Journal of Computer Science

Original Research Paper

Highly Efficient Architecture for Scalable Focused Crawling Using Incremental Parallel Web Crawler

[1]P. Jaganathan and [2]T. Karthikeyan

[1]*Department of Computer Application, PSNA College of Engineering and Technology, Dindigul, India*
[2]*Department of Research and Development Centre, Bharathiar University, Coimbatore, India*

Article history
Received: 09-02-2014
Revised: 17-02-2014
Accepted: 04-08-2014

Corresponding Author:
P. Jaganathan
Department of Computer Application, PSNA College of Engineering and Technology, Dindigul, India
Email: sudharsan@axiip.com

Abstract: With the growing industrial impact over the recent years in computer science, data mining has established itself as one of the most important disciplines. In the fast growing Web and in an appropriate amount of time, locating the resources that are precise and relevant is a huge challenge for the all-purpose single process crawlers, which makes the enhanced and the convincing algorithm in demand. Gradually Large scale search engines frequently update their index and in a timely behavior which are not capable to present such information. In this study a scalable focused crawling is proposed with an incremental parallel Web crawler, the Web pages can be crawled concurrently that are relevant to multiple pre-defined topics. Furthermore, to solve the issue of URL distribution, a compound decision model based on multi-objective decision making method is introduced, which will consider multiple factors synthetically such as load balance and relevance, the update frequency issue can be solved by the local repository decision. The result shows that our proposed system will efficiently produce high quality, relevance and freshness with significantly low memory requirement.

Keywords: Focused Crawler, Incremental Web Crawler, URL Distribution Issue, Load Balance, Relevance

Introduction

A program that retrieves and stores Web pages from the Web is called as a Web crawler. Unprecedented scaling challenges for all-purpose single-process crawlers' plays the major role in the growth of World Wide Web as said by (Chakrabarti *et al*., 1999; Kumar *et al*., 2013). To finish the downloading pages in a reasonable amount of time, a new hypertext resource discovery system is used which is called as a focused crawler, which selectively seek out pages and the set of topics which are relevant pre-defined. Another new crawler called parallel crawler is proposed which crawl the multiple processes in parallel as said by (Balamurugan *et al*., 2012) Due to the high dynamic nature of Web documents, to acquire useful information and to integrate data, local repository freshness should be maintained, this makes the web pages to crawl consistently. There is a significant waste of time and space, whenever we make full crawling as said by qiang (Zhu, 2007; Mannar Mannan *et al*., 2014). To overcome this incremental crawler was proposed, instead of crawling all web pages, it selectively and incrementally updates the local repository. From this it is clear that the crawler should have the certain objectives.

The Web pages crawled should have high quality, high relevance and high freshness. To achieve these objectives in this study we proposed an optimized novel architecture for the incremental parallel crawler based on focused crawling as said by (Cho and Garcia-Molina, 2002). The major contribution of this study is summarized as follows: First, an optimized novel architecture based on focused crawling for incremental parallel crawler is proposed, which helps to crawl the Web pages that are relevant to multiple pre-defined topics concurrently. Then the solution is found in incremental parallel crawler for core issues like URL distribution and the update frequency as said by (Shkapenyuk and Suel, 2002; Avraam and Anagnostopoulos, 2011) and to compute the URL priority, a novel approach is proposed to selectively fetch higher quality relevant information, in which old and new URLs are differently treated. Then in the proposed architecture, they implemented the second level master,

which will avoid the overlapping issues and also reduces the cost of communication and space largely.

Focused Crawlers over General Crawlers

General Crawler is used mainly for search engines, whose ultimate aim is to meet out the general demand of common users by increasing the web resource coverage rate as said by (Dey *et al*., 2010; Sun *et al*., 2008). The problem exists in general crawlers are as follows:

- A large amount of useful information is downloaded by general crawler, at the time of maximizing web resource coverage rate
- Web pages written in JavaScript are of huge numbers, by tag matching it is impossible to extract new URLs, since these link URLS are generated by JavaScript
- Most general crawler does not support attribute search, it support only keyword search

Focused Crawler can solve the problems faced in the general crawlers, relevant to the subject it selects the link URL and useless information is filtered. Even after filtering most useless link pages, still useless information remains in huge numbers. Further, crawlers retrieve pages at rapid speed to keep the search engine indices up-to date. Thus the single search engine of crawling behaviour causes 60GB of daily load to the web. To enhance the coverage and to reduce the bandwidth usage, Parallel and distributed crawling was purposed. This system supports load distribution and localization, but not for declining the load.

