Design and Evaluation of Focused Crawling System

Jyh-Jong Tsay Jung-Shian Liu Zhi-Ming Lai Chen-Yang Shin
National Chung Cheng University
Department of Computer Science
and Information Engineering
Chiayi 621, Taiwan, ROC.
email: tsay@cs.ccu.edu.tw

Abstract

The enormous growth of the world wide web in the recent years has made it important to discover resources efficiently. In this paper, we develop and evaluate approaches for focused crawling whose objective is to retrieve particular topical portions of the World Wide Web quickly without having to visit all web pages.
Keyword: Focused Crawler,Text Categorization,Information Retrieval

1 Introduction

The web environment provides a huge amount of data that offers us more chances than we to find out what we are looking for. Thus many search engines such as *Google*[6], *Yahoo*[7], and *AltaVista*[8] have bloomed in recent years. But these general search engines often produced low quality results. Thus in recent years, topic-specific search engines arisen from this perspective. Topic-specific search engines are constructed according to some domain knowledge. They can be more powerful than general search engines when users knows what they need belongs to a certain topic domain. For example, ResearchIndex[12] is the full-text indexing of scientific literature that aims to improve the dissemination and feedback of scientific literature and to provide improvements on functionality, usability, availability, cost, comprehensiveness, efficiency, and timeliness.

In this paper, we study how context information affects focused crawling, and propose an algorithm incorporating with context information to improve the performance of focused crawling system. The remainder of this paper is organized as follows. Section 2 gives preliminaries for focused crawling and reviews some related work. Section 3 presents our strategy for ordering URLs in crawling. Section 4 describes our system architecture. Section 5 gives experimental results. And section 6 concludes this paper and gives further remarks.

1.1 Related Work

Jason Rennie et. al.[3] created a topic-specific spider to find scientific papers from the web. They claimed that the creation of efficient web spiders is best solved by reinforcement learning, a branch of machine learning that concerns itself with optimal sequential decision making. One strength of reinforcement learning is that it provides a formalism for measuring the utility of actions that give no immediate benefit, but do give benefit in the future.
Charu C. Aggarwal et. al.[20] propose a novel concept of intelligent crawling which actually learns characteristics of the linkage structure of the world wide web while performing the crawling. Specifically, the intelligent crawler uses the inlinking web page content, candidate URL structure, or other behaviors of the inlinking web pages or siblings in order to estimate the probability that a candidate is useful for a given crawl.
M. Diligenti et. al.[2] demonstrate the credit assignment for the focused crawler can be significantly improved by equipping the crawler with the capability of modeling the context graph. A context graph encodes a limited number of the parent links around the target web page, which apparently keep the relationship between content and layer information with a specific topic.

1.2 Our Contributions

Our approach is based on the assumption that similar pages have links pointing to related pages. We

propose ordering strategies that combine information from page contents, hyperlinks, query words and context graphs.

- **Hot Word:** We observe if the link's anchor text contain query word , this link often is valuable. So we use query word as "Hot Word". Computing similiarity between Hot Word and link's anchor text as Hot Word Score. If $score > \theta_h$, it is highly possible that the page pointed to by this link is a target page.We then put this link in queue 0
- **Link Information:** In M. Diligenti et. al.[2],Queue is not a priority queue. So we use link information , let our queue be a priority queue.We compute similiarity between out link's anchor text and anchor texts from all hyperlinks in L_i.L_i be the set of hyperlinks pointing to some pages in layer i for $0 \leq i \leq N$.

2 Preliminaries

2.1 Web Crawler

Web crawlers are widely used today. For examples: Crawlers for the major search engines attempt to visit most web pages in order to build context indexes (e.g., *Google*, *Altavista*, *InfoSeek, Excite*, and *Lycos*[6, 8, 9, 10, 11]). Other crawlers may also visit many pages, but may look only for certain types of information (e.g., e-mail addresses). Sometime, web crawlers are annoying because certain crawler implementations can overload networks and servers. Therefore, a standard for robot exclusion has been proposed by Martijn Koster in 1993[13]. It urges robot writers to obey the rules defined by webmasters in a special file "robots.txt".

