Implicit Link Analysis for Small Web Search

Gui-Rong Xue[1] Hua-Jun Zeng[2] Zheng Chen[2] Wei-Ying Ma[2] Hong-Jiang Zhang[2] Chao-Jun Lu[1]

[1]Computer Science and Engineering
Shanghai Jiao-Tong University
Shanghai 200030, P.R.China
grxue@sjtu.edu.cn, cj-lu@cs.sjtu.edu.cn

[2]Microsoft Research Asia
5F, Sigma Center, 49 Zhichun Road
Beijing 100080, P.R.China
{i-hjzeng, zhengc, wyma, hjzhang}@microsoft.com

ABSTRACT

Current Web search engines generally impose link analysis-based re-ranking on web-page retrieval. However, the same techniques, when applied directly to small web search such as intranet and site search, cannot achieve the same performance because their link structures are different from the global Web. In this paper, we propose an approach to constructing implicit links by mining users' access patterns, and then apply a modified PageRank algorithm to re-rank web-pages for small web search. Our experimental results indicate that the proposed method outperforms content-based method by 16%, explicit link-based PageRank by 20% and DirectHit by 14%, respectively.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - *search process, retrieval models*; H.2.8 [**Database Management**]: Database Applications - *Data mining*

General Terms

Algorithms, Experimentation

Keywords

Information retrieval, web search, link analysis, log mining

1. Introduction

Global search engines such as Google [14] and AltaVista [2] are widely used by people to find information directly on the Web. Nevertheless, to get more specific and up-to-date information, people often go directly to some specific websites and conduct site search. Intranet search is another increasingly important area for an organization to manage and search its own documents. In both cases, search occurs in a closed sub-space of the Web, which is called *small web search* in this paper.

Existing small web search engines generally use the same technologies as those used in global search engines. However, their performances are problematic. As reported in a Forrester survey [15], current site-specific search engines fail to deliver all the relevant content, instead returning too much irrelevant content to meet the user's information needs. In the survey, the search facilities of 50 websites were tested, but none of them received a satisfactory result. Furthermore, in the TREC-8 Small Web Task, little benefit is obtained from the use of link-based methods [16], which also indicates the low performance of exiting search technologies in small web search.

The reason of this low performance lies in several aspects. First, it has been known that ranking by content-based similarity faces difficulties such as shortness and ambiguity of queries. Second, link analysis technologies which have been successful applied to enhance global web search could not be directly applied because the link structure of a small web is different from the global Web. We will explain this difference in detail in Section 3. Third, even though it is understood that users' access logs contain valuable information about web-pages' importance, so far few successful efforts in this direction were made except DirectHit [13].

In this paper, a method is proposed to generate implicit link structure based on user access pattern mining from Web logs. We demonstrate how implicit links could be inferred and extracted. The link analysis algorithms then use these implicit links to compute rank scores of web-pages for improving the search performance. The experimental results reveal that generated implicit links contain more recommendation relationships than explicit links. The search experiments conducted on Berkeley website illustrate that our method outperforms the content-based method by 16%, explicit link-based PageRank by 20% and DirectHit by 14%, respectively.

The rest of this paper is organized as follows. In Section 2, we introduce some related works on small web search and link analysis. In Section 3, we present the basic link structure of a small web, and describe the mining technology to construct implicit link structure and the ranking process. Our experimental results are presented in Section 4. Finally, conclusions and future works are discussed in Section 5.

2. Related Work

Many researches and products have been devoted to small web search. Most of them use the "full text search" technologies which retrieve a large amount of documents containing the same keywords to the query and rank them by keyword-similarity. For example, AltaVista provide a content-based site search engine [1]; Berkeley's Cha-Cha search engine organizes the search results into some categories to reflect the underlying intranet structure [9]; and the navigation system by M. Levence et al. [23] returns a sequence of linked pages to help the end-user to browse the search results. These systems do not use link analysis in re-ranking the search results, thus there is still head room for improving the search precision.

SIGIR'03, July 28–August 1, 2003, Toronto, Canada.

Link analysis technology has been widely used to analyze the pages' importance, such as HITS [20] and PageRank [27][6]. In both algorithms, the Web is represented a directed graph G={V, E}, where V stands for web-pages w_i, and E stands for the hyperlinks $l_{i,j}$ within two pages. In HITS algorithm, each web-page w_i has both a hub score h_i and an authority score a_i. The hub score of w_i is the sum of all the authority scores of pages that are pointed by w_i; the authority score of w_i is the sum of all the hub scores of pages that point to w_i, as shown in the following equation.

$$a_i = \sum_{j:l_{j,i}\in E} h_j , \qquad h_i = \sum_{j:l_{i,j}\in E} a_j \tag{1}$$

The final authority and hub scores of every web-page could be obtained through an iterative update process.

PageRank is a core algorithm of Google [14] which measures the importance of web-pages. It uses the whole linkage graph of the Web to compute universal query-independent rank value for each page. PageRank algorithm models the users' browsing model as a random surfing model, which assumes that a user either follow a link from the current page or jump to a random page in the graph. The PageRank of a page w_i is then computed by the following equation:

$$PR(w_i) = \frac{\varepsilon}{n} + (1-\varepsilon) \times \sum_{l_{j,i}\in E} PR(w_j) / \text{outdegree}(w_j) \tag{2}$$

where ε is a dampening factor, which is usually set between 0.1 and 0.2 [18]; n is the number of nodes in G; and out-degree(w_j) is the number of the edges leaving page w_j, i.e., the number of hyperlinks on page w_j. The PageRank could be computed by an iterative algorithm and corresponds to the primary eigenvector of a matrix derived from adjacency matrix of the available portion of the Web.

