

# **DESIGN OF A NOVEL INCREMENTAL PARALLEL WEBCRAWLER**

*Synopsis of the Thesis to be submitted in fulfillment of the requirements for the  
degree of*

**DOCTOR OF PHILOSOPHY**

By

**DIVAKAR YADAV**



Department of Computer Science & Engineering and Information Technology  
JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY UNIVERSITY

A-10, SECTOR-62, NOIDA, INDIA

January, 2010

# DESIGN OF A NOVEL INCREMENTAL PARALLEL WEBCRAWLER

---

World Wide Web (WWW) is a vast repository of interlinked hypertext documents known as web pages. A hypertext document consists of both, the contents and the hyperlinks to related documents [1]. Users access these hypertext documents via a software known as web browser. It is used to view the web pages that may contain information in form of text, images, videos and other multimedia. The documents are navigated using hyperlinks, also known as Uniform Resource Locators (URLs). Though the concept of hypertext is much older but WWW was originated after Tim Berners-Lee, an English physicist who wrote a proposal using hypertext to link and access information in 1990 [2]. Since then websites were being created around the world using hypertext mark up languages and connected through Internet.

Now it has become an integral part of human life to use Internet to access the information from WWW. The current population of the world is approximately 6.77 billion out of which approximately 1.67 billion people (24.7%) use Internet [3]. In fact from .36 billion in 2000, the number of Internet users has increased to 1.67 billion in 2009 i.e. an increase of 362% from 2000 to 2009. Same growth rate is expected in future too. In Asia alone, around .7 billion people use Internet that is approximately 42.2% of worldwide Internet users. India where approximately 0.08 billion people use Internet, is the third largest country of Internet users in Asia after China and Japan. Thus it is not far away when one will start feeling that life is incomplete without Internet.

Since its inception in 1990, World Wide Web has grown exponentially in size. As of today, it is estimated that it contains approximately 50 billion publicly accessible/indexable web documents [4] distributed all over the world on thousands of web servers. It is very difficult to search information from such a huge collection of web documents on World Wide Web as the web pages/documents are not organized as books on shelves in a library, nor are web pages completely catalogued at one central location. It is not guaranteed that users will be able to

retrieve information even after knowing where to look for information by knowing its URLs as Web is constantly changing [5-6]. Therefore, there was a need to develop information retrieval tools to search the required information from WWW. Information retrieval tools are divided into three categories as follows:

- Web directories
- Meta search engines
- Search engines

Web documents are organized in a hierarchical taxonomy tree on the basis of topics and subtopics in Web directories. To access information from a Web directory on a topic, it is necessary to traverse a path in the taxonomy tree from the root to the desired node in the tree. The hierarchical tree is organized in such a way that general topics are sub-divided into more specified topics if one follows towards the lower order from the root of the hierarchical tree. Arrangement of the web documents in this hierarchical way helps even a non-expert user to easily access the information but the basic problem with a Web directory is that the hierarchical tree is maintained manually and therefore only a small fraction of the Web is covered. Another problem in the Web directory is that the depth of an item in the hierarchical tree is not based on the access pattern [7]. As a result, longer path may be followed to retrieve more relevant information in comparison to less relevant information if it is down the order in the tree. However, this problem is not faced in search engines which use flat approach to access information. So user can get the most relevant information in one go in response to an appropriate search query.

Meta search engines do not maintain their own indexes/repository rather they are developed to exploit the best features of many search engines. They provide a single interface where user queries are issued and later on sent to many search engines. Results obtained from these multiple search engines are compiled [8-9] to get the final result. The documents are ranked after eliminating duplicates and the final results are displayed to the user.

Internet would have not become so popular if search engines would not have been developed. Starting in 1994, a number of search engines were launched, including *AltaVista*, *Excite*, *Infoseek*, *Inktomi*, *Lycos*, and of course the evergreen, *Yahoo* and *Google*. Most of these search engines save a copy of the web pages in their central repository and then make appropriate indexes of them for later search/retrieval of information. User interface, Query engine, Indexer, Crawlers and Repository are the basic components of search engines [10-12].

