# THE ULTIMATE SEARCH ENGINE: AN OPTIMAL RETRIEVAL SYSTEM

**Hisham.KH. Abujalban***

**B. Sriram***

**Dr. Moayad A. Fadhil****

**Abstract**

Currently various search engines are available in the internet. Still always, due to technological developments and requirements, new systems are developed. The search engines play a vital role in web technology in information retrieval processes. The currently available search engines are utilizing the full potentials of the technology. Still, the users expect some advance search system that provides easy retrieval methods and that provides correct results using advanced ranking system. The users' expectations on information retrieval processes need to be significantly met by the search engine providers. The providers need to develop an alternative optimized retrieval routes to compete in the market. In this paper, we have identified various information retrieval processes that enhance the search engine extraction methodologies. Also, we have proposed a model system that provides an optimal retrieval strategy using link based and term based ranking algorithms and enables the users to identify their required solution easily. We have divided the system with two major parts: Offline Warehousing and Online Querying. The proposed model uses various algorithms which helps the users to retrieve the required information in the beginning of the search page itself.

***Keywords:*** *Search Engines, Information Retrieval Processes, Online Search Systems, Optimal Retrieval System, Crawling and Indexing*

* *Faculty, IST Department, Sur University College*

** *Head of Software Engineering Department, Faculty of Information Technology, University of Philadelphia, Jordan.*

**Introduction:**

The internet has a distributed collection of billions of pages that shall be accessed by anyone. The unprecedented ease of access to information has revolutionized the information seeking processes and knowledge acquisition. The internet technology provides the facility to interact between the information providers and information seekers. The ultimate seeking process requires a search engine that accepts the user's query and returns billionth of results in nanoseconds.

The design of good search engine confronts many competitions. The web contains huge volume of information that is relevant and reliable. But some information is irrelevant and unreliable. Due to competitions, the search engines must carefully decide what webpages to be retrieved, and in which order they have to be displayed so that the users shall get their desired results conveniently. The information seeker should feel comfortable to use the search engine in his information seeking and knowledge gaining activities. The users' expectations are that the search engines should show the reliable and precise results in the top priority in order to get the correct results immediately.

Currently various search engines are available in the market. They provide the information in less than the seconds. These search engines use various information retrieval models to retrieve the information from the various webpages. The user has made to check for the exactness of the desirable requirements. As the webpages have numerous words, the search terms given by the users must be particular and specific. Then only the search shall succeed.

In this research, after studying and analyzing various search engines and their retrieval models, we have proposed a new model for information retrieval from the internet. Also, we have proposed a system that combines locator based and context based ranking algorithms. Fadhil (2006) studied and discussed about the optimal retrieval systems [4]. He has introduced new modified ranking system that had two subsystems: Term-Based Ranking, Link-Based Ranking. He also defined the corresponding algorithms for the optimal retrieval. We have used these algorithms in construction of our new search engine system.

**Purposes:**

1. Enhancing the indexing processes by extracting several processes.
2. Implementing new ranking module based on locator based and content based algorithm.
3. Designing new search engine based on the findings

**Literature Review**

Various algorithms related to search engines have been studied and suggested [2, 4, 9]. A generalized link based algorithm had been discussed and the generalized practical implications were suggested by Greets et al [5]. They associated their algorithm with respect to relational databases and set of queries a unique weighted directed graph. Pavalam et al [10] studied the usages of web crawling algorithms used on searching. They said that the usage is web crawler algorithms determine the relevancy based on the factors frequency and keyword location. This process is iterative, as the user's required results have been obtained.

According to Menczer et al [8], topical crawlers are used to address the scalability limitations of universal search engines. They analyzed computational complexity of crawlers and found that evolutionary crawlers high efficient and scalable.Bharat [1]described an extension to Search Engines to explicitly maintain user searchcontext as they look for information, on many topics, using many Search Engines,and over many sessions. The paper presents an extension to Search Engines called***SearchPad***that makes it possible to keep track of 'search context' explicitly.

Kim and Cho [6] presented the importance of personalized search engines in finding web documents for specific users. They proposed a search engine using link structure and fuzzy concept network. The search engine finds relevant documents in which user is interested in, and reorders itwith respect to the user's interests. The Search Engine finds authoritative and hubsources for a user query using link structures.

Keyhanipour et al [7] suggested that meta – search engines could be considered as an interface on the top of local search engines to provide uniform access to many local search engines. Their experimental results showed that the concerning user preferences as wellas integration of decision lists has significant effect on reducing users' time andeffort for finding the required information.

