Simulation Study of Language Specific Web Crawling

Kulwadee SOMBOONVIWAT[†] Takayuki TAMURA[†, ††] and Masaru KITSUREGAWA[†]

[†] Institute of Industrial Science, The University of Tokyo
[††] Information Technology R&D Center, Mitsubishi Electric Corporation

E-mail: [†] {kulwadee,tamura,kitsure@tkl.iis.u-tokyo.ac.jp}

Abstract The Web has been recognized as an important part of our cultural heritage. Many nations started archiving national web spaces for future generations. A key technology for data acquisition employed by these archiving projects is web crawling. Crawling cultural and/or linguistic specific resources from the borderless Web raises many challenging issues. In this paper, we investigate various approaches for language specific web crawling and evaluate them on the Web Crawling Simulator.

Keyword Language Specific Web Crawling, Web Crawling Simulator, Web Archiving

1. Introduction

The increasing importance of the World Wide Web as being a part of our cultural heritage is a key motivation for many nations to start archiving the Web for future generations. One of the key technologies for automatic accumulation of web pages is *web crawling*. Collecting language specific resources from the Web raises many issues. Two important issues which will be addressed in this paper are:

1) How to automatically identify the language of a given web page?
2) What is a suitable strategy for language specific web crawling?

As for the first issue, the language of a web page can be determined from its character encoding scheme. In this paper, character encoding scheme will be identified by 1) applying a charset detector tool, or 2) checking the charset property supplied in the HTML's META tag.

Next, for the second issue, we propose two language specific web crawling strategies which are based on the idea of *focused crawling*. Focused crawling [1] is one of the most important methods for automatically gathering topic specific resources. By exploiting a topical locality in the Web, a focused crawler selectively retrieves web pages that are likely to be most relevant to the specified topic. In order to adapt the focused crawling techniques for crawling language specific resource, it is necessary to investigate whether the Web has a language locality property. From the examination made on our dataset, we found several evidences that support the existence of language locality in the Web region corresponding to our dataset.

To avoid social and technical difficulties associated with the evaluation of web crawling strategies, we will investigate crawling strategies by using a web crawling simulator. The web crawling simulator is a trace-driven simulation system for evaluating web crawling strategies. Logically, a virtual web space is constructed from the information available in the input crawl logs. The simulator generates requests for web pages to the virtual web space, according to the specified web crawling strategy. The virtual web space gives the properties of the requested web page, such as page's character set and download time, as a response to each request.

We did experiments using two datasets of different languages: Japanese dataset (about 110 million URLs) and Thai dataset (about 14 million URLs). While the Japanese dataset is a representative of a highly language specific web space, the Thai dataset is a representative of a web space with low degree of language specificity.

Two crawling strategies will be evaluated: 1) a simple strategy, and 2) a limited distance strategy. The simple strategy assigns priority to each URL in the URL queue based on relevance score of its referrer. There are two modes in this strategy: soft-focused and hard-focused. From our simulation results, the soft-focused strategy achieves 100% crawl coverage but it requires a large amount of memory to be available for maintaining the URL queue. We use the limited distance strategy to cope with this problem. In the limited distance strategy, a crawler is allowed to go through irrelevant pages in a limited length specified by a parameter N. According to our simulation results, the URL queue's size can be kept compact by specifying a suitable value of parameter N.

The paper is organized as follows: Section 2 presents a background of language specific web crawling. Section 3 describes the essences of language specific web crawling: a classifier, and a crawling strategy. In Section 4, the architecture of the crawling simulator is explained. Subsequently, in Section 5, we report the simulation results and our evaluations. Finally, Section 6 concludes the paper and discusses our future works.

2. Background

In this section, we present the overview of focused web crawling concepts and the discussion of some related works. The following concepts will be reviewed: focused crawling, and focused crawling with tunneling.

2.1 Focused Crawling

Web crawling is the process of automatically downloading new and updated web pages by following the hyperlinks. A web crawler is a program used to facilitate this process.

A general-purpose web crawler tries to collect all accessible web documents. Such a strategy runs into scalability problems, since the Web contains a tremendous amount of documents, and it continues to grow at the rapid rate.

Focused crawling was proposed by Chakrabarti et al. [1]. The goal of a focused crawler is to selectively seek out pages that are relevant to the predefined topics of interest. The idea of focused crawling is based on an assumption that there is a *topical locality* in the Web i.e. web pages on the same topic are likely to be located near in the Web graph.