URL Distribution Challenges

One of the most important issues in parallel crawlers is URL distribution. URL-hash-based or host-name-hash-based are the most earliest distributed crawlers, in which the computation process is easy and the loading balance is guaranteed due to the randomness of the hash. The distributed crawler ignores the URLs' relevance and thereby it leads to the URLs belonging to the same topics which are being distributed to the different crawlers. URLs with the same domain name are distributed to the same crawlers or the crawlers in the same group. However, traditional suffix naming conventions are not followed by every URL or domain name. Due to a number of websites or documents aren't distributed uniformly, which leads to unbalance load. In our proposed URL distributed model, multiple factors are considered including the load balance, relevance's and so on. Due to the more dynamic nature of the Web, the web pages downloaded by crawlers will be obsolete quick. It is imperative for crawlers to decide on which the crawling policy that keeps the local Repository as up-to-date as possible. In Directory-Based downloading policy (DB), identifies that if a sampled Web page has been updated, all Web pages in the directory of sampled Web pages will be crawled, in each download cycle the crawler uniformly re-downloads Web pages at random (Rand) manner. The existing Web pages are divided into clusters in Cluster Level Sampling algorithm (CLS), then for each cluster the re-crawling frequency decision is depends upon the sample set of Web pages.

A variety of algorithms are proposed for building focused crawlers to maintain the quality of web documents fetched and for keeping the crawling scope within desired domain, New URLs obtained during the crawling process are used to update the learned model at certain periods. (UNB) The Link Structure based Focused Crawler (LSFC) is proposed; it uses the page relevance and link the scoring for irrelevant pages. All the above works are referred to the crawlers for full crawling. A novel ranking model is introduced in our proposed incremental crawler architecture, where all the different factors are considered for new URL and old URL and thereby make ordering as more reasonable one.

Scalable Focused Crawling using Incremental Parallel Web Crawler

A novel architecture of the incremental parallel crawler based on focused crawling is proposed to overcome the drawbacks said by (Vellingiri and Pandian, 2011; Wu and Lai, 2010; Tyagi and Gupta, 2010) and relevant web pages are crawled concurrently which are relevant to multiple pre-defined topics. In our proposed architecture, we added a second level master, in the same topic it masters the crawlers and thereby overlapping issues are avoided, which largely reduces the space and the cost of communication. In the incremental crawl, N time full-crawl is implemented, which has some features.

First, all the old and new URLs are sent to the ranking model by URL dispatcher. Each new URL is verified from the repository before computing the priority, to know whether the URL dispatcher has been already downloaded or not. If the URL is already downloaded, then the retrieved URL is discarded when found the corresponding document in the repository. Second, according to the URL distribution algorithm, the seed URL selected by URL distributors is sorted from the queue which assigns it to client crawler. This process continues until the sorted URL becomes empty.

Third, under the control of a second level master, web pages are fetched by each client crawler. Fourth, after the document is downloaded, to extract it, client crawler passes it to the embedded URLs which send to new URL queue. Concurrently corresponding URLs and the crawled web documents in the repository are stored by client crawlers. Then according to the update frequency decision model, the old URLs queue can be acquired as shown in Fig. 1.