**Proposed System Architecture**

The proposed system of search engine has been shown in Figure 1.

The system has been divided into two major parts: Offline Warehousing and Online Querying. The parts are independent to each other. They get connected based on the users' request.

**Part 1: Offline Warehousing**

This is internal part of the search engine that runs independently at providers end. This part is major part of the system that collects the data from the various webpages and stores them in the database for users' retrieval. It has the following subsystems and databases.

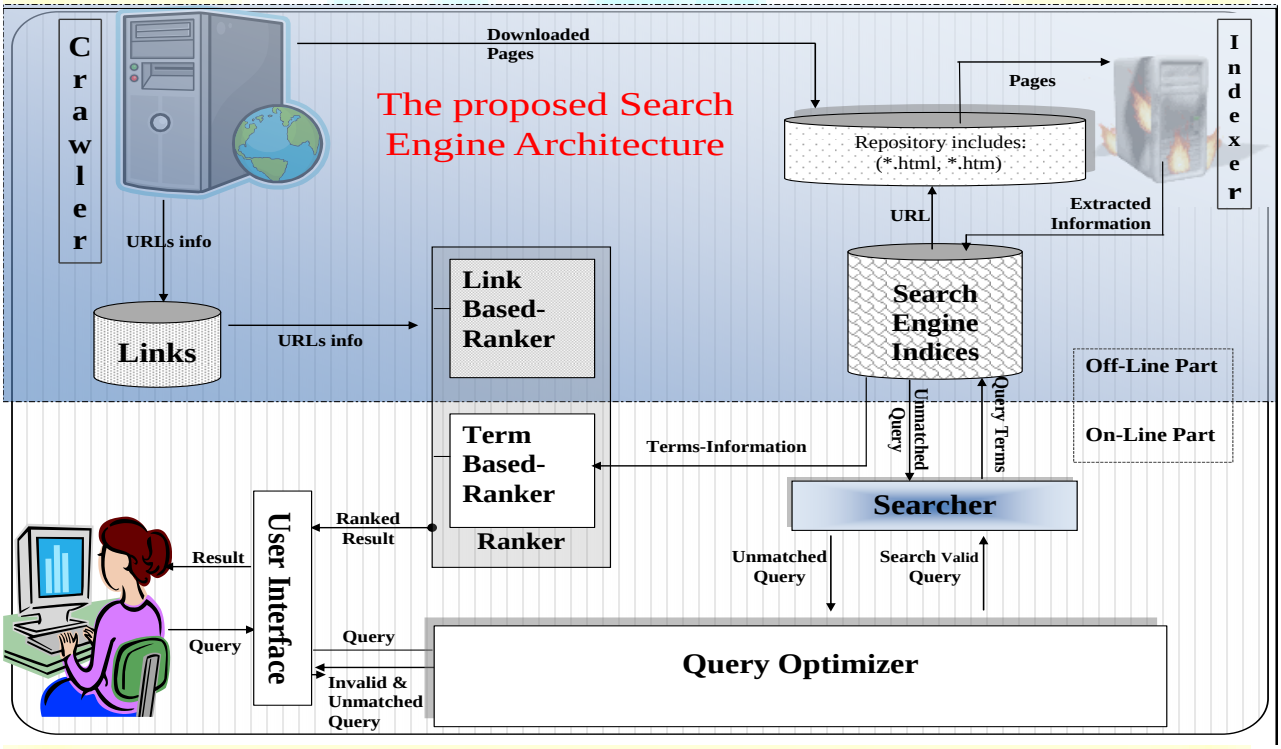



Fig 1: Proposed System

**Locator Divider:**

This is an application program that identifies the Uniform Resource Locators (URLs) and divides them into their parts. The search engine database will contain a table for saving the locators with their respective parts. The table will contain the following fields: protocol, host name, domain name, root folders, subfolder, sub subfolder, document name and file extension. Based on the

URLs specifications, the fields will be filled. If any part does not exist in the URL, the field will be left blank. In the first step, the URLs are normalized to general format. The web address will divided with help of special characters used in the web addresses such as colon (:), slash (/), double slash (//), backslash (\), and period (.). The parts will be stored into respective fields for future references and retrieval. The Locator Divider Algorithm is shown is Figure 2. The purpose of this locator divider is give more weightages to the indices and ranking the respective webpages at top ranking so that the information seeker shall get the desired result in the beginning of the page itself.

