International Journal of Engineering Research and General Science Volume 2, Issue 4, June-July, 2014
ISSN 2091-2730

Design of a Focused Crawler Based on Dynamic Computation of Topic Specific Weight Table

Meenu [1], Priyanka Singla[1], Rakesh Batra[1]
[1]Dept. Of Computer Science & Engineering, YMCA Institute of Engineering and Technology, Faridabad, India
E-mail: mahi.batra11@gmail.com

Abstract - Focused Crawler aims to select relevant web pages from internet. These pages are relevant to some predefined topics. Previous focused crawlers have a problem of not keeping track of user interest and goals .The topic weight table is calculated only once statically and that is less sensitive to potential changes in environment. To address this problem we design a focused crawler based on dynamic computation of topic keywords and their weights. This weight table is constructed according to user query. To check the similarity of web page with respect to the topic keywords, a cosine similarity function is used and priority of extracted links is calculated.

Keywords - Crawler; Focused Crawler; Topic Weight Table; Link Score; Search Engine; page score; user.

1. INTRODUCTION

Web Crawler is a continuously running program which downloads pages periodically from World Wide Web. It is also known as Web Spider or as a Wanderer. These downloaded pages are indexed and stored in a database. Later on, these pages are used by search engine to find the information related to search query. A recent study estimate that the size of visible web has passed over billions of documents and list is still increasing. Due to enormous growth and changing in the web, it becomes difficult for a search engine to keep up index fresh. Even a popular search engine like Google crawl only 40 % of whole web [1]. So to avoid this problem, we need a crawler which crawl a specific and relevant subset of World Wide Web. There is need of a crawler which efficiently and effectively works with respect to limiting resource and time [7].Focused crawler is a crawler also known as topical web crawler which downloads only relevant pages from World Wide Web. These pages are relevant to the set of topics defined. It was first introduced by Chakrabarti et al.[2]. Focused crawler predicts the relevancy of page at two places: (a) before downloading (b) after downloading. Before downloading it predicts the relevancy of page by seeing the anchor text of links; this approach is given by Pinkerton [3][8], also known as link based analysis and after downloading by seeing the content of page, known as content based analysis. Relevant pages are stored in a database and their contained URL is added to URL queue. However, most focused crawler use local search algorithm such as best-first search or breadth – first search to determine the order by which target URL are visited[4]. Focused crawler is a useful for application such as distributed processing of web. It is also used in personal search engine, web database and commercial intelligence.

In this paper a focused crawler has been designed based on dynamic computation of topic keywords & their weights. It constructs topic weight table according to user query. Thus, it allows the final collection to address the user information needs. The outline of this paper is as follows: sections 2 provide the brief discussion of existing crawler and issues related to that crawler. Section 3 describes our proposed work. In Section 4 results have been obtained and compare with existing crawler and give some experimental result. In Section 5 conclusion and suggestion for future direction have been presented.

2. RELATED WORK

Focused Crawler is heavily depending upon topical locality phenomenon [5]. Topics offer a good mechanism for evaluating the relevancy of page. Topic is the set of keywords with their associated weights. The topic vector can be written as given in equation (1).

$$\text{Topic} = \{(k_1, w1), (k_2, w2) \ldots (k_n, w_n)\} \quad (1)$$

Here k_1, k_2.., k_n are keywords and w_1, $w_2 \ldots$ w_n are the weights associated with these keywords. Topics may be obtained from different sources such as asking user to specify them. But users are unwilling to specify the topics because of requirement of additional effort and time. Anshika Pal et al. [9] proposed a method for topic specific weight table construction. Topic name is given to Google web search engine and first few results are retrieved. After that term frequency and document frequency of words are calculated and each word is assigned weight $w_i = tf * df$. After that their weights are normalized using the following eq. (2).

$$W_{i+1} = W_i / Wmax \quad (2)$$

Where, Wmax is the maximum weight assigned to any keyword and W_{i+1} is the new weight assigned to each keyword. After construction of topic specific weight table, page relevancy is calculated based on content and link analysis as proposed in [5]. A critical look at the available literature indicates the following limitations:

1. The present crawlers rely on Keywords weights which are computed once statically.
2. Results are less relevant to user interest and goals.
3. It is not sensitive to potential alteration in the environment.

In order to make search results more relevant to user interest a mechanism has been proposed which dynamically computes topic specific weight table. And dynamic computation leads to a higher relevancy of a page according to user.

3. PROPOSED ARCHITECTURE

In order to get results according to user interest and more sensitive to potential changes in the environment section 3.1 describe dynamically construction of topic specific weight table.

