Temporal Ranking of Search Engine Results

Adam Jatowt[1], Yukiko Kawai[1], and Katsumi Tanaka[2]

[1] National Institute of Information and Communications Technology,
3-5 Hikaridai, Seika-cho, Soraku-gun, 619-0289, Kyoto, Japan
{adam, yukiko}@nict.go.jp
[2] Graduate School of Informatics, Kyoto University,
Yoshida-Honmachi, Sakyo-ku, 606-8501, Kyoto, Japan
ktanaka@i.kyoto-u.ac.jp

Abstract. Existing search engines contain the picture of the Web from the past and their ranking algorithms are based on data crawled some time ago. However, a user requires not only relevant but also fresh information. We have developed a method for adjusting the ranking of search engine results from the point of view of page freshness and relevance. It uses an algorithm that post-processes search engine results based on the changed contents of the pages. By analyzing archived versions of web pages we estimate temporal qualities of pages, that is, general freshness and relevance of the page to the query topic over certain time frames. For the top quality web pages, their content differences between past snapshots of the pages indexed by a search engine and their present versions are analyzed. Basing on these differences the algorithm assigns new ranks to the web pages without the need to maintain a constantly updated index of web documents.

1 Introduction

The Web has become the biggest information repository in the world, thus crawling the content of the Web takes a long time. Because web documents can change often, it is difficult to retrieve the freshest information. One strategy for coping with this problem is to customize the crawling patterns to match the frequency of changes to particular web pages. This works reasonably well for pages that change regularly and predictably, like news pages. However, the majority of web resources change in an arbitrary fashion. Thus, search engines cannot retrieve all recently modified content in real time and consequently contain older versions of web pages in their indexes. For example, according to one study, Google [11] crawls web documents with an average delay of one month [19]. This is unacceptable to users who require fresh and relevant information. They often cannot obtain information that is sufficiently fresh simply because the pages have not been crawled over frequently enough. Consequently, users may receive information that is out-of-date or already known to them, or that may even be no longer correct. Search engines generally calculate page ranks using cached versions of pages, which might be obsolete. Thus, the highest ranks may be assigned to the pages that are “popular” and relevant to the query topic but may not necessarily contain the freshest information.

M. Kitsuregawa et al. (Eds.): WISE 2005, LNCS 3806, pp. 43 – 52, 2005.

Ranking of pages, nowadays, is to large extent based on link structure. Algorithms like PageRank [5] help to order the Web and determine the importance of web documents on the basis of their in-link numbers. While such algorithms have been proven to be very effective for finding popularity and importance of documents, they are ineffective for detecting fresh information. It is not necessarily true that web documents with a large number of in-links have good temporal characteristics and consequently are kept up-to-date. There are many stale pages, i.e., pages that are rarely changed; yet they have high ranks assigned by search engines. Obviously, stale web pages are less likely to be interesting and may contain obsolete content with incorrect information. Documents can also be abandoned or their temporal characteristics might change, so that they become more static in terms of new content updates. By conventional link analysis we may not detect such pages because the number of links to the pages may change very slowly. However the decrease in the temporal quality of such web pages can be detected by our proposed system.

One way to determine which web pages are up-to-date and which are not is to analyze their content at several time points in relation to real world changes. High quality pages should have fresh content relevant to their topics. However, precisely determining what is new and up-to-date and what not for any arbitrary query would require computers containing a reliable and continuously up-dated external knowledge base about the world, which is still impossible with the current state of technology.

We thus take a simpler approach by analyzing how changes in web documents are related to the query to approximately estimate if pages are up-to-date and relevant to the query topic. First, we evaluate the temporal relevance of web pages during lengthy, previous periods and select the pages with the best temporal characteristics, i.e., the ones that are frequently changed and are on-topic. It is likely that these pages will also have fresh content later at the time of a query. Then, when a query is entered, we analyze the differences between the present versions of these pages and versions indexed by a search engine. Thus, our method attempts to detect inconsistencies between the state of web documents indexed by a search engine and the current state of web documents. This data is used to estimate the degree of modification needed to make the search engine rankings reflect the freshness and relevance of the content to the query. The rankings are modified accordingly, and the results are sent to the user.

Thus, our method enhances the rankings calculated by traditional search engines. By modifying the rankings based on temporal analysis, it helps the user retrieve fresh, relevant content. We assume here that search engines contain all past data that they have accumulated during crawling, so they can rank web pages in real time.

