



An Improved Architecture for Parallel Focused Crawler Using Mobile Agents

Dr. Naresh Kumar

Assistant Professor, Maharaja Surajmal Institute of Technology (MSIT),
Janakpuri, New Delhi, India

Abstract: In day to day life Web has become a crucial part of our life. The exponential growth of the web provides the problems in terms of load on the network, time consumption, bandwidth preservation and refreshment of the repository. It has become the major concern for both search engine developers and its users also. So, to overcome above said problems, this paper proposes Parallel Focused crawler which has the features of Parallel as well as Focused Crawler. It will download the web pages related to the specific domain and reject the ir-relevant pages.

Keywords: Search engine, dynamic web pages, Internet traffic, WWW, mobile crawler, parallel and focused crawler

I. Introduction

Today most important source of information is WWW which is increasing from thousands of pages to more than billions of pages [1]. According to a latest survey of 20 July 2016 of [2] the size of indexed repository is reached to 4.75 billion pages. The Search Mechanisms (SM) like Search Engines (SE) and Meta Search Engine are designated to search the required information from these repositories [3]. These SM uses a process called Crawler [4] (also known as bot, spider, worm, walker and wanderer) for searching the information from WWW. But due to large number of pages available in the repository general crawler downloads irrelevant pages in bulk [5]. As a study of [6], 40% contents of the web are changed daily. Moreover, 40% of Internet traffic is due to crawlers of different SM [7]. Lots of suggestions were given and adopted while designing the crawlers to overcome these problems [8][9][10][11]. But still the problems are remains same. So, this paper proposes an alternate approach to maintain the repository with important pages and reduce the load on the network.

II. Related Work

An approach called PDFC was used in [3] where the concept of ASCII count was used to resolve the problem of load on the network, refreshment of repository and bandwidth utilizations. They implement the proposed approach using Java programming language using Netbeans IDE 7.0.1. The experimental results of PDFC show the detection of minor to minor change like even a single full stop was also detected. Moreover, load on the network and bandwidth preservation was also improved to some extent.

In [10], the web page change behavior was checked for 30 days then probability of page change is calculated. The main objectives of the authors were to analyze the load on the network, page change behavior and protection of bandwidth. For this purpose they use three things – number of URLs, number of keywords and last modified date. The results of the proposed approach shows 63% change in page change behavior which straightaway declares the reduction of 37% load on the network. It also shows reduction in usage of bandwidth by using compression at the server end.

To reduce the traffic load on the network an uninterrupted crawling process was proposed in [12]. The outdated pages were avoided by using a three step algorithm. The authors resolve the problems of frequent image changes by converting image to bit map and calculating i-value of the image. The proposed approach has been implemented and applied on the contents of a web page. The result of this approach shows the addition of six words in the page. Divakar Yadav et al. proposed a parallel crawler architecture with Web page change detection in [13]. The main stress of this paper was to check whether the contents or structure of Web page has been changed or not. The proposed architecture consists of two main components i.e. multi-threaded server (MTS) and a client crawler. MTS was responsible for downloading the Web pages whenever requested by the clients. They provide no inter client communication. Any Web page that had been changed since the last crawl was considered for downloading. A Web page was considered as changed if any change in page structure, change in text contents, or change in image was observed. The concept of tree was used to detect the change in page structure. They compared old and new tree developed for corresponding Web page. If both are same then the pages was compared on change in text contents. At this stage they considered ASCII value of characters, frequency of terms, count of total characters and distinct character in the Web page. If again these were also same then the comparison was made on the bases of images. For this purpose they developed a code for images to determine whether they get any change or not. Generally, a change in a link to an image hyperlink will be reflected in the label of the hyperlink for that image. But what

happens if the text does not get any change but the image gets replaced with another image? In this case, it remained undetected. So, they proposed a formula for detection of change in the image. At the end in their conclusion they claimed about the achievement of reduction in redundancy of Web pages, reduction in load on the network and updation of central repository with updated Web page only.

Another focused crawling approach was also used in [14] which can crawl the pages on a specific topic. The querying policy was applied on the syllabus of computer science. The result of proposed approach shows good level of precision and relevancy of results.

III. Problem Formulation

The above discussed crawling approaches have the following problems:-

- Iterative nature of repository updation affects the efficiency of the crawler.
- The use of trees is a very complicated approach and slow in nature.
- How many times a web pages get changed since the last crawl is not determined in [3][10] while providing the updated page in SE repository?
- How many no. of mobile agents are required to be created so that maximum efficiency will be determined?
- While crawling; if one mobile agent is overloaded and another is free than how to utilize free agent optimally?

