AN EXTENDED MODEL FOR EFFECTIVE MIGRATING PARALLEL WEB CRAWLING WITH DOMAIN SPECIFIC AND INCREMENTAL CRAWLING

Md. Faizan Farooqui[1], Dr. Md. Rizwan Beg[2] and Dr. Md. Qasim Rafiq[3]

1Research Scholar, Department of Computer Application,
Integral University, Lucknow 226026, India
faizan_farooqui2000@yahoo.com
2Professor & Head, Department of CS&E, Integral University,
Lucknow 226026, India
rizwanbeg@gmail.com
3Chairman, Department of Computer Engineering,
Aligarh Muslim University, Aligarh, India
mqrafiq@hotmail.com

ABSTRACT

The size of the internet is large and it had grown enormously search engines are the tools for Web site navigation and search. Search engines maintain indices for web documents and provide search facilities by continuously downloading Web pages for processing. This process of downloading web pages is known as web crawling. In this paper we propose the architecture for Effective Migrating Parallel Web Crawling approach with domain specific and incremental crawling strategy that makes web crawling system more effective and efficient. The major advantages of migrating parallel web crawler are that the analysis portion of the crawling process is done locally at the residence of data rather than inside the Web search engine repository. This significantly reduces network load and traffic which in turn improves the performance, effectiveness and efficiency of the crawling process. The another advantage of migrating parallel crawler is that as the size of the Web grows, it becomes necessary to parallelize a crawling process, in order to finish downloading web pages in a comparatively shorter time. Domain specific crawling will yield high quality pages. The crawling process will migrate to host or server with specific domain and start downloading pages within specific domain. Incremental crawling will keep the pages in local database fresh thus increasing the quality of downloaded pages.

Keywords

Web crawling, parallel migrating web crawler, search engine,

1. INTRODUCTION

The Internet is a global system of interconnected computer networks. The searching and indexing tasks for the web are currently handled from specialized web applications called search engines. The modern search engines can be divided into three parts they are the publically available search engine, the data- base and the web crawling system. A web crawler is an automated program that browses the Web in a methodological manner. The process of traversing the web is called Web crawling. Web crawler starts with a queue of known URLs to visit. As it visits these pages, it scans them for links to other web pages. The Web Crawler consists of Crawler Manager, Robot.txt downloader, Spider and Link Extractor. A Crawl Manager takes a set of URLs from Link

DOI : 10.5121/ijwsc.2012.3308

extractor and sends the next URLs to the DNS resolver to obtain its IP address. Robot.txt file are the means by which web authors express their wish as to which pages they want the crawler to avoid. Link extractor extract URLs from the links in the downloaded pages and sends the URLs to the crawler manager for downloading afterwards.

2. LITERATURE SURVEY

In [13], the author demonstrated an efficient approach to the "download-first process-later" strategy of existing search engines by using mobile crawlers. In [14] author has implemented UbiCrawler, a scalable distributed and fault-tolerant web crawler. In [15] author presented the architecture of PARCAHYD which is an ongoing project aimed at designing of a Parallel Crawler based on Augmented Hypertext Documents. In [16] the author studied how an effective parallel crawler is designed. As the size of the Web grows, it becomes imperative to parallelize a crawling process. In [17] the author proposes Mercator, which is a scalable, extensible crawler. [18] Presented Google, a prototype of a large scale search engine which makes heavy use of the structure present in hypertext. [19] Aims at designing and implementing a parallel migrating crawler in which the work of a crawler is divided amongst a number of independent.

3. MOTIVATION: PROBLEMS IN GENERIC WEB CRAWLERS

According to [1], Web crawlers of big commercial search engines crawl up to 10 million pages per day. Assuming an average page size of 6K [2], the crawling activities of a single commercial search engine adds a daily load of 60GB to the Web.

Scaling Issues: One of the first Web search engines, the World Wide Web Worm [3], was introduced in 1994 and used an index of 110,000 Web pages. Big commercial search engines in 1998 claimed to index up to 110 million pages [1]. The Web is expected to grow further at an exponential speed, doubling its size (in terms of number of pages) in less than a year [4].

Efficiency Issues: Current Web crawlers download all these irrelevant pages because traditional crawling techniques cannot analyze the page content prior to page download [13].

Index Quality Issues: Current commercial search engines maintain Web indices of up to several hundred million pages [1].

