Designing Efficient Topic-Driven Web Crawlers

Yiqiao Wang, Eleni Stroulia
Computing Science Department
University of Alberta
Edmonton, AB T6G 2H1, Canada
{stroulia, yiqiao}@cs.ualberta.ca

Abstract

Crawlers are essential to web search engines for retrieving high quality web pages automatically and efficiently based on developer defined notions of importance and quality. Due to rapid growth of World-Wide Web and limited resources available to crawlers, developing good crawling strategies and evaluating them are still big challenges. In this paper, we do a comprehensive study of existing and proposed crawling strategies done by other research works. We have developed a topic-driven crawler that uses combinations of two different strategies in evaluating page importance during the crawl.

1. Introduction

Crawlers, also known as robots, bots, spiders and wanderers, are programs that retrieve web pages autonomously. Crawlers are most commonly used by search engines. Search engines index the pages retrieved by crawlers and return highly ranked pages to answer user queries. Because of the dynamic nature of web, its huge increase in size, and limited bandwidth, storage, and computational resources available to web crawlers, it is impossible for a web crawler to crawl the entire internet. It is even difficult for the crawlers to monitor the crawled portion of the internet for changes [1]. Thus it is crucial to develop efficient crawling strategies for a crawler to retrieve more important, high quality, and relevant pages first before they visit other pages.

A crawler starts off with an URL and retrieves an initial page, P_0. It saves and parses P_0, extracts its out-going links, saves the links into a queue which is known as crawler frontier [1], and chooses from the queue the next link to visit [1,2]. Crawlers store the pages traversed in their own buffer storage. When the buffer is full, different crawlers use different methods to delete the least important page in the buffer to make room for newly traversed pages using different notions of importance [1]. This process of retrieving new pages from previously retrieved ones goes on. The key challenge during the crawl is to identify the next most important link to crawl from the crawler frontier [1].

There are two types of crawlers: standard (general) crawlers and topic-driven (focus) crawlers [1,3]. And there are generally two kinds of algorithms for determining the importance of a page: link-based algorithms and content-based (similarity-based) algorithms [1,3,4,5].

In this paper, I propose to design and implement a topic-driven crawler that combines link-based approach with content-based approach in evaluating the importance of a web page. The crawler can use up to two different crawling strategies including analyses based on similarity (content-based), and based on information we can get from the URL and text surrounding the URL. Users can easily specify from crawling application interface whether s/he wants both strategies to be use and how much credit to give to the results given by each crawling strategy.

The performance of the crawler will be affected by the combination of crawling strategies chosen and the weight assigned to each strategy used. We will experiment with different combinations of strategies and see the differences between different combinations.

The remainder of the paper is organized as follows: section 2 discusses some background and related work as how to achieve efficient crawling in the web and how to evaluate crawlers; since the crawling module is a part of our internet application project, section 3 gives the overview of our project and discusses where the crawler modules fits in our project; section 4 discusses in detail how the crawler is designed and implemented, and how we propose to evaluate the crawler; section 5 lists some of the future work; section 6 outlines the conclusions that can be drawn from this work.

2. Background and Related Work

There are two types of crawlers: Standard crawlers and topic-driven crawlers. Standard crawlers are query based and they are more common in early days, examples include FishSearch and WebCrawler [1]. General crawlers try to collect and index all accessible Web documents to be able to answer all possible ad-hoc queries [3]. On the contrary, a focused crawler tries to selectively seek, acquire, and index pages that are relevant to its predefined set of topics and tries to avoid Web pages that seem to be irrelevant to the crawl. The topics are specified by exemplary documents [3]. Charkrabarti *et al.* uses the metaphor that generic crawlers and search engines are like public libraries. They try to satisfy everyone. While topic-driven crawler are like "university research libraries" where they allow serious users to search for a topic they are interested in depth [3]. [1] proposes to feed crawlers with both relevant and irrelevant documents as positive and negative examples. [3] claims that focused crawlers are more efficient, but they are more difficult to develop since the crawlers have to make additional decisions of the relevance of a web page during the crawl [1].