Another advantage of our approach is that by analyzing archives of web pages we estimate the general relevance of web documents. Thus, we can find whether web pages are devoted to the given topic or whether they just simply mention query words in some point of time in the context of different topic. It seems that currently used ranking algorithms like PageRank rely too much on measuring importance of web documents by link structure analysis and too little on the actual content. This is partially due to the fact that it is difficult for computers to understand the content of the page and its relevance degree to the query, when there is no metadata provided, without using costly natural language processing techniques. However, it often happens that for a given query we obtain results that are not exclusively related to the query topic. Instead we might get the pages that have high PageRank value and they just

contain the query terms only in the search time. On the other hand, web documents that are devoted to the topic and contain always fresh and informative content that is relevant to the query may not have high ranks in the search engine results especially if they concern rare topics.

The next section discusses related research. Section 3 presents our method for temporal ranking of web pages, and Section 4 presents the results of experiments we conducted to evaluate the proposed method. We conclude with a short summary.

2 Related Research

Studies of the characteristics of the Web (e.g., [4,6]) have confirmed the common belief that the Web is a highly dynamic environment and have attempted to estimate the frequencies of web page changes or to predict their updating patterns. For example, more than 40% of web pages in the "com" domain are changed everyday [6].

Some web pages have adopted a push work style in which information about their updates is sent to interested users. This requires users to pre-select the web pages and submit specific requests to receive information about changes of possible interest. However, the majority of web pages still use the conventional pull work style in which users have to visit the appropriate sites each time they want to search for fresh data.

Several automatic change detection and monitoring systems have been proposed [8,13,3]. Most require the user to provide the URLs of target resources and to specify a notification method and several tracking parameters (monitoring frequency, types or locations of changes of interest, etc.). The well-known AIDE [8] uses the HTMLdiff algorithm to present different types of changes from archived web page versions in a graphical way. WebVigiL [13] is an information monitoring and notification system for detecting customized changes in semi-structured and unstructured documents. Finally, ChangeDetector [3] uses machine learning techniques for effectively monitoring entire web sites. The disadvantage of traditional change detection systems is that they require a user to identify beforehand potentially interesting web sites or the type of information that he or she wants to receive in the future.

Google News [10] is a popular automatic news retrieval system. It tracks more than 4000 news sources and displays the latest news related to a query. However, only news-type web pages are analyzed for fresh information. Thus, users receive fresh information from a limited number of news sources.

While there have been a few proposed modifications to the currently used ranking algorithms to incorporate the temporal dimension [1,2], there seems to be no easy way to maintain a fresh search engine index. One proposal was to use a distributed search engine that collects web pages locally [18]. While this could probably provide fresher information more quickly, it is unlikely that distributed search engines will soon replace the existing search engines.

The concept of continual queries was conceived to alleviate the impact of changing web content on information retrieval (e.g., [15,7]). Continual query systems guarantee the freshness of web pages by continuously monitoring them with appropriate frequencies. The crawling schedules of these systems are optimized to minimize resource allocation while keeping web pages as fresh as possible. Consequently, the responses to queries are more up-to-date and are generated at less cost. Our approach is different in that we examine changes in the contents of web documents over long

time periods by using archived pages. Then we utilize these results in query time for ranking pages basing on the differences between their present versions and versions stored by search engines. Additionally, rather than estimating only the frequencies of the content changes of pages in the past, we also focus on the relevance of these changes to the query topic. Web pages modified frequently are considered "active" documents; however, changes that are not related to the query topic are usually of little value to the user.

3 Ranking Method

The basic schema of the system that is using our proposed algorithm is displayed in Figure 1. When a user enters a query, it is forwarded to a search engine, which returns a list of results. The system takes a certain number, *N*, of the top URLs. The number is limited since analyzing all pages for an arbitrary query would be too costly. Past versions of each of these pages are then retrieved from a web page archive. Since retrieving all available archived pages could be too time consuming, only those pages that match some specified time period are retrieved. The number of web pages and the length of the time period can be specified by the user.

