ISSN (e): **2250 – 3005** || *Volume, 08 || Issue, 5|| May – 2018 ||*
International Journal of Computational Engineering Research (IJCER)

A Novel Technique for URL Sorting and Indexing

Huma Siddiqui[1], Dr. Nidhi Tyagi[2]
[1]*Student of M.Tech department of CSE MIET, Meerut*
[2] *Professor, department of CSE, MIET, Meerut*
Corresponding Author: Huma Siddiqui

ABSTRACT
Indexing in search engines is an effective area in present scenario. With the dynamic nature of the World Wide Web, search engine have a difficulty in getting the relevant data to provide result to user query in shorter time. This paper proposes the technique for URL sorting and indexing. The proposed system explain that there are millions of websites, so from where get the list of websites, rank of websites, and also how many rank will included in search. It uses a Quick sort and B+ search. Sorting of documents will be done by quick sort by giving rank to the documents and for faster accessing of the keyword along with the indexing will be done by B+ search. The relevant word that comes from online dictionary improves the search result.
Keywords: Search Engine, Quick Sort, B+ tree.

Date of Submission: 05-05-2018 Date of acceptance: 21-05-2018

I. INTRODUCTION:

The WorldWideWeb is an area where information and other web resources are identified by Uniform Resource Locators (URLs), interlinked by hypertext links, and can be accessed via the Internet [5]. It is a primary tool for millions of people to interact on internet allows users to access data that is stored on the computers that are connected to the internet using standard interface software. Search engine enables to get information on the internet. It searches the data by simply entering specified keywords and it returns a list of the relevant data where the keywords were found [6]. A Web Crawler that goes around the internet collecting and storing data in a database for further analysis and arrangement. Web crawlers are mainly used to create a replica of all the visited pages by a search engine for later processing. It will index the downloaded pages to provide fast searches. It is also called robots and spiders.

Search engine indexer make easier in indexing relevant data in shorter time. An indexer collects the data, parses and stores data in index which is used by the search engine. It is used for providing the result for search result. And web pages that are stored within the search engine index that will shows on the search engine results page. But the fast accessing of keywords stored in an index is a major issue for performances of Web Search Engines. Indexing is performed on the web pages after they have been stored into a repository by the crawler. The present architecture of search engine indexing is done on the basis of the terms of the document and consists of an array of the posting lists where each posting list is associated with a term and contains the term as well as the identifiers of the documents containing that term. The current search engines use terms to retrieve documents from it [1].

Today the size of the web is increasing with a tremendous rate. So it is very difficult for web user to fulfill his or her information need. Since the user enters only a combination of keywords to search information without giving a thought about the context, search engines therefore search without concerning the user's context of search; providing results which may or may not fulfill the requirements. The basic aim is to select the best collection of information according to users need. The existing search engines adopt different strategies for computing the words frequency in the web documents. If higher frequency words match with the topic keyword, then the document is considered to be relevant [2]. But they generally do not generally search the data in minimum time.

Quick sort is a highly efficient sorting algorithm and is based on partitioning of array of data into smaller arrays. It adopt divide and conquer technique. Quick sort partitions an array and then calls itself recursively to sort the resulting sub arrays [7].

B+ tree often used in the implementation of database indexes. Each node of the tree contains an ordered list of keys and pointers at lower level nodes. To search for or insert an element into the tree, one loads up the root

node, finds the adjacent keys that are search- for value is between, and follows the corresponding pointer to the next node in the tree, recursively eventually leads to the desired value [8].

II. RELATED WORK:

In this section review of previous literature has been discussed. In the field of index organization and maintenance, many algorithms and techniques have been proposed but they seem to be less efficient in accessing the index.

The paper [1] introduces "A New Context Based Indexing in Search Engines Using Binary Search Tree". It uses the BST and stores the context information for every keyword and availability of the keyword in the documents. Their proposed architecture will provide a list of relevant document (With the help of context) for the user query. However their proposed architecture has to be implemented on a corpus with large number of documents in different contexts. But this architecture is somewhere is time consuming.

Another researcher was [2] this paper introduces "Context Indexing in Search Engine using Binary Search Tree [2] proposes a technique for indexing the keywords extracted from the web documents along with their context .The Binary Search Tree based indexing technique, is able to support dynamic indexing and improves the performance in terms of accuracy and efficiency for retrieving more, relevant documents as per the user's requirements since the context of the various keywords is also stored along with them. Thus, the indexing technique provides a fast access to document context and structure along with an optimized searching.

The research paper [3] introduces a double indexing mechanism for search engines based on campus Net. The proposed mechanism has document index as well as word index. The document index is based on, where the documents do the clustering, and ordered by the position in each document.