3.1 Topic Weight Table Construction:

Fig.1 shows the Process of topic specific weight table construction

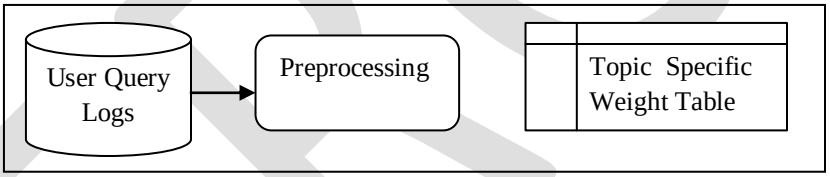

Fig. 1: Topic Specific Weight Table Construction

User Query log is a place where all the queries fired by users are stored. In preprocessing stage, tokens are extracted from queries and normalization of token is done. After that topic specific weight table is constructed. The topic specific weight table is reconstructed after fixed interval of time by applying same procedure .The algorithm related to weight table construction is given below.

3.1 .1 Weight Table Construction algorithm

As we know that user query is the most important source of knowing user interest. For knowing the relevancy of page according to user interest, it constructs weight table with the help of user query. The proposed algorithm is given below.

***Step 1*:** Crawler access the user queries from user query logs.
Step 2: Tokenize the queries collected by crawler. /* Tokenization is the task of chopping a query in to pieces called tokens. It will throw away certain characters such as punctuation.*/
Step 3: Drop common terms such as stop words. /* Process of dropping common words is known as stop listing. These words have a little value for knowing the user interest. Most of the commonly used stop words are " for", "it", "the", "a", "can", "do", "did", "will", "shall" etc. */
***Step 4*:** Do linguistic processing such as lemmatization/stemming and producing a list of normalized tokens. /* The main aim of both stemming and lemmatization is to reduce inflectional form and sometimes derivationally related form of a word to a common base form. */

***Step 5*:** Sort the terms alphabetically either in ascending or in descending order.

***Step 6*:** Multiple occurrences of same terms are merged. This step also records some statistics such as query frequency which is the number of queries which contain each term.

***Step 7*:** After that it calculate the weight of each term as given in equation (3).

$$W_i(new) = (1-\alpha)qf + \alpha W_i(old) \quad (3)$$

Here, α is constant whose value lies in between $0<\alpha \leq 0.5$, qf is the query frequency of each term and $W_{i(old)}$ is the weight of term if that term occur in previous weight table if not occurred previously then taken it as 0 and $W_i(new)$ is the current weight of term.

***Step 8*:** Terms whose weight is greater than or equal to threshold value is taken as keyword for knowing the relevancy of a page.

Example: Suppose we have set of following sample queries taken from query log.

Q1: Kejriwal new manifesto for lok sabha polls 2014.
Q2: Manifesto of BJP for lok sabha polls 2014.
Q3: lok sabha polls 2014 dates

Apply step by step procedure on the above given queries. Up to step 6 following result will come.

Terms	Query Frequency
"BJP"	1
"date"	1
"kejriwal "	1
"lok sabha"	3
"manifesto"	2
"new"	1
"poll"	3
"2014"	3

Let us suppose α=0.5 and w_i(old) of terms is 0.By applying the formula given in equation (3) following results will come.

Terms	Weight
"BJP"	0.5
"date"	0.5
"kejriwal"	0.5
"lok sabha"	1.5
"manifesto"	1.0
"new"	0.5
"poll"	1.5
"2014"	1.5

Let us suppose threshold value for topic keyword weight is 1.0. Thus final topic specific weight table is given in Table I

Table I: Weight Table

No	Keyword	Weight
1	lok sabha	1.5
2	poll	1.5
3	manifesto	1.0
4	2014	1.5

This Weight Table is used by relevancy calculator as shown in fig.2.

3.2 Crawling Process

The process of crawling after topic specific weight table construction is shown in fig. 2. The detailed description of each component is given below.

3.2.1 Seed URLs Generation

Here, seed URLs are generated by one search engine www.threesearch.com. We put the topic keyword here and it show the result of three most popular search engine Google, Yahoo, MSN .We take the seed URLs which are common in all the three search engine. Initially these URLs are given to URL Frontier .

3.2.2 URL Frontier

It is a data structure that contains all URLs that remain to be downloaded. Here, it is used priority queue instead of simple queue.