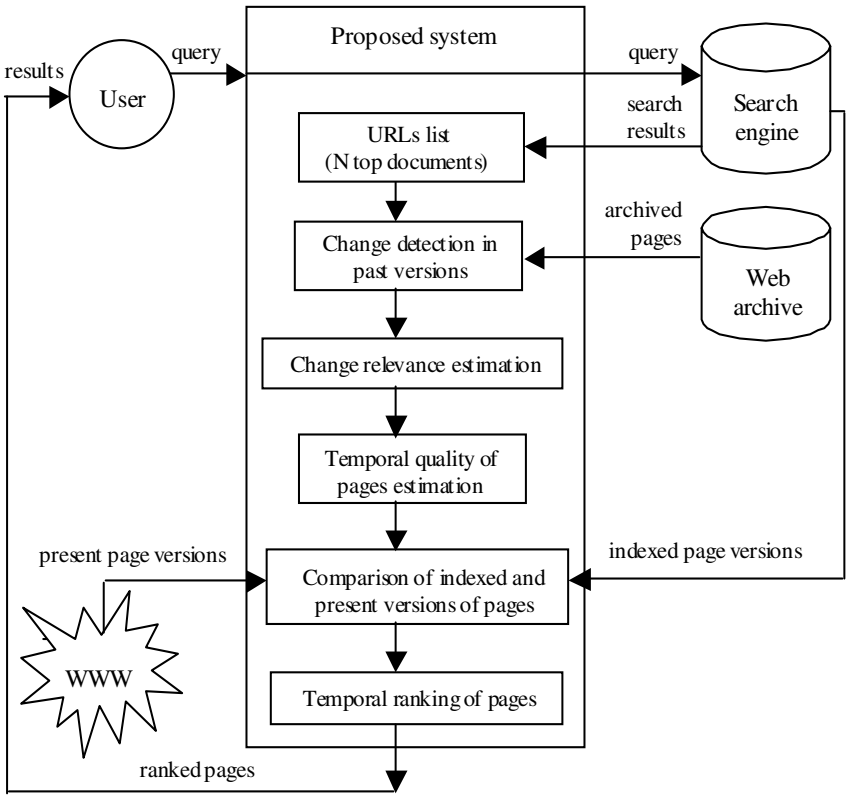

Fig. 1. Basic schema of the proposed system

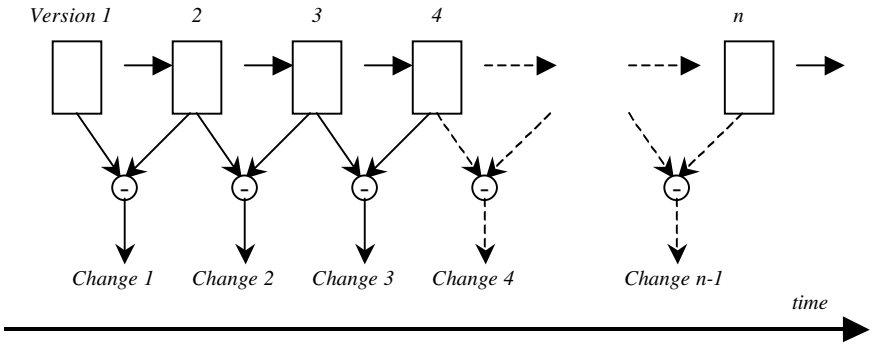

Fig. 2. Comparison of past web page versions to identify changes

The next step is to identify the changes in the past versions of documents. For simplicity, we assume that web documents have fixed structures. Since content changes are more important to users then other types of changes, like presentation or structural ones [9], only content changes are considered in the proposed method. HTML tags, links, and multimedia are discarded. As shown in Figure 2, consecutive versions of web pages are compared for changes. The change comparison is done at the sentence level. Sentences from every consecutive pair of versions of each web document are analyzed to identify added sentences. If a particular sentence appeared only in the later version, it is treated as an addition. To avoid detecting minor sentence modifications (e.g., a few changed words and grammatical corrections), a similarity comparison algorithm is used to examine the types of words and their order in the analyzed sentences. Analysis of the number and order of common words in each pair of sentences produces a similarity value for each sentence pair. A sentence is considered to be a changed one if it has a lower similarity value than some predefined threshold.

Through this comparison, previous additions to the web documents are detected. After these changes are extracted, standard text processing steps such as stemming and stop-word filtering are applied.

We want to find the relevance of changes to the topic represented by the query. To do so, first we employ vector-based representation of changed contents and the query topic. A vector is computed for the whole changed content in each web document version. Query vector is constructed from the query words issued previously by the user; however, it can be also expanded by adding terms related to the topic.