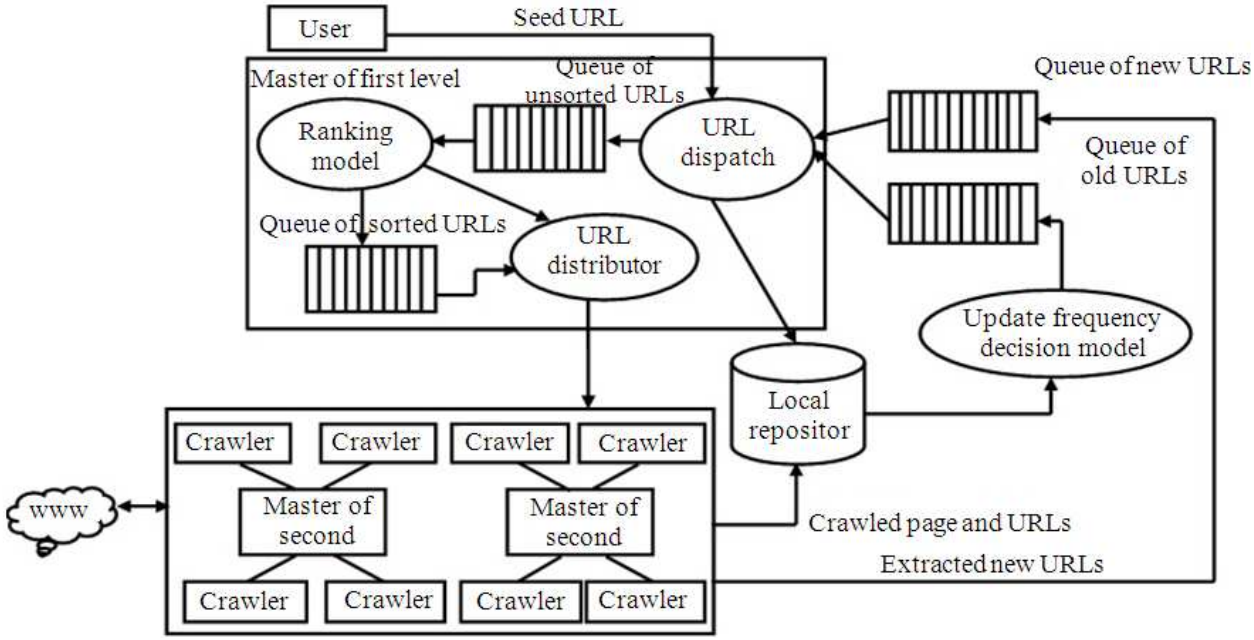

Fig. 1. Scalable Architecture for Incremental Parallel Web Crawler

Ranking Model

It is imperative to prioritize the crawling links, in order to fetch relevant higher quality information selectively and to compute the score of each URL the multi-objective decision making method is also used. Different factors are considered to make the order reasonable for new and old URL. The Following factors are considered for old URLs:

- Finding Web page and the pre-defined topic relevance
- In the recent K times, Average number of new URLs gets extracted
- The data source quality containing the URL
- In the forward link count it considers the number of present URLs in the web page
- In the backlink count, the local repository URLs are pointing to this URL

Following factors are considered for new URLs:

- It contains relevance between pre-defined topics and parental pages and also it considers relevance between URL anchor texts and predefined topics and relevance between URL hyperlinks and pre-defined topics
- URL potential ability
- The URL has the quality of the data source
- The page rank value

Crawling Process

The Crawling process consists of New ordered Queues, Scheduler, Site ordering module, URL Collector, URL Queues/Known URLs, Link Extractor, Multithreaded Downloader, Link analyzer. Based on the customized web page rank, the set of URLs to be downloaded which is supplied by the scheduler. In the latest ordered queues the URLs are saved. The set of URLs based on customized page rank Saved by Latest ordered Queues. The customized page rank of the web page is given by site ordering module. The set of already known URLs is called Known URLs. They are treated as seed URLs. From URL collector, the Multithread Downloader takes a URL and downloads the related WebPages to store it in the local repository. By opening the connections to different servers the Downloader component fetches files from the web. The URL collector maintains the web URL from the downloaded web pages. Link Extractor is used to extract the URL from the downloaded web pages. Link analyzer is used to verify the extracted web URLs by the link extractor.

The URLs gets rejected if they found similarity in the URL and for further processing it won't be forwarded. To save downloading pages it requires little memory space while executing web crawling process. Local repository is owned by each crawling process. In the repository the downloaded pages are saved by the web crawling process and the crawling

process is running in the storage area of the machine. To make refinement decision the Ranking Module constantly scans the local database and the known URLs. The local repository is filtered by the Ranking Module. The less important web page will be rejected by the Ranking Module from the local repository to make the space for the new web pages. The set of URLs in the local collection is called locally collected URLs. The local repository is maintained fresh by the Updated Module, web pages are selected by crawlers to increase the freshness and this result is called as an update decision.