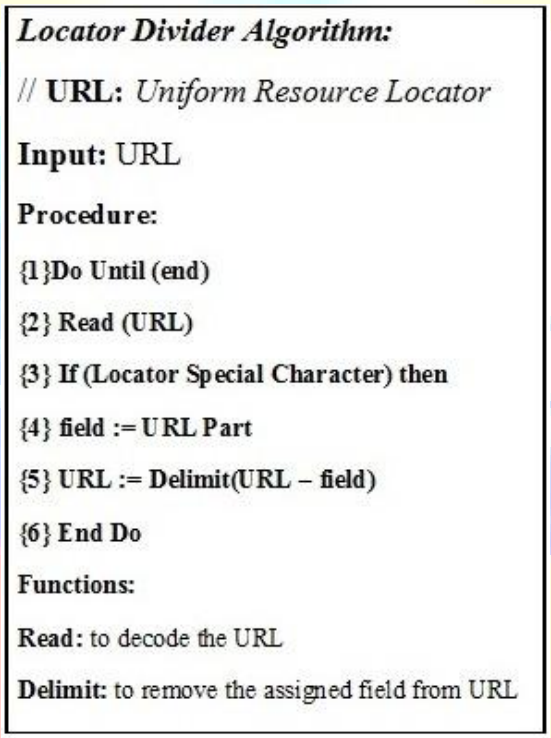



Fig 2: Locator Divider Algorithm

**Crawler:**

Crawler is a software application that systematically browses the webpages and the related information. Based on the on the information retrieval the crawler will create a web indexing. This will follow the general crawler policies such as Selection Policy, Revisit Policy, Politeness Policy and Parallelization Policy (Wiki, 2013). The crawler shall follow any type crawling algorithms suggested by Cho et al (1998) [3], Sheng (2012) [11]to index the webpages and store it into database. Once the webpages have been crawled, file type and URLs will be stored into

the search engine indices. Then the crawler dequeue another webpage from the URL list to repeat the process. The crawler stops when no more un-crawled webpages exist.

**Indexer:**

The indexer is one of the most complicated and critical process in constructing the search engines. Indexer is used to analyze the crawled webpage contents, keeping about each word occurred in the individual webpages. The real texts are divided into stream of words by translating the texts into sequence of character codes. Translating real text into a stream of words is usually partly dealt with by implementing an appropriate transformation function in the first step of an indexing algorithm. This first step can be organized into the sequence of methods shown below:

**Document Conversion**

It is the first step in translating process which refers to any method that transfers an input text into a sequence of character codes.

**Parsing**

Parsing uses an input sequence of characters to determine word boundaries, and outputs words as (d , p , w). Finding a word boundary can be simple if special characters such as space, full stops, and semicolons are used. These special characters and other such symbols are usually excluded from an index, because their usefulness is limited when searching with methods that are based on the occurrence of words. Determining the end or the beginning of a word can also be a hard task in other situations.

**Word Transformation**

Word transformation concerns different words with equivalent meanings. It is sometimes appropriate to return documents, which contain these equivalent words, as result documents, even if they were not present in the initial query.

In this research, an enhanced Indexer is proposed, which will lead to enhance the quality of the extracted information that must be more accurate to be useful in describing the content of the

indexed Web pages. Once the enhanced indexer reads the webpages as a normal text file, it performs the following steps.

**Page Parsing**

Most Search Engines are used to convert the whole Web page into a pure text page then parse it partially. In parsing step (in the Enhanced Indexer), each page is fetched from the repository and parsed into a text file which contains the most important HTML tags. These tags are support to the Indexer in the next steps in order to extract the information which is formed, then form accurate attributes for each word occurring in the indexed page.

**Extracting keywords and their attributes**

In this step, after parsing process is completed successfully, the lexical tokens are generated from the text file of the Web pages. Each token is checked whether it is a HTML tag or a real word. If the token is HTML tag it will be processed in the next step, but if the token is a real word a negative dictionary (a dictionary that contains the stop words) is used to remove the noise words that do not reflect the content of the Web page. The negative dictionary (Stop-Word) exists in the Search Engine Indices.