Since direct application of link analysis in a small web could not exhibit good result, some systems change their focuses to users' usage information. For example, DirectHit [13] is one of the representative examples, which harnesses millions of human decisions by millions of daily Internet searchers to provide more relevant and better organized search results. DirectHit's site ranking system, which is based on the concepts of "click popularity" and "stickiness," is currently used by Lycos, Hotbot, MSN, Infospace, About.com and about 20 other search engines. The underlying assumption is that the more a web-page is visited, the higher it is ranked according to particular queries. Although this method is intuitive, it also has some restrictions. One of the problems is that it needs a large amount of user logs and only works for some popular queries. Another problem is that it is easy to fall into a quick positive feedback loop when access to a popular resource leads to its higher rank, which in turn leads to an even higher number accesses to it.

There are also some works on utilizing the usage data in link analysis. For example, Miller et al. [25] propose a method to use the usage data to modify the adjacency matrix in Kleinberg's HITS algorithm (which is called modified-HITS). It replaces the adjacency matrix M with a link matrix M', which assign the weight between nodes (pages) based on users' usage data collected from web-server logs. This method is somewhat similar to our proposed solution, but there are some problems with it. First, this method do not separate the user logs into sessions based on their tasks, thus it is inevitably that many noise data will be introduced into the link matrix. Moreover, Web users often follow different paths to reach a same goal. If only adjacent pages are treated as related, the underlying relationship could not be discovered.

3. Link Analysis for Small Web

Link analysis algorithms such as PageRank [27] and HITS [20] use eigenvector calculations to identify authoritative pages based on hyperlink structures. The intuition is that a page with high in-degree is highly recommended, and should have a high rank score. However, there is a basic assumption underlying those link analysis algorithms: the whole Web is a citation graph (see the left plot in Figure 1), and each hyperlink represents a citation or recommendation relationship. Formally, there is the following *recommendation assumption* [18]: a hyperlink in page X pointed to page Y stands for the recommendation of page Y by the author of page X.

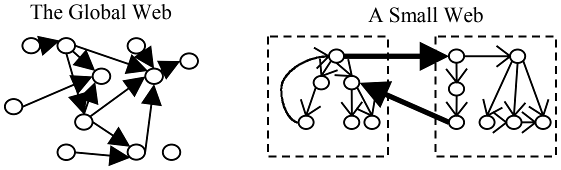

Figure 1: Link structure of the global Web and a small web.

For the global Web, the recommendation assumption is generally correct because hyperlinks encode a considerable amount of authors' judgment. Of course some hyperlinks are created not for the recommendation purpose, but their influence could be filtered or reduced to an ignorable level.

However, the recommendation assumption is commonly invalid in the case of a small web. As depicted in the right part of Figure 1, the majority of hyperlinks in a small web are more "regular" than that in the global Web. Most links are from a parent node to children nodes, between sibling nodes, or from leaves to the root (e.g. "Back to Home"). The reason is primarily because hyperlinks in a small web are created by a small number of authors; and the purpose of the hyperlinks is usually to organize the content into a hierarchical or linear structure. Thus the in-degree measure does not reflect the authority of pages, making the existing link analysis algorithms not suitable for small web search.

In a small web, hyperlinks could be divided into *navigational links* and *recommendation links*. The latter is useful for link analysis to enhance search. However, only filtering out navigational links from all hyperlinks is inadequate for our purpose because the remaining recommendation links are incomplete. In other words, there are many *implicit recommendation links* (hereafter called "implicit links" for short) in a small web that could be discovered by mining user access pattern. In the following, we will discuss the method for extracting implicit recommendation links in a small web.

3.1 Implicit Links Construction

Generally a web can be modeled as a directed graph G = (V, E) where V = {w_i | 1≤i≤n} is the set of vertices representing all the pages in the web, and E encompasses the set of links between the pages. $l_{i,j}\in E$ is used to denote that there exists a link between the page w_i and w_j.

We propose to construct a new *implicit link* graph instead of the original *explicit link* graph in a small web. This new graph is a weighted directed graph $G' = (V, E')$, where V is same as above but E' encompasses the implicit links between pages. Furthermore, each implicit link $l_{i,j} \in E'$ is associated with a new parameter $P(w_j|w_i)$ denoting the conditional probability of the page w_j to be visited given current page w_i.

Our goal is to extract implicit links E' by analyzing the observed users' browsing behaviors. The idea is that we can assume E' controls how the user traverses in the small web. Based on the implicit link graph G' and explicit link graph G, we can assume there exists a generative model for user log. The entire user log consists of a number of browsing sessions $S = \{s_1, s_2, s_3, \ldots)$. Each session is generated by the following steps:

(1) Randomly select a page w_i from V as the starting point;

(2) Generate an implicit path (w_i, w_j, w_k, …) according to the implicit links E' and the associated probabilities, where we assume each selection of implicit link is independent on previous selections;

(3) For each pair of adjacent pages w_i and w_j in the implicit path, randomly select a set of in-between pages $w_{x1}, w_{x2}, \ldots, w_{xm}$ according to the explicit links E to form an explicit path (w_i, $w_{x1}, w_{x2}, \ldots, w_{xm}, w_j$).