Although, a simple term frequency weighting method could be utilized here, we are using the weighting method shown in Equation 1. A word is weighted more highly the more frequently it occurs in the changed content of a past page version on the whole and also the more frequently it occurs in many different sentences in this changed content.

$$W_i = \frac{n^i}{n^{words}+1} * \frac{N^{sen}+N^i}{N^{sen}} \tag{1}$$

Here, W_i is the weight of a word i in the changed part of a web page, n^{words} is the total number of words in the changed part, and n^i denotes the number of instances of the word i. N^{sen} is the number of sentences in the changed part, and N^i is the number

of sentences containing the word *i*. Thus, the first part of Equation 1 is related to a term frequency of the word *i*, whereas the second one is related to its sentence frequency in the change of the page.

Next, the cosine similarities between each change vector and the query vector are computed. Additionally, we calculate the size of each change in relation to the size of the whole page and the time differences between consecutive page snapshots. All of this data is used to estimate a temporal quality value for each page. The formula for computing the temporal quality of a page is shown below.

$$TQ^n = \frac{1}{\sum_{j=1}^{j=n-1} \frac{1}{\left(T^{present} - T_j\right)}} * \sum_{j=1}^{j=n-1} \left(\frac{1}{\left(T^{present} - T_j^{past}\right)} * \frac{\cos(A^c_{(j,j+1)}, Q)}{\left(T_{j+1} - T_j\right)} * \left(1 + \frac{S^c_{(j,j+1)}}{S_j}\right) \right) \quad (2)$$

TQ^n is the measure of temporal quality of a page calculated over its n past versions, $A^c_{(j,j+1)}$ is the vector of added changes between the j and $j+1$ versions of the page, $\cos(A^c_{(j,j+1)},Q)$ is the cosine similarity between vector $A^c_{(j,j+1)}$ and query vector Q, $S^c_{(j,j+1)}$ is the size of the change between the j and $j+1$ versions of the page, and S_j is the total size of the j version expressed as the number of words. Finally, T_j and T_{j+1} are the dates of the consecutive past versions of the page, and $T^{present}$ is the time when the query was issued. As shown by Equation 2, temporal quality depends on the relevance of its changed content to the query topic, on the size of the changes, and on the time differences between consecutive changes. Small changes are less likely to be as attractive as large changes if we assume the frequency of updating is constant. Additionally, the temporal quality of a page is higher if the page is usually modified relatively quickly. It also depends more on later changes than on older ones as the weight assigned to each change depends on its temporal distance from the present moment. Equation 2 is used to identify and discriminate slow-changing web documents as well as those that have changes not related to the query topic. The higher quality a page is, the greater the likelihood is that its actual content is fresh and relevant at the time of the query.

After the temporal qualities of the web pages have been estimated, the results from the search engine are analyzed again, and the dates of the last crawls of the candidate documents are checked. These dates are usually displayed close to a link to each URL on the search engine results page. Depending on the dates of last crawls and on the values of temporal qualities the system decides which web documents should be analyzed. If a candidate document has high temporal quality, and the date of the last crawl is quite old, the page is processed further. Thus, a page is considered for the next step of the algorithm if the result of multiplying its temporal quality value by the time elapsed since the last crawl is higher than some specified threshold value. In this way, only the pages that have highest probability of having new, relevant content at the time of the query are retrieved and analyzed. Thus, the cost of ranking is decreased.

The next step is to compare the indexed and present versions of the documents for changes. The indexed versions are extracted from a search engine database while the present versions are downloaded at the time of the query. Google [11], MSN Search [16] and other search engines offer cached document versions. This is done to provide page content when users cannot access a given web page. They can instead download the cached version of the document. Our algorithm uses the cached pages in order to compare how the web pages changed since the last crawl. We can see the difference between the state of the web pages seen at the present moment and the state seen by

the search engine ranking mechanism. Thus, in this way the system can correct ranks assigned by search engines in case of some relevant information appearing or disappearing from a given web page.

When the present and indexed versions of the pages are compared, both addition and deletion type changes are extracted this time. If a sentence is found in the indexed page version and not in the present version, it is treated as a deletion. Additions mean that new content has been added that may or may not contain fresh information regarding the query. Deletions have an equally important, although negative, effect on the quality of the overall ranking. The present version of a page may not contain some piece of information related to the query because it was deleted since the last time the page was fetched. Or it cannot contain query word at all. In such cases, the user may have difficulty understanding the relationship between the content of the page and the query.

After we have found inserted and deleted sentences, we group them to form deleted and inserted change for each web document. For each type of change, we compute a vector using the weighting method in Equation 1: A_i and D_i denote the vectors of additions and deletions for page *i*.