Distributed URL Model

For choosing an optimal crawler for a given URL, comprehensive URL distribution model was made, in which multiple factors are taken into consideration. Generally, assume that the factors are f1, f2 …ft and their corresponding evaluated value for a crawler are g1, g2…gt (0…GI…1) and their weights of the factors are w1, w2 ….wet and by the formula then the evaluated value can be computed. Finally according to the total evaluated value the rank of the crawlers are estimated and then select the optimal crawler.

Factors Selection

For a given URL the optimal crawler will be selected according to the following factors:

- CPU: The basic frequency is taken into consideration here
- Hard disk Capacity and memory Capacity
- Loading rate: Loading rate is defined as the ratio of the number of crawl tasks to memory capacity
- Network bandwidth: Most commonly it is expressed in terms of bits per second (bps)
- Network distance: The network distance is known as latency. It is defined as the specific amount of time it takes for a single block of data to travel from its originating source to a network compute
- Relevance of the URL: We make a difference between a new URL and an old URL and its detailed formula is described
- Potential ability of URL: From the given URL it has an ability to crawl new URLs

Weights and Evaluation

The method of taking many conflicting objectives into consideration scientifically and reasonably and then makes a decision is called Multi-objective decision making method. The issues of URL distribution Factors considered are in contradiction with each other. The factors cannot be measured in a uniform standard, in which the incommensurability of the multi objective decision making method is used. One of multi-objective decision making methods is an Analytic Hierarchy Process (AHP), which is used for calculating the weights and evaluating values.

Evaluation Metrics

In the proposed incremental parallel crawler the user gets required information within an acceptable time, its ultimate aim is to bring high quality, high relevance and high freshness. The performance of the proposed architecture can be evaluated using two metrics.

Efficiency

The time taken to complete the fixed number of tasks and the maximum number of tasks completed in unit time are used to measure the efficiency, by assuming the number of tasks as N and time needed to complete all the tasks as T.

$$\text{Efficiency} = N/T$$

Freshness

The number of up-to-date Web pages in the local repository is the freshness, in a set of web pages. The up-to-date means that the locally stored image of the page and its counterpart at the source are exactly same:

$$F(U;t) = \begin{cases} 1 & if\ p_i\ is up\_to\_date at time \\ 0 & otherwise \end{cases}$$

The freshness of the entire local copy at time t is:

$$F(U;t) = \frac{1}{U}\sum_{pi \in U} F(pi;t)$$

Results

We carried out extensive experiments on a large dataset to evaluate the architecture of the incremental parallel crawler based on focused crawling and the various parameter settings were proposed. In this section, the performance of proposed architecture will be evaluated from the Aspects of Efficiency and Freshness.

Experiments for Efficiency

To evaluate the efficiency comparative testing technique is used. In this our architecture is considered with simple single crawler and parallel. Over 10,000 URLs are crawled in each test and the time consumption

also calculated. The performance of the parallel crawler is higher when comparing single crawler. With the increase of the crawl tasks, our parallel system has a big advantage over DSP.

Experiments for Freshness

Before applying UFG based re-crawling algorithm, an important question to be answered is the availability of units in the data set and the value of K. Before evaluating architecture freshness we should estimate the value of k. For different number of units 'k' value of freshness is shown in Fig. 2. K gets increased when the value of freshness goes up. Freshness increases at a much slower rate when k passes 40. This shows that k does not have a significant impact on freshness when k>40.

Discussion

Normally the channel data rate should be twice the bandwidth. The channel data rate will be 8 KBPS, if there a 4 KHz of channel without noise. Existing crawlers takes 100 seconds to transmit data, but our proposed crawler takes only 60 sec for without compression and with compression it takes only 21 seconds. Bandwidth meter pro is used to measure the bandwidth consumption. Existing crawler consumes 130 KHz of bandwidth, while our proposed crawler consumes only 110 KHz as shown in Fig. 3. Hence by reducing the network traffic, our proposed crawler preserves the bandwidth.