In other words, this model control the generation of the user log based on implicit links and explicit links. The final user log contains abundant information of all implicit links. Thus we can extract implicit links by analyzing the observed explicit paths in the user log.

We use a simple two-item sequential pattern mining method to discover possible implicit links. This method uses a gliding window to move over each explicit path, generating all the ordered pairs and counting the occurrence of each distinct pair. The gliding window size represents the maximum interval a user clicks between the source page and the target page. For example, for an explicit path (w_{i1}, w_{i2}, w_{i3}, …, w_{ik}), our method generates pairs ($i1$, $i2$), ($i1$,$i3$), …, ($i1$, ik), ($i2$,$i3$), …, ($i2$, ik), … If one of the pairs, e.g. (i, j), corresponds to an implicit link ($l_{i,j} \in E'$), paths of the pattern (w_i, …, w_j) should occur frequently in the log, with different in-between pages.

All possible ordered pairs and their frequency are calculated from all the browsing sessions S, and infrequent occurrences are filtered by a minimum support threshold. Precisely, the support of an item i, denoted as *supp*(i), refers to the percentage of the sessions that contain the item i. The support of a two-item pair (i, j), denoted *supp*(i, j), is defined in a similar way. A two-item ordered pair is frequent if its support *supp*(i, j)≥ *min-supp*, where *min_supp* is a user specified number.

After two-item sequential patterns are generated, they are used to update the implicit link graph $G' = (V, E')$ described previously. All the weights of edges in E' are initialized to zero. For each two-item sequential pattern (i, j), we add its support *supp*(i, j) to the weight of the edge $l_{i,j}$. All the weights are normalized to represent the real probability. The resulting graph is used in a subsequent link analysis algorithm.

It is clear that numerous redundant pairs other than real implicit links may also be included because the user's browsing has to follow the explicit links. However, based on our statistical analysis, this effect is small. Specifically, if the connectivity in a small web is high and the users have no significant bias in selecting paths, implicit links could be separated from explicit links by setting an appropriate minimum support. For the detail of the analysis, please refer to the Appendix A.

3.2 Applying PageRank to Implicit Links

After obtaining the implicit link structure, we apply a link analysis algorithm similar to PageRank [27] to re-rank the web-pages for a small web search. We construct a matrix to describe the graph. In particular, assume the graph contains n pages. The $n \times n$ adjacency matrix is denoted by $\mathbf{A}$ and the entries $A[i, j]$ is defined to be the weight of the implicit links $l_{i,j}$.

The adjacency matrix is used to compute the rank score of each page. In an "ideal" form, the rank score PR_i of page w_i is evaluated by a function on the rank scores of all the pages that point to page w_i :

$$PR_i = \sum_{j:l_{ji} \in E} PR_j \cdot A[j,i] \qquad (3)$$

This recursive definition gives each page a fraction of the rank of each page pointing to it—inversely weighted by the strength of the links of that page. The above equation can be written in the form of matrix as:

$$\overrightarrow{PR} = \boldsymbol{A}\overrightarrow{PR} \qquad (4)$$

However, in practice, many pages have no in-links (or the weight of them is 0), and the eigenvector of the above equation is mostly zero. Therefore, the basic model is modified to obtain an "actual model" using *random walk*. Upon browsing a web-page, with the probability 1-ε, a user randomly chooses one of the links on the current page and jumps to the page it links to; and with the probability ε, the user "reset" by jumping to a web-page picked uniformly and at random from the collection. Therefore the ranking formula is modified to the following form:

$$PR_i = \frac{\varepsilon}{n} + (1-\varepsilon) \sum_{j:l_{j,i} \in E} PR_j \cdot A[j,i] \qquad (5)$$

Or, in the matrix form:

$$\overrightarrow{PR} = \frac{\varepsilon}{n}\vec{e} + (1-\varepsilon)\boldsymbol{A}\overrightarrow{PR} \qquad (6)$$

where $\vec{e}$ is the vector of all 1's, and ε (0<ε<1) is a parameter. In our experiment, we set ε to 0.15. Instead of computing an eigenvector, the simplest iterative method—Jacobi iteration is used to resolve the equation.

The re-ranking mechanism is based on two types of linear combination: the score based re-ranking and the order based re-ranking. The score based re-ranking uses a linear combination of content-based similarity score and the PageRank value of all web-pages:

$$Score(w) = \alpha Sim + (1-\alpha)\, PR \quad (\alpha \in [0, 1]) \qquad (7)$$

where *Sim* is the content-based similarity between web-pages and query words, and *PR* is the PageRank value.

The order based re-ranking is based on the rank orders of the web-pages. Here we use a linear combination of pages' positions in two lists in which one list is sorted by similarity scores the other list is sorted by PageRank values. That is,

$$Score(w) = \alpha O_{Sim} + (1-\alpha) O_{PR} \quad (\alpha \in [0, 1]) \tag{8}$$

where O_{Sim} and O_{PR} are positions of page w in similarity score list and PageRank list, respectively.

Our experimental result indicated that the order-based re-ranking performs better than the score-based re-ranking.

4. Experiments

In this section, we discuss the experimental data set, our evaluation metrics, and the result of our study based on those metrics.

4.1 Implicit Link Generation

The goal of our work is to improve small web search by analyzing the user access log in the web site. To conduct the experiments, the website at http://www.cs.berkeley.edu/logs/ with 4-month click-thru logs was used. Before mining for the user access patterns on this log, we preprocess the log by performing data cleaning, session identification and consecutive repetitions elimination. Data cleaning is done by simply filtering out the access entries for embedded objects such as images and scripts. Afterward, users are distinguished by their IP address, i.e. we assume that consecutive accesses from the same IP during a certain time interval are from a same user.