To access information from the WWW, users provide search queries in the search engine's interface. For the search query provided, search results, in the order of their relevance [13, 23], are displayed on the screen. On behalf of the search engine, it is the query engine which processes the search queries for getting the relevant documents stored in the search engines database/repository.

In fact the databases/repositories for search engines are maintained with the help of Web crawlers. Web crawlers also known as spiders, robot, web pot etc, are programs that traverse the Web and download web documents [14]. The working of Web crawlers is initiated with initial set of URLs known as *seed URLs*. They download web documents for the seed URLs and extract new links present in the downloaded documents. The downloaded web documents are stored and properly indexed in the repository so that with the help of their indexes they may later be retrieved when required. The extracted URLs from the downloaded web page are verified to know whether their corresponding documents have already been downloaded or not. If they are not already downloaded, the URLs are again assigned to crawlers for further downloading. This process is repeated till no more URLs are left for downloading or target numbers of documents are being downloaded. Millions of web pages are downloaded per day by a crawler to achieve the target.

From above discussions it is very clear that a search engine is the most popular information retrieval tool having the following objectives:

1. It should explore and download web documents from WWW as much as possible.

2. It should bring high quality documents so that the user gets the required relevant information within acceptable time.
3. The documents must be displayed in the order of their relevance with respect to the user query.
4. As the web documents are very much dynamic in nature, search engine should update its repository as frequently as possible. The ideal case would be of synchronizing updation of repository with the web document's actual change frequency.

To satisfy the first objective i.e. to cover the Web as much as possible, nowadays search engines do not depend on a single but on multiple crawlers that execute in parallel to achieve the target. While working in parallel, crawlers still face many challenging problems such as overlapping, quality and network bandwidth that need to be addressed.

Search engines employ ranking algorithms to meet second and third objectives mentioned above. The most popular algorithm is the back link count, proposed by Sergey Brin and Lawrence Page, the Google founders in 1998 [2]. Though back link count helps in efficiently displaying the documents in the order of their relevance, it fails to bring quality documents. The reason being that it needs to have image of entire Web in terms of back link count over the entire Web, which is not possible for the crawler especially when its database is in the starting or growing stage.

The fourth objective of search engine is to keep its database up-to-date with respect to the web pages maintained at Web server end. The optimum case will be if the updating frequency is synchronized with web page's change frequency. In fact it is almost impossible to find the exact change frequencies of web documents as they get changed at random and follow the Poisson process [15]. Nevertheless it is equally important to find whether a document has changed or not.

A critical look at the available literature indicates the following issues that need to be addressed:

#### **Issue 1: Overlapping of web documents**

Overlap problem occurs when multiple crawlers running in parallel download the same web document multiple times due to the reason that one web crawler may not be aware of another

having already downloaded the same page. Also many organizations mirror their documents on multiple servers to avoid arbitrary server corruption [21-22]. In such a situation, crawlers may also unnecessarily download many copies of the same document.

### **Issue 2: Quality of downloaded web documents**

The quality of downloaded documents can be ensured only when web pages of high relevance are downloaded by the crawlers. Therefore, to download such relevant web pages at the earliest, multiple crawlers running in parallel must have global image of collectively downloaded web pages so that the redundancy in terms of duplicate documents may be avoided.

### **Issue 3: Network bandwidth/traffic problem**

In order to maintain the quality, the crawling process is carried out using either of the following approaches:

- Crawlers can be generously allowed to communicate among themselves or
- They can not be allowed to communicate among themselves at all.

In the first approach network traffic will increase because crawlers communicate among themselves more frequently to reduce the overlap problem whereas in second approach, if they are not allowed at all to communicate then as a result same web documents may be downloaded multiple times thereby consuming the network bandwidth. Thus both approaches put extra burden on network traffic.

