Volume 4, Issue 7, July 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Search optimization technique for Domain Specific Parallel Crawler

Anita Saini , Vinit Kumar
Student, CS Shobhit
University, Meerut-250110,
India

Nidhi Tyagi
Associate Professor, CS Shobhit
University, Meerut-250110,
India

*Abstract— **Architectural framework of World Wide Web is used for accessing linked documents spread out over millions of machines all over the Internet. Web is a system that makes exchange of data on the internet easy and efficient. Due to the exponential growth of web, it has become a challenge to traverse all URLs in the web documents and handle these documents, so it is necessary to optimize the parallelize crawling process. In domain specific parallel crawler different domains are distributed among crawler for getting fast result. The crawler crawls the web periodically to maintain the freshness of repository but due to large amount of data, the relevant information does not update frequently. This paper proposes a novel technique that uses a Selection Factor algorithm for optimizing the search in Domain Specific Parallel Crawler and provide relevant information frequently in repository.***

Keywords— Search Engine, Parallel Crawler, Domain Specific Parallel Crawler, Selection Factor Algorithm, Page Rank

I. INTRODUCTION

The Internet is a vast array of interconnected computer systems[1], providing an assortment of different services. Basically internet refers to a useful destination for getting updated and accurate information. The information on the web is increasing in the form of billion and trillion web pages around the world[5]. Web search engines are used to find specific web pages on the WWW. Largest search engines, like Google and Altavista, cover only limited parts of the web and much of their data are out of date several months of the year[12]. Crawling Strategies are used to provide relevant information which are changes frequently on internet.Web crawler is a system for the bulk downloading of web pages from the web world[7]. Starting with a set of seed URLs, crawlers extract URLs appearing in the retrieved pages, and store pages in a repository[3]. The crawler has to deal with two main responsibilities i.e. downloading the new pages, and keeping the previously downloaded pages fresh.. Domain Specific Crawler makes the crawling task more effective in terms of relevancy and load sharing. In this paper, a novel technique is designed of Domain Specific Parallel Crawler for optimizing search technique on WWW, which improve the efficiency of the search engine. For doing this an algorithm is designed.

II. BACKGROUND

Searching for an information is a primary activity on the web and about 80% of the web users use search engines to retrieve information from the web[11]. Current day search and categorization services cover only a portion of the Web called the publicly indexable Web(PIW)[14]. A general web search engine has three parts a crawler, indexer and query engine. Web search engines work by sending out a spider to fetch as many documents as possible, indexer reads these documents and creates an indexed based on the words contained in each document and query engine is responsible for receiving query and filling requests from results[9]. Web search engines[10] employ crawlers to continuously collect web pages from the web. General architecture of search engine is given below in figure1.

There are different types of crawlers Parallel Crawlers, Distributed Crawlers, Focused Crawlers and Incremental Crawlers. A parallel crawler may consists of multiple crawling processes, which we refer to as C-proc's. Each C-proc performs the basic task that a single-process crawler conducts[2]. The issues and challenges of parallel crawlers are multiple downloading of pages, quality of pages and increased bandwidth consumption[8,15]. A Distributed Crawler [16] is a web crawler that operates simultaneous crawling agents. Each crawling agent runs on a different computer, in principle some crawling agents can be on different geographical or network locations. The main Challenges of distributed crawlers are assignment of URLs among different agents, effective way of partitioning the collection, load balancing and efficient cache design. A Focused Crawler[17,18] is designed to gather documents on a specific topic that are relevant from a pre-defined set of topics. Challenges of focused crawlers are missing relevant pages, maintaining freshness of database and absence of particular context.

The incremental crawler continuously crawls the web, revisiting pages periodically. During its continuous crawl, it may also purge some pages in the local collection, in order to make room for newly crawled pages[6,13]. The drawbacks of Incremental crawler how to keep the local collection fresh and how we improve the quality of the local collection[4,9].