In order to handle the case of multi-users with the same IP, IP addresses whose page hits count exceeds some threshold are removed. The consecutive entries are then grouped into a browsing session. Different grouping criteria have been modeled and compared in [10]. In this paper, we choose the "overtime" criterion similar to [10], i.e., a new session starts when the duration of the whole group of traversals exceeds a time threshold. Consecutive repetitions within a session are then eliminated, e.g., session (*A, A, B, C, C, C*) is compacted to (*A, B, C*). After preprocessing, the log contains about 300,000 transactions, 170,000 pages and 60,000 users.

The original web-pages and link structure is downloaded from the website. About 170,000 pages are downloaded and indexed by the Okapi system [30]. For each page, an HTML parser is applied to removing tags and extracting links in pages. Finally, we got 216,748 hyperlinks in total.

We fixed several parameters for the rest experiments. i.e. window size as 4, minimum support threshold as 7, using support-weighted adjacent matrix, and order-based re-ranking for search. These parameters are determined based on an extensive experiment which will be discussed in Section 4.3. Our algorithm is compared with several state-of-the-art algorithms including full text search, explicit link-based PageRank, DirectHit, and modified-HITS algorithm [25].

After two-item sequential pattern mining, 336,812 implicit links are generated. Figure 2 shows there are 22,122 links that are both in the explicit links and the generated implicit links. That is, 11% of the links are overlapped, which is small.

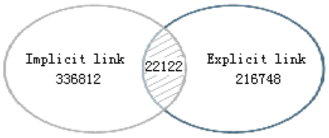

Figure 2: Overlap of implicit links and explicit links.

Some evidences should be given to prove that the implicit links satisfy the recommendation assumption. We develop a prediction model by using implicit links, which is similar to [32]. This model predicts whether a page will be visited by a user in the next step. The prediction accuracy directly reflects the correctness of the page recommendation. We take 4/5 of the log data as the training data to create implicit links and 1/5 of the data as testing data. The following precision is used as our evaluation metrics.

$$\text{Prediction precision}=\frac{P^+}{(P^+ + P^-)} \tag{9}$$

where P^+ and P^- are the numbers of correct and incorrect predictions, respectively.

As stated previously in Section 3.2, the implicit links are generated by the restriction of the minimum support. The higher the support of the implicit link, the higher the probability of the linked pages accessed at the same session. As shown in Figure 3, the prediction precision monotonously increases as the minimum support increases. It indicates that our implicit link is accurate and reflects user's behaviors and interests.

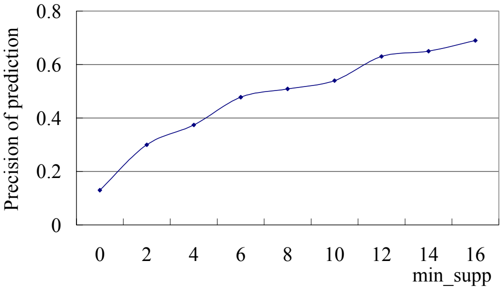

Figure 3: Precision of page prediction by implicit links.

The quality of implicit links is evaluated from human perspective. We randomly select three subsets which contain 375 implicit links in total. Seven volunteer graduated students who are familiar with the subjects of the pages are chosen as our evaluation subjects. They are asked to evaluate whether the implicit links are recommendation links according to the content of the pages. As shown in the upper part of Table 1, about 67% of implicit links in average are recommendation links. Another three subsets selected from explicit links is shown in the lower part of Table 1, where the average recommendation link ratio is just about 39%.

Table 1: Recommendation links in implicit and explicit links.

Subset	Implicit link	Recomm. link	Ratio
1	128	87	0.68
2	114	82	0.72
3	133	84	0.63
Average			**0.67**
Subset	Explicit link	Recomm. link	Ratio
1	107	47	0.44
2	84	26	0.31
3	99	42	0.42
Average			**0.39**

Several examples of these implicit links are shown in Table 2. For example, the fourth implicit link "Xuanlong's course: CS188" → "Wilensky's course: CS188" represents the same course taught by different instructors. When the user visited the page "Xuanlong's course: CS188", page "Wilensky's course: CS188" could be recommended. Table 2 also shows that parts of the implicit links are overlapped with the explicit links, which are created by the author and satisfy the recommendation assumption.

Table 2: Examples of the implicit links.

#	Source Page	Target Page	Explanation	Exp. Link?
1.	Book: Artificial Intelligence: A Modern Approach	The book's slide	A book and its slides	Y
2.	Jordan's Homepage	Andrew Ng's Homepage	Teacher and student	Y
3.	Various pictures	Landscape photographs	Picture	N
4.	Xuanlong's course: CS188	Wilensky's course: CS188	Same course	N
5.	Anthony Joseph's Homepage	Brian Harvey's HomePage	People in Vision group	N
6.	AI on the Web	Machine learning software	Machine learning	N
7.	Sequin's course: CS284	Sequin's course: CS285	Course of same person	N

4.2 Search Result

Because our system only re-ranks the results of the full text search engines, the global search precision is not changed. However, the precision of top matches is improved. Given a query *Q*, let *R* be the set of the relevant pages to the query and |*R*| be the number of pages. Assume the system generates a result set, we only take the top 30 from the result set as *A*. The precision of search is defined as:

$$Precision = \frac{|R \cap A|}{|A|} \tag{10}$$

In order to evaluate our method effectively, we propose a new evaluation methodology: the degree of authority. Given a query, we ask the seven volunteers to identify the top 10 authoritative pages according to a human perspective ranking of all the results. The set of 10 authoritative web-pages is denoted by *M* and the set of top 10 results returned by search engines is denoted by *N*.

$$Authority = \frac{|M \cap N|}{|M|} \tag{11}$$

The precision measures the degree to which the algorithm produces an accurate result; while the authority measures the ability of the algorithm to produce the pages that are most likely to be visited by the user or the authority measurement is more relevant to user's satisfactory degree on the performance of small web search engine.