2.2 Focused Crawler

A focused crawler(or topic-specific spider) is a crawler aiming to search and retrieve web pages from the world wide web that is related to a specific topic. Rather than collecting and indexing all accessible Web documents, a focused crawler analyzes its crawling boundary to be mostly relevant for the crawling so that irrelevant regions of the Web can be skipped. This leads to significant savings in hardware and network bandwidth resource.

The simplest focused crawler uses a fixed model of the relevancy class, typically encoded as a classifier, to evaluate the retrieved documents for topical relevancy. The classifier is either provided by the user in the form of query terms, or can be built from a set of seed documents. An ideal focused crawler traverses the minimal number of irrelevant documents and retrieves the maximal number of relevant pages. In figure1 , the pseudo code of a general focused crawler is given. During the process of crawling, a lot of hyperlinks are retrieved. Each link is assigned the score of the document to which it leads. A major open problem in focused crawling is to assign proper credit to all pages along a crawl route to yield highly relevant documents.

```
Focused Crawling algorithm
enqueue(url_queue, starting_url);
While(not empty(url_queue)){
    url = dequeue(url_queue);
    page = crawl_page(url);
    enqueue(crawled_url,url);
    url_list = extract_urls(page);
    for each u in url_list{
        if u not in url_queue and u not in crawled_url
            enqueue(url_queue,u);
    }
    reorder_queue(url_queue);
}
```

Figure 1. The general focused crawler in pseudo-code.

3 Our approach

A focused crawler should visit on-topic pages as early as possible and avoid traversing off-topic areas of the Web. Hence, it's the most important task for a focused crawler to measure whether a page can lead to a lot of on-topic pages. Our approach bases on context graphs[2] and improves the ordering strategies by incorporation information from hyperlinks and query words.Figure2.2 is an overview of our crawling algorithm. Follow is a pseudo code. Decide which queue this link be assigned to and their priority in each queue.

3.1 Hot Word Similiarity

Let h be a hyperlink, q be the query, and p be the web page containing h. We first compute a *hot-word score*, denoted as $hot(h, q)$, of h which is the similarity between anchor text of h in p and query words in q. For example: query word is "碩士班甄試" and anchor text is "研究所甄試". We use bigram to compute their similarity score. If $hot(h, q) > \theta_h$, it is highly possible that the page pointed to by h is a target page. We then

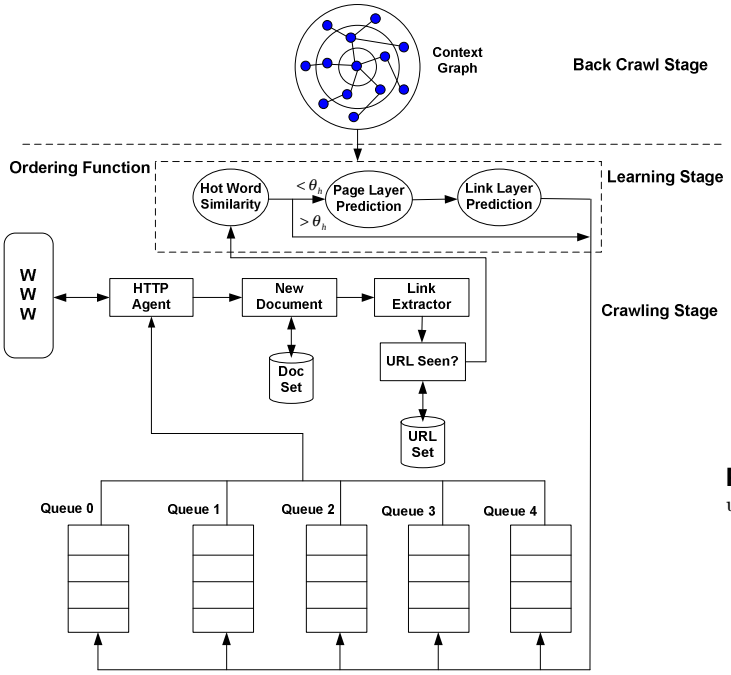

Figure 2. Overview of our focused crawling system.

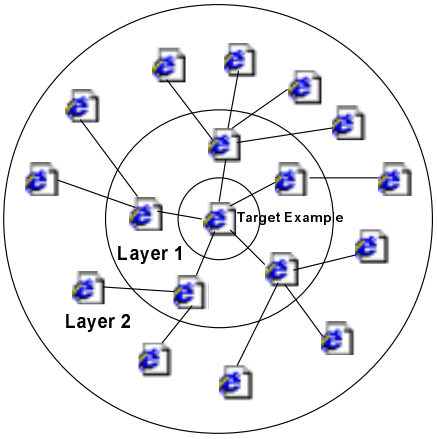

Figure 3. A context graph represents how a target document can be accessed from the web.