### **Issue 4: Change of web documents**

Changing of web documents is a continuous process. Of course, the frequency of change varies from document to document. This change must be reflected at the search engine repository failing which a user may have to access an obsolete web document.

### **Main Contribution of the Thesis**

The focus of this thesis is to investigate the issues in concern to parallel crawling and search engines database updating.

1. A survey has been conducted to know the users search trend on WWW. It has helped to identify the major problems users face while searching required information from the Internet.
2. A novel architecture for incremental parallel web crawler has been designed that helps to reduce the overlap and network bandwidth problem among crawlers while working in parallel. The proposed crawler (see Figure 1) has a client server based architecture consisting of following main components:
  - Multi Threaded server
  - Client crawlers
  - Change detection module

The Multi Threaded (MT) server is the main coordinating component of the architecture. On its own, it does not download any web document but manages a connection pool with client machines which actually download the web documents. URL dispatcher, ranking module, URL distributor, URL allocator, indexer and repository are the sub components of MT Server. The salient features of the MT server are as given below:

- It establishes communication links between MT server and client crawlers.
- If a client crawler loses its connection with the MT server, the whole system does not get disrupted as other client crawlers remain functional.
- The system is scalable and it is possible to add-up new client crawlers/machines to an ongoing crawling system without any temporary halt of the overall system. The MT server automatically starts allocation of URLs to the newly added client crawler without any inherent effects on the other client processes or the whole system.
- Once the communication is established between the MT server and the clients, the *URL allocator* sends seed URL to a client crawler in order to download the corresponding web document and waits for its response.

