An Improved Generic Crawler using Poisson Fit Distribution

Thangaraj M.
Associate Professor, Madurai Kamaraj University,
Madurai -21

Sivagaminathan P. G.
Research Scholar, Bharathiar University,
Coimbatore-46

ABSTRACT

The remarkable growth of Internet populates the World Wide Web to contain huge web data which is unexplored to whom it is intended for worth extraction and assimilation into knowledge. Retrieving potential information from web data needs a broad-spectrum crawler to collect relevant documents and metadata. Breadth first crawler algorithm is presented to fetch related web documents essential to create a web archive for alias extraction. In this paper, it is proved that the upgraded crawler generates better random depth rather than predetermined depth crawling. Contributing different mean values to this function enabled crawler it is possible to generate dynamic random depth.

Keywords

Breadth First Search, Parsing, Multi-threading, *Probability Mass Function, Frontier, Virtual web

1. INTRODUCTION

The number of web pages has been growing exponentially in the web invariably in different disciplines. Web search engines came into existence in order to make accessing and searching easier in a user point of view. Every search engine internally maintains databases of HTML documents with a well defined index. This database of HTML document is maintained by special software called Web Crawler. There are popular general purpose crawlers like *RBSE*, *BingBot*, *PolyBot*, *WebFountain*, *GoogleBot*, *WebRACE*, *World Wide Web Worm*, *Yahoo Slurp* and *GM Crawl*. Few open source crawlers are *Web Sphinx [26]*, *Nutch*, *Scrapy*, *Seeks*, and *Xapian*. A tool called *Lucene* is a open source IR library used for text indexing and searching. Categories of crawlers are batch crawler, incremental crawler, focused crawlers. Special software poses [25] challenges due to large volume of web data, fast rate of change, dynamic page generation and accessing hidden pages.

Among thousands of trillions of web users, nearly 30% percent of search engine queries are about person names [11], places and objects in real world. Further, it is increased with modern smart phones, palmtop and other handheld devices. People inquiry mind often foresee to access any kind of information within short span of time. A2E stands for Automatic Alias Extraction [9] experimented on different datasets for information retrieval. A2E comprises [22] of three software components a) Generic Crawler b) Alias extraction engine c) GUI form. Crawler uses breadth first traversal method [16] to fetch related URL's from web documents on a web graph stored in a queue for further task to be carried out in searching. The second component alias extraction engine catches concurrent text in accessed page and build indices for each pattern that appears in a group of URL's stored in database by automatically eliminating stemming and stop words. The third component is used to provide query processing support to user output.

It is a pre-requisite for the tool to provide a most relevant seed URL pertaining to the search query and it is user centered. It means irrelevant documents are eliminated to some extent possible by choosing the potential URL for searching task. While retrieving relevant documents, it also allows few irrelevant documents getting stored in database. In this method, a web archive is created to perform alias extraction using regular expression. It is obvious that regular expression parses web documents rapidly and it need not read the entire web page content for locating aliases. Hence, Generic crawler has been designed to create a simulated web environment as in Figure 1. In this method, if the URL selection is not proper obviously it adds large garbage to the web archive. However, rapid alias searching and extraction is done in web archive using regular expression pattern matching technique followed by candidate alias extraction.

Web Crawler is one of the important components in any search engine and Information Retrieval task. Web Crawler is a software program which, navigates the web and extract new pages for storage in search engine database. The job of this crawler is to traverse the graph [6][17] for parsing, elimination of duplicate pages[8], robot exclusion, and downloading pages as directed by the scheduler in Figure 1 of architecture.

Crawlers are otherwise known as Web Ants, Web Robot, Wanderers, Bots, Worms, and Automatic Indexers. Commercial search engines often fine tune the crawling methodology and rebuild indexing techniques very frequently to make their search faster and to remain popular in online market. Popular search engines use parallel crawlers to achieve better and faster results. Some of the popular search engines are Google, Yahoo, Altavista, AOL, msn, opera, Overture, Netscape, Excite, AskJeeves, InkTomi, Mozilla Firefox.

2. RELATED WORK

2.1 Generic Crawler used in *A2E* Tool

This research work commenced with an objective to extract aliases for a given set of entities using formal co-occurring patterns as available in the web corpus. Normally, any keyword based shallow pattern extraction brings more irrelevant documents when a query is given for searching. Proposed approach with regular expression is a bit enhancement in searching. Crawler used in this method brings better results when collecting relevant pages through proper URL selection meeting out selection policy on crawlers. Multi-threaded downloader is capable of generating maximum of ten threads to navigate hyperlinks across the breadth of a page. Timeout period in this method is fixed threshold of 30 seconds. It means time gap between thread requests and getting a negative response from web server to fetch an URL is termed as timeout period. If timeout exceeds, previously waiting thread moves to next available URL.