The focused crawling system has three main components: a **classifier** which determines relevance of crawled pages, a **distiller** which identifies *hubs*, i.e. pages with large lists of links to relevant web pages, and a **crawler** which is controlled by the classifier and the distiller.

Initially, the users need to provide a topical taxonomy (such as Yahoo!, and The Open Directory Project), and example URLs of interest to the focused crawler. The example URLs get automatically classified onto various categories in the topical taxonomy. During this process, the user can correct the automatic classification, add new categories to the taxonomy and mark some of the categories as “good” (i.e. relevant). These refinements made by the users will be integrated into the statistical class model of the classifier.

The crawler obtains a relevance score of a given page from the classifier. When the crawler is in **soft-focused** mode, it uses the relevance score of the crawled page to assign the priority values for unvisited URLs extracted from that page. The unvisited URLs are then added to the URL queue. In the **hard-focused** mode, for a crawled page p, the classifier will classify p to a leaf node c^* in the taxonomy. If any parents of c^* are marked as “good” by the user, then the URLs from the crawled page p are extracted and added to the URL queue.

The distiller component in the focused crawling system employs a modified version of Kleinberg’s algorithm [8] to find topical hubs. The distiller is executed intermittently and/or concurrently during the crawl process. The priority values of URLs identified as hubs and their immediate neighbors are raised.

2.2 Focused Crawling with Tunneling

An important weakness of the focused crawler is its inability to tunnel toward the on-topic pages by following a path of off-topic pages [6]. Several techniques have been proposed to add the tunneling capability to the focused crawler [4, 7].

The context focused crawler uses a best-first search heuristic. The classifiers learn the layers representing a set of pages that are at some distance to the pages in the target class (layer 0) [4]. The crawler keeps a dedicated queue for each layer. While executing a crawl loop, the next URL to be visited by the crawler is chosen from the nearest nonempty queue. Although this approach clearly solves the problem of tunneling, its major limitation is the requirement to construct a context graph which, in turn, requires reverse links of the seed sets to exist at a known search engine. In this paper, we use a limited distance strategy (Section 3.3.2) to remedy this weakness of the focused crawler and also to limit the size of the URL queue.

3. Language Specific Web Crawling

We will adapt the focused crawling technique to language specific web crawling. As stated earlier in Section 2.1, focused crawling assumes topical locality in the Web. Consequently, prior to applying the focused crawling technique to language specific web crawling, it is necessary to ensure or at least show some evidences of *language locality* in the Web.

We sampled a number of web pages from Thai dataset (see Section 5.1). The key observations are as follows.

1) In most cases, Thai web pages are linked by other Thai web pages.
2) In some cases, Thai web pages are reachable only through non-Thai web pages.
3) In some cases, Thai web pages are mislabeled as non-Thai web pages.

The above observations give support to our assumption about the existence of language locality in the Web. In the following subsections, we will describe the essences of language specific web crawling.

3.1 System Architecture

In the first version of a language specific web crawler has two main components: a **crawler** and a **classifier**. The crawler is responsible for the basic functionality of a web crawling system e.g. downloading, URL extraction, and URL queue management. The classifier makes relevance judgments on crawled pages to decide on link expansion. The following subsection describes methods for relevance judgments used in the language specific web crawling.

3.2 Page Relevance Determination

In language specific web crawling, a given page is considered relevant if it is written in the target language. Two methods for automatically determine the language of a web page will be employed in our system.

- Checking character encoding schemes of web pages.

Character encoding scheme specified in the HTML's META declaration can be used to indicate the language of a given web page. For example,
<META http-equiv="Content-Type" content="text/html; charset=EUC-JP"> means that the document's character encoding scheme is "EUC-JP" which corresponds to Japanese language. Table 1 shows the character encoding schemes corresponding to Japanese and Thai languages.

Language	Character Encoding Scheme (charset name)
Japanese	EUC-JP, SHIFT_JIS, ISO-2022-JP
Thai	TIS-620,WINDOWS-874,ISO-8859-11

Table 1 Languages and their corresponding character encoding schemes.

- Analyzing character distribution of web pages.

A freely available language detector tool, such as the Mozilla Charset Detector [9, 10], can be used to analyze the distribution of bytes of a web page to guess its character encoding schemes.