IV. Proposed Parallel Focus Crawler

The mobile crawlers are sent to the Remote Site (RS) where they can retrieve the document, process them and remove the unchanged pages. Lastly, the modified pages are compressed and forwarded to the SE. Figure 1 shows the architecture of proposed Parallel Focused Crawler (PFC). The main components of PFC are Crawler Executor (CE), Link Extractor (LE), Data Manager (DM), File Creator (FC), De-Compressor, Load Adjuster (LA) and Crawling Process (CP). Each of these components is discussed below:

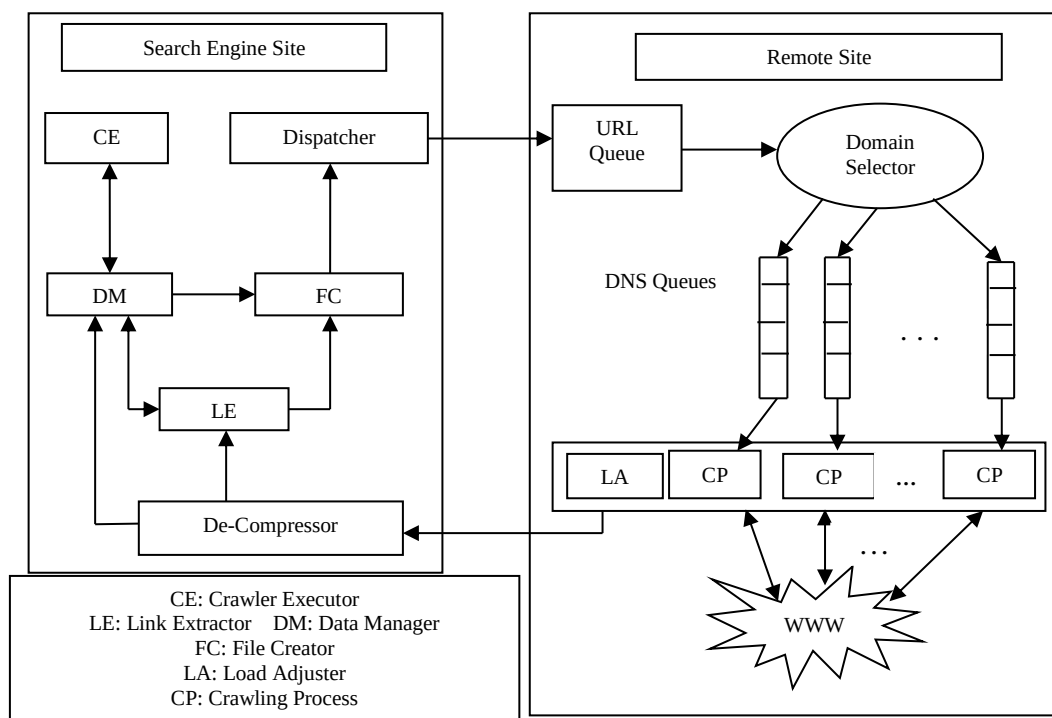
(a) Crawler Executor: CE creates required number of mobile crawlers, provide URLs to it for crawling by taking the information from Page Change Estimator (PCE) which is available in DM.

(b) Domain Selector: DS takes the URL from URL queue and determines its corresponding domain name like .edu, .com etc. After this each URL is forwarded to the corresponding DNS queues for further processing. The time of downloading and size of each document to be crawled may vary from one URL to another URL.

(c) Data Manager: DM is central repository of SE which contains and maintains all the pages crawl by the crawler. This module has all the modules which are available in Statistical Database of [3].

(d) Page Change Estimator: PCE is maintained at the SE side and is used to predict the dynamic nature of the page. In short it is used to check the probability of page change. The parameters and method of change detection used by PCE are same as used by Frequency Change Estimator in [3].

Figure 1: Proposed Parallel Focused Crawler



(e) File Creator: CE creates one FC for each mobile agent that contains the information about each web page to be crawl. The remaining contents of FC are same as it available in ODF of [3].

(f) Link Extractor: LE takes the original web page from De-Compressor and extracts the available links. It then checks the document corresponding to the extracted link is available in the DM or not? If the link is not then it send this URL to FC for further processing otherwise its time of change will determine whether to send this URL to FC or not.

(g) Load Adjuster: This module will work on server side. The main task of this module is to adjust the load of overloaded CP. If any CP has heavy load of crawling and another is free then LA assigns some load of overloaded CP to other CP which is currently free. Further, if CP download the web page and detect any change in the downloaded web page then it will compress the page and then send the page to SE.

V. Working of PFC

The working of proposed architecture of PFC is divided into two cases: (1) When PFC runs for the first time (2) When PFC starts re-crawling the Web. Each of these cases is explained with sufficient detail given below:

(a) When PFC runs for the first time: The crawling process is initiated by providing the initial URL and CE generates an MC for downloading the corresponding webpage. Dispatcher then gives this URL to URL queue available in the RS. The Domain Selector (DS) takes the URL from the URL Queue and determines the domain of the URL by using Domain Name Server. It then stores it in the Domain Name Server Queue (DNS Queue) corresponding to the domain. Numbers of DNS Queue are equal to the number of domains. Then CP extract the document corresponding to the URL. Then this webpage is forwarded to the LA module where ASCII value of the web page is determined. When there is no ASCII count available with the MC then LA assumes that it is a new document so it will download the document without any comparison. At the SE side document will be received by the De-Compressor which will decompress the document and sent one copy of it to DM and one copy to LE for further processing.