Further in context based indexing in search engines using ontology [4], the index construction is done on the basis of the context using ontology. The context repository, thesaurus and ontology repository are used by the indexer to identify the context of the document.

The above techniques showed indexing methods for search engine. But some where they are not effective techniques.

III. PROPOSED WORK

The architecture shown in fig 1, of URL sorting and indexing, crawled web pages are store in repository. The indexes for the stored web pages hold the valuable compressed information Web page contains the URL's. The list of URL's is provided from ICANN.

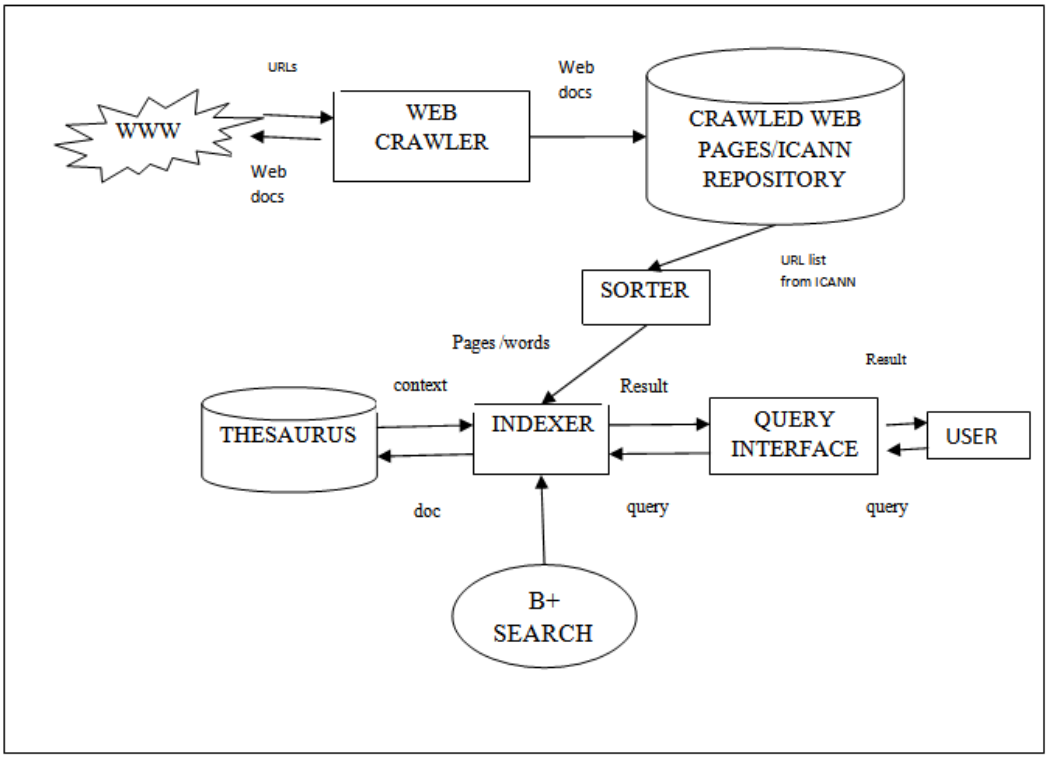

Figure 1: Proposed architecture of URL sorting and indexing

- **ICANN:** It is the organization of the Internet's global Domain Name System (DNS), including policy development for internationalization of the DNS system, introduction of new generic top-level domains (TLDs). From ICANN we get the list of URLs.
- **Sorter:** Here sorting of URLs is done by applying Quick sort technique by site rank. If ranking is same then sorting is done by providing priority to the domain by using the order.
- **Thesaurus:** It is the on-line dictionary for getting the concise definitions, indicative meaning shared by synonyms, and sentences showing how words are used in context.
- **Indexing:** The indexing of search engine indexer will be done by using B+ Tree. All the related words are come from online dictionary.
- **Query interface:** This module will read the query imputed by the user. It access the indexer to find the documents which are relevant for the user's query and also provide an interface to show the searched results to the user.

Algorithm:
Step1. Get list of URLs from ICANN repository.
Step2. Sort these URLs list using site rank by applying quick sort, if ranking is same then sort the domain by using following priority:

- .com
- .org
- .gov
- .net
- .mil
- Last include all other

Step3. Get first URL from the sorted list and crawl all pages of that URL for title tag ad meta tag. Again sort the pages according to the title and meta tag.
Step4. Create data base of that page with words found in title and meta tag. Sort the data of page and store the word list in data base.
Step5. Read second URL of the list then go to 3, 4 steps.
Step6. To search the data in the data base use context indexing B+ tree structure, which is fastest binary search tree with multidata?
This algorithm will search the documents quickly. Quick sort include the efficient average case compared to any sort algorithm, as well as the well-designed recursive definition, and the popularity due to its high efficiency. The quick sort produces the most effective and widely used method of sorting a list of any document size. And B+ tree is easy to implement.