Although using language detector tools to determine document language is more reliable than using HTML's META tag supplied by an author of the document, some languages, such as Thai, are not supported by these tools. Therefore, in our experiments on Thai dataset, we will determine a page's language by checking the character encoding schemes identified in the HTML's META tag. For the experiments on Japanese dataset, the Mozilla Charset Detector will be used to determine language of a given web page.

3.3 Priority Assignment Strategy

In this paper, we propose two strategies for priority assignment: a simple strategy, and a limited-distance strategy. The simple strategy is based on the work of focused crawling. The limited-distance strategy's idea is similar to that of focused crawling with tunneling. We will now describe each strategy in more detail.

3.3.1 Simple Strategy

The simple strategy is adapted from focused crawling. In this strategy, the priority value of each URL will be assigned based on the relevance score of the referrer (or parent) page. Relevance score of the referrer page can be determined using the methods described in Section 3.2. For example, in the case of Thai language specific crawling, if the character encoding scheme of a given web page corresponds to Thai, then the relevance score of the web page will be equal to 1. Otherwise, the relevance score of that web page will be equal to 0.

As in the case of focused crawling, there are two crawling modes in our simple strategy i.e.

- Hard focused mode.

The crawler will follow URLs found in a document only if the document is written in the target language (relevance score = 1).

- Soft focused mode.

The crawler does not eliminate any URLs. It uses the relevance score of the crawled page to assign priority values to the extracted URLs.

Table 2 summarizes the key concept of our simple strategy. In the following subsection we will introduce the limited distance strategy.

Mode	Relevant Referrer	Irrelevant Referrer
Hard-focused	Add extracted links to URL queue	Discard extracted links
Soft-focused	Add extracted links to URL queue with *high* priority values	Add extracted links to URL queue with *low* priority values

Table 2 Simple Strategy.

3.3.2 Limited Distance Strategy

In the limited distance strategy, the crawler is allowed to proceed along the same path until a number of irrelevant pages, say *N*, are encountered consecutively (see Figure 1 for an illustration). The priority values of URLs are determined differently according to modes being employed:

- *Non-prioritized Limited Distance.*

All URLs are assigned equal priority values.

- *Prioritized Limited Distance.*

In this mode, priority values are assigned based on distance from the latest relevant referrer page in the current crawl path. If the URL is close to relevant page then it will be assigned the high priority value. Otherwise, if the URL is far from relevant page then it will be assigned the low priority value.

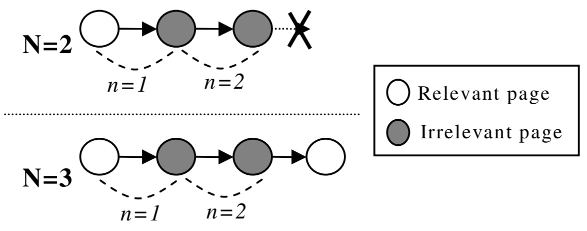


Figure 1 Limited Distance Strategy.

3.4 Evaluation Metrics

In this paper, we will use the following metrics to measure the performance of the language specific web crawling strategies.

Harvest Rate (precision) is the fraction of crawled pages that are relevant. Therefore, in the case of language specific web crawling, if 50 pages written in the target language are found in the first 500 pages crawled then we have a harvest rate of 10% at 500 pages.

Coverage (recall) is the fraction of relevant pages that are crawled. Because it is extremely difficult to determine the exact number of relevant documents in the real Web environment, many indirect indicators for estimating recall have been proposed, such as the target recall and robustness [5]. Nevertheless, in our case, we will evaluate our crawling strategy by conducting crawling simulations. Thereby, the number of relevant documents can be determined beforehand by analyzing the input crawl logs. Thus, in this paper, the crawl coverage will be measured using the explicit recall.

4. Web Crawling Simulator

Evaluating crawling strategies under the real dynamic web environment poses the following issues.

1) Social issues:

A good crawler needs to be polite to the remote web servers. It is an ill-manner to let the buggy crawler annoy the remote servers. We need to test a crawler extensively before launching it into the Web.

2) Technical issues:

Because the web is very dynamic, it is impossible to ensure that all strategies are compared under the same conditions and to repeat the evaluation experiments.

3) Resource issues:

Web crawling consumes time, network bandwidth and storage space. The requirement to test the crawler on large portion of the Internet causes a high cost to be assigned to the operation.