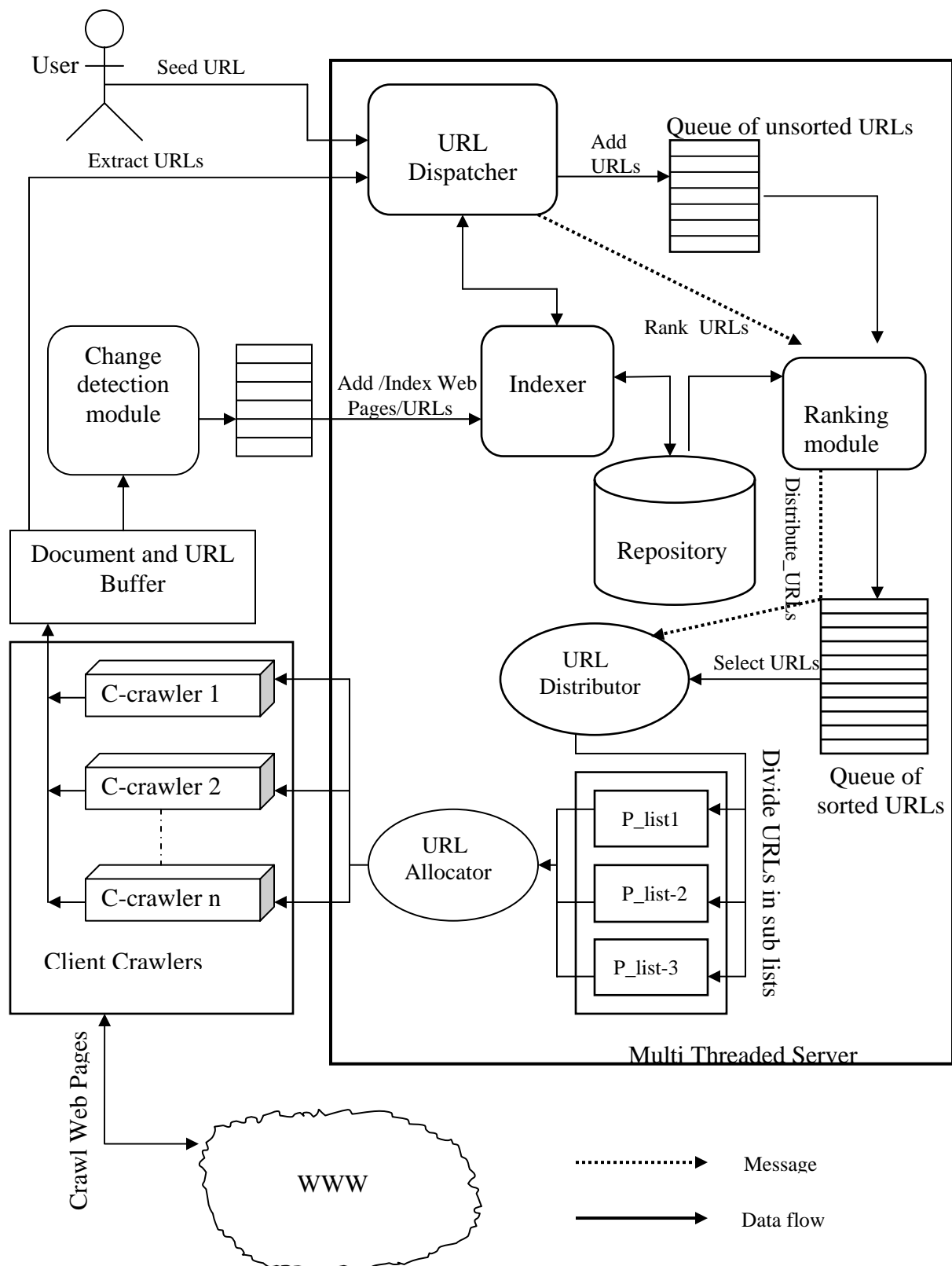


Figure 1: Proposed Architecture of Incremental Parallel WebCrawler

The client crawler responds to MT server in the form of downloaded web document as well as embedded URLs extracted from the document. The *URL dispatcher* adds these URLs to an unsorted queue from where they are stored in a *sorted queue* by ranking module. From *sorted queue*, the URLs are further divided in three sub-lists by *URL distributor*, based on their rank. Finally these URLs are selected by *URL allocator* and are assigned to suitable client crawlers to carry forward the crawling process.

- In order to achieve parallelization, the MT server maintains independent threads with each client and carries out the tasks of receiving and sending URLs. The server ensures that it does not lose the connection with any client machine and in case of disruption, it automatically stops assigning URLs to that client machine, thereby avoiding data losses.
- Once a normal crawling routine is established, the server routinely receives lists from the clients, sorts them by priority and resends each URL to the suitable client crawlers.
- It coordinates functioning of all its individual components such as URL dispatcher, ranking module, URL distributor, URL allocator, indexer etc.
- It provides required data to change detection module too, to find whether a page is changed or not.

The client crawlers collectively refer to all the different instances of client machines interacting with each other through server. The number of clients may vary depending on the availability of resources and the scale of actual implementation. It may be noted that there are no direct links among client crawlers and therefore, all communications take place through the server.

3. A ranking method has been developed which is used to rank the unvisited URLs and helps to decide their order of visit. This ensures that more relevant documents are visited first enabling the crawlers to download high quality documents.

4. Additionally, methods for change detection among web documents have been developed. These methods can be used to decide whether two versions of a web document are same or different and thus help to decide whether the existing web document is required to be replaced or not. The change detection methods help for detecting the following major types of changes:
- Structural changes
  - Presentation changes
  - Content level changes

The structural change detection methods also help in detecting presentation changes as well. As the presentation of web documents gets modified through insertion/deletion/modification of the tag structure, the methods proposed for *structural change detection* also work efficiently and guarantee to detect the structural as well as presentation changes.