4. MIGRATING PARALLEL WEB CRAWLER

The Migrating Parallel Crawler system consists of Central Crawler, Crawl Frontiers, and Local Database of each Crawl Frontier and Centralized Database. It is responsibility of central crawler to receiving the URL input from the applications and forwards the URLs to the available migrating crawling process. Crawling process migrated to different machines to increase the system performance. Local database of each crawl frontier are buffers that locally collect the data. This data is transferred to the central crawler after compression and filtering which reduces the network bandwidth overhead. The central crawler has a centralized database which will contain the documents collected by the crawl frontiers independently. The main advantages of the migrating parallel crawler approach are Localized Data Access, Remote Page Selection, Remote Page Filtering, Remote Page Compression, Scalability, Network-load dispersion, Network-load reduction.

5. ISSUES IN MIGRATING PARALLEL WEB CRAWLER

The following issues are important in the study of a migrating parallel crawler interesting:

Communication bandwidth: Communication is really important so as to prevent overlap, or to improve the quality of the downloaded content, crawling processes need to communicate and co-ordinate with each other to improve quality thus consuming valuable communication bandwidth.

Quality: Prime objective of migrating parallel crawler is that to download the "important" pages first to improve the "quality" of the downloaded pages.

Overlap: When multiple processes run in parallel to download pages, it is possible that different processes download the same page multiple times. One process may download the same page that other process may have already downloaded, one process may not be aware that another process has downloaded the same page.

6. COORDINATION IN MIGRATING PARALLEL WEB CRAWLER

The coordination is done in order to prevent overlap and also crawling processes need to coordinate with each other so that the correct and quality pages are downloaded. This coordination can be achieved in following ways:

Independent: In this method Crawling processes may download pages independent of each other without any coordination. The downloaded pages may overlap in this case, but we assume that overlap is not significant enough.

Static assignment: In this method the Web is partitioned into sub partitions and each sub partition is assigned to each Crawling processes before the actual crawling process may begin. This type of arrangement is called static assignment.

Dynamic assignment: In this coordination scheme there is a central coordinator that divides the Web into small partitions and each partition is dynamically assigned to a Crawling processes for further downloading.

7. CRAWLING MODES FOR STATIC ASSIGNMENT IN MIGRATING PARALLEL WEB CRAWLER

In this scheme of static assignment, each Crawling processes are responsible for a certain partition of the Web and have to download pages within the partition. There may be possibility that some pages in the partition may have links to pages in another partition. Such types of links are referred to as an inter-partition link.

Firewall mode: Each Crawling processes strictly download the pages within its partition and inter-partition links are not followed. In firewall mode all inter-partition links are either thrown away or ignored.

Cross-over mode: In this mode each Crawling processes download pages within its partition. It follows inter-partition links when links within its partitions are exhausted.

Exchange mode: In this mode Crawling processes periodically exchange inter-partition links. Crawling processes are not allowed to follow inter-partition links [16].

Table 1. Comparison of three Crawling Modes[16]

Mode	**Coverage**	**overlap**	**Quality**	**Communication**
Firewall	Bad	Good	Bad	Good
Cross-Over	Good	Bad	Bad	Good
Exchange	Good	Good	Good	Bad

8. EVALUATION MODELS IN MIGRATING PARALLEL WEB CRAWLER

This section deals with metrics that we define to quantify the advantages or disadvantages of migrating parallel crawler.

Coverage: the coverage is defined as I/U, where U is the number of pages downloaded by migrating parallel crawler and I is number of unique pages downloaded by the migrating parallel crawler.

Overlap: The overlap is defined as (N−I) /I. Where, N is number of pages downloaded by the crawler, and I is the number of unique downloaded pages by migrating parallel crawler.

Quality: The quality of downloaded pages is defined as $|AN \cap PN|/|PN|$. The migrating parallel crawler downloads the important N pages, and PN is that set of N pages. AN is set of N pages that an actual migrating parallel crawler would download, which is different from PN.

Communication overhead: The communication overhead is defined as E/I. Where I in total pages downloaded and E represents exchanged inter-partition URLs.

Table 2 summarizes different crawling techniques with their performance attributes. Crawling process of migrating parallel crawlers move to the resource which needs to be crawled to take the advantage of localized data access. After accessing a resource, migrating parallel crawler move on to the next machine or server and send result of crawling to central database in compressed form. From the table it is evident that Migrating parallel crawlers are more efficient in terms of network load reduction, I/O performance, reducing communication bandwidth, network load dispersion etc.