To avoid above issues, we evaluated our crawling strategies on the crawling simulator. The Web Crawling Simulator is a trace-driven simulation system which utilizes crawl logs to logically create a virtual web space for the crawler.

The architecture of the crawling simulator is shown in Figure 2. The crawling simulator consists of the following components: simulator, visitor, classifier, observer, URL queue, crawl logs, and link database.

A **simulator** initializes and controls the execution of the system.

A **visitor** simulates various operations of a crawler i.e. managing the URL queue, downloading of web pages, and extracting new URLs.

A **classifier** determines relevance of a given page by applying methods described in Section 3.2.

An **observer** is an implementation of the web crawling strategy to be evaluated.

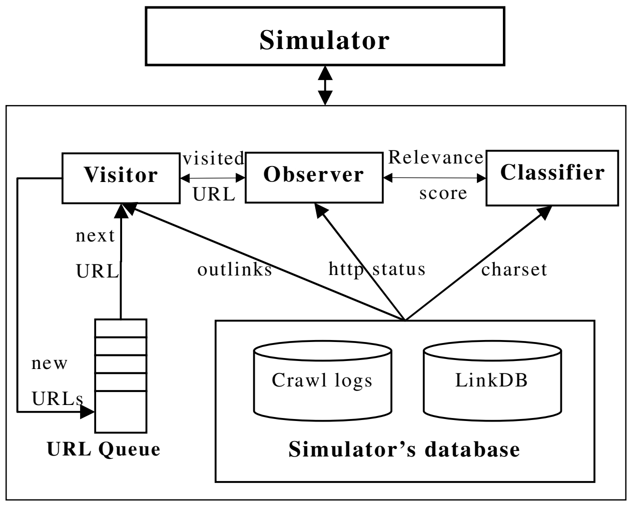


Figure 2 The Web Crawling Simulator.

It must be noted that the first version of the crawling simulator is still simple. It has been implemented with the omission of details such as elapsed time and per-server queue typically found in a real-world web crawler.

5. Experiments

In this section, we report experimental results and evaluations of the proposed language specific crawling strategies (described in Section 3.3).

5.1 Datasets: Japanese, Thai

In order to create the virtual Web environment, the simulator needs access to the database of crawl logs. These crawl logs were acquired by actually crawling the Web to get the snapshot of the real Web space. Two sets of crawl logs will be used in our experiments: 1) Japanese dataset (about 110 million URLs) and 2) Thai dataset (about 14 million URLs).

In the case of Japanese dataset, we used a combination of hard focused with limited distance strategies to limit the crawl radius into a region of Japanese web space. In the case of Thai dataset, a combination of soft focused with limited distance strategy was used to obtain the snapshot of Thai web space. Table 3 shows statistics of Thai and Japanese datasets.

The ratio of relevant pages to total pages in a dataset is an indicator of the language specificity of a dataset. Datasets with high degree of language specificity are not suitable for evaluating language specific web crawling strategies because 1) there will be little space for making an improvement, and 2) performance comparison between different crawling strategies may be difficult.

There are about 1.4 million relevant HTML pages out of 3.8 million HTML pages within the Thai dataset (relevance ratio is about 35%). For the Japanese dataset, 68 million pages out of 95 million pages are relevant (relevance ratio is about 71%). While Japanese dataset is a better representation of the real Web space (due to its larger size), it has quite a high degree of language specificity. Nevertheless, before obtaining evidences from the simulation result, we cannot conclude that the Japanese dataset is inappropriate for evaluating web crawling strategies. The discussion about this issue will be presented later in Section 5.2.1.

	Thai	**Japanese**
Relevant HTML pages	1,467,643	67,983,623
Irrelevant HTML pages	2,419,301	27,200,355
Total HTML pages	**3,886,944**	**95,183,978**

Table 3 Characteristics of experimental datasets Note that in this table we show only the number of pages with OK status (200).