The scheduler is an algorithm controls multi-threaded crawlers to ensure optimized usage of downloading maximum number of pages in a minute at the same token without system crashes, concern over other resources and web servers. Program of this kind consumes client CPU time, network bandwidth to download pages, memory and web servers. Crawler is designed at par with the following crawling policies such as selection policy, politeness policy, and parallelization policy. This method experimented on a single processor machine downloading 6 pages per minute using multi-threaded program written in VB.NET. However, alias extraction in local archive uses a single query search instead of multithreaded architecture for safety purpose. To avoid system crashes, server slow down designers should make sure web server should not be overloaded, and also to abide by robot exclusion policy of the respective web servers. Speed of the parallel crawler and downloading voluminous relevant pages would certainly increase retrieval efficiency in alias extraction. The downloaded pages are held in database in compressed form along with relevant meta-data which act as a web archive altogether as in Figure 1.

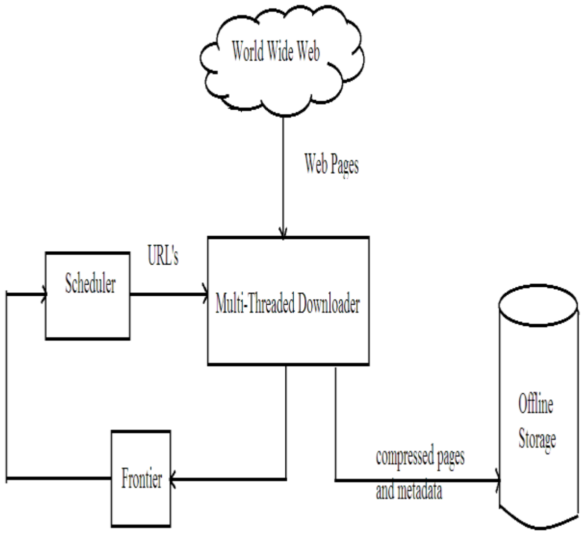

Figure 1: Architecture of Generic Crawler

2.2 Breadth First Search

This breadth first searching method [16][19][24] is otherwise known as blind traversing approach since it works blindly as it selects next URL from the Frontier. Crawling links are selected in the order in which it is encountered in the frontier. In a real-life extraction, where billions of documents are required to produce meaningful results, this search method allows few unwanted pages getting in to the offline [5] storage. This blind traversal performs well when most relevant seeds are selected for each pattern-name pair for efficient extraction.

2.3 Design Feature of Generic Crawler

Table 1: Existing A2E Crawler Properties

Crawler Characteristics	Description
List of Sites avoided while Crawling	Robot exclusion policy in Web Servers, and Wikipedia resources. Eliminates pictures, audio and video content but stores only textual matters
List of Selected Sites for the Crawl Celebrity Name Dataset Drug Name Dataset	www.espncricinfo.com, www.foursquare.com, www.cricketcountry.com, www.webindia123.com , www.imdb.com, www.quora.com etc., (i.e) Seed URL for each pattern-name pair www.medicalnewstoday.com, www.medications.com/sinusitis, www.drugs-forum.com, www.healthboards.com, etc.,
Maximum number of Crawler Levels	Ten levels
Keywords to Limit Crawl Expansion	Pre-defined Patterns
Maximum number of URL’s to Visit	No limitation in the Breadth of Graph G
Maximum number of Relevant URL’s	Maximum twenty, Relevancy is checked during seed selection

Timeout Period to fetch an URL	Waiting time of a thread to fetch an URL is 30 seconds. If exceeds, thread moves to next URL
Crawler Characteristics	Description
Number of Crawlers	Ten multithreaded Crawlers or Parallel Crawlers
Maximum time for a Crawler run	No time fixed in algorithm
Threshold value for including document	No Threshold
The Degree of Overlap between Clusters	NA
Maximum number of Clusters	No Indexing at crawler level, hence no clusters used
Robot exclusion Policy	Checks "robot.txt" in every web server for permission before Crawling
Spam Filtering	No spam rejection
Politeness Policy	Server should not be overloaded due to multiple crawler request, Crawls only permitted Web Sites and Pages
Parallelization Policy	Maximizing download rate, Minimizing overhead at the same time avoiding repeated downloads
Maximum Downloading time of a page	No limit
Number of pages crawled per minute	6 pages

It is evident that Wikipedia contains authentic information and references for all domains. However, permission is denied by Wikipedia for any public robot to crawl. This crawler fetches web pages with textual content alone by avoiding pictures, video, and audio files for efficiency reasons as per the property stated in Table 1.