**Inverted Indexing**

In the final step, each prepared token will be processed according token algorithm.While the reading head does not reach the end of the file the Indexer will read the ith token (where *i*refer to the word's number within the page). If the read tokeni is tag it will be processed by theprocedure *tag_process*(tokeni) which extracts the attributes of its related word(s). Each tag has different steps of processing which depends on its type. If the token is word and not anunwanted word (not *stop*(wordi)) then, word descriptor will be extracted, and the word's word_id (identification number of a word in the Lexicon) will be fetched to store the word_id and word descriptor attributes in the Inverted Index Table in Search Engine Indices. In the Inverted Index a mapping between the word_id and the page_id where the word occurs is done.

**Repository**:

This will contain the collection of webpages downloaded by the crawler. All the webpages are indexed using enhanced indexing system.

**Search Engine Indices:**

It is the database that contains all the tables using various algorithms.

**Ranker**

The ranker system ranks the information crawled, read, analyzed and retrieved. The ranker ranks the webpages based on locator based algorithm and context based algorithm. The context based algorithm is divided into two parts: Link Based Ranking and Term Based Ranking (Fadhil, 2006). Link based ranker will work in Offline Warehousing mode. The page ranks of the webpages are calculated inbound and outbound links. Term based ranking will work in Online Querying mode. This will provide the more relevant pages in the first place.

**Links**

The links are the repository that contains the details of the extracted URLs and page link rank and table.

**Part 2: Online Querying**

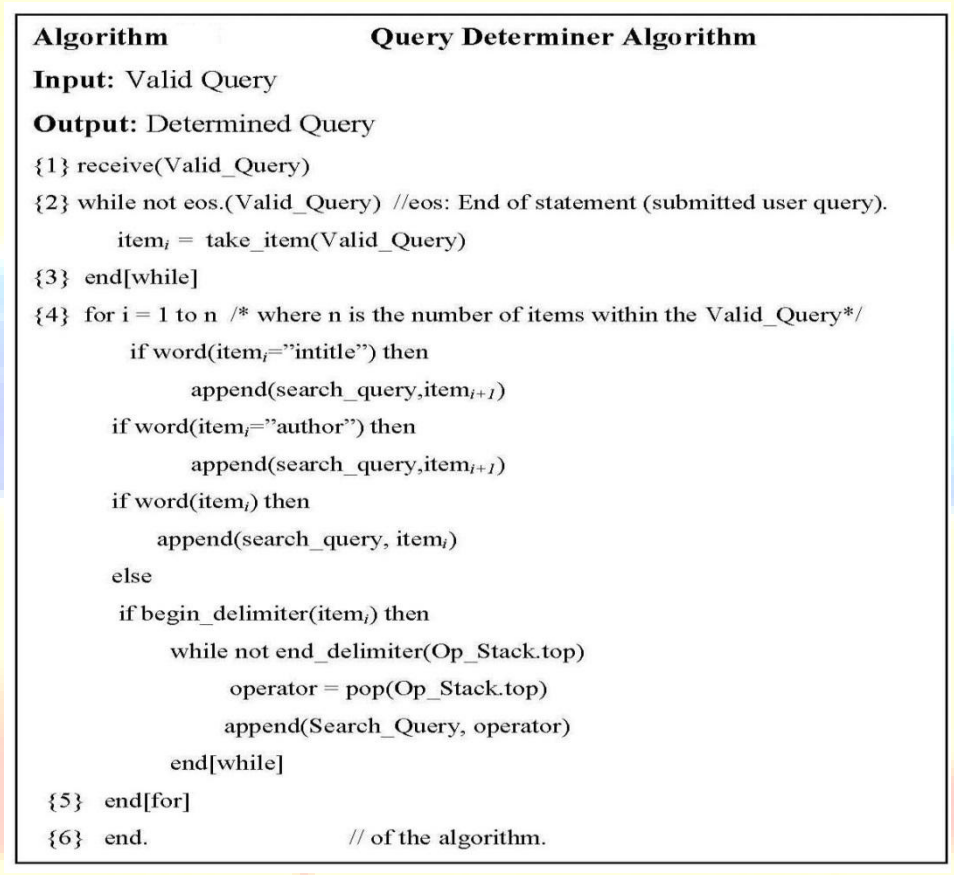
This is the part that works while the searcher is online. This part depends on the query that is given and searched by the users.The part consists of the following.

**User Interface**

This is the main webpage of the search engine. This has two parts: one for querying and another for answers. The user will search for the required information through the querying interface. The searched results will be shown in the answer interface. The proposed Search Engine supports complex query interface, which includes Boolean operators and other features, such as phrase search and title search. For the answer interface, the proposed Search Engine usually returns pages in the order of relevance to the query. In other words, the page that has the higher ranking score appears on the top of the list. Typically, each result entry in the list includes a title of the page, a URL, a brief summary.