研究	究生	士班	甄試
試招	所簡	系招	榜公
簡介	系辦	類系	院碩
碩士	屆考	公告	所碩
國立	考題	招生	大學
學校	學系	班甄	在職
學年	歷屆	放榜	年度

Figure 4. Top 28 terms in the feature distribution for the topic "碩士班甄試" in all layers.

put h in queue 0. If $hot(h,q) < \theta_h$, we put h in queue k, where k is the link distance predicted by our Page Layer Prediction classifier based on the content of p.

3.2 Page Layer Prediction

See figure3.Given examples of target documents, we construct a context graph for each one. Each document appearing in the context graphs is assigned a label that is the minimum link distance from that document to a target document. These documents with their labels form a training set to train a classifier that predicts the link distance from a new document to target documents. A Naive Bayes classifier is used to predict link distance.

The preliminary step for classification is data collection. The next step is term extraction and selection. We use the 2-gram-based model to select candidate terms for documents in Chinese language. Then we use χ^2 - *statistic* method to select representative and informative terms for each class such that selected terms can let classifier distinguish one class from the others. In [4],[5] Tsay and Wang did an extensive comparison of several measures for term selection in Chinese text categorization, such as odds ratio, IG, MI and CHI. In their experiment, CHI achieves the best performance while combined with Naive Bayes classifier, hence we use CHI as term selection method in our crawling system.Figure4 show Top 28 terms in the feature distribution for the topic "碩士班甄試".Such as "甄試" is a meaningful term for the topic "碩士班甄試". The Bayesian approach classifying a new instance is to assign the class with the maximum probability, C_{NB}:

$$C_{NB} = \underset{C_k \in C}{argmax} P(C_k|Doc)$$

3.3 Link Layer Prediction

Each queue is a priority queue in which hyperlinks are ordered according to their similarity scores computed by anchor texts as follows(see figure5). Let G be the context graphs derived from the training examples. The web page appearing in G are partitioned to layer 0, layer 1, ..., layer N, where a page is in layer i if its link distance to target pages is i. L_i be the set of hyperlinks pointing point to a page in layer i for $0 \leq i \leq N$, and L_{N+1} be the set of pages not in any L_i, $0 \leq i \leq N$. Score $sim(h, L_i)$ is defined as the average cosine similarity between the anchor text of h and all anchor texts of all hyperlinks in L_i. Let

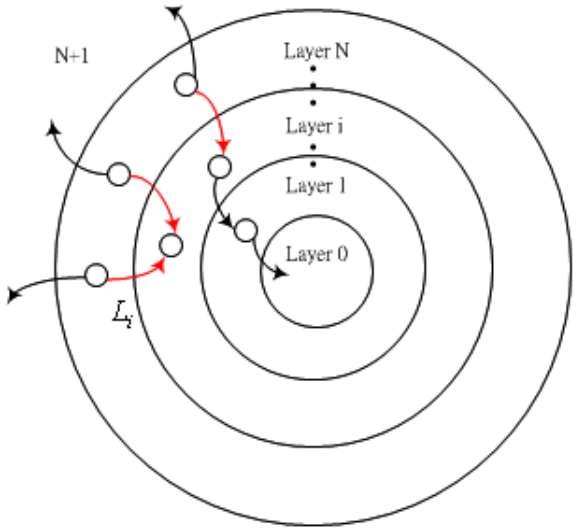

Figure 5. $sim(h, L_i)$ the average cosine similiarity between the anchor text of h and all texts of all hyeperlinks in L_i Let $i = argmax_{0 \le j \le N+1} sim(h, L_j)$

```
Our Crawling algorithm
enqueue(url_queue[N+1], starting_url);
while (not empty(url_queue)) {
    url = dequeue(url_queue);
    page = crawl_page(url);
    enqueue(crawled_url, url);
    url_list = extract_urls(page);
    for each u in url_list{
        if u not in url_queue and u not in crawled_url {
            if hot(u) > θ
                n = 0;
            else
                n = predict(page);
            enqueue(url_queue[n], u);
            reorder_queue(url_queue[n]);
        }
    }
}
```

Figure 6. Our focused crawling algorithm.