The volunteers were asked to evaluate both precision and authority of search results for the selected 10 queries (which are *Jordan, Vision, Artificial Intelligence, Bayesian Network, wireless Network, Security, Reinforcement, HCI, Data Mining,* and *CS188*). The final relevance judgment for each document is decided by majority votes. Figure 4 shows the comparison of our approach with full text search, PageRank, DirectHit and modified-HITS algorithms. Here *iPR*, *ePR*, *mH* and *DH* correspond to implicit link-based PageRank, explicit link-based PageRank, modified-HITS, and DirectHit, respectively. The right-most label "Avg" stands for the average value for the 10 queries. As can be seen from Figure 4, our algorithm outperforms the other 4 algorithms. The average improvement of precision over the full text is 16%, PageRank 20%, DirectHit 14% and modified-HITS 12%. While the average improvement of authority over the full text is 24%, PageRank 26%, DirectHit 15% and modified-HITS 14%.

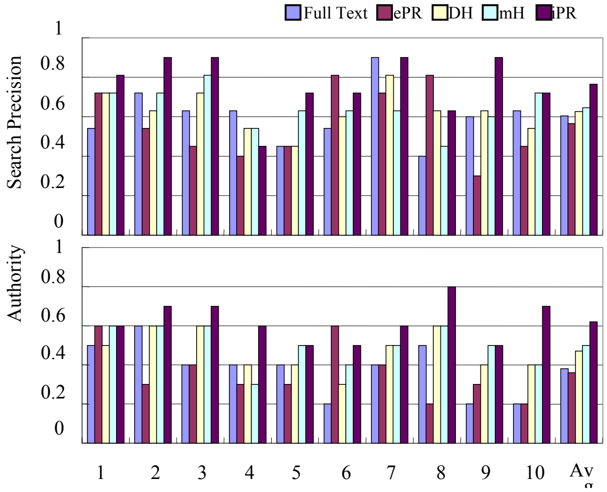

Figure 4: Precision and authority of the different ranking methods.

From Figure 4, we found that the performance of explicit link-based PageRank is even worse than that of the full text search technique, demonstrating the unreliability of explicit link structure of this website.

In Figure 4, DirectHit has a medium performance in all the algorithms. DirectHit outperforms full text search because it takes usage information into account. However, DirectHit could not reveal the real authoritativeness of web-pages. The experiment also shows that DirectHit only improves a part of popular queries' precision. So the average precision is not as good as our algorithm.

Table 3: Ranks of query "vision" in different method.

Web-page Descriptions	*iPR*	*ePR*	*mH*	*DH*
UC Berkeley Computer Vision Group	1	41	2	8
David Forsyth's Book: Computer Vision	2	94	1	4
David Forsyth's Book: Computer Vision(3rd Draft)	3	9	3	10
A workshop on Vision and Graphics	4	44	20	1
UC Berkeley Computer Vision Group	5	2	13	7
CS 280 Home Page	6	14	10	11
Thomas Leung's Publications	7	55	4	31
Jitendra Malik's Brief Biography	8	17	7	6
An overview of Grouping and Perceptual Organization	9	5	21	5
David Forsyth's Homepage	10	87	29	35
A paper of Phil	13	1	6	9
Kim' ZuWhan resume	18	3	5	2
A slide of Landay's talk about Notepals	37	4	33	14
John A. Haddon's publication	39	23	41	13
A slide of Landay's talk about Notepals	41	6	42	18
Chris Bregler's Publications	44	27	8	42
Course: Appearance Models for Computer Graphics and Vision	51	63	47	3
Reference of Object Recognition	62	59	9	17

The modified-HITS algorithm achieves higher performance than full text search, DirectHit and explicit link-based PageRank. In fact, this algorithm is a special case of our proposed algorithm when we set the minimum support threshold to 0 and window size to 1. As we mentioned earlier, when the minimum support threshold is 0, a lot of noise data will be created; and when the window size is 1, we will miss many useful links which will also affect the performance.

Table 3 shows the top 10 pages for the query "vision." We also found that the results from implicit link-based PageRank are more authoritative than that from the modified-HITS. To give an example, "ANSI Common Lisp" is a page ranked high by explicit link-based PageRank because contains numerous out-links and in-links. But according the user logs, this page is rarely.

The convergence curve of several algorithms is shown in Figure 5. The gap represents the difference of the sum of pages' scores from previous iteration. In this figure, the difference of PageRank values between consecutive iterations drops significantly after 7 iterations and shows a strong tendency toward zero. This shows the convergence of our algorithm in a practical way.