3.2.3 Web page downloader

This module built the connection with the internet and downloads the pages corresponding to the given URL using appropriate network protocol and store temporarily in document buffer. After that it gives signal: Something to test to the content seen test

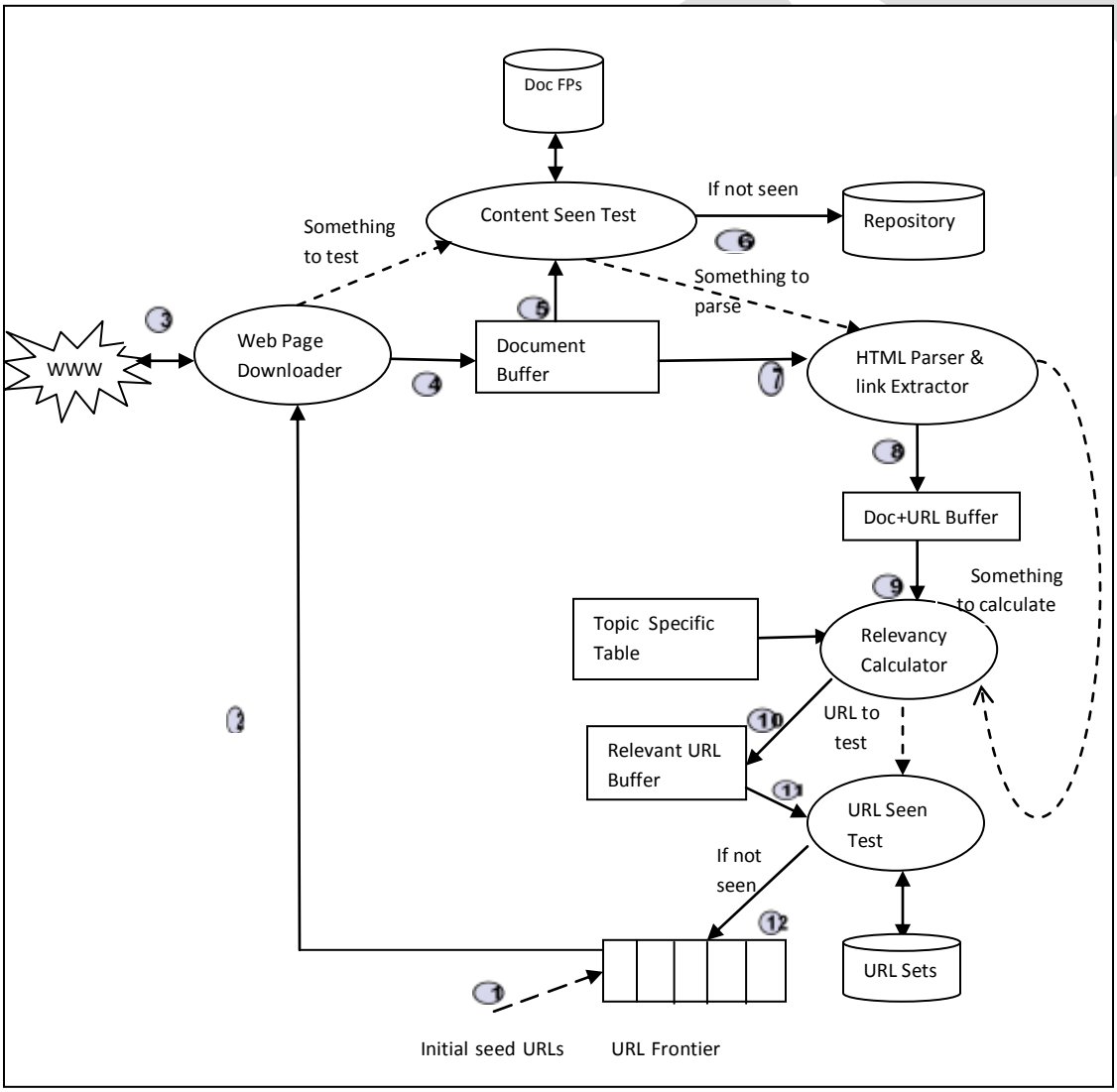

Fig.2 Crawling Process

3.2.4 Content Seen Test

Many documents available on the web under multiple different URLS .These effects will cause any crawler to download same document multiple times. To prevent downloading of a document more than once, web crawler wishes to perform content seen test. Using content seen test it is possible to suppress link extraction from mirrored pages which may result in significant reduction in number of pages that needs to be downloaded. The content seen test will be expensive if we match complete documents. In order to save space and time, we maintain a data structure called document fingerprint set that store 64 bit checksum of the content of each downloaded document. If the document is downloaded before, it reject that page and next document is downloaded by downloader otherwise page is stored in repository and signal is given to HTML Parser and Link Extractor

3.2.5 HTML Parser & Link Extractor

Once a page has been downloaded we need to parse its content to extract the information that will guide the future possible path of the crawler. In order to extract hyperlink URL from a web page, anchor tags and other related information we can use these parser. After that page, its extracted links and other information are stored in buffer and signal: Something to calculate is given to Relevancy Calculator.