Table 2. Performance Attributes for Different Web Crawlers

	High performance distributed web crawler	Incremental crawler	Parallel crawler	Agent based	Migrating parallel Crawler	Domain Specific Crawler	Mobile Crawler	Breadth First Crawling
Robustness	++	--	--	--	++	--	++	--
Flexibility	++	--	--	++	++	--	--	--
Manageability	++	--	--	++	++	--	--	--
I/O Performance	++	--	--	++	++	--	--	--
Network resources	++	--	--	--	++	--	++	--
OS limits	++	--	--	--	++	--	++	--

Higher performance	++	--	--	--	++	--	++	--
Extensibility	--	++	--	--	--	--	++	--
Incremental crawling	--	++	--	--	--	--	--	--
Reasonable cost	++	--	--	--	--	--	--	++
Overlapping	--	--	++	++	++	++	++	--
Communication bandwidth	--	--	++	--	++	++	++	--
Network load dispersion	--	--	++	--	++	++	++	--
Network load reduction	--	--	++	++	++	++	++	--
Freshness	--	++	--	--	--	--	--	--
Page rank	--	--	--	--	--	--	--	++
Scalability	--	--	--	--	--	++	--	--
Load sharing	--	--	--	--	--	++	++	--
High quality	--	--	--	--	--	--	--	++
Agent oriented	--	--	--	++	++	--	--	--

9. PROPOSED ARCHITECTURE OF MIGRATING PARALLEL WEB CRAWLER

A migrating parallel crawler consists of several multiple crawling processes which migrate to source of data for example Web server or host before the crawling process is actually started on that Web server or host. Migrating parallel crawlers move to resource and take the advantages of localized access of data. Migrating parallel crawler after accessing a resource migrates to the next host or server or to their home system. Each migrating parallel crawling process performs the tasks of single crawler that it downloads pages from the Web server or host, stores the pages in the local database, extracts URLs from the content of downloaded pages and follows the extracted URLs or links. When Crawling process run on the same local area network and communicate through a high speed interconnection network then it is known as intra-site migrating parallel web crawler. When Crawling process run at geographically distant locations connected by the Internet or a wide area network then it is known as distributed migrating parallel web crawler. The architecture consists of central coordinator system and crawling process.

9.1 Central Coordinator System

Central coordinator system consists of Central Crawl Manager, Crawling Application, Web Cache, Central Database or Repository, URL Dispatcher, URL Distributor, Domain Selector. Central crawl manager is the central part of the crawler all other components are started and register with the central crawl manager to offer or request services. It acts as the core of the system. Central Crawl Manager's objective is to download pages in the order specified by the crawling application. The central crawler has a list of URLs to crawl. After getting the URLs of files, central crawl manager queries the DNS resolvers for the IP addresses of the host or servers. The central manager then takes into consideration robots.txt files in the root directory of the web server. The central crawler a set of available crawling process which have registered themselves with the central crawl manager which logically migrated to different domain specific locations. The central crawl manager assigns different URLs to all the crawl process in domain specific manner and in turn the crawling processes begin to crawl the received URL in breadth first fashion and pages are downloaded by multi threaded downloaded. It is the responsibility of the crawling application

to check for duplicate pages. The crawling application considers in this paper is a breadth-first crawl strategy. Breadth first search gives high quality pages thus improving the quality of downloaded pages. The crawling process actually crawls in breadth first manner. The DNS resolver uses the GNU adns asynchronous DNS client library to access a DNS server usually collocated on the same machine. A web cache stores web resources in anticipation of future requests. Web cache works on the principle that the most popular resource is likely to be requested in the future. The advantages of Web Cache are reduced network bandwidth, less user perceived delay, less load on the origin servers. Web cache provides quick response time. When the information resides in web cache, the request is satisfied by a cache, the content no longer has to travel across the Internet. Central database or Repository stores the list of URLs received and downloaded web pages which are collection of the documents downloaded by crawling process. Web pages can be stored in compressed form. Each document is provided with a unique number called document identifier denoted by doc_id. The documents are stored along with doc_id, length of document and URL. This helps with data consistency and makes data storage much easy and efficient. Data is stored in the central data base in hierarchical tree manner.

Each node or the parent node of the tree is the domain and the child node of the parent node is the sub domain. Each node consists of domain name and its corresponding URLs. The database is continuously updated for future use by adding new URL. Central Database communicates with the database of the search engine. URL dispatcher reads the URL database and retrieves the URL from the index table maintained in the central repository and forwards it for crawling. URL distributor classifies the URLs on the basis of domains. It then distributes the URL to the concerned domain specific crawling process for crawling. The URL distributor balances the load on the crawling process. Domain selector identifies the domains of the links which are extracted by the link extractor. The links belonging to specific domain are stored in the corresponding domain table. The domain selector also forwards the domain specific URL to the URL dispatcher for further crawling.