The bar chart in Fig. 4. Shows that, out of 100 pages on the average, only 60 (19+41) pages have been changed.

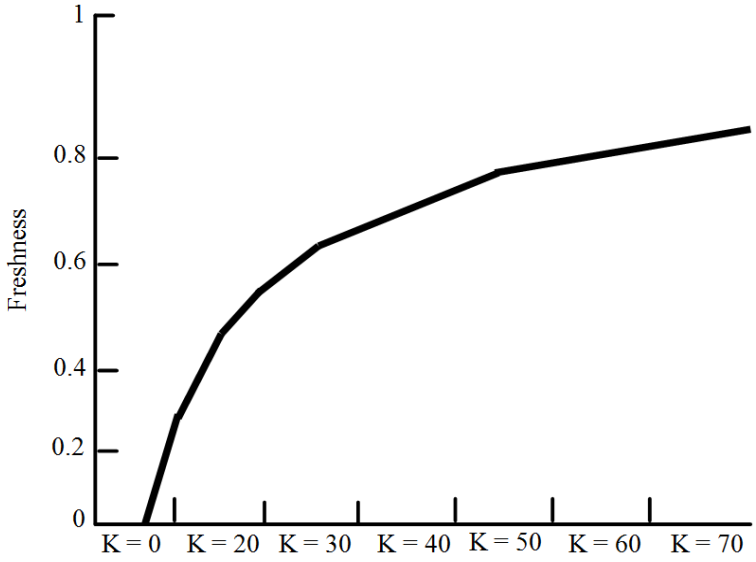

Fig. 2. Corresponding freshness for various values of K

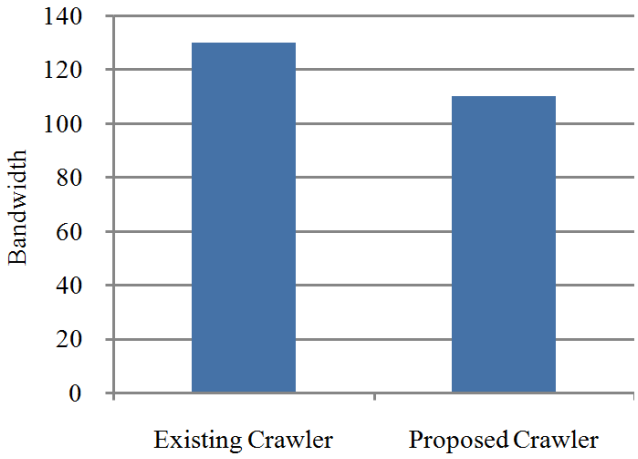

Fig. 3. Bandwidth comparison

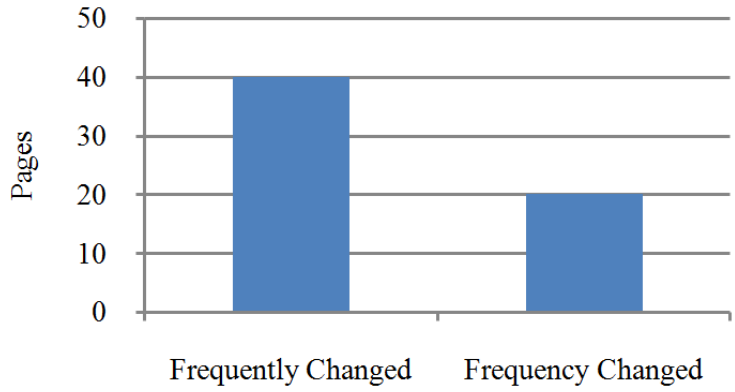

Fig. 4. Incremental parallel web crawling

Conclusion

For many applications in the web sources high quality and high freshness are necessary. The other crawling method will bring out significant waste of time to maintain the data. The rapid growth of the web makes it as a challenge to traverse all URLs and it's difficult to refresh, changes of 40% pages daily, since its URL is very large. In this study, a novel incremental parallel Web crawler for focused crawler is proposed and also this study has presented a novel Parallel Domain Focused Crawler for reduction in load on the network.