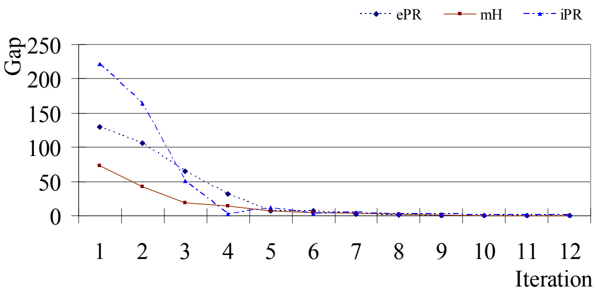

Figure 5: Convergence of different ranking models.

4.3 Parameter Selection

As we mentioned, several parameters are used in the experiments, such as window size = 4, minimum support threshold = 7, using support-weighted adjacent matrix, and using order-based re-ranking. Here we provide experiments for setting those parameters.

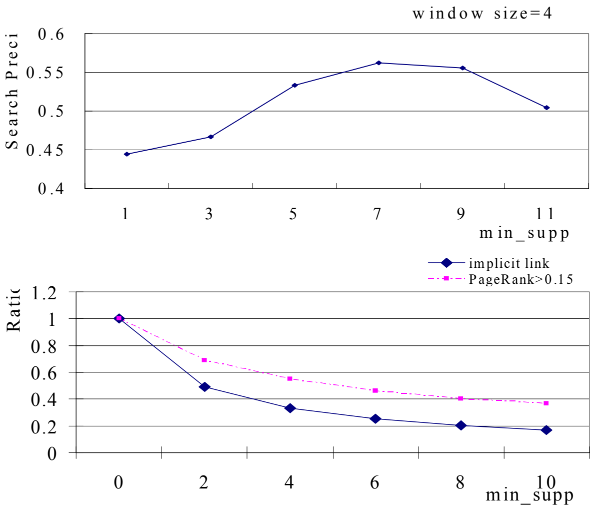

Figure 6: Search precision and implicit link number with different *min_supp*.

In order to choose the most suitable support threshold for mining user access pattern, we asked the seven volunteers to test on 5 queries for each support. The 5 queries are *Machine Leaning*, *Web Mining*, *Graphics*, *OOP*, and *Database Concurrency Control*). Then we compute average precision of the top 30 documents. As shown in upper graph of Figure 6, the system achieved the best search precision when the minimum support is 7. It also shows that the system performance dramatically drops while it is less than 4 or higher than 10. From our observations, the reason can be explained as follows. First, when the minimum support is too small, user's random behaviors are counted and the number of the implicit links is large (as shown in bottom graph of Figure 6, where the ratio denote the proportion of link number of current min_supp to the total link number), which will introduce more irrelevant links to affect the ranking results. Second, high support results in the missing of some potentially important but infrequent implicit links. The bottom graph of Figure 6 also shows that, by increasing the support value, the number of implicit links decreases, leading to the decrease of the number of pages whose PageRank is larger than 0.15(ε=0.15). When the *min_supp* is large, the impact of the PageRank on the search result is very weak.

The second experiment is to test the impact of the window size. Figure 7 shows the impact of different window size on search precision. The evaluation method is same as above. From Figure 7, we found that the precision increases when the window size changes from 1 to 4, which proves the analysis of Berkhin et al. [4] that a user may click several times to get what he wanted. On the other hand, by analyzing the effect of window size on the number of implicit links, we found that more noisy implicit links are created if the window size is large. So if we continue to increase the window size, the performance may decrease. Furthermore, we calculate the interval distribution for our mined implicit links. As shown in Figure 8, about 13.7% of the implicit link is accessed in one step, 26% in two steps, 24% in three steps, and so on.

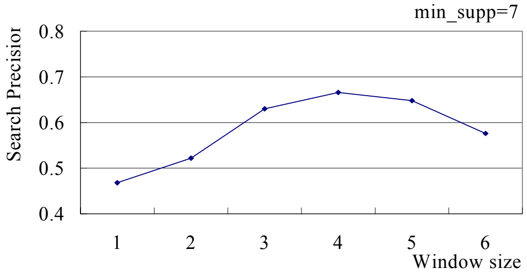

Figure 7: Precision of different window size.

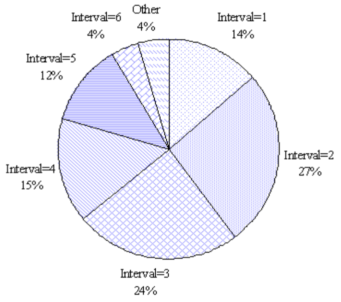

Figure 8: Interval distribution of implicit links.

When constructing the adjacent matrix, there are two choices to set up the weight of the matrix: the weight with 0 or 1 (called 0-1 weighted) or the weight with the support of the implicit link (called support-weighted). We asked the 7 volunteers to test on 10 queries (which are *Machine Leaning, Web Mining, Graphics, OOP, Database Concurrency Control, Classification, Titanium, Distributed Database System, Parallel Algorithm,* and *Mobile*), and evaluated the average precision for the top 30 documents. As can be seen from Figure 9, the support-weighted method achieves better search precision compared to 0-1 weighted method in average. The improvement may be due to the fact that the support-weighted method has stronger recommendation than the 0-1 weighted.

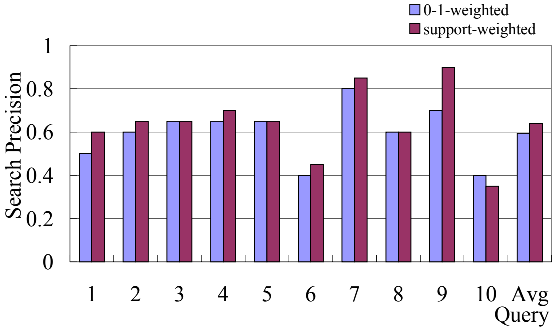

Figure 9: Precision of different weighting methods.