IV. IMPLEMENTATION AND RESULT:

The proposed work has been implemented in java using NetBeans open source IDE.
Here are some snapshots
a. User interface, user enters the search keyword

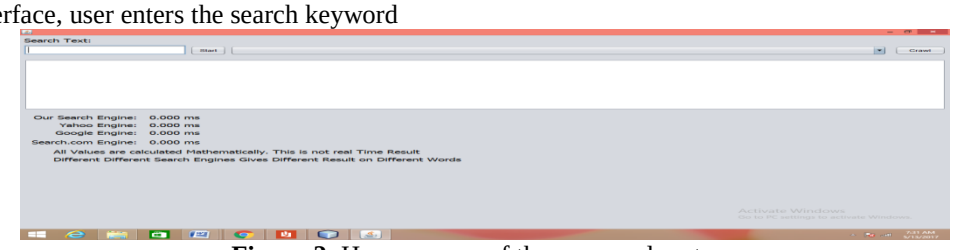

Figure 2: Home screen of the proposed system

b. User enters any keyword "current" in search text box and click on start button.

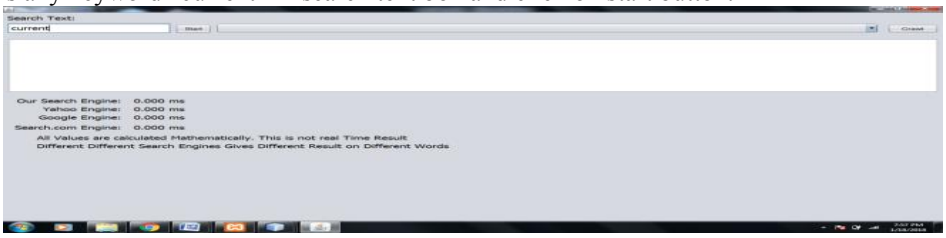

Figure 3: Enter keyword text box

c. On submitting keyword "current", this keyword reflected in adjacent text box.

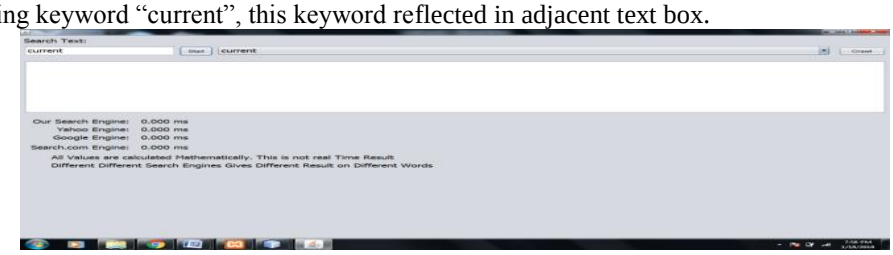

Figure 4: Crawling of keyword

e. All the relevant URLs will be shown. Also the time taken by different search engines also shown which is calculated by mathematical formula.

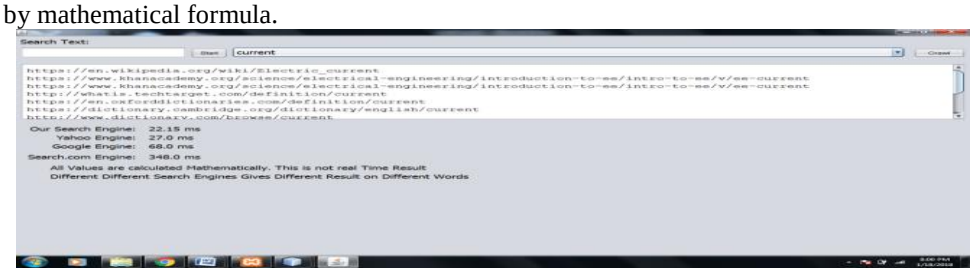

Figure 5: Time taken by the Proposed system and other search engines

f. Now select any URL from list and copy this URL.