5.2 Experimental Results

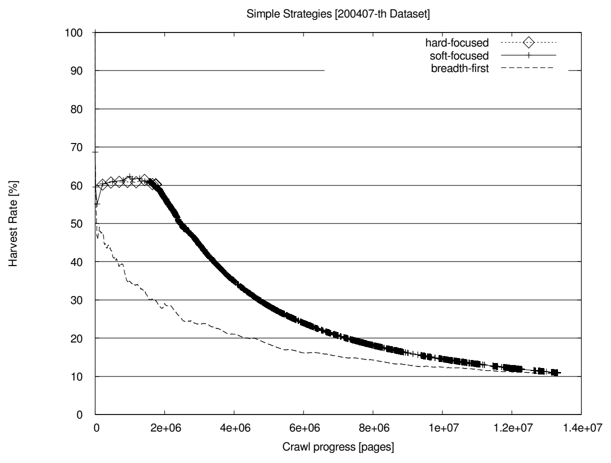
5.2.1 Simple Strategy

Figure 3 and 4 show the simulation results of a simple strategy in hard-focused and soft-focused mode on Thai and Japanese dataset respectively.

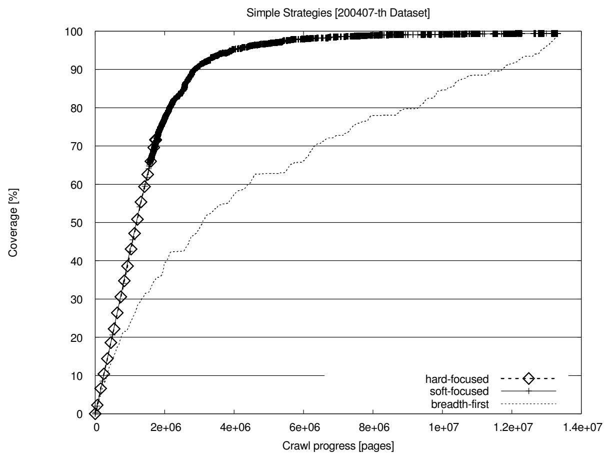
Let us consider the results on Thai dataset in Figure 3. Both modes of the simple strategy give higher harvest rate than the breadth-first strategy, and yield 60% harvest rate during the first 2 million pages

While the soft-focused mode can reach 100% coverage after crawling about 14 million pages, the hard-focused mode stops earlier and obtains only about 70% of relevant pages. This is caused by abandonment of URLs from irrelevant referrers imposed in the hard-focused mode.

Evaluations of the simple strategy on Japanese dataset give the results (Figure 4) which are consistent with those on Thai dataset. However, harvest rates of all strategies are too high (even the breadth-first strategy yields >70% harvest rate.). The reason is that the Japanese dataset is already kept sufficiently relevant i.e. the dataset has a high degree of language specificity. Because it seems to be difficult to significantly improve the crawl performance on Japanese dataset with additional focusing strategies, in the subsequent experiments we will test crawling strategies with the Thai dataset only.

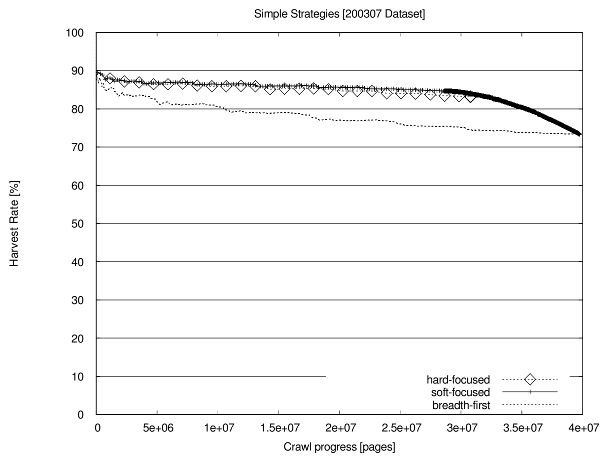


(a) Harvest rate

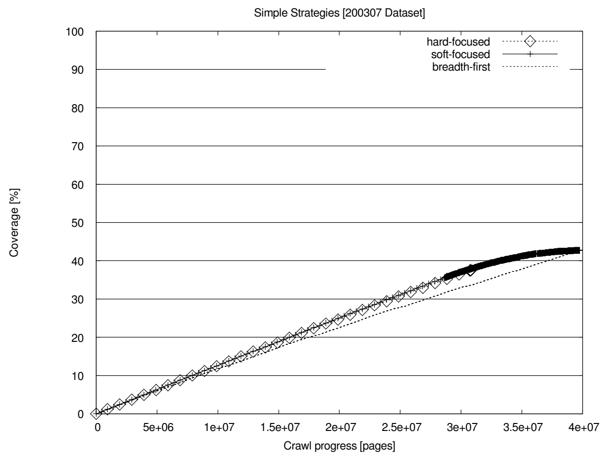


(b) Coverage

Figure 3 Simulation results of the Simple Strategy on Thai dataset.