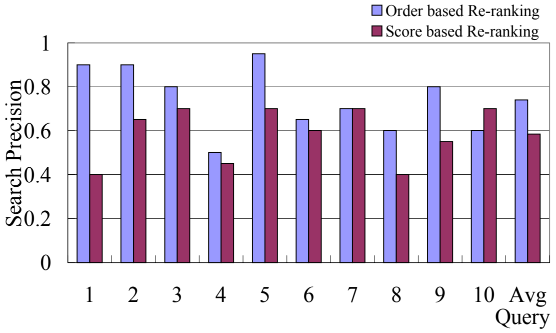

Figure 10: Precision of different re-ranking mechanisms.

The last experiment is to measure the different re-ranking mechanisms, i.e. the score-based re-ranking and the order-based re-ranking. We also asked the 7 volunteers to test on the same 10 queries as above and calculated the average precision for the top 30 documents. As can be seen in Figure 10, the order-based re-ranking outperforms the score-based re-ranking. In our experiments, we noticed that few of results have high similarity score and PageRank score and there is little difference between similarity and PageRank scores among search results Therefor a linear combination of the similarity score cannot achieve good results.

5. Conclusion and Future Work

Conventional link analysis algorithms such as PageRank do not work well when directly applied to analyze the link structure in a small web such as a web site or an intranet. This is because the recommendation assumption for hyperlinks is commonly invalid in a small web. In this paper, we propose a method to construct implicit links by mining users' access patterns, and then apply a modified PageRank algorithm to re-rank search result. The experimental results showed that our method can effectively improve the search performance.

Since PageRank is an algorithm which is query independent, in the future we plan to improve our algorithm so that it can provide topic sensitive ranking. Another possible direction is to use our mined implicit links to improve web-pages clustering accuracy. Existing solutions on clustering web-pages are based on the content and explicit links of web-pages. As we noted earlier, the explicit links in a specific website is only for content organization, so it is difficult to achieve good clustering result by this kind of links. We plan to combine our mined implicit links and the contents of web-pages together to cluster web-pages.

Furthermore, to improve the design of website, we may discover the gap between the designer's expectation and the visitor's behavior by comparing the importances of web-pages calculated from explicit links and implicit links, and then suggest modifications to the site structure to make it more effective for browsing and navigation.

6. References

[1] Agrawal R. and Srikant R. Mining sequential patterns. In Proc. of ICDE'95, 3-14, Taiwan, March 1995.

[2] AltaVista. http://www.altavista.com.

[3] Baeza-Yates R. and Ribeiro-Neto B. *Modern Information Retrieval*. Addison-Wesley, 1999.

[4] Berkhin P., Becher J. D. and Randall D. J. Interactive path analysis of web site traffic. In Proc. of the 7th SIGKDD, 414-419, San Francisco, California, 2001.

[5] Borodin A., Roberts G. O., Rosenthal J. S., and Tsaparas P. Finding authorities and hubs from link structures on the World Wide Web. In Proc. of WWW10, 415-429, Hong Kong, May 2001.

[6] Brin S. and Page L. The anatomy of a large-scale hypertextual web search engine. In Proc. of WWW7, 107-117, Brisbane, Australia, April 1998.

[7] Broder A., Kumar R., Maghoul F., Raghavan P., Rajagopalan S., Stata R., Tomkins A., and Wiener J. Graph structure in the Web. In Proc. of WWW9, 309-320, Amsterdam, May 2000.

[8] Chakrabarti S., Dom B. E., Gibson D., Kleinberg J., Kunar R., Raghavan P., Rajagopalan S., and Tomkins A. Mining the link structure of the World Wide Web, IEEE Computer, 32(8):60-67, August 1999.

[9] Chen M., Hearst M., Hong J. and Lin J. Cha-Cha: a system for organizing intranet search results. In Proceedings of the 2nd USITS, Boulder, CO, October 1999.

[10] Cooley R., Mobasher B. and Srivastava J. Data preparation for mining World Wide Web browsing patterns. Knowledge and Information Systems, 1(1):5-32, 1999.

[11] Cooley R., Tan P.-N. and Srivastava J. Discovery of interesting usage patterns from web data. In Proc. of WEBKDD'99, 163-182, August 1999.

[12] Craswell N., Hawking D., and Robertson S. Effective site finding using link anchor information. In Proc. of SIGIR'01, 250-257, September 2001.

[13] DirectHit. http://www.directhit.com.

[14] Google. http://www.google.com.

[15] Hagen P., Manning H. and Paul Y. Must search stink? The Forrester report, Forrester, June 2000.

[16] Hawking D., Voorhees E., Bailey P. and Craswell N. Overview of TREC-8 web track. In Proc. of TREC-8, 131-150, Gaithersburg MD, November 1999.

[17] Hearst M. Next generation web search: setting our sites. IEEE Data Engineering Bulletin, 23(3):38-48, September 2000.

[18] Henzinger M. R. Link analysis in web information retrieval. IEEE Data Engineering Bulletin, 23(3):3-8, September 2000.

[19] Kessler M. M. Bibliographic coupling between scientific papers. American Documentation, 14(1):10-25, 1963.