Two different schemes, *document tree based* and *document fingerprint based* have been developed for *structural change* detection. The *document tree based* scheme works efficiently and guarantees to detect structural changes. Apart from providing details about the structural changes within documents, it also helps in locating the area of major/minor change as well. For instance, the information about at what level how many nodes have been inserted/deleted is also reported. In *document fingerprint based* scheme, two separate fingerprints in the form of strings for each version of web document are generated on the basis of their tag structure. The first fingerprint contains the set of first characters of the tags in the order they appear in the web page whereas the second fingerprint contains the set of last characters of the tag in the order they appear in the web page. Time and space complexity in document tree based scheme is higher than the *fingerprint based* scheme. Generation and comparison of fingerprints in the form of strings, require less time than to generate and compare two trees. Also, the space required to store the node details (as node consists of multiple fields) in the document tree is high

in comparison to space required to store fingerprints. Though fingerprint based scheme is efficient in terms of space and time complexity, it also guarantees to detect structural changes even at micro level but the details about the changes can not be reported, for which document tree based scheme is better.

Similar to structural change detection scheme, two different schemes, *Root Mean Square* based and *Checksum based* have been developed for content level change detection. Both schemes are based on ASCII principle of symbols and both generate checksum for entire web page and also for different paragraphs. The checksum for entire web page helps to draw the picture about content level changes at page level whereas through paragraph checksum, changes at micro level i.e. paragraph level can be detected. Though, both the schemes are efficient and guarantee to detect changes at micro level also but the former scheme considers changes in contents uniformly whereas the later scheme assigns different weightage to different contents present in the web page. In this work, the font size has been considered for assigning the weightage. If the contents having bigger font size gets changed, it contributes more in comparison to contents having smaller font size.

The thesis is organized chapter wise as follows:

**Chapter 1:** This chapter is devoted to introduction about World Wide Web, Internet, Search engines and Web crawlers. It has been discussed as to how the WWW evolved and has become the source of information sought by users. The introduction of various search tools used for searching the information from Internet and challenges faced by them has been discussed.

**Chapter 2:** In this chapter, a discussion on various searching tools is given. The general architecture of a search engine along with its major components is given. A detailed discussion of web crawlers and their types is also discussed. Additionally, some popular searching algorithms such as Fish, Shark, Breadth first search, Depth first search, and Best first search have been discussed in this chapter. This chapter also discusses about the existing change detection techniques used to detect whether versions of web documents have been changed or not. Finally,

based on the literature reviewed the major challenges being faced by parallel web crawler and their change detection techniques have been identified, providing the basis for the work to be carried out.

**Chapter 3:** This chapter deals in detail about a survey which was conducted to identify the users search behavior on WWW while using search engines. The various problems faced by them were identified. Based on the responses received and the analysis carried thereof, inferences were drawn.

**Chapter 4:** In this chapter a novel architecture for incremental parallel web crawler has been proposed. The proposed architecture not only minimizes the overlap problem but also avoids unnecessary communications among client crawlers along with maintaining quality of collectively downloaded web pages. The details about the various components and their working have been provided. While concluding the chapter, a comparison between the proposed and existing architecture of parallel crawlers has been made.

**Chapter 5:** In this chapter, mechanisms that determine whether a web document has been changed over the time from its previous version or not and by what amount, have been proposed. The hallmark of the proposed mechanisms is that changes occurring at micro level i.e. paragraph levels are also identified with a capability to identify three major types of changes namely: content level changes, structural changes and presentation changes. At the end of this chapter the performance of the proposed algorithms with some well known change detection algorithms has been compared.

**Chapter 6:** It is the last chapter of the thesis in which conclusion and future work have been discussed. It has been concluded that the architecture of incremental parallel WebCrawler, designed in this work, has achieved the objectives identified during the literature survey.

The change detection methods for Structural, Presentation and Content level changes have been implemented and their efficacy has been tested. In the end the publications which have been

referred during the work have been listed followed by two appendices. Appendix A provides the details of implementation of parallel crawler and Appendix B contains the web pages which were used to test the efficiency of change detection methods.

The issues of identifying changes in an embedded image and detection of behavioral changes in web documents have been suggested for the work to be carried out in future.