The strategy used by a crawler to select a link for traversal ties to the goals of crawlers [1]. The strategies are also affected by the types of web sites the crawlers crawl on and what is considered important to those web sites. Since even human experts' notion of quality and importance of a web page vary widely [4], different crawlers use different algorithms to judge the importance of a page. However, there has been little empirical evaluation of these algorithms.

2.1 Page importance

There are in general two ways to judge the importance of a web page: link-based and similarity-based [1,2,4]. Link based measurements are based on the intuition that the counting of links reflect the credibility of a web page [1,4]. Much research has been done in link-based analysis. Similarity-based approaches take into account the similarity between the contents of a document and the topic. The query in question is pattern matched against a web page.

2.1.1 Link-based measurements for page importance

In-degree, out-degree, PageRank, hubs and authorities are the most commonly used link-based measurements for page importance [1].
In-degree counts the number of backlinks to a web page; out-degree counts the number of links that emanate from a web page. They both consider the importance of each link equally. Pages with higher counts are considered to be more valuable and authoritative. PageRank does not consider all links equally. It recursively computes a document's score based on the scores of documents that point to it. PageRank can be thought of as a model of user behavior because "the PageRank of a page represents the probability that a random surfer will be on that page at any given time" [1, 2, 3, 4, 5].

Other link-based algorithms utilize the notions of authority and hubs. An authoritative document is one that many other documents link to. A hub is a document that links to many other documents for a given topic. Park claims that "hubs and authorities stand in mutually reinforcing relationship: a good authority is a document that is linked to by many good hubs, and a good hub is a document that links to many authorities" [4].

The research done by Amentos shows that in-degree, authority, and PageRank are effective at identifying high authority pages as judged by human experts [1,4]. Cho *et al.* shows that PageRank works better than in-degree and depth first algorithm when either pages with many backlinks or with high PageRank are sought [2].

2.1.2 Similarity-based measurements for page importance

Similarity has been well studied in Information Retrieval (IR) and has been applied to WWW environment [2]. In similarity-based algorithms, a document P and a query Q are viewed as two m-dimensional vectors which contain the entire collection of vocabulary. Each vector element represents the importance of a word in P or Q. If the word does not appear in P or Q, this value is set to zero. Otherwise, the value is set to represent the importance of a word [2]. This value takes into account the frequency a word appears in the document and in the vocabulary, and the location of a word in the document. Similarity is then calculated as inner product of P and Q vectors [2]. Other research has suggested that an alternative is to use the cosine similarity measure which is the inner product of the normalized vectors [1,2].

Chakrabarti *et al*. suggests to use average "harvest ratio" to judge page relevance, i.e. the average number of relevant pages retrieved over different time slices of the crawl [1].

Other measurements that have been considered include pattern matching the query with anchor text of URL, text around URL, location and font of matched key words within the document [1,5]. Google search engine considers anchor text of a URL. Anchor text helps the crawlers to retrieve documents that cannot be indexed by a text-based search engine, such as images, programs, and databases [5].

Various researches have also suggested combining link-based criteria with similarity-based criteria in accessing page quality. PageRank alone is link-based, i.e. it does not consider the content of a document when deciding the importance of the document. The original proposed PageRank was intended to be used in combination with similarity-based criteria to rank retrieved set of documents. This is how PageRank is used in Google search engine [1,5]. Presently, PageRank is also used to guide crawlers in accessing page quality. Menczer *et al.* proposes a crawler that uses PageRank score to prioritize a link in crawl frontier. The crawler then follows the link with the highest PageRank and it calculates the similarity score for the document with regards to the topic. The crawler maintains a buffer to store crawled pages. If the buffer is full, the page with lowest similarity score is removed [1].

2.2 Evaluating Crawlers

A "perfect" crawler should locate pages that are recent, novel, and authoritative, and it should be efficient, robust and re-locatable. A crawler is robust if the URLs and servers crawled over time overlaps to a great degree when started on different subsets of seed set [1,3]. A re-locatable crawler is capable of executing on a remote host on the network [6].

It is difficult to evaluate the performance of crawlers based on all the above criteria. Firstly, even human experts do not agree on what is a quality page in a specific web site [4]. Secondly, web crawlers have different roles and they are good for different things. Various researches has shown that PageRank, if used alone, is good at identifying authoritative pages while it is too general for topic-driven tasks; InfoSpider [1] locates pages with novelty. BestFirst [1] is good at topic-driven tasks; crawlers based on hub/authority algorithms monitor easily crawled pages for recent relevant changes; and crawlers that use combination of different policies seem work well as well [1,4] .