3.2.6 Relevancy Calculator

This module calculates the relevancy of page corresponding to topic keyword in the table by using equation (4). Here, it uses cosine similarity to calculate the relevancy of page:

$$\text{Relevancy(t,p)} = \frac{\sum_{i=1}^{n} CWi\ (t) \times \text{CWi}\ (\text{p})}{\sqrt{\sum_{i\in t} W_i^2(t)} \times \sqrt{(\sum_{i\in t} W_i^2(p)}} \quad (4)$$

Where , CWi(t) and CWi(p)are the weight of i-th common keyword in weight table t and web page p respectively, and Wi(t) and Wi(p) are the weight of keyword in web page p and weight table t respectively. If the relevancy score of page is greater than threshold value then Link Score of its extracting links are calculated by using equation (5).

$$\text{LinkScore (k)} = \alpha + \beta + \gamma + \omega \quad (5)$$

Where LinkScore (k) is score of link k, α= URLScore (k) is the relevancy between topic keywords and href information of k, β = AnchorScore (k) is the relevancy between topic keywords and anchor text of k, γ=ParentScore (k) is the page relevancy score of page from which link was extracted and ω=SurroundingScore (k) is the relevancy between text surrounding the link and topic keyword. The links whose score is greater than threshold is considered to be relevant. Relevant URLs and their score is stored in relevant URL buffer and signal is given to process URL seen test.

3.2.7 URL Seen Test:

In the course of extracting links, crawler may encounter duplicate URLs .To avoid downloaded of document more than once URL seen test must be performed on extracted links before adding to URL Frontier .To perform URL seen test , We store all URLs seen by crawler in canonical form in a table called URL set. To save space and time, it does not store textual representation of each URL in the URL set but uses a fixed size checksum which are stored in disk. To increase the efficiency, we keep in-memory cache of most popular URLs.

4. EXPERIMENTAL RESULTS

Generally, Harvest-ratio is used to measure the performance of focused crawler. Harvest-ratio is also known as precision metric of crawler. It can be defined as percentage of crawled pages that are relevant to specific topic.

$$\text{Harvest} - \text{Ratio} = \frac{\#\ \text{Of Relevant Pages}}{\#\ \text{Of Downloaded Pages}} \quad (6)$$

The harvest-ratio of present crawler has been calculated by using formula given in equation (6) and compared with the basic crawler and focused crawler based on static weight table. Table II shows the harvest ratio of three crawlers at different no. of crawled pages.

Table II

No. of Crawled pages	Harvest-Ratio		
	Basic Crawler	Focused Crawler Based On Static Computation Of Topic Weight Table	Focused Crawler Based On Dynamic Computation Of Topic Weight Table
500	0.3	0.8	1
1000	0.25	0.78	0.95
1500	0.27	0.79	0.92
2000	0.2	0.77	0.9
2500	0.18	0.74	0.88
3000	0.19	0.75	0.91
3500	0.15	0.73	0.87
4000	0.15	0.7	0.87

The illustrated crawled result is shown on two dimensional graphs where x-axis is the number of crawled pages and y-axis is the Harvest-Ratio calculated as shown in fig. 3