(a) Harvest Rate



(b) Coverage

Figure 4 Simulation results of the Simple Strategy on Japanese dataset.

From the simulation results presented so far, the soft-focused mode seems to be the best strategy to pursue. But the implementation of the soft-focused strategy may not be feasible in the "real" world crawling. In soft-focused mode, all to-visited URLs will be kept in the queue. In our simulated Web environment which consists of about 14 million pages (OK + non-OK pages), the size of URL queue during the crawl progress is as shown in Figure 5. The maximum queue size of soft-focused mode is about 8 million URLs, far greater than the maximum queue size of hard-focused mode crawling (about 1 million URLs). Scaling up this to the case of the "real" Web, we would end up with the exhaustion of physical space for the URL queue.

Therefore, some mechanisms for limiting the number of URLs submitted to the URL queue are needed. Although discarding of URLs imposed in hard-focused mode can effectively limit the URL queue's size (as can be seen from Figure 3 and 4), it is too strict. The limited distance strategy, which will be presented in the next section, lessens this strictness by permitting a crawler to follow links found in irrelevant referrers for a maximum distance of *N* irrelevant nodes.

5.2.2 Limited Distance Strategy

From Section 3.3.2, two modes of priority assignments were proposed for the limited distance strategy: non-prioritized and prioritized modes.

The results for non-prioritized mode are shown in Figure 6. In these simulations we are using N=1, 2, 3, 4. Figure 6(a) shows the measurement of URL queue's size during the crawl progress; Figure 6(b) shows the harvest rate obtained by each mode; and Figure 6(c) shows the corresponding coverage for each mode of the strategy.

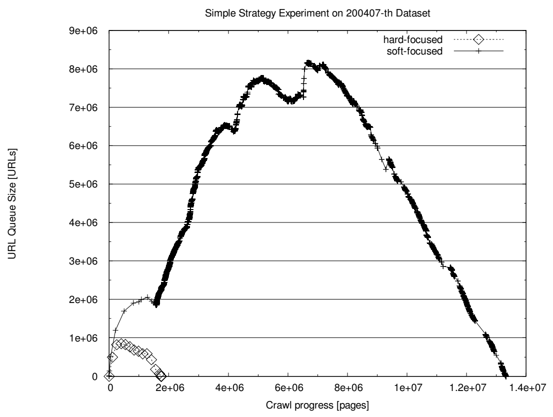


Figure 5 Size of URL Queue while running the simulations for the Simple Strategy.

It can be clearly seen that the size of the URL queue can be controlled by changing the parameter *N* of the strategy. If *N* increases then the queue size will be larger. Likewise, the coverage also increases while increasing *N*. By specifying a suitable value of *N*, the URL queue can be kept compact while still obtaining almost the same percentage of crawl coverage as in the case of soft-focused mode of the simple strategy. Nevertheless, from the result in Figure 6(b), as the parameter *N* is being increased, the harvest rate of the crawl will be lower. Therefore, although increasing parameter *N* may enlarge our crawl coverage, setting too high value of *N* is not beneficial to the crawl performance.

Figure 7 is the simulation results for a prioritized mode of the limited distance strategy. In analogous to the non-prioritized mode, the URL queue size can be controlled by specifying an appropriate value of the parameter *N*. However, this time, both the crawl coverage and the harvest rate do not vary by the value of *N*.

To conclude, the “non-prioritized” limited distance adds ability to limit size of the URL queue to the crawler; and the introduction of “prioritized” limited distance helps solving the weakness of the “non-prioritized” strategy (i.e. the decreasing of harvest rate while *N* is being increased).

6. Conclusion

In this paper, we proposed a language specific web crawling technique for supporting the national and/or language specific web archive. The idea of focused crawler (Chakrabarti et.al.[1]) was adapted to language specific web crawling with an assumption that there exists “language locality” in the Web.

To avoid difficulties associated in testing a crawler on the Web, we have evaluated our crawling strategies on the crawling simulator. Two strategies have been proposed and evaluated in this paper: 1) Simple Strategy, and 2) Limited Distance Strategy.