**Keywords:** World Wide Web (WWW), Internet, Search engine, Web document/page, Parallel web crawler, Search algorithm, MT server, Client crawler, Overlapping, Network band width, Ranking, Structural change, content level change, Presentation change, Change detection, Web page relevance, Fingerprint, Checksum.

#### **References:**

- [1] Berners-Lee, Tim, “*The World Wide Web: Past, Present and Future*”, MIT USA, Aug 1996.
- [2] Berners-Lee, Tim, and Cailliau, CN, R., “*WorldWideWeb: Proposal for a Hypertext Project*” CERN October 1990, available at: <http://www.w3.org/Proposal.html>.
- [3] “*Internet World Stats. Worldwide internet users*”, available at: <http://www.internetworldstats.com> accessed on Sept 15, 2009.
- [4] Maurice de Kunder, “*Size of the World Wide Web*”, Available at: <http://www.worldwidewebsite.com> (accessed on 15 Dec 2009).
- [5] Adar, Eytan, Teevan, Jaime, Dumais, Susan T., and Elsas, Jonathan L. , “*The web changes everything: understanding the dynamics of web content*”, In Proceedings of the Second ACM International Conference on Web Search and Data Mining, pp. 282-289, February 2009.
- [6] Brewington, B. and Cybenko, G., “*How Dynamic is the Web*”, In Proceeding of 9th International World Wide Web Conference, pp.264-296, 2000.
- [7] Gerstel, O., et al “*Reducing human interactions in Web directory searches*”, ACM Transactions on Information Systems, Vol. 25, No. 4, Article 20, July 2007.
- [8] *MetaCrawler Search Engine*, available at: <http://www.metacrawler.com>.
- [9] *Dogpile Meta Search Engine*, available at: <http://www.dogpile.com>.
- [10] Saeid, Asadi and Hamid, R. Jamali, “*Shifts in Search and Future Engine Development: A Review of Past, Present Trends in Research on Search Engines*”, Webology, Volume 1, Number 2, December, 2004.

- [11] Monica Peshave and Kamyar Dezhgosha, “*How Search Engines Work and a Web Crawler Application*”. Department of Computer Science, University of Illinois, Springfield USA.
- [12] CHO, A., J., Garcia-Molina Hector, Paepcke, A., and Raghvan, S., “*Searching the Web*”. ACM Transactions on Internet Technology, Vol. 1, No. 1, pp. 2–43, August 2001.
- [13] Page, Lawrence, Brin, Sergey, Motwani, Rajeev, and Winograd, Terry, “*The PageRank Citation Ranking: Bringing Order to the Web*”, Technical Report, Stanford University InfoLab, 1999.
- [14] Cho, Junghoo and Garcia-Molina, Hector, “*The Evolution of the Web and Implications for an Incremental Crawler*”, Proceedings of the 26th International Conference on Very Large Data Bases, pp. 200 – 209, 2000.
- [15] Cho, Junghoo, Angeles, Los, and Garcia-Molina, Hector, “*Effective Page Refresh Policies for Web Crawlers*”, ACM Transactions on Database Systems, Volume 28, Issue 4, pp. 390 – 426, December 2003.
- [16] Liu, Ling, Pu, and Carlton, Tang, Wei, “*WebCQ – Detecting and Delivering Information Changes on the Web*”, In Proceedings of the ninth international conference on Information and knowledge management, McLean, Virginia, United States, pp. 512 - 519, 2000.
- [17] Douglass, Fred, Ball, Thomas, Cheny, Yih-Farn, and Koutsoufios, Eleftherios, “*The AT & T Internet Difference Engine: Tracking and Viewing Changes on the Web*”, World Wide Web, 1(1): 27-44, January 1998.
- [18] Junghoo Cho & Hector Garcia-Molina, “*Parallel Crawlers*”, Proceedings of the 11th international conference on World Wide Web WWW '02, Honolulu, Hawaii, USA. ACM Press, pp. 124 – 135, 2002.
- [19] Paolo Boldi, Bruno Codenotti, Massimo Santini, and Sebastiano Vigna, “*Ubicrawler: A scalable fully distributed Web crawler*”, In Proceeding of The Eighth Australian World Wide Web Conference, 2002.
- [20] Brin, Sergey, and Page, Lawrence, “*The Anatomy of a large scale hyper textual web Search Engine*”, In Proceedings of the Seventh World-Wide Web Conference, 1998.
- [21] Bharat, K. and Border, A., “*Mirror, mirror on the web: A study of host pairs with replicated content*”, In Proceedings of the Eighth International Conference on the World-Wide Web, 1999.
- [22] Manning, C.D., Raghavan, Prabhakar, and Schütze, Hinrich, “*An Introduction to Information Retrieval*”, Online edition, Cambridge University Press Cambridge, England, 2009.
- [23] Paolo Boldi, Massimo Santini, Sebastiano Vigna, “*PageRank as a Function of the Damping Factor*”, In the proceedings of the 14th international conference on World Wide Web, pp. 557-566, May2005.
- [24] Hersovici, M., Jacovi, M., Maarek, Y., Pelleg, D., Shtalheim, M. and Ur Sigalit, “*The Shark-Search Algorithm – an application: tailored web site mapping*”, Computer