(b) When PDFC starts re-crawling the Web: Once the process of crawling is initiated, as discussed in Case (1). The URLs extracted from the first webpage are checked whether they are already traversed or not. If they are not traversed earlier then Case 1 is repeated otherwise they are forwarded to the FC. FC will check whether the probability of change is reached or not. If it is yes then CE generates MC and sends the URL and associated code to the URL queue via Dispatcher. DS places it into its corresponding DNS Queue. The downloaded web page given to the LA to calculate and compares the old and new ASCII count of the web page. If both old and new ASCII count is found same then the webpage is rejected otherwise it compresses the webpage and sends it to the SE. This process is repeated until all the DNS Queue becomes empty.

VI. Conclusion

This paper has proposed an architecture to resolve the problems related to the Parallel Domain Focused Crawler. It combines parallel and focused crawling techniques to reduce the load on the network. Further it will also help to maintain the fresh and up to date pages in the central repository of SE. Moreover this architecture will also be helpful in bandwidth preservation. The proposed architecture has the characteristics like Full Distribution, Scalability, Load Balancing and Reliability. All these parameter is easy to achieve because PFC provides only refreshed page(s) and forward them on the network in compressed form.

VII. Future Work

The author continually exercising on improving the proposed architecture. But, prior to implementing it several issues like crawling communication between different processes, load adjuster and better compression technique need to be identified so that better results can be achieved.

VIII. References

- [1] V. Shkapenyuk and T. Suel, "Design and Implementation of A High Performance Distributed Web Crawler", in Proceedings of the 18th International Conference on Data Engineering, San Jose, California. IEEE CS Press, pp. 357-368, 2002.
- [2] URL: www.worldwidewebsite.com
- [3] Rajender Nath and Naresh Kumar, "A Novel Parallel Domain Focused Crawler for reduction in Load on the Network", in International Journal Of Computational Engineering Research (ijceronline.com) Vol. 2 Issue. 7, pp. 77-84, November 2012.
- [4] Jungoo Cho, Hector Gracia-Molina, "Parallel Crawlers", in the proceedings of 11th international World Wide Web conference, Technical report, UCLA Computer science, 2002.
- [5] Naresh Kumar et al., "Web Crawler Challenges and Their Solutions", in National Conference on Advances in Computer Science and Engineering (ACSE-2016), Department of Computer Science and Applications, Kurukshetra University Kurukshetra, April 2016.
- [6] J. Cho and Garcia-Molina H., "Estimating Frequency of Change", in ACM Transactions on Internet Technology (TOIT), vol. 3, no. 3, pp. 256-290, 2003.
- [7] X. Yuan and Harms J., "An Efficient Scheme to Remove Crawler Traffic from the Internet", in Proceedings of the 11th International Conferences on Computer Communications and Networks, pp. 90-95, 2002.
- [8] Nidhi Tyagi & Deepti Gupta, "A Novel Architecture for Domain Specific Parallel Crawler", in Nidhi Tyagi et. al. / Indian Journal of Computer Science and Engineering Vol 1 issue 1, pp 44-53, 2010.

- [9] Odysseas Papapetrou and George Samaras, "Minimizing the Network Distance in Distributed Web Crawling", R. Meersman, Z. Tari (Eds.): CoopIS/DOA/ODBASE 2004, LNCS 3290, pp. 581–596, 2004.
- [10] Rajender Nath and Satinder Bal, "A Novel Mobile Crawler System Based on Filtering off Non-Modified Pages for Reducing Load on the Network", in The International Arab Journal of Information Technology, Vol. 8, No. 3, July 2011.
- [11] Rajender Nath and Naresh Kumar, "A Novel Architecture for Parallel Crawler based on Focused Crawling", in International Multiconference on Intelligent Systems, Sustainable, New and Renewable Energy Technology & Nanotechnology (IISN 2012), pp.36-39, March 2012.
- [12] Manuel Alvarez et al., "Deep Bot: A focused Crawler for Accessing Hidden Web Content", Published in DEECS 2007: June 12, 2007, San Diego, California, USA. ACM 978-1-59593-856-5/07/06.
- [13] Divakar Yadav, Sharma A. K. and Gupta J. P., "Parallel Crawler Architecture and Web Page Change Detection", in World Scientific and Engineering Academy and Society (WSEAS), Transactions on Computers, ISSN: 1109-2750, Issue 7, pp. 929-940, 2008.
- [14] T. Guilherme de Assis · Alberto H. F. Laender, Marcos André Gonçalves · Altigran S. da Silva, "A Genre-Aware Approach to Focused Crawling", in WorldWide Web (2009) 12:285–319, DOI 10.1007/s11280-009-0063-7.