To estimate the relevance of the changes to the query topic, we calculate cosine similarities between the query vector and the change vectors for each page. Calculation of the new rank is based on the cosine similarities, the time elapsed since the last crawl over the page, the rank assigned by the search engine, and the size of the changes (Equation 3).

$$R_i^{new} = \left[\frac{\cos(A_i,Q) - \alpha * \cos(D_i,Q) + 1}{\beta * \left(T^{present} - T_i^{indexed}\right) + 1}\right] * \left[1 + \gamma * \frac{N - R_i^{se} + 1}{N}\right] * \left[1 + \eta * \left(\frac{S_i^{a}}{S_i^{indexed}} + \mu * \frac{S_i^{d}}{S_i^{indexed}}\right)\right] \quad (3)$$

Here, R_i^{new} is the new ranking value of a page *i*, $cos(A_i,Q)$ is the cosine similarity between the vector of additions for the page and the query vector, and $cos(D_i,Q)$ is the cosine similarity between the vector of deletions and the query vector. R_i^{se} is the original rank assigned to the page by a search engine, $T_i^{indexed}$ is the time elapsed since the search engine indexed the page, and $T^{present}$ is the present time when the query is issued. S_i^a, S_i^d, and $S_i^{indexed}$ denote the number of words in additions, deletions, and in the indexed version of the page, respectively. Finally, $\alpha, \beta, \gamma, \eta, \mu$ are the weights used to adjust the effects of the features on the new ranking. Their values range from *0* to *1* for β, γ, η and *–1* to *1* for α, μ.

The first part of Equation 3 shows how much the present version of a given page differs from its cached version in relation to the user's query and the time that has elapsed. We use the present time instead of the latest update time because the former means the time when the temporal ranking is being constructed. Using it enables us to determine the time difference between the current version of the page and the one indexed by the search engine. Additionally, the latter is not always available. The second part of Equation 3 takes into account the original ranking of the page in the search engine results. It enables a user to decide on the level of trust for the rankings decided by the search engine and to choose to which extent search engine ranks should be modified. Finally, the last part of Equation 3 depends on the sizes of both types of changes with respect to the size of the cached page.

By assigning suitable weights in Equation 3, we can obtain rankings from different viewpoints. Thus, it is possible to rank documents according to their additions simi-

larity and deletions dissimilarity to the query vector, the time elapsed since the last crawl, the original search engine ranking, and the size of both types of changes. For example, we may look for documents that have large additions related to the query that were made shortly after the last crawl. Or we may search for documents that have both types of changes relevant to the query without considering the time difference between page versions.

Finally, the pages are sorted based on their new ranks. The final results include first the pages that had ranks computed by Equation 3 and later the rest of the pages from the whole set *N* of documents sorted by their temporal quality values (Equation 2). Then, the results are sent to the user. The original ranks assigned by the search engine are also displayed for each URL. Additionally, snippets of web page content in which the query words appear may also be shown.

4 Experiments

We implemented our method in Java 2.01 on a workstation with a Pentium M CPU (1.7 GHz) and 2.0 GB of RAM. We used a TomCat web server [20] and Porter Stemmer in Java [17]. Poorly written HTML code was repaired using the Java module of the JTidy HTML correction tool [14].

The past versions of web pages were obtained from the Internet Archive website [12]. Unfortunately, the pages provided were at least six months old, so we could not test the system using the latest page versions. We have retrieved past versions of web pages for the time frames of 6 months, one year and two years. For some relatively new pages that had not much history we analyzed up to the earliest page version available inside the given time frame. However, few such young pages appeared in the top results of the search engine results for the queries that we have used. Another problem was that some pages had more previous versions crawled by Internet Archive than others.

For simplicity, in our implementation, we have computed query vector using only query words so we did not expand the query vector by additional related terms.

We present here the results for two queries: “Indiana Jones” and “Japanese economy” that were issued to the Google search engine. We compared the results obtained using our method with results obtained from subjects. The subjects were asked to evaluate and rank the top ten web pages found by the search engine from the point of view of freshness and relevance. Table 1 shows the Spearman rank correlation coefficients for “Indiana Jones” query between the subject results and the search engine results, r_{ser}^{s}, and between the subject results and the proposed method results, r_{sys}^{s} for different time frames. It can be seen that our method achieves significantly higher correlation for all

Table 1. Spearman rank correlation coefficients for “Indiana Jones” query

Time frame	r_{ser}^{s}	r_{sys}^{s}
6 months	0.06	0.35
12 months	0.06	0.35
24 months	0.06	0.29

three time periods. For example, for the time frame equal to one year, the subjects judged "Welcome to TheRaider.net" (www.theraider.net) to be the page with the most relevant changes and fresh content related to the query among the top ten results and ranked it first. However, it was ranked sixth in the search engine results.