Menczer *et al.* proposed three crawler evaluation mechanisms which include uses of classifiers, retrieval system and mean topic similarity [1]. Cho *et al.* evaluate their crawlers based on the percentage of high quality pages, judged by human experts, crawled during a period of time [2].

3. Overview of Internet Application Project

For this project, our group has reengineered the homepage of computing science department at the University of Alberta and added other functionality to it. One of the most important improvements we made to these web pages are that they are generated dynamically on the fly using information taken from the database. Another important improvement we made is that we added some semantics to the web pages by allowing students to view courses by categories, by areas, or by professors. Also the web pages are designed and generated following usability guidelines. We have the following modules in our project:

- Security
  Security module checks for user access privileges using password authentication. Different users have different access privileges to other application modules
- Web Crawler (accessible by professors)
  This is our customised crawler that allows users to specify the crawling strategies used for web page retrieval. The output of this crawler is feed to information retrieval package described below.
- Information Retrieval (accessible by course co-ordinator)
  Information retrieval parses files that are retrieved from newsgroup and pages that are crawled by the crawler. It then ranks all the pages and return the pages ranked above a minimal threshold to the browser to answer user queries.
- Course web page generation (accessible by professors)
  Professors can enter course information about a course, and edit the information.
- Teaching Algorithm (accessible by course co-ordinator)
  This module allows course co-ordinator to assign courses to professors using past assignment history, professor preferences, and various constraints.

4. Designing Topic-Driven Crawlers

4.1 Crawler that Combines Various Crawling Strategies

Because topic-driven crawlers are more specialized in certain topics and because of their promises to search deeper in a topic in hypertextual environments [1,4], the crawler we implemented is topic-driven. Specifically, the crawler uses combinations of two different crawling strategies to compute the importance of a page in the frontier and select the most important page for next crawl. These strategies include similarity-based analysis (calculation of similarity score) and link-based analysis (analysis of information get from URL and keywords).

Figure 2 shows the user interface of the crawler. From the interface, users can enter example URLs and seed URLs. Pages retrieved from example URLs are parsed and they form a topic. Starting URLs specify where the crawler is started. Users can also select up to two page importance metrics to use during a crawl: similarity and link based approaches. In the drop down box, each crawling strategy offers three options to

the users including “not considered”, “most important”, and “second important”. If “not considered” is selected, the corresponding measurement criteria will not be used. Otherwise, the measurement is used with the importance level selected by the users.