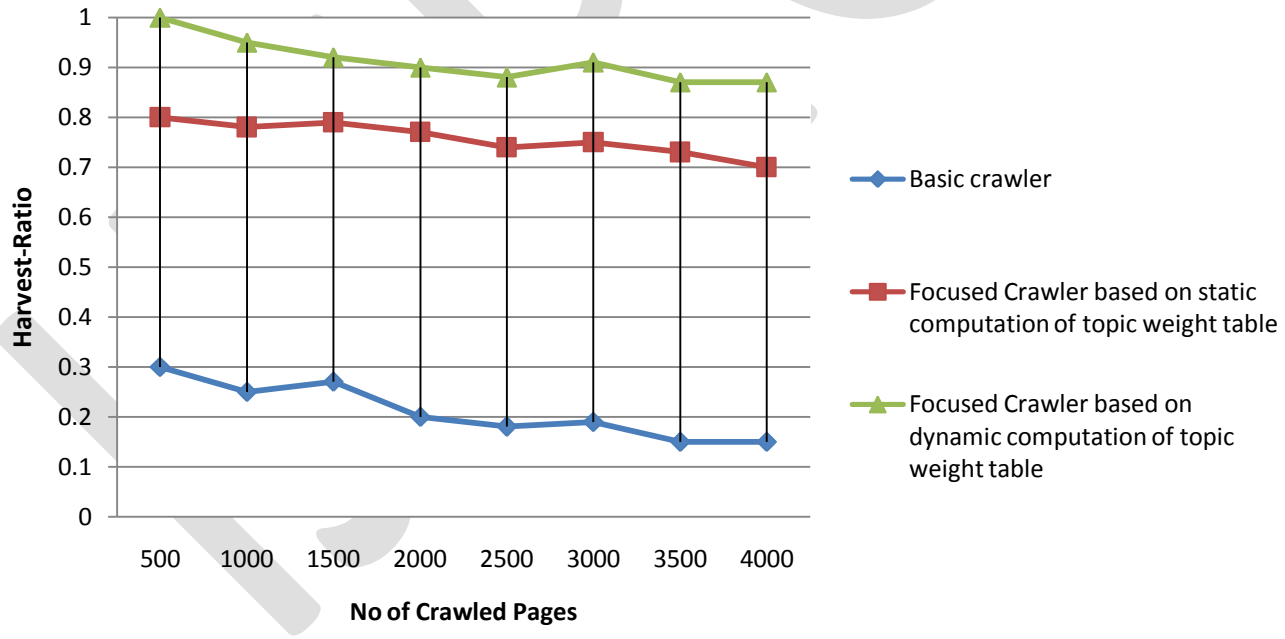

Fig. 3: Experimental Result

5. CONCLUSION AND FUTURE WORK

Focused Crawler is the main component of special search engine. A focused crawler selectively seeks out and downloads web pages that are relevant to the search topic. Our approach is based on dynamic computation of topic specific weight table of focused crawler. Here, Weight table is built according to user query. Thus, it gives results which are more relevant to user. This approach does not consider the context related to keyword of topic. In our Future work, we will try to consider context related to keywords and also do

code optimization because crawler efficiency not only depend to retrieve maximum relevant page but also to finish the operation as soon as possible.

REFERENCES:

[1] S. Lawrence and L. Giles, "Accessibility and distribution of information the web[j]," *Nature*, vol. 400, pp. 107–109, 1999.

[2] S. Chakrabarti, M. Van Den Berg, B. Dom, "Focused Crawling: A New Approach to Topic specific web resource discovery", Proc. Of 8th International WWW conference, Toronto, Canada, May, 1999.

[3] B. Pinkerton, "Finding what people want: Experiences with the web crawler" in In Proceedings of the First International World-Wide Web Conference, Geneva, Switzerland, May 1994.

[4] Ah Chung Tsoi , Daniele Forsali ,Marco Gori, Markus Hagenbuchner and Franco Scarselli, " A simple Focused Crawler " , WWW 2003 ACM, 2003.

[5] X.Chen and X. Zhang, "HAWK: A Focused Crawler with Content and Link Analysis", Proc. IEEE International Conf. on e-Business Engineering, 2008.

[6] Li Wei-jiang, Ru Hua-suo, Zhao Tie-jun,Zang Wen-mao ," A New Algorithm of Topical Crawler", Second International Workshop on Computer Science and Engineering 2009.

[7] M. Kumar and R. Vig, "Design of CORE: context ontology rule enhanced focused web crawler", International Conference on Advances in Computing, Communication and Control (ICAC3"09) pp. 494-497, 2009.

[8] Brin, S. and Page, L. (1998) "The Anatomy of a Large- Scale Hyper textual Web Search Engine," Computer Networks and ISDN Systems, 30(1–7).

[9]Anshika Pal , Deepak Singh Tomar , S.C Shrivastava " Effective Focused Crawler Based on Content and Link structure Analysis", International Journal Of Computer Science and Information Security", vol.2,no.1.june 2009.

[10]Debashis Hati , Amritesh Kumar ," An Approach for Identifying URLs Based on Division Score and Link Score in Focused Crawler" International Journal of Computer Applications , Volume 2 – No.3, May 2010.

[11]Jaytrilok Choudhary and Devshri Roy "A Priority Based Focused Web Crawler" International Journal of Computer Engineering and Technology, Volume 4, Issue 4, July-August 2013.

[12] Mohsen Jamali, Hassan Sayyadi, Babak Bagheri Hariri and Hassan Abolhassani," A Method for Focused Crawling Using Combination of Link Structure and Content Similarity"