$i = argmax_{0 \le j \le N+1} sim(h, L_j)$. The priority of h in queue k is assigned as $(i, sim(h, L_i))$. All hyperlinks in the same queue are ordered by i first, and then by $sim(h, L_i)$ for those with the same i. Note that we use English words and Chinese bigrams as index terms to compute cosine similarity[18].

4 System Architecture

We implemented our system on Microsoft Windows 2000 operation system, using Java language with JDK 1.3.1[16]. There are four distinct stages to use the algorithm when using this system:

- **Collecting Data :** In this stage, web pages and context graphs are collected to train topic specific features.When constructing context graphs for target pages, we need to know the whole web structure in advance. One solution to know the structure is to look up the web graph structure database. We use special search mechanism, link search, provided by Google and AltaVista to construct the structure. Figure 8 shows it on Google.
- **Training classifier:** By using the data collected in previous stage, the classifier is trained to have the ability to rank documents according to its relevancy to the specific topic. We also provide an approach to help the classifier to rank documents more precisely. our system tries to train Naive

```
dequeue(queue) :
remove the element at the beginning of the first non-empty
queue and return it.
hot(url) :
return similarity between anchor text of url and query word.
sim(url, L) :
score = average similarities of url's anchor_text vector and all
        document vectors in L
return score;
predict(page) :
results = classifier_predict(page);
winner_layer = results[0];
winner_layer_score = results[1];
if winner_layer_score < threshold
    winner_layer = N+1;
return winner_layer;
reorder_queue(queue) :
for each u in url_queue{
    i = arg max_{0≤j≤N+1} sim(u, L_j) ;
    priority of u = (i, sim(u, L_i)) ;
}
sort url_queue by i first;
if the same i
    sort by sim(u, L_i)
```

Figure 7. Function description of our focused crawling algorithm.

Figure 8. Link search mechanism provided by Google search engine.

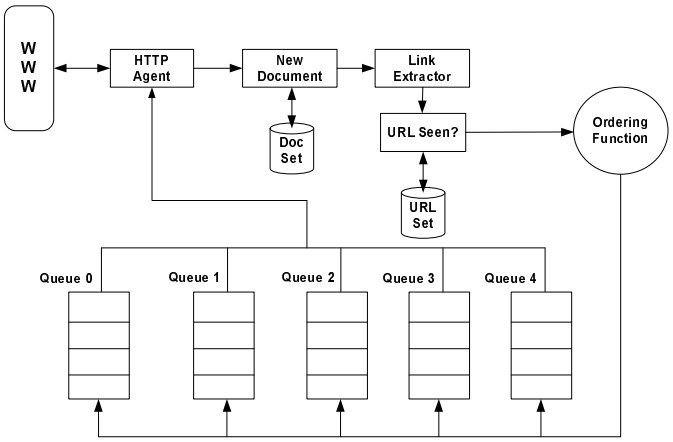

Figure 9. The crawling stage: the focused crawler uses the ordering function to predict the possibility of finding topically relevant documents.

Bayes classifiers and build ordering functions, for further use. We use Naive Bayes classifier to predict page layer and build ordering function for link layer prediction. Our goal is to be able to use classifiers to assign a web page to a set of $N+2$ classes corresponding to the layers $0, 1, 2, \cdots, N$ and a category "other", and combine ordering function to order all URLs in the priority queue. In our experiment, we set N to 4.

- **Crawling Pages:** Our crawler use both the trained classifier and anchor information to rank all the web pages and crawls the web according to this ranking.
- **Ranking Output:** A mechanism for ranking the output crawled pages is used in this stage. It ranks all the output pages in the order of their relevancy to user specified topic. After the crawling phase, our focused crawler can retrieve many web pages from the Web. However, it is difficult for user to discover and exploit these web pages. Therefore, we provide a ranking mechanism to rank these retrieved pages by calculating their relevance with respect to the focused topics.

5 Evaluation

In this chapter, we will compare the performances of our crawling system to two crawlers, one is a breadth-first crawler[21] and the other is a traditional context-graph crawler. Currently, we choose three topics "碩士班甄試" and "旅遊" to be evaluated.