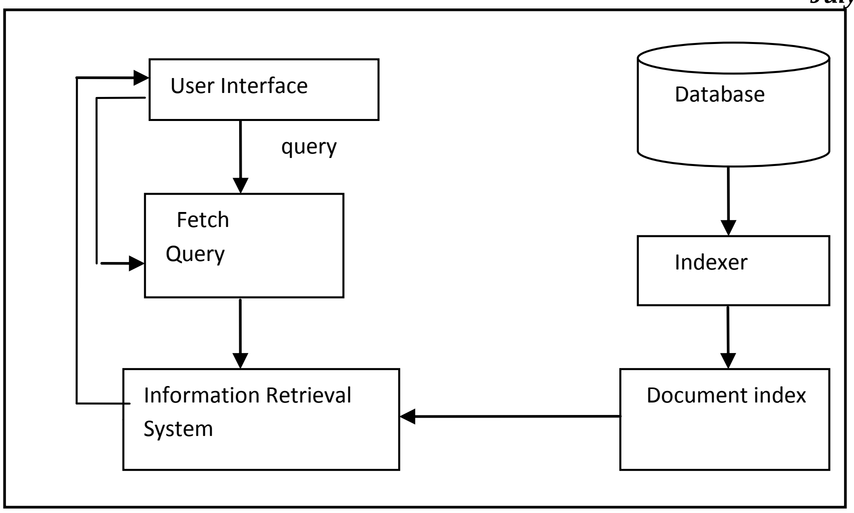

Figure1: General Architecture of a Web Search Engine

The modified parallel crawler is domain specific crawler which takes documents and divide them among different crawlers according to their domain and crawls the documents in parallel[19]. Benefits of Domain Specific Parallel Crawlers are full distribution of data, scalability, load balancing and reliability .The proposed architecture for Domain Specific Parallel Crawler reduce the task of crawling and handles URLs in more specific way by neglecting less specific web pages on the basis of Selection Factor and page rank.

III. PROPOSED ARCHITECTURE

In traditional crawling strategy, the pages from all over the web are brought to the search engine side and then processed, and only after analyzing the page it can be concluded that whether the page is useful or not. Studies suggest that most of the times, the downloaded page is not useful in the sense that it has not updated since it last crawls. The large volume implies that the crawler can only download a limited number of the Web pages within a given time, so it needs to prioritize its downloads. The high rate of change implies that the pages might have already been updated or even deleted. To increase the rate of relevant information updation, an algorithm is used which provide best result in crawling when it is occur periodically. The Proposed technique architecture is given below in figure2.