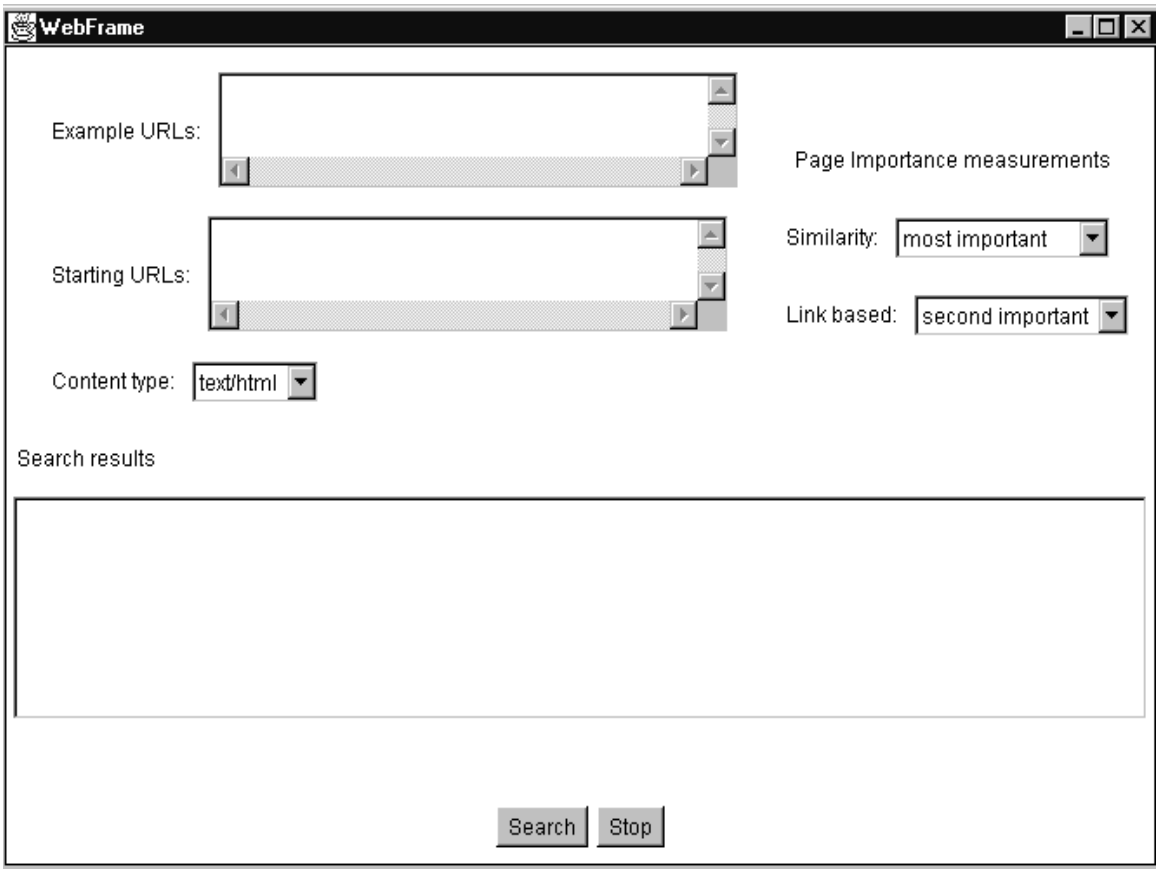

Figure 2. Crawler Interface

Figure 3 offers simplified pseudo-code description of the algorithm. For similarity based analysis, two vectors are maintained at all times. One vector for the topic (Q), the other for the pages the crawler analyzes (P). Each vector element contains the number of times a word appears in a document. Similarity is defined using cosine product of a topic vector and a page vector [1].

$$Sim(q, p) = \frac{\sum_{k \in p \cap q} f_{kq} f_{kp}}{\sum_{k \in p} f_{kp}^2 \sum_{k \in q} f_{kq}^2} \qquad (1)$$

Where k are words that are common both to P and Q. f_{kq} is number of times a word appears in q and f_{kp} is the number of times a word appears in p.

The second measure criteria is based on the information we get from URLs, and the location and font of the keywords. It considers the following factors which are also similarity based:

- Anchor text of URL
- Text surrounding the URL.
- String value of URL

A score based on the above information is also computed for each page. Pages that satisfy the above testing criteria are ranked higher. This strategy might be useful because we want to start our crawler on the homepages of other computing science departments. Given course notes of some specific research area from the University of Alberta, we want to see that the crawler finds similar course notes from other universities. Because most course notes from other universities are in ppt or pdf format, so the best one can expect a crawler to return is the higher level html page that contains these course notes. An efficient way to find these top level index pages are to look at the text value surrounding the URL, string value of the URL, and anchor text of the URL.

The algorithm first parses all example pages to form a topic and adds all the seed URLs to its crawler frontier. An importance score for each page in the crawler frontier is calculated. The importance score is a function of similarity score (calculated using equation 1) and score calculated using information gathered from URLs. This function depends on the importance level of each crawling strategy specified by the user. At any given time, the crawler selects from the frontier the page with the highest importance score to traverse next. We specify a search limit (MAX_PAGES in figure 3) for the crawler: the number of links the crawler should visit before it stops. So if we put the least important pages at the end of the queue, then we guarantee that more important pages will always be processed first. And if the crawler stops, we know that the links left in the queue are less important ones.

```
combination_crawl (example urls, starting urls,  user preferences) {

form topic(example urls);

while (#frontier > 0 and visited < MAX_PAGES) {
    foreach link (frontier) {
       doc := fetch_new_document (link);
       sim(doc, topic) := compute_sim_score();
       linkScore(doc) := compute_linkScore();
       score(page) := user_defined_fuction (sim, pageRank, linkScore);
   }
    sortQueue(frontier);
    link := deleteFromQueueHead(frontier);
    doc := fetch_new_document (link);

    foreach outlink (extract_new_links(doc)) {
        enqueue (frontier, outlink);
  }
 }
}
```