Furthermore, the model of URL distribution is based on the method of multi-objective decision making and by introducing the update frequency graph, it update the frequency of the local repository detection model. To start downloading, the crawling process will migrate to the host or server. The crawling process will migrate to the host or server to start downloading. Incremental crawling will increase the quality of downloaded pages by keeping the local database fresh. The experimental results show that our proposed architecture can efficiently yield high quality, relevance and freshness.

Funding Information

The authors have no support or funding to report.

Author's Contributions

All authors equally contributed in this work.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Avraam, I. and I. Anagnostopoulos, 2011. A Comparison over Focused Web Crawling Strategies. Proceedings of the 15th Panhellenic Conference on Informatics, Sept.30 -Oct. 2, IEEE Xplore Press, Kastonia, pp: 245-249. DOI: 10.1109/PCI.2011.53

Balamurugan, M., J. Bhuvana and S.C. Pandian, 2012. Privacy preserved collaborative secure multiparty data mining. J. Comput. Sci., 8: 872-878. DOI: 10.3844/jcssp.2012.872.878

Chakrabarti, S., M.V.D. Berg and B. Dom, 1999. Focused crawling: A new approach to topic-specific web resource discovery. Proceedings of the 8th International World Wide Web Conference, (WWC' 99), Elsevier North-Holland, New York, pp: 1623-1640. DOI: 10.1016/S1389-1286(99)00052-3

Cho, J. and H. Garcia-Molina, 2002. Parallel crawlers. Proceedings of the 11th international conference on World Wide Web, May 07-11, Honolulu, HI, ACM, New York, pp: 124-135. DOI: 10.1145/511446.511464

Dey, M.K., H.M.S. Chowdhury, D. Shamanta and K.E.U. Ahmed, 2010. Focused web crawling: A framework for crawling of country based financial data. Proceedings of the 2nd IEEE International Conference on Information and Financial Engineering, Sept. 17-19, IEEE Xplore Press, Chongqing, pp: 409-412. DOI: 10.1109/ICIFE.2010.5609387

Kumar, M.V., G.T. Arasu and V. Palanisamy, 2013. Analysis of intelligent data mining for information extraction using java agent development environment platform. J. Comput. Sci., 9: 1451-1455. DOI: 10.3844/jcssp.2013.1451.1455

Mannar Mannan, J., M. Sundarambal and S. Raghul, 2014. Selection of ontology for web service description language to ontology web language conversion. J. Comput. Sci., 10: 45-53. DOI: 10.3844/jcssp.2013.45.53

Shkapenyuk, V. and T. Suel, 2002. Design and implementation of a high-performance distributed Web Crawler Proceedings of the 18th International Conference on Data Engineering, (CDE' 02), IEEE Computer Society, ACM, Washington, pp: 357-368.

Sun, Y., P. Jin and L. Yue, 2008. A Framework of a Hybrid Focused Web Crawler. Proceedings of the 2nd International Conference on Future Generation Communication and Networking Symposia, Dec. 13-15, IEEE Xplore Press, Sanya, pp: 50-53. DOI: 10.1109/FGCNS.2008.73

Tyagi, N. and D. Gupta, 2010. A novel architecture for domain specific parallel crawler. Indian J. Comput. Sci. Eng., 1: 44-53.

Vellingiri, J. and S.C. Pandian, 2011. A novel technique for web log mining with better data cleaning and transaction identification. J. Comput. Sci., 7: pp.683-689. DOI: 10.3844/jcssp.2011.683.689

Wu, M. and J. Lai, 2010. The research and implementation of parallel web crawler in cluster. Proceedings of the International Conference on Computational and Information Sciences, Dec. 17-19, IEEE Xplore Press, Chengdu, pp: 704-708. DOI: 10.1109/ICCIS.2010.175

Zhu, Q., 2007. An Algorithm OFC for the Focused Web Crawler. Proceedings of the International Conference on Machine Learning and Cybernetics, Aug. 19-22, IEEE Xplore Press, Hong Kong, 4059-4063. DOI: 10.1109/ICMLC.2007.4370856