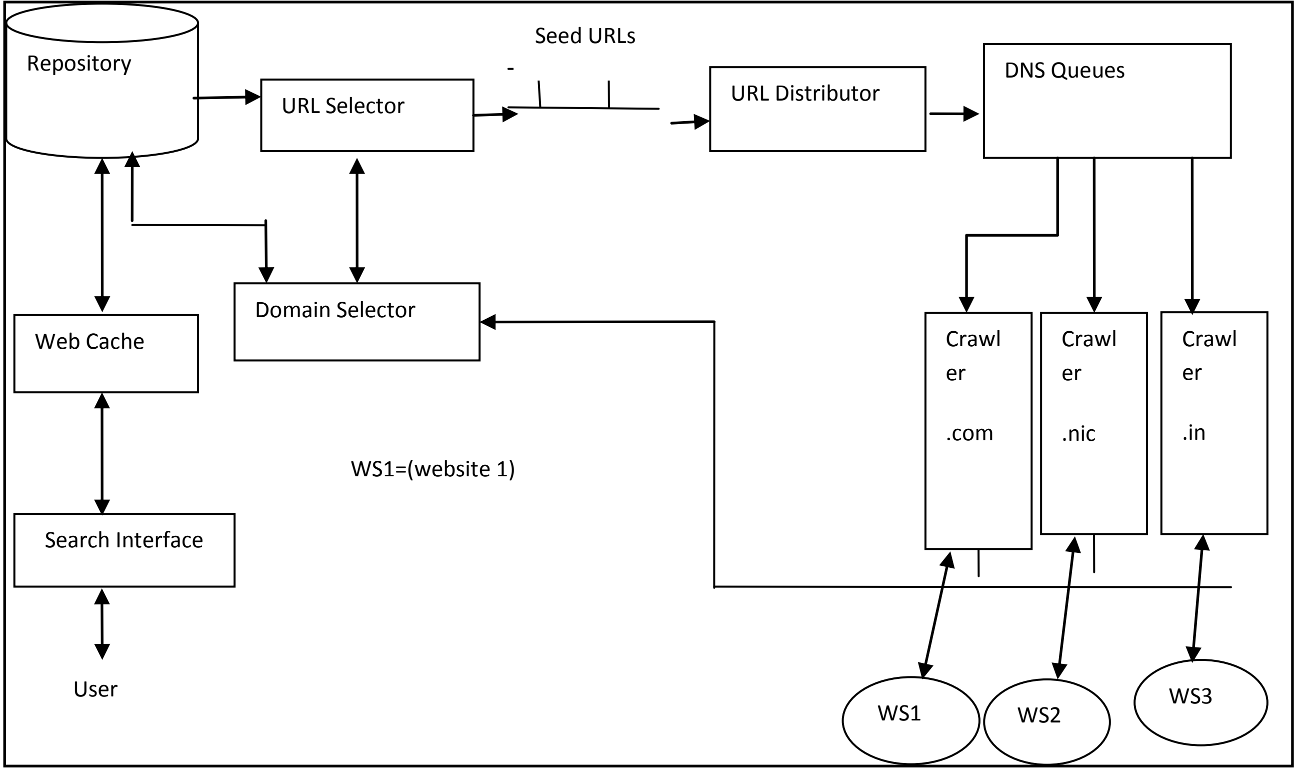

Figure.2: Proposed architecture of optimized Domain Specific Parallel Crawler.

Description of Modules

Crawler Modules

This is the most important module of the entire proposed architecture.

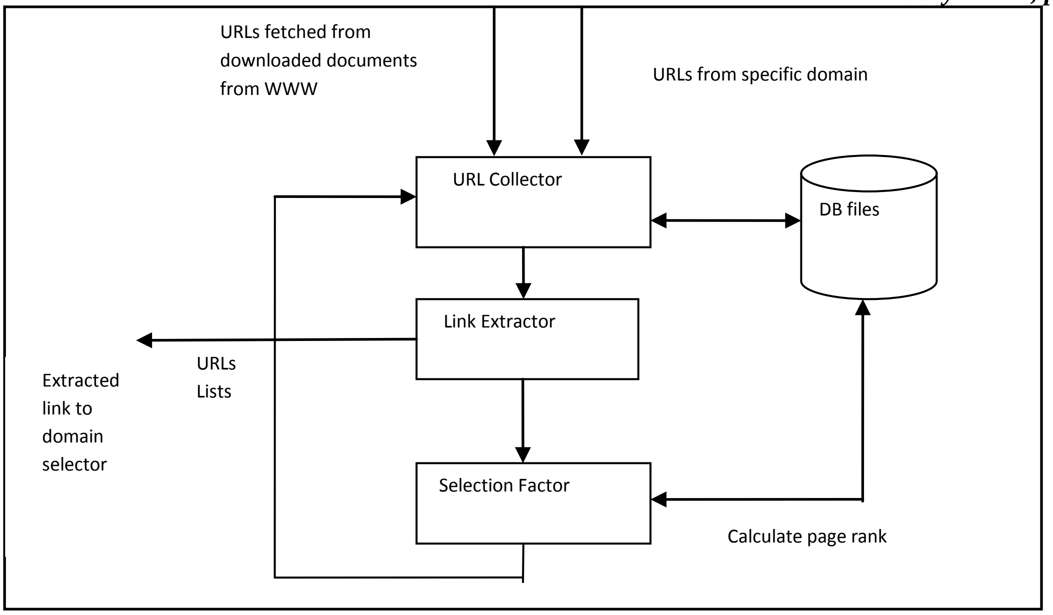
URLs fetched from downloaded documents from WWW
URLs from specific domain
URL Collector
DB files
Link Extractor
URLs Lists
Extracted link to domain selector
Selection Factor
Calculate page rank

Figure.3: Crawler module

The working of the crawler module involves following steps:

(1) Fetch URL (2) Download Web Page (3) Extract URL
(4) Forward URL

The figure.3 shows the various modules of the crawler, firstly it fetches the url from the seed urls and extracts the links from the urls with the help of link extractor. After extracting link selection factor is computed by taking keywords from the index table which has already stored in repository. These keywords are matched with downloaded documents found in repository, a frequency is measured and stored in queue temporarily. Page rank is calculated for those documents which are stored in queue and stored in database as a file. This file is further used for crawling the web pages periodically. The extracted links are forwarded to domain selector.