Figure 3: Pseudo-code of the crawler that uses combination of three crawling strategies

4. 2 Evaluating the Crawler

I have run three sets of the experiments on the crawler. The example URLs are some of the course notes of COMPUT301 that are offered at the University of Alberta, and I started the crawler on the computing science department homepage of Simon Fraser University. Each experiment uses a different combination of crawling strategies. The first experiment uses similarity ranking as most important and link based ranking as second important. The second run uses similarity based ranking alone, while the last experiment uses link based ranking alone.

Because of the crawler was using different page evaluation metrics, it visits links in different orders and get different sets of files before it stops. If similarity based approach is used alone, it returns the most number of web pages before it stops and using link based approach alone returns approximately the same amount of web pages that are returned by using two strategies together.

The crawler using similarity based approach alone goes into detailed course notes a lot faster than the other two crawlers did. It retrieves the higher level web page that contains a list of the course notes first and then retrieves these course notes before it continues parsing the higher level index page. I think this is because of detailed course notes have higher similarity scores computed against the example course notes than the scores of

higher level pages which contain a bunch of links and links are not considered in this case.

The crawler that uses link based approach alone tends to finish parsing the entire high level pages that contain links to all the courses before it goes into any specific page. This is reasonable because in this case, link is the only information it considers and similarity between the two documents does not count. Most detailed course notes have no or little outgoing links, and the higher level index pages are favoured in this case.

Using the combination of link based and similarity based approach seems to average things out. Since the crawler is considering both page content and the link to a page, the order that it visits high level pages and detailed course notes intervene. Because the combination of strategies does not favour any kind of pages, it takes a fairer view of the hyperlink hierarchy.

In general, web crawlers that are implemented for different web sites have different notion of importance and goals. When considering which strategy a crawler should use, the rule of the thumb probably is to ask first what's the most important goal of the web crawler in question.

5. Future work

Because of the limited time and resources available, I did not do extensive experiments of the web crawler. And there are more sophisticated crawling strategies that the crawler could have used. The future work would be trying other crawling strategies such as PageRank, which has a reputation for locating high authoritative pages. Also more experiments are needed to draw more interesting conclusions.

6. Conclusion

In this paper, we have surveyed the current status of research in developing efficient web crawlers. We have presented the implementation of a topic-driven crawler that uses different combinations of importance metrics for judging page quality. Each importance metrics used has been proven to be efficient from previous work. We have run some experiments and drawn some conclusions of each experiment. Little empirical work has been done in evaluating different algorithms and the researches have been presenting different conclusions. The conclusion that we have drawn from our work adds up to our knowledge of the performance of each crawling strategy.

References:

[1]. F. Menczer, G. Pant, P. Srinivasan and M. Ruiz. Evaluating Topic-Driven Web Crawlers. In *Proceedings of the 24th Annual International ACM/SIGIR Conference, New Orleans, USA*, 2001.

[2]. J. Cho, H. Garcia-Molina, and L.Page. Efficient crawling through URL ordering. In *Proceedings of the Seventh International World Wide Web Conference, Brisbane, Australia*, 1998.

[3]. S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: A new approach to topic-specific web resource discovery. In *Proceedings of 8th International World Wide Web Conference*, 1999.

[4]. B. Amento, L. Terveen, and W. Hill. Does "authority" mean quality? Predicating expert quality ratings of web documents. In *Proceedings of 8th International World Wide Web Conference*,1999.

[5]. S. Brin and L.Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International World Wide Web Conference, Brisbane, Australia*, 1998.

[6]. Robert C. Miller and Krishna Bharat. SPHINX: A Framework for Creating Personal, Site-Specific Web Crawlers. In *Proceedings of the Seventh International World Wide Web Conference, Brisbane, Australia*, 1998.

[7]. T. Haveliwala. Efficient Computation of PageRank. Technical report, Stanford Database Group, 1999.

`