Networks and ISDN systems, Special Issue on 7th WWW conference, Brisbane, Australia, 30(1-7), 1998.

- [25] Najork, M., and Wiener, J. L. , “*Breadth-First Search Crawling Yields High-Quality Pages*”, In Proc. of 10th International World Wide Web Conference, Hong Kong, China, 2001.
- [26] Jamali, Mohsen, Sayyadi, Hassan, Hariri, Babak, Bagheri and Abolhassani, Hassan, “*A Method for Focused Crawling Using Combination of Link Structure and Content Similarity*”, Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, pp. 753-756, 2006.

### List of Author’s Publications:

1. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Parallel Crawler Architecture and Web Page Change Detection Techniques*”, WSEAS Transaction on Computers, Issue 7, Vol 7, July 2008. Page(s):929-941, ISSN: 1109-2750 (**Indexed in Scopus**).
2. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Topical Web Crawling Using Weighted Anchor Text and Web Page Change Detection Techniques*”, WSEAS Transaction on Information Science and Applications, Issue 2, Vol 6, ISSN:1790-0832, pp.263-275, Jan 2009(**Indexed in Scopus**).
3. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Web Page Change detection*”, Proceeding of the International Conference on Web Intelligence Systems (ICWIS09), Chennai ,India, Jan 2009, ISBN 978-81-8487-013-8.Page(s):48-58.
4. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Topical Web Crawling Using Weighted Anchor Text*”, Proceeding of the International Conference on Web Intelligence Systems (ICWIS09), Chennai ,India, Jan 2009, ISBN 978-81-8487-013-8.Page(s):145-152.
5. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Change Detection in Web pages*”, IEEE Proceeding of 10th International Conference on IT, Dec 17-20, 07, Rourkela (India). ISBN: 0-7695-3068-0, Page(s) 265-270 (**Indexed in Scopus**).
6. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Architecture for parallel crawler and algorithm for web page change detection*”, IEEE Proceeding of 10th International Conference on IT, Dec 17-20, 07, Rourkela (India). ISBN: 0-7695-3068-0 Page(s) 258-264 (**Indexed in Scopus**).
7. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Incremental Parallel Web crawler*”, IAENG International Journal of Computer Science (reviewers comment received resubmitted, **journal is indexed in Scopus**).
8. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Optical Character Recognition for Hindi Script Using Neural Network Approach*”, IAENG International Journal of Computer Science (reviewers comment received resubmitted, **journal is indexed in Scopus**).
9. Yadav, Divakar, Sharma, A.K., and Gupta, J.P., “*Users Search Trend on WWW and Their Analysis*” (to be communicated).