The Input of a crawler is a URL for personal name alias extraction and drug name alias extraction. Every unique name-pattern pair needs a separate URL for fetching the relevant documents essential for alias pattern matching and extraction. For Instance, '*rahul dravid better known as ** ' is a text pattern need to be located in sports related web documents in order to get meaningful results. Hence, the user should make sure that right URL is given for each name-pattern pair. Likewise there are some other patterns, such as '*[name] also known as **', "** aka [name]*", "*[name] aka **", "*[name] popularly known as **" etc., were used.

The crawler is designed so as to traverse unlimited nodes across the breadth but with a constant depth of ten. The advantage of this method is that it makes use of multi-threaded parallel crawling techniques to speed up breadth first traversal. The output of this crawler is a database containing offline textual web pages in compressed form along with Metadata for further processing and analytics.

All web servers contain "*Robot.txt"* file assigned by the web master, to check whether permission is granted to crawl root directory, sub-directories or folders, specific site and specific page. If the permission is denied [23] at the root directory, none of the web documents in that server is accessible by crawler robots. In some servers, root directory of a server is permitted but folders are forbidden based on web administrator policy decision.

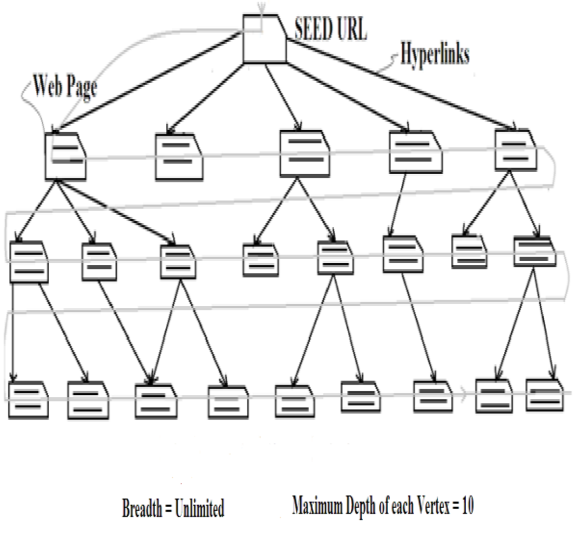

Figure 2: Breadth First Traversal with Fixed Depth

Breadth first algorithm works on a level by level as in Figure 2. Breadth First Search is used only to collect URL's from a page. It starts collecting URL from the seed URL continue searching at all neighboring vertices of G at the same level. When all the hyperlinks are crawled at the same level, it moves to next level and it repeats the same. This Breadth first search is preferable where objective is found in the shallower parts in a deeper tree [12][20]. Breadth first search is not suited for game tree kind of problems where there are multiple branches leading to the same objective with the same length [2].

2.4 Breadth First Crawler Pseudo Code with Fixed Depth

Output database table comprises of three columns PARENT_URL, CHILD_URL and PARENT-DATA. Initially seed URL is assigned to PARENT_URL. As in algorithm 1, seed page is completely parsed and algorithm collects N number of hyperlinks across the breadth of graph G. It traverses each and every hyperlink breadth wise as many as available in the document in the order of appearance. When all pages of a seed are visited, breadth first traversal moves to next level of vertices recursively to a depth of ten for each vertex. Here, each visited web site is a vertex and traversed hyperlinks as edges in graph G.

In General, crawler runs continuously forever and terminates only when there are no more links on web. This could be true for few seed URL's but not all the time. Consequently, time frame cannot be estimated to traverse the entire web even for a single seed. Mostly, crawler continues to run even with a single link for indefinite period and comes to halt only when it is explicitly closed by the user. The fixed depth is being counted for each and every vertex at various levels of breadth first graph. In algorithm 1, each CHILD-URL will in turn becomes seed URL for further recursive search with a depth of 10 and thus it takes indefinite time to halt. PARENT_DATA column stores web documents in binary form.

In this research, for alias extraction 3.20 GB of sports related documents and 3.15 GB of drug related documents were collected offline for pattern matching, candidate alias extraction.