Downloader

It takes a URL from URLs list and downloads the corresponding web document to store it in the local repository. It sends the signal to extract the web addresses from the world wide web and require more URLs for processing. In Domain Specific crawling threads download web pages independently without communication between them. Each crawling thread gets a seed URL from the respective DNS queues.

URL Distributor

URL distributor maintains information on the domains identification of the URL. It takes a URL from seed URL queue and distributes the URLs to the concerned queue of domain specific crawler module for further processing. It uses the algorithm for distributing the URLs.

Domain Selector

It identifies the domains of the links extracted by the link extractor and forward the URL in the repository for storage, under the corresponding domain table is the main task of domain selector.

Web Cache

A web cache is a mechanism for the temporary storage of web documents, such as HTML pages & images, to reduce bandwidth usage, server load & perceived lag. It stores copies of documents passing through it subsequent requests may be satisfied from the cache if certain conditions are met.

Repository

A repository is a collection of resources that can be accessed to retrieve information. Repositories often consist of several databases tied together by a common search engine.

Selection Factor

Selection factor is calculated by the keywords matched percentage of documents and then downloads documents and stored in queue according to priority then check page rank and after that download the document. The formula is given below.

$$SF = \frac{sum\ of\ \%\ of\ \ document}{total\ \ no.of\ \ documents\ *100}$$

Algorithm Selection Factor ()

```
{
1. While(Queue!=empty)
2. Fetch the  keywords from the downloaded documents
3. Match these keywords to already stored keywords in index table
            3.1  if(these keywords are matched)
                    Then find matched keyword %
                          3.1.1 After finding % apply SF formula and stored result in queue.
                               SF= F(say frequency)
                           3.1.2 Now calculate frequency of document by
                                F1= (1-% of document)
                           3..1.3 Now check these frequencies with standard frequency.
                                 F<=f1,f2,f3……………fn.
                            3.1.4 After checking it select the doc and stored in ascending order in a queue
                  Else
                3.2 Fetched keywords stored in index table.
4. go to step 1
5. After selecting document calculate the page rank of each document.
6. Higher ranking document download first.
}
```

IV. CALCULATION OF DERIVED FORMULA

Let us consider that keywords are already stored in index table in repository. Keywords are fetched from index table and then matching these keywords from already stored document in repository. Find the percentage of matching keywords. Matching % is given below:

$$F = (1 - \%\, of\ matched\ keyword) \quad \ldots Eqn(1)$$

$document\ id$	$\%\, of\ matched\ keyword$	$frequency$
$doc1$	20%	$(1-0.2)=0.8$
$doc2$	30%	$(1-0.3)=0.7$
$doc3$	40%	$(1-0.4)=0.6$
$doc4$	60%	$(1-0.6)=0.4$
$doc5$	75%	$(1-0.75)=0.25$
$doc6$	80%	$(1-0.8)=0.2$
$doc7$	95%	$(1-0.95)=0.15$
$doc8$	82%	$(1-0.82)=0.18$
$doc9$	50%	$(1-0.5)=0.5$
$doc10$	10%	$(1-0.1)=0.9$

$$Selection\ Factor(SF) = \frac{sum\ of\ \%\, of\ all\ documents}{total\ no.of\ documents * 100} \quad \ldots\ Eqn(2)$$

$$= \frac{20+30+40+60+75+80+82+50+10+95}{10*100}$$

$$= 0.542(say\ frequency)$$

$$F = 0.542(Standard\ frequency) \quad \ldots\ Eqn(3)$$

$$f1 \le F, f2 \le F, f3 \le F \ldots\ldots\ldots\ldots\ldots\ldots fn \le F. \quad \ldots\ Eqn(4)$$

Now, compare f1, f2, f3, f4, f5, f6, f7, f8, f9, f10 with F from equation (4), these frequencies should be less than F, greater frequency documents are neglected and lower frequency documents are stored in ascending order in a queue in repository.

After applying eqn(4), selected frequencies are $f7, f8, f6, f5, f4, f9$.