Table 2 shows results for the "Japanese economy" query. Similarly to Table 1 we can see that the results are slightly worse for longer time frames. This is probably due to the drift in topics of web pages for longer time periods.

Table 2. Spearman rank correlation coefficients for "Japanese economy" query

Time frame	r_{ser}^{s}	r_{sys}^{s}
6 months	0.23	0.78
12 months	0.23	0.78
24 months	0.23	0.63

5 Conclusions

In this paper we have investigated the temporal relevance issue of web pages and presented a novel method for fresh information retrieval from the Web basing on the search engine results. Our approach reorders the results returned by a web search engine so that the pages with fresher and relevant information are ranked higher. It does this by evaluating temporal characteristics and qualities of documents utilizing the archived versions of web pages. This enables identification of the documents that are more likely to contain fresh content that is relevant to the query topic. For such pages we examine the differences in present and indexed by the search engine versions of web documents. In our proposal the search engine does not have to maintain a continuously updated index of all web pages; rather it needs to fetch selected web documents at the time of query. Additionally, we find web documents related to the query topic not only at a single time point but also during a given time period. Such pages can be recommended to users interested in a given topic.

References

1. Amitay, E., Carmel, D., Herscovici, M., Lempel, R., and Soffer A.: Trend Detection Through Temporal Link Analysis. Journal of The American Society for Information Science and Technology, 55: 2004, 1–12
2. Baeza-Yates, R., Saint-Jean, F., and Castillo C.: Web Structure, Age and Page Quality. "String Processing and Information Retrieval", Springer, Lecture Notes in Computer Science (LNCS 2476) 117–130
3. Boyapati, V., Chevrier, K., Finkel, A., Glance, N., Pierce, T., Stokton, R., and Whitmer, C.: ChangeDetector™: A site level monitoring tool for WWW. Proceedings of 11th International WWW Conference. Honolulu, Hawaii, USA, 2002, 570-579
4. Brewington, E. B. and Cybenko, G.: How Dynamic is the Web? Proceedings of the 9th International World Wide Web Conference. Amsterdam, The Netherlands, 2000, 257-276

5. Brin, S. and Page, L.: The anatomy of a large-scale hypertextual web search engine. Proceedings of the 7th World Wide Web Conference, pages 107–117, 1998, Australia
6. Cho, J. and Garcia-Molina, H.: The Evolution of the Web and Implications for an Incremental Crawler. Proceedings of the 26th International Conference on Very Large Databases (VLDB). Cairo, Egypt, 2000, 200-209
7. Cho, J. and Ntoulas, A.: Effective Change Detection Using Sampling. Proceedings of the 28th VLDB Conference. Hong Kong, SAR China, 2002
8. Douglis, F., et al.: AT&T Internet difference engine: Tracking and Viewing Changes on the Web. World Wide Web, vol. 1 (1). Kluwer Academic Publishers, 1998, 27- 44
9. Francisco-Revilla, L., Shipman, F., Furuta, R., Karadkar, U. and Arora, A. Perception of Content, Structure, and Presentation Changes in Web-based Hypertext. Proceedings of the 12th ACM Conference on Hypertext and Hypermedia (Hypertext '01), Aarhus, Denmark, ACM Press, 2001, 205–214
10. Google News: http://news.google.com
11. Google Search Engine: http://www.google.com
12. Internet Archive: http://www.archive.org
13. Jacob, J., et al.: WebVigiL: An approach to just-in-time information propagation in large network-centric environments. Web Dynamics Book. Springer-Verlag, 2003
14. JTidy: http://jtidy.sourceforge.net
15. Liu, L., Pu, C., and Tang, W.: Continual Queries for Internet Scale Event-Driven Information Delivery. IEEE Knowledge and Data Engineering 11 (4). Special Issue on Web Technology, 1999, 610–628
16. MSN search: http://search.msn.com
17. Porter Stemmer in Java: http://www.tartarus.org/~martin/PorterStemmer/java.txt
18. Sato, N., Uehara, M., and Sakai, Y.: Temporal Ranking for Fresh Information Retrieval. Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages. Sapporo, Japan, 2003, 116–123
19. Search Engine Statistics: Freshness Showdown, http://searchengineshowdown.com/stats/freshness.shtml
20. Tomcat Apache: http://jakarta.apache.org/tomcat/