Topic: 碩士班甄試	
Extracted term size: 61,383	
Layer	**Number of nodes**
0	26
1	71
2	115
3	122
4	196
Total	530

Figure 10. Statistics: the context graph for the topic "碩士班甄試".

Topic: 旅遊	
Extracted term size: 177,306	
Layer	**Number of nodes**
0	9
1	104
2	193
3	299
4	552
Total	1157

Figure 11. Statistics: the context graph for the topic "旅遊".

路旅	遊學	國家	生活
旅行	交通	遊網	遊商
天地	特惠	飯店	服務
宅急	急便	浪漫	旅遊
遊局	香港	訂房	資訊
合作	實旅	春節	諮詢
店訂	合旅	行社	期遊

Figure 12. Top 28 terms in the feature distribution for the topic "旅遊" in all layers.

5.1 Experiment Setup

The context graph data set is summarized in Table 10, Table 11 respectively. All the 2-gram terms are extracted from HTML documents, exclusive of HTML tags and their attributes.

All the on-topic documents are selected by user from the documents returned by general text-based search engine. For example, when constructing the context graph for the topic "碩士班甄試", our system submits query such as "碩士班甄試" to Google search engine, and extracts the top 100 relevant documents. Then we choose 26 documents as seeds, and iteratively find their back-links to construct their context graphs.

In these three tables, we observe some interesting characteristics. First, we found the topic "旅遊" has much more terms than the topic "碩士班甄試". Secondly, pages related to the topic "旅遊" have more back-links than those related to the topic "碩士班甄試". This suggests that there is a bigger "旅遊" community in the Web than the one of "碩士班甄試". In fact, there are many "旅遊" web pages in the Web and usually these pages have much higher in-degree than "旅遊" pages.

In the training stage, we use the above two training data to build Naive Bayes classifiers and ordering function. In order to obtain some understanding of the effectiveness of this approach, we list the top 28 ranked terms below in Figure 4 and Figure 12.

As we can see in Figure 4 and Figure 12, many of these terms are very predictive and uniquely describe the specified topic. These terms are likely to be used to describe the contents of the target topics. They are good predictors for the specified topics.

5.2 Experiment Results

We compare the performance of our focused crawler to two other crawlers.We introduce three examples, three topics: "碩士班甄試" and "旅遊"

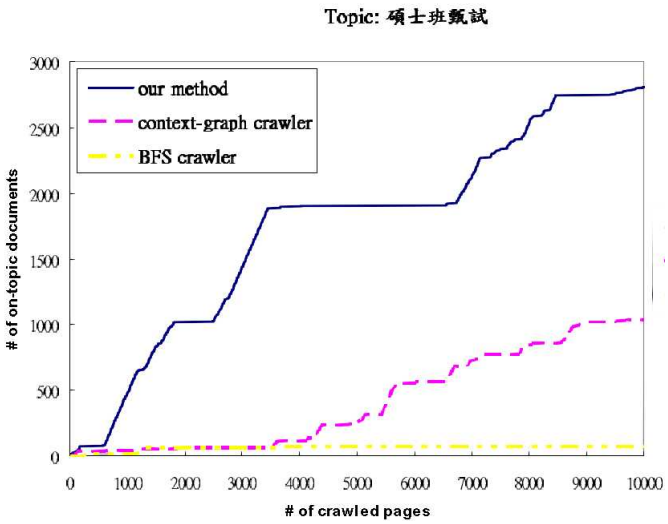

Figure 13. Both the context-graph crawler and our focused crawler are more effective than a traditional breadth-first crawler when retrieving "碩士班甄試" documents.

- **The result of the Topic(碩士班甄試):** See figure13,we found a interesting phenomenon the curve is not smooth. The increase of the curve is very little between 3500 pages and 7000 pages.We consider that when focused crawler is directed to off-topic domain , it must do a likely BFS search to search more page in order to find the next on-topic domain.Because this topic is a small domain topic, the distribution of on-topic pages is more loose.See figure14,the performance of dump result is close to the perfect performance.Figure15 show top rank of pages,we can see that even 1023th page is also on-topic.