**Query Optimizer**

Query optimizer is the part that checks the query of the user for validation and determination. The query of the user will be optimized with error free terms. The special characters and the unwanted words will be removed from the query. The query determiner will determine the relationship between each two consecutive terms. Query Determiner Algorithm is shown in figure 3.



**Algorithm** **Query Determiner Algorithm**

**Input:** Valid Query

**Output:** Determined Query

```
{1} receive(Valid_Query)
{2} while not eos.(Valid_Query)  //eos: End of statement (submitted user query).
        item_i = take_item(Valid_Query)
{3}  end[while]
{4}  for i = 1 to n  /* where n is the number of items within the Valid_Query*/
         if word(item_i="intitle") then
                append(search_query,item_i+1)
       if word(item_i="author") then
                append(search_query,item_i+1)
       if word(item_i) then
            append(search_query, item_i)
       else
        if begin_delimiter(item_i) then
             while not end_delimiter(Op_Stack.top)
                  operator = pop(Op_Stack.top)
                 append(Search_Query, operator)
             end[while]
 {5}  end[for]
 {6}  end.                          // of the algorithm.
```

Fig 3: Query Determiner Algorithm

**Searcher**

The searcher is the core of relevant pages selection process. It receives the query from theQuery Optimizer Subsystem, and then builds SQL statements to search the Inverted Index Table in the Search Engine Indices for a match with any keywords to retrieve word_id. Depending on the retrieved word_id the required information will be extracted.

**Conclusion**

The corresponding system entity relationship diagram is shown in figure 4.