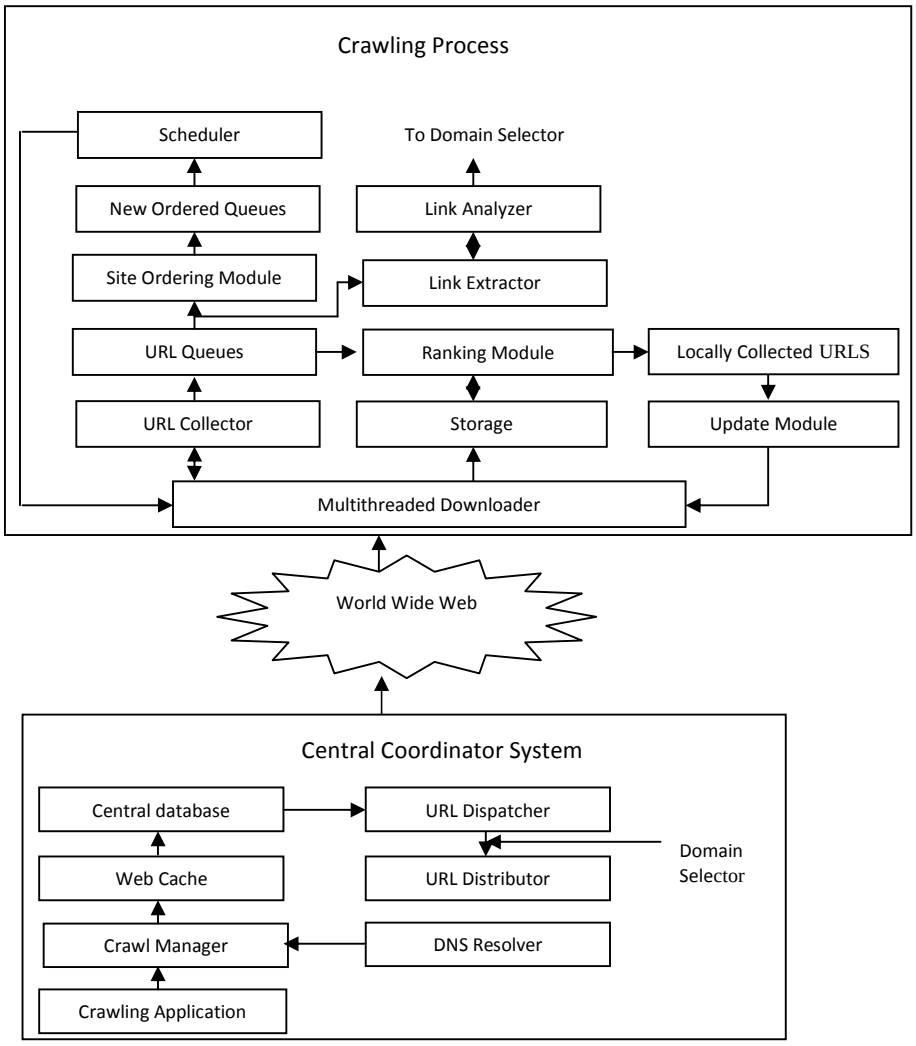

Figure: Proposed Architecture of Migrating Parallel Crawler

9.2 Crawling Process

Crawling process consists of Scheduler, New ordered Queues, Site ordering module, URL Queues/ KnownURLs, Multi threaded Downloader, URL Collector, Link Extractor, Link analyzer. Scheduler provides the set of URLs to be downloaded based on the modified page rank. The URLs are stored in new ordered queues. New ordered Queues stores the set of URLs based on modified page rank. Site ordering module provides the modified page rank of the page. KnownURLs are the set of already known URLS. They can be treated as seed URLs. Mutli threaded downloader takes a URL from URL collector and downloads the corresponding web-page to store it in the local repository or database. The downloader component fetches files from the web by opening up to connections to different servers. URL collector keeps the URL from the downloaded pages. Link Extractor extracts the link from the downloaded pages. Link analyzer checks the extracted URLs by the link extractor. If there is similarity in the URL then such URLs are discarded and not further forwarded for processing. When the crawling process runs it requires some memory space to save the downloaded web pages. Each crawling process has its own local storage. The crawling process saves the downloaded web pages in this database. It is the

storage area of the machine on which the crawling process is running. The Ranking Module constantly scans the knownURLs and the local database to make the refinement decision. The Ranking Module refines the local database. The Ranking Module discards the less important page from the local database to make space for the new page. LocallycollectedURLs are the set of URLs in the local collection. The Update Module maintains the local database fresh. In order to maintain the local database fresh, the crawler has to select the page that will increase the freshness most significantly this decision is known as update decision.