- **The result of the Topic(旅遊):** See figure16,this curve is more smooth than cure in topic of "碩士班甄試". Because the topic of "旅遊" is a big domain, it have more pages and links and many pages are linked by each other. So this kind of this topic is easy to crawl. Just using a suitable start point , it will perform well. And shows that our focused crawler performed even better than other crawlers. See figure17,The ranking result for topic "旅遊" is also better. Figure 18 show that even 1003th page is aslo on topic.

6 Conclusion

In this paper, we proposed an ordering strategy that combines information from page contents, hyperlinks, query words and context graphs. Experiments show

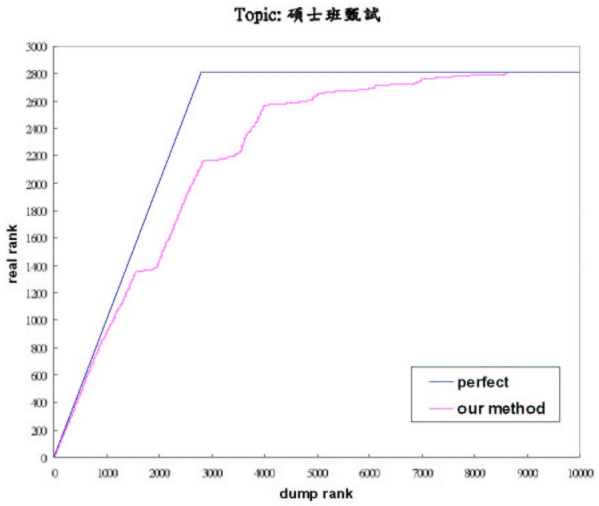

Figure 14. The dump ranking result of the downloaded documents computed using our ranking function.

Rank:1
Link:http://www.ccu.edu.tw/~exams/exam2/s2_new.htm
Title:國立中正大學九十一學年度碩士班甄試暨學士班申請入學招生公告

Rank:501
Link:http://www.thu.edu.tw/grade/91m/
Title:東海大學九十一學年度碩士班招生公告

Rank:1023
Link:http://www.lib.nthu.edu.tw/exam/87/eecs.html
Title:清華大學八十七學年度研究所試　--資訊電機學院

Figure 15. The ranking result examples for topic "碩士班甄試"

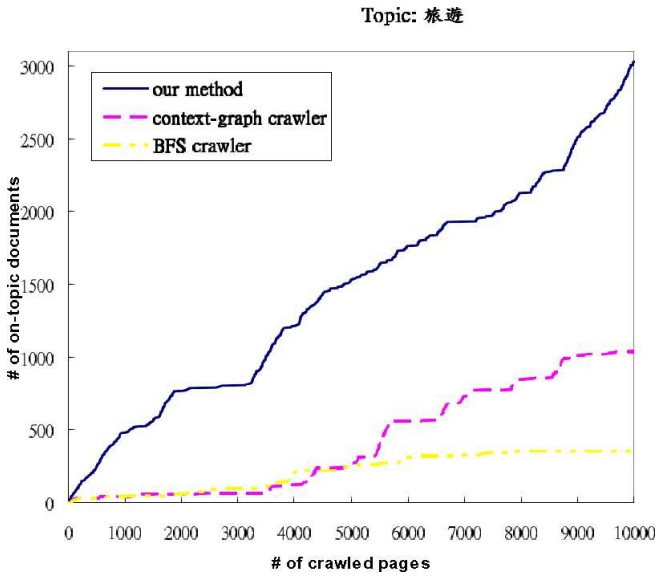

Figure 16. Both the context-graph crawler and our focused crawler are more effective than a traditional breadth-first crawler when retrieving "旅遊" documents.

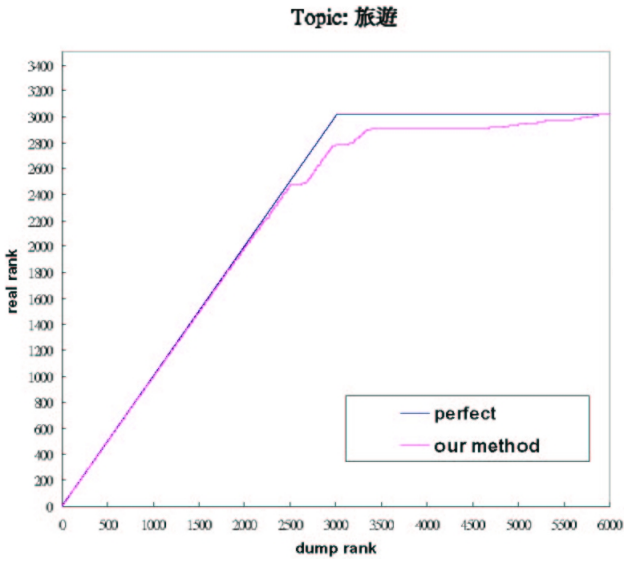

Figure 17. The dump ranking result of the downloaded documents computed using our ranking function.