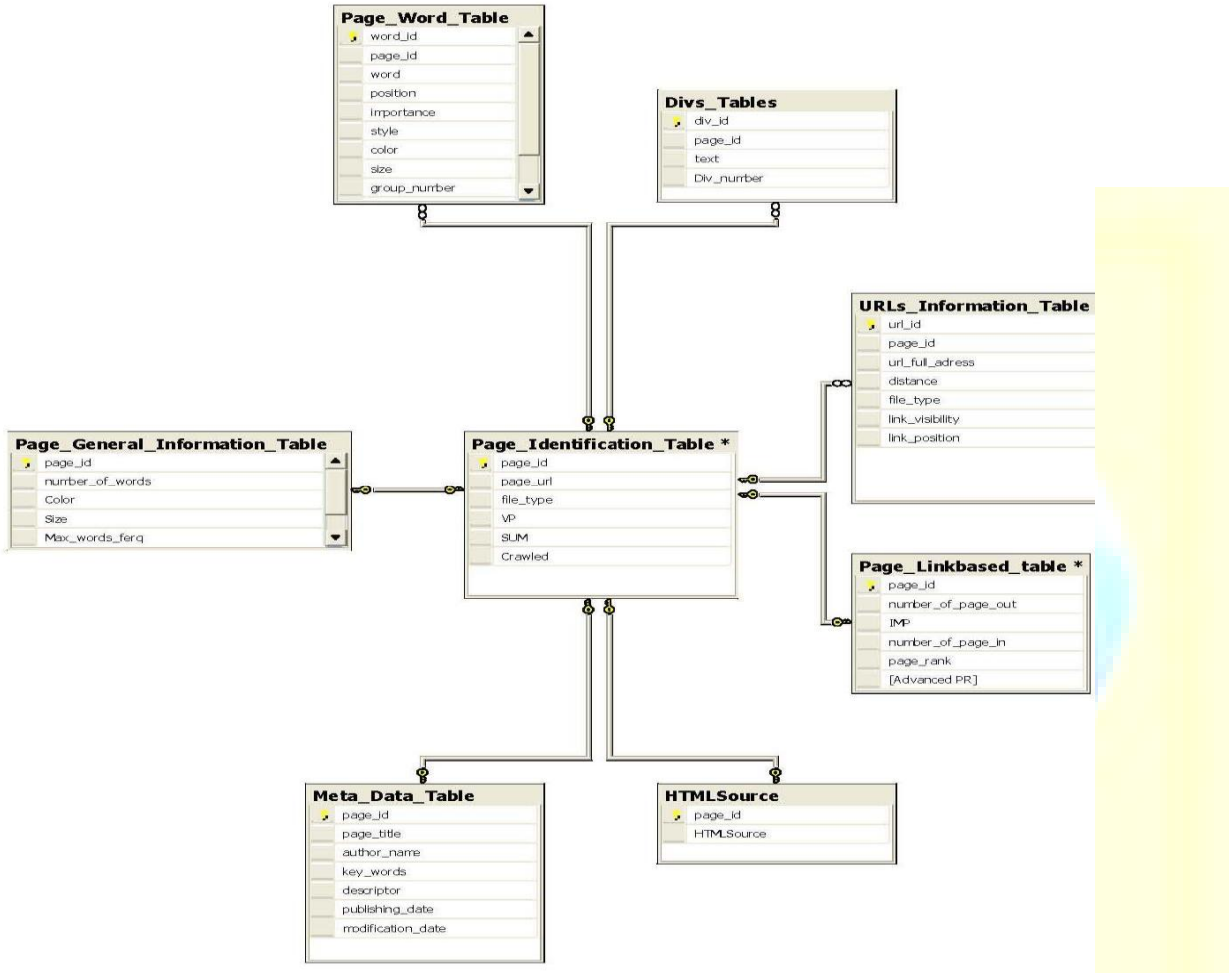



Fig 3: System Entity Relationship Diagram

The term based ranker will rank the pages based on the search words and its position in the webpages. If the search word exists in the URL table domain name, the ranker will give more weightage to the particular link in such a way that those will be shown in the first page. Similarly the ranker term based ranker will check the search word and its existence in the URL table. If the search word(s) exist in the URL table, those results will be shown in the first page. The further ranking will be based on the repetition of the search words in the particular webpage. The webpages that contain maximum number of repeated search words will be given higher priority.

Thus, this system will rank the webpages with close to user requirements and will displaythe results accordingly. This system will benefit the users to retrieve the desired results immediately with higher ranking.

**Acknowledgement**

We would like to thank the Dean and the management of Sur University College for their financial and technical supports. Also we would like to thank the Sur University College IST department staff members for their valuable inputs and supports.

**Reference:**

[1]. Bharat, K. (2000). Search Pad: Explicit Capture of Search Context to Support Web Search. *Computer Networks.*Vol 33, Iss 1 – 6, June 2000, pp: 493 – 501.

[2]. Brin. S and Page. L (1998). The anatomy of a large scale hyper textual Web search engine. ComputerNetworks and ISDN Systems, 30:107–117.

[3]. Cho, J., Molina, H.G., Page, L. Efficient Crawling through URL Ordering. Visited and retrieved from http://oak.cs.ucla.edu/~cho/papers/cho-order.pdf

[4]. Fadhil. M.A. (2006). Enhancement of Ranking and Query Optimizer in Internet Search Engine, *Al-Rafidain University College Journal for Science*, Issue 19, pp: 99 – 122.http://www.iasj.net/iasj?func=fulltext&aId=44133

[5]. Greets, F., Mannila and Terzi, E, (2004). Relational Link-Based Ranking. *Proceedings of the 30th VLDB Conference.* Canada.

[6]. Kim, K.J., Cho, S.B. (2007), Personalized mining of web documentsusing link, *Applied Soft Computing*, 7 pages 398–410.

[7]. Keyhanipour, A. H, Moshiri, B, Kazemian, M., Piroozmand, M, and Lucas. C. (2007), Aggregation of web search engines based on users' preferences inWebFusion, Knowledge-Based Systems 20, pages 321–328.

[8]. Menczer, F., Pant, G., and Srinivasan, P., (2003). Topical Web Crawlers: Evaluating Adaptive Algorithm. *ACM Transactions on Internet Technology*, V(N), pp: 1 – 38.

[9]. Ng. A., Zheng. A., and Jordan. M. (2001). Stable algorithms for link analysis. In SIGIR.

[10]. Pavalam, S.M., Raja,S.V.K., Akorli,F. K. and Jawahar, M. (2011). A Survey of Web Crawlers Algorithms. *International Journal of Computer Science Issues.* 8(6). pp: 309 – 313.

[11]. Sheng, C., Zhang, N., Tao, Y., Jin, X. (2012). Optimal Algorithms for Crawling a Hidden Database in the Web. *Proceedings of the VLDB Endowment*, Vol 5, No 11.

[12]. Wiki. Visited and retrieved from http://en.wikipedia.org/wiki/Web_crawler