10. EVALUATION METHOD

The performance of the proposed migrating parallel web crawler is measured by obtaining top 1000 URLs returned by modified page update algorithm will be used. Minimum number of overlapping pages of specific domain and more relevant pages will be considered. The fresh pages will be considered by using the incremental module. Less traffic on the network will be ensured. Less bandwidth will be consumed as less traffic will travel over the network.

11. RESULT

The crawled pages will be domain specific. Fresh pages will be downloaded as change frequency is taken into consideration. High quality of pages will be downloaded as crawling processes are performing in breadth first manner. Breadth first crawling improves the quality of downloaded pages.

12. CONCLUSIONS

In this paper we proposed an Extended Model for Effective Migrating Parallel Web Crawling with Domain Specific and Incremental Crawling. Domain specific crawling will yield high quality pages. The crawling process will migrate to host or server with specific domain and start downloading pages within specific domain. Incremental crawling will keep the pages in local database fresh thus increasing the quality of downloaded pages.

The research directions in migrating parallel crawler include:

• Security could be introduced in migrating parallel crawlers
• Migrating parallel crawler could be made polite
• Location awareness could be introduced in migrating parallel crawlers

This future work will deal with the problem of quick searching and downloading the data. The data will be collected and analyzed with the help of tables and graphs.

ACKNOWLEDGMENTS

First and foremost, our sincere thanks goes to Prof. Syed Wasim Akhtar, Honorable Vice Chancellor, Integral University, Lucknow. Prof Akhtar has given us unconditional support in many aspects, which enabled us to work on the exciting and challenging field of Software Engineering. We would also like to give our special thanks to Prof. T. Usmani, Pro Vice Chancellor, Integral University. His encouragement, help, and care were remarkable during the past few years. We are also grateful to Prof S. M. Iqbal Chief Academic Consultant, Integral University. Prof Iqbal provided us with valuable thoughts for our research work. My gratitude also goes to Dr. Irfan Ali Khan, Registrar, Integral University for his constructive comments and shared experience.

REFERENCES

[1] D. Sullivan, "Search Engine Watch," Mecklermedia, 1998.
[2] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," Stanford University, Stanford, CA, Technical Report, 1997.
[3] O. A. McBryan, "GENVL and WWW: Tools for Taming the Web," in Proceedings of the First International Conference on the World Wide Web, Geneva, Switzerland, 1994.
[4] B. Kahle, "Archiving the Internet," Scientific American, 1996.
[5] J. Gosling and H. McGilton, "The Java Language Environment," Sun Microsystems, Mountain View, CA, White Paper, April 1996.
[6] J. E. White, Mobile Agents, MIT Press, Cambridge, MA, 1996.
[7] C. G. Harrison, D. M. Chess, and A. Kershenbaum, "Mobile Agents: Are they a good idea?," IBM Research Division, T.J. Watson Research Center, White Plains, NY, Research Report, September 1996.
[8] H. S. Nwana, "Software Agents: An Overview," Knowledge Engineering Review, Cambridge University Press, 11:3, pp. , 1996.
[9] M. Wooldridge, "Intelligent Agents: Theory and Practice," Knowledge Engineering Review, Cambridge University Press, 10:2, pp. , 1995.
[10] P. Maes, "Modeling Adaptive Autonomous Agents," MIT Media Laboratory, Cambridge, MA, Research Report, May 1994.
[11] P. Maes, "Intelligent Software," Scientific American, 273:3, pp. , 1995.
[12] T. Finin, Y. Labrou, and J. Mayfield, "KQML as an agent communication language," University of Maryland Baltimore County, Baltimore, MD, September 1994.
[13] Oachim Hammer , Jan Fiedler "Using Mobile Crawlers to Search the Web Efficiently" in 2000
[14] Paolo Boldi, Bruno Codenotti, Massimo Santini, Sebastiano Vigna, "UbiCrawler: A Scalable Fully Distributed Web Crawler" in 2002
[15] A.K. Sharma, J.P. Gupta, D. P. Aggarwal, "PARCAHYDE: An Architecture of a Parallel Crawler based on Augmented HypertextDocuments" in 2010
[16] J. Cho and H.Garcia-Molina, "Parallel crawlers". In Proceedings of the Eleventh International World Wide Web Conference, 2002, pp. 124 – 135
[17] A. Heydon and M. Najork, "Mercator: A scalable, extensible web crawler". World Wide Web, vol. 2, no. 4, pp. 219 -229, 1999.
[18] Akansha Singh , Krishna Kant Singh , "Faster and Efficient Web Crawling with Parallel Migrating Web Crawler" in 2010
[19] Min Wu, Junliang Lai, "The Research and Implementation of parallel web crawler in cluster" in International Conference on Computational and Information Sciences 2010