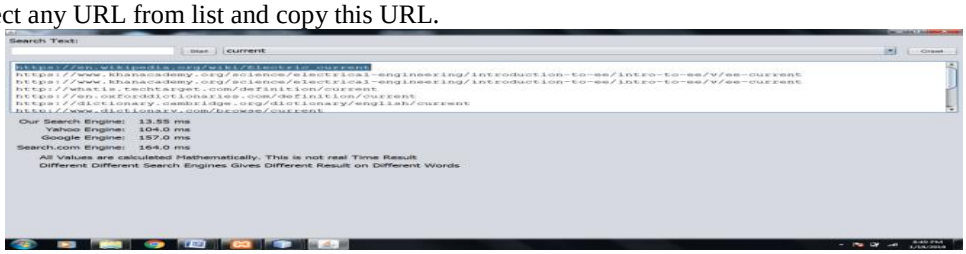

Figure 6: selection of URL

g. Open the URL on any browser (Google Chrome, internet explorer etc).

Figure 7: Web page open in the browser

V. RESULT ANALYSIS:

Table1: Comparison of time taken by search engines on different keywords

Keywords search	Time taken by Google in (ms)	Time taken by yahoo in (ms)	Time taken by search.com in (ms)	Time taken by Proposed Search Engine in (ms)
Apple	211.0	216.0	245.0	210.0
Delhi	64.0	266.0	121.0	22.15
Current	68.0	27.0	348.0	22.0

All values are computed mathematically. Different search engines give various results on keywords. Fig 7 depicts the graphical comparison for the time taken by the proposed system and other search engine.

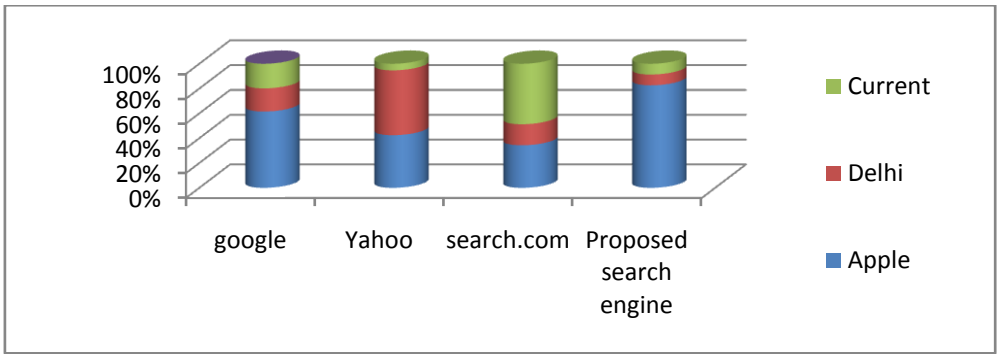

Figure 7: Graph shows time taken by the project and other search engine

VI. RESULT AND CONCLUSION

In this paper, the proposed technique for URL sorting and context based indexing provides the advantage of fast searching of B+ Tree. It shows best relevant result in minimum time, as the both algorithm quick sort and B+ Tree has been used. Complexity of quick sort is O(nlogn) where as B+ Tree search optimizes the search with the complexity of log(n). the efficiency of the search engine has been improve by this technique.

REFERENCES

[1]. Aparna Humad, Vikas Solanki " A New Context Based Indexing in Search Engines Using Binary Search Tree", " International Journal of Latest Trends in Engineering and Technology (IJLTET)" Vol. 4 Issue 1 .
[2]. Charu Kathuria, Goutam Datta, Vanditaa Kaul "Context Indexing Using Search Tree", "International Journal on Computer Science and Engineering " ISSN : 0975-3397 Vol. 5 No. 06 Jun 2013.
[3]. S.M. Shafi, Rafiq.A Rather, " Precision and Recall of Five Search Engines for Retrieval of Scholarly Information in the Field of Biotechnology", "Webology", Volume 2, Number 2, August, 2005.
[4]. Parul Gupta , Dr A.K Sharma, "Context based Indexing in Search Engines using Ontology", "International Journal of Computer Applications" (0975 – 8887) Volume 1 – No. 14
[5]. Huma Siddiqui, Dr. Nidhi Tyagi "Web Documents Indexing Techniques- A Review", "International Journal of Advanced Research in Computer Science and Software Engineering" Volume 6, Issue 8, August 2016 ISSN: 2277 128X
[6]. Anchal Jain , Nidhi Tyagi , "Context Based Web Indexing For Semantic Web" , "IOSR Journal of Computer Engineering (IOSR-JCE)" e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 12, Issue 4
[7]. https://en.wikipedia.org/wiki/Data_structure
[8]. https://en.wikipedia.org/wiki/B-tree

Huma Siddiqui "A Novel Technique for URL Sorting and Indexing." International Journal of Computational Engineering Research (IJCER), vol. 08, no. 05, 2018, pp. 22-26.