The simple strategy has two variations i.e. soft-focused and hard-focused. The soft-focused mode is an ideal strategy with perfect crawl coverage but impractical space requirement for maintaining the URL queue.

The limited distance strategy achieves high crawl coverage as in soft-focused strategy with less space requirements by discarding only some URLs with low probabilities of being relevant to the crawl (the probabilities are inferred from the distance to the latest relevant referrer page in the crawl path).

Although we could achieve the highest harvest rate of about 60% on Thai dataset during the first part of the crawl, no strategy was able to maintain this rate.

For the future works, we will conduct more simulations on a larger dataset with a wider range of crawling strategies. We also would like to enhance our crawling simulator by incorporating transfer delays and access intervals in the simulation.

References

[1] Soumen Chakrabarti, Martin van den Berg, and Byron Dom, “Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery”. Computer Networks, 31(11-16):1623-1640, 1999.

[2] Soumen Chakrabarti, Martin van den Berg, and Byron Dom, “Distributed hypertext resource discovery through examples”. Proc. of 25th Int. Conf. on Very Large Data Bases, pages 375-386, September 1999.

[3] J. Cho, H. Garcia-Molina, and L. Page, “Efficient Crawling Through URL Ordering”. Computer Networks, 30(1-7):161-172, 1998.

[4] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, "Focused crawling using context graphs". Proc. 26th International Conference on Very Large Data Bases (VLDB2000), pages 527-534, Cairo, Egypt, 2000.

[5] G. Pant, P. Srinivasan, F. Menczer, “Crawling the Web”. in M. Levene and A. Poulovassilis, editors: Web Dynamics, Springer-Verlag, 2003.

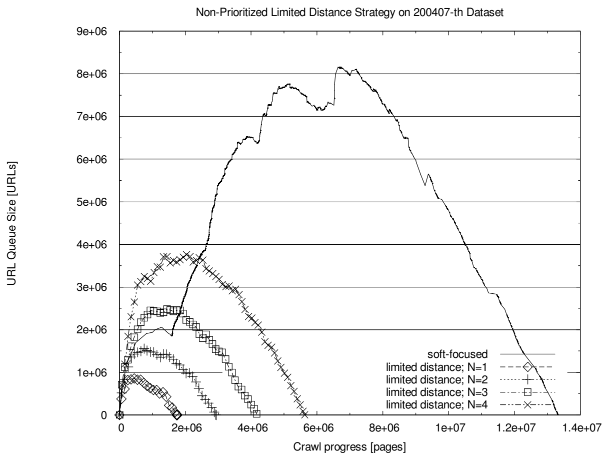
[6] A.A. Barfourosh et al., “Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition”, tech. report CSTR-4291, Computer Science Dept., Univ. of Maryland, 2002.

[7] A. McCallum et al., "Building Domain-Specific Search Engines with Machine Learning Techniques", Proc. AAAI Spring Symp. Intelligent Agents in Cyberspace, AAAI Press, 1999, pp. 28-39.

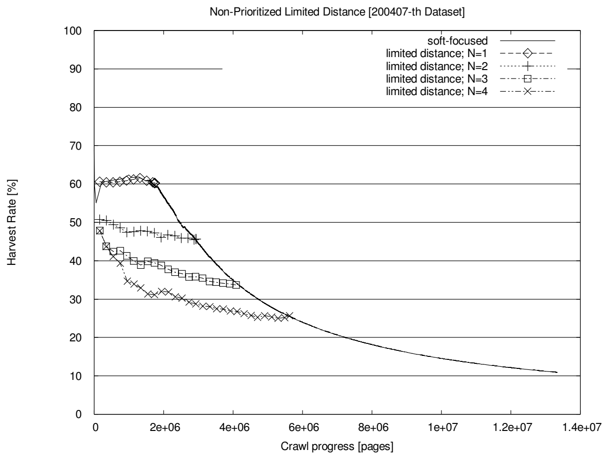
[8] J. Kleinberg, “Authoritative sources in a hyperlinked environment”. Journal of the ACM, 46(5):604-632, 1999.