[20] Kleinberg J. M. Authoritative sources in a hyperlinked environment. Journal of the ACM, 46(5):604-632, 1999.

[21] Kleinberg J. M., Kumar R., Raghavan P., Rajagopalan S. and Tomkins A. S. The Web as a graph: measurements, models, and methods. In Proc. of COCOON'99, 26-28, Tokyo, 1999.

[22] Levene M. and Loizou G. Web interaction and the navigation problem in hypertext. Encyclopedia of Microcomputers, 28(7):381-398, Marcel Dekker, NY, 2001.

[23] Levene M. and Wheeldon R. A web site navigation engine. In Proc. of WWW10, Hong Kong, May 2001.

[24] Mannila H., Toivonen H., and Verkamo A. I. Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1(3):259-289, November 1997.

[25] Miller J. C., Rae G. and Schaefer F. Modifications of Kleinberg's HITS algorithm using matrix exponentiation and web log records. In Proc. of SIGIR'01, 444-445, New Orleans, September 2001.

[26] Nakayama T., Kato H. and Yamane Y. Discovering the gap between web site designers' expectations and users' behavior. In Proc. of WWW9, Amsterdam, May 2000.

[27] Page L., Brin S., Motwani R. and Winograd T. The PageRank citation ranking: bringing order to the Web. Technical report, Stanford University Database Group, 1998.

[28] Pei J., Han J., Mortazavi-asl B., and Zhu H. Mining access patterns efficiently from web logs. In Proc. of PAKDD'00, 396-407, Kyoto, April 2000.

[29] Raghavan P. Social networks: from the Web to the enterprise. IEEE Internet Computing, 6(1):91-94, Feburary 2002.

[30] Robertson S. E., Walker S., Beaulieu M. M., and Gatford M., Payne A. 1995. Okapi at TREC-4. In Proc. of TREC-4, 73-96, NIST Special Publication 500-236, October 1996.

[31] Srikant R. and Yang Y. Mining web logs to improve website organization. In Proc. of WWW10, Hong Kong, May 2001.

[32] Yang Q., Zhang H. H. and Li T. Mining web logs for prediction models in WWW caching and prefetching. In Proc. of KDD'01, 473-478, August 2001.

Appendix A

To get the insight into how the redundant pairs in Section 4 affect the mining result, we conduct the following probabilistic analysis.

For the simplicity of explanation, we assume the explicit graph G is a completely connected graph, i.e. every page has hyperlinks to all other pages. Thus the number of implicit links is far less than the number of explicit links, i.e. $|E'|<<|E|$. Furthermore, we assume each web-page occurs only once in an explicit path, i.e., users never visit a same web-page in a session. Elaborating the explicit path generation of the step (3) in Section 4, for an adjacent pair (w_i, w_j) in the implicit path, we start from the page w_i and select web-pages one by one according to the following random process.

(a) Select the target page w_j with probability p ($0<p<1$); select another page $w_x \neq w_j$, w_i with probability $1\text{-}p$.

(b) If we have arrived at w_j, stop; else go to (a)

For explicit paths of different length, the probabilities could be easily calculated as in Table 1, where w_{x1}, w_{x2}, …≠ w_i, w_j and w_{x1}, w_{x2}, … are different from each other.

Table 4: Probabilities of explicit paths according to an implicit link (*i*, *j*), given current page *w*$_i$.

Paths	Probability
$w_i \rightarrow w_j$	p
$w_i \rightarrow w_{x1} \rightarrow w_j$	$(1\text{-}p)p$
$w_i \rightarrow w_{x1} \rightarrow w_{x2} \rightarrow w_j$	$(1\text{-}p)^2p$
…	…

Therefore, the probability that an arbitrary explicit link $l_{x,y}$ is included in the path is about $[(1\text{-}p)^2p+(1\text{-}p)^3p+(1\text{-}p)^4p+\ldots]/n^2 = (1\text{-}p)^2/n^2$ where n=$|V|$ and n^2 is the number of distinct pairs (n>>2). This probability is calculated given current implicit link $l_{i,j}$ whose probability is $P(w_j|w_i)P(w_i)$, thus the probability of having a path containing w_i, w_x, w_y, and w_i in this order is $P(w_j|w_i)P(w_i)(1\text{-}p)^2/n^2$. Intuitively, this joint probability is the contribution of implicit link $l_{i,j}$ to the probability of explicit link $l_{x,y}$.

For the same explicit link $l_{x,y}$, we could sum up contributions of all the implicit links, and get the total probability of this explicit link $P(w_y|w_x)P(w_x) \approx (1\text{-}p)^2/n^2$. Here we ignore the contribution of implicit links with one end being w_x or w_y because $|E'|<<|E|$. Compared to explicit link probability, the average probability of an implicit link $l_{i,j}$ is $P(w_j|w_i)P(w_i) \approx 1/n^2$. In other words, the average probability of explicit links is about $(1\text{-}p)^2$ of that of implicit links. Thus by two-item sequential pattern mining, implicit links could be separated from explicit links by setting an appropriate minimum support. Furthermore, if the variance of implicit link probabilities are relatively larger than the variance of explicit link probabilities, i.e. the users have no significant bias in selecting paths, most two-item access patterns obtained from web log mining with the highest support values will be implicit links.

The above analysis is based on a strict assumption that the explicit link graph is completely connected. This assumption is generally not true in practice. However, if the connectivity in a small web is high, the mining result will still be satisfactory. In some small webs, the existence of a search page or a site map dramatically increases the connectivity of a web-site.