Now compute the page rank for these documents and store in database. These selected documents are used when crawling task is take place again i.e. when crawling is occur periodically only relevant information is updated in repository

V. CONCLUSION

Selection of documents for downloading can be optimize by using Selection Factor algorithm on the basis of keywords stored in repository. This proposed architecture using keywords matched percentage and then compute results which are stored in queue and further used for computing page rank, which helps to improve result of domain specific parallel crawler. When crawler crawls the web periodically then only relevant information is updated frequently in repository. The speed of crawling web pages is increased and reliability of the system is also increased due to using multiple crawlers for crawling task.

VI. FUTURE WORK

The implementation of proposed system enhance the efficiency of the system in future .The computation of Selection Factor increase the rate of downloaded documents in which user shows more interest. This work can be extended in future by improving selection factor algorithm. Secondly, in this we also include the load sharing factor of Hadoop algorithms in cluster balancing. It is very vast research field in crawler. Thirdly, we can also extend the architecture with distribution of URLs to specific crawlers in minimum amount of time and provide crawling security.

REFERENCES

[1] Satinder Bal Gupta, “The Issues & Challenges with web crawlers”, International Journal of Information Technology & Systems,Vol.1,No.1., 2012.

[2] Junghoo cho, Hector Garcia- Molina, “Parallel Crawler”,University of california,Los Angeles, 2002.

[3] Nidhi Tyagi, Ram Kumar Rana, “A Novel Architecture of Ontology-based Semantic Web Crawler”,International Journal of Computer Applications(0975-8887) Volume 44-No18, 2012.

[4] http://www.cameratim.com/computing/internet-primer/the-internet

[5] Niraj Singhal, Ashutosh Dixit, R.P Agarwal, A.K Sharma, “Regulating Frequency of a Migrating Web Crawler based on Users Interest”, International Journal of Engineering and Technolog, 2012.

[6] Hollander, "Google's Page Rank Algorithm to Better Internet Searching", University of Minnesota, Morris Computer Science Seminar, 2003.

[7] Christopher Olston and M. Najork, ”Foundations & Trends in Information Retrieval”, Vol. 4, No.3 175-246, 2010.

[8] Internet Archive, “http://archive.org/.

[9] J. Cho and Garcia-Molina, H. The evolution of the web and implications for an incremental crawler. In proceedings of the Twenty-sixth International Conference on Very Large Databases, Cairo, Egypt, 2000.

[10] Lovekesh kumar Desai, ”A Distributed Approach to Crawl Domain Specific Hidden Web”. Computer Science Theses. Paper 47, Page No. 1, 2007.

[11] Kobayashi, M. and Takeda, K.”Information Retrieval of the Web” ACM Computing Surveys,2000.

[12] Suel, V. Shkapenyuk, "Design and Implementation of a High-Performance Distributed Web Crawler" In Proceedings of the 18th International Conference on Data Engineering (ICDE'02), San Jose, CA Feb. 26--March 1, pages 357—368, 2002.

[13] Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill Book Co., New York, (1983).

[14] Steve Lawrence and C.Lee Giles, ”Searching the world wide web “.Science, 280(5360):98.

[15] Nath, R. Bal, S. and Singh, M. Load Reducing Techniques on the Websites and Other Resources: A comparative Study and Future Research Directions. Journal of Advanced Research in Computer Engineering, pp. 39-49, 2007.

[16] Shkapenyuk, V. and Suel, T., ”Design and implementation of a high performance distributed web crawler”, In Proceedings of the 18th International Conference on Data Engineering (ICDE), San Jose, California. IEEE CS Press, pp. 357-368, 2002.

[17] S. Chakrabarti, M. van den Berg, and B. Dom. Focused Crawling: A New Approach for Topic-Specific Resource Discovery. WWW Conference, 1999.

[18] S. Chakrabarti, K. Punera, and M. Subramanyam, “ Accelerated focused crawling through online relevance feedback”. In Proc. of 11th Intl. Conf. on World Wide Web, pages 148–159, 2002.

[19] Nidhi Tyagi and Deepti Gupta, ”A Novel Architecture for Domain Specific Parallel Crawler”, Indian Journal of Computer Science and Engineering,Vol.1 No.1 44-53,