Rank:1
Link:http://www.tisnet.net.tw/cgi-bin/dir/398e5e70/
Title:休閒旅遊

Rank:501
Link:http://www.settour.com.tw/package/
Title:旅遊行程

Rank:1003
Link:http://www.famoustour.com.tw/j-tour.htm
Title:北南越八日逍遙遊

Figure 18. The ranking result examples for topic "旅遊"

that our approach can obtain relevant pages faster than BFS and focused crawler only using context graphs. However, there are still some further improvements for our system.

- Handle Dynamic web page and complicated HTML layout
- More scalable
- More precise relevancy metrics
- Support Multiple languages

References

[1] Junghoo Cho, Hector Garcia-Molina, Lawrence Page. *Efficient Crawling Through URL Ordering.* In Proceedings of the 7th World Wide Web conference (WWW7), Brisbane, Australia, April 1998.

[2] M. Diligenti, F.M. Coetzee, S. Lawrence, C. L. Giles, M. Gori. *Focused Crawling Using Context Graphs.* In Proceedings of the 26th VLDB Conference, Cairo, Egypt, 2000.

[3] Jason Rennie, Andrew McCallum. *Efficient Web Spidering with Reinforcement Learning.* International Conference on Machine Learning (ICML98)1998.

[4] Jyh-Jong Tsay, Jing-Doo Wang. *Comparing Classifiers for Automatic Chinese Text Categorization.* Proceeding of National Computer Symposium 1999(NCS'99), R.O.C, Dec 20,21,1999.

[5] Jyh-Jong Tsay, Jing-Doo Wang. *Term Selection with Distributional Clustering for Chinese Text Classification using N-grams.* ROCLING 1999, Page 151-170,August 24.25,1999.

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

[7] Yahoo!. http://www.yahoo.com/

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

[9] InfoSeek. http://www.infoseek.com/

[10] Excite. http://www.excite.com/

[11] Lycos. http://www.lycos.com/

[12] ResearchIndex. http://citeseer.nj.nec.com/cs/

[13] Martijn Koster. *Robots Exclusion.* http://www.robotstxt.org/wc/exclusion.html

[14] Sriram Raghavan, Hector Garcia-Molina. *Crawling the Hidden Web.* In the Proceedings of the 27th Intl. Conf. on Very Large Databases (VLDB), pp. 129-138, September 2001.

[15] Yiming Yang, Jan O. Pedersen. *A re-examination of text categorizaion methods.* In Proceedings of 22th Ann Int ACM SIGIR Conference on Research and Development Information Retrieval(SIGIR'94), pages 42-49, 1999.

[16] JAVA SUN. http://java.sun.com/

[17] Cora search engine. http://www.whizbang.com/

[18] G. Salton. *Automatic Text Processing.* Addison Wesley, Massachusetts, 1989.

[19] Budi Yuwono, Savio L. Lam, Jerry H. Ying, Dik L. Lee. *A World Wide Web Resource Discovery System.* The Fourth International WWW Conference Boston, USA, December 11-14, 1995.

[20] Charu C. Aggarwal, Fatima Al-Garawi, Philip S. Yu. *Intelligent Crawling on the World Wide Web with Arbitrary Predicates.* In Proceedings of the Tenth International World Wide Web Conference, Hong Kong, May 1-5 2001.

[21] M. Najork, J. L. Wiener. *Breadth-First Search Crawling Yields High-Quality Pages.* Proceedings of the 10th International World Wide Web Conference, May 2001.

[22] Martin Ester, Matthias Gross, Hans-Peter Kriegel. *Focused Web Crawling: A Generic Framework for Specifying the User Interest and for Adaptive Crawling Strategies.* 27th Int. Conf. on Very Large Databases (VLDB 2001), Rom, Italien, 2001.