International Conference on System Sciences*, January 1997.

[7] L. Gravano et. al. The efficiency of Gloss for the text database discovery problem. In *ACM SIGMOD'94*, 1994.

[8] L. Gravano et. al. Precision and recall of Gloss estimators for database discovery. In *PDIS'94*, 1994.

[9] L. Gravano et. al. Generalizing Gloss to Vector–Space databases and broker hierarchies. In *VLDB'95*, May 1995.

[10] H. Lieberman. Letizia: An agent that assists Web browsing. In *International Joint conference on Artificial Intelligence*, 1995.

[11] G. Salton. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Mass:Add-Wesley, 1989.