[9] The Mozilla Charset Detectors. http://www.mozilla.org/projects/intl/chardet.html

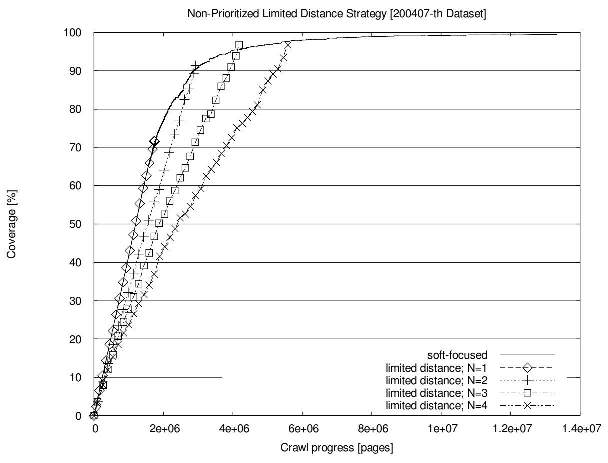
[10] Shanjian Li, Katsuhiko Momoi, "A composite approach to language/encoding detection", In 19th International Unicode Conference, 2001.



(a) URL queue's size

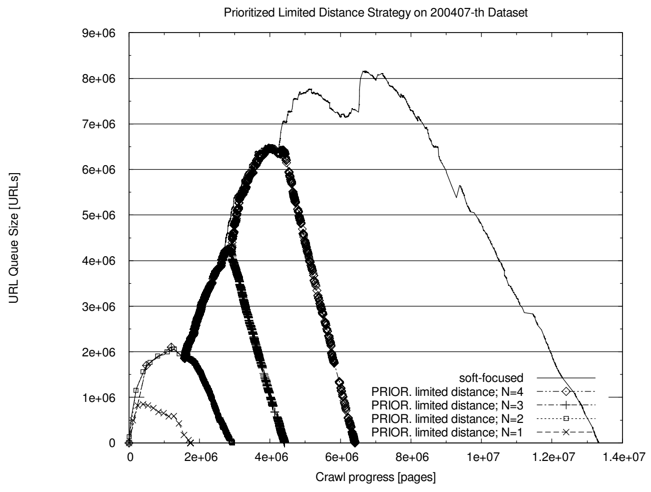


(b) Harvest rate

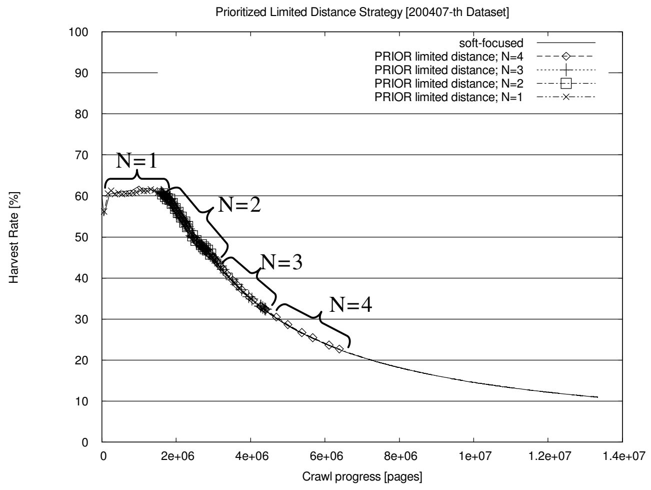


(c) Coverage

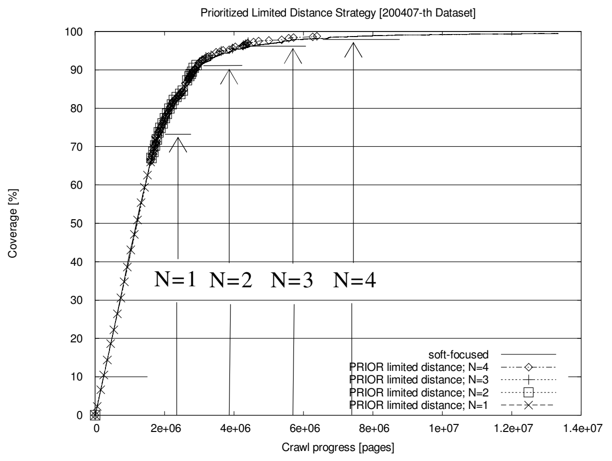
Figure 6 Non-Prioritized Limited Distance Strategy.



(a) URL queue's size



(b) Harvest rate



(c) Coverage

Figure 7 Prioritized Limited Distance Strategy.