The efficiency of Information Retrieval task named alias extraction had been proved with statistical measures precision, recall and F-score.

```
/* Vertices Visited Once and Ensure Duplicate Page Rejection*/
Pseudo code WebCrawler1.0
MAX_DEPTH_LEVEL=10; d=0;
PARENT_URL=Seed_URL              /*Input to the Algorithm through a form*/
Frontier=Enqueue_URL(PARENT_URL)
Repeat(d<=MAX_DEPTH_LEVEL)
BEGIN
   Repeat(Frontier not Empty)
   Begin
      Dequeue URL from Frontier
      Find the IP address of its Hostname
      Download the Page and Store in PARENT_DATA column of table in
      compressed form
      Parse the Web Page and Extract all Hyperlinks contained in it
      Assign the First Occurring Hyperlink to CHILD_URL column of table
      NEXT_LEVEL_URL=CHILD_URL
                     /*set a counter to hold number of hyperlinks in a Web Page*/
         Ctr=number of Hyperlinks in a Page
         Repeat (Ctr!=0)
         Begin
            Insert Hyperlink (URL) in to the Frontier based on order of
            appearance on Page
            Ctr=Ctr -1
         End
      d=d+1              /*increment depth by 1*/
                        /*starting address of level d is assigned as new PARENT_URL*/
      PARENT_URL=NEXT_LEVEL_URL
      FRONTIER=Enqueue_URL(PARENT_URL)
   End
END                              /* Output stored in a Database of three columns */
```

Algorithm 1 – Existing Algorithm with Fixed Depth

3. PROPOSED WORK

The improvement in the existing breadth first crawler could well be achieved by invoking a Poisson probability distribution [27][28] or probability mass function to return an optimized random number ranging from 0 to 35.

In probability theory and statistics, a probability mass function (PMF) is a function that gives the probability that a discrete random variable is exactly equal to some value. The probability mass function of a discrete random variable is a function that defines the probabilities that the random variable takes particular values in its range.

3.1 Probability of events for a Poisson distribution

An event can occur 0, 1, 2 times in an interval . The average number of events in an interval is λ.

```
Function Int Depth_PMF( λ )
/*floating point array P[ ] holding Probability Values */
/* X=0,1,2,3,4…………35 */
/* λ  Average Depth of Existing Crawler*/
/* Using λ, function generates a random number from 0 to 35
1.          I =1; X=0; depth=0.0;Xfact=1;
2.          REPEAT(I<=35)
3.                  Repeat(X<=I)
4.                      IF(X=0) THEN
5.                          Xfact=1
6.                      ELSE
7.                          Xfact = Xfact * I
8.                          X=X+1
9.                      ENDIF
10.                 End
11.                 $P[I]= (e^{-\lambda} \times \lambda^{I})/Xfact$      /*Probability of $P_x( X)$
12.                 I=I+1
13.                 IF(P[I]==0.00) THEN    //*correct to 2 decimal places
14.                     depth=I;        /* converges to 0.00, I is assigned as depth*/
15.                     Return(Ceil(depth));
12.                 ENDIF
13.         END
```

Algorithm 2 – Probability Mass Function

λ is the event rate or rate parameter. The probability of observing x events in an interval, is given by the equation

$$P_x(X) = (e^{-\lambda} \times \lambda^{x}) / (x!)$$

λ - Average of event per interval

e - 2.71828 (Euler's Number) the base of Natural logarithm

x - Takes value 0,1,2,……..35

x! - x=(x)×(x-1)×(x-2)……×(2)×(1)

Where x = (0,1,2,3,4………..35), Random Depth is calculated by depth = x when P(x) reaches 0.0 as probability.

```
Pseudo code WebCrawler2.0
/* Revised crawler with random depth generation*/
1.      Display "Input Mean value between 0  to 35" ,
2.      Accept  mean
3.      MAX_DEPTH_LEVEL=Depth_PMF(mean);
/*  Call  steps from 2 to 26 from WebCrawler1.0 */
```

Algorithm3 – Crawler with Poisson Probability Distribution

In the existing method, the average depth of graph G is considered as λ, (i.e) λ=10. Traversing the deep web leads to infinite values and it is never ending. Hence, this function generates random depth ranging from 0 to 35.

This is done with an intention to maximize downloadable pages from each visited page while crawling. Once the local archive size is large with relevant documents, chances of retrieval will yield better result than the fixed depth crawler.

3.2 Tabular Values Obtained from Probability Mass Function

Table 2: Probable value converges when x=19

Mean λ =10, Random Number = 19

X	0	1	2	3	4	5	6	..	18	19	33	34	35
$P_x(X)$	0.00	0.00	0.00	0.01	0.02	0.04	0.06	..	0.01	0.00	0.00	0.00	0.00

Table 3: Probable value converges when x=25

Mean λ = 15, Random Number = 25

X	0	1	2	3	4	5	6	7	.	24	25	..	34	35
$P_x(X)$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	..	0.01	0.00	..	0.00	0.00

Table 4: Probable value converges when x=32

Mean λ=20, Random Number=32

X	0	1	2	--	10	11	12	13	..	30	31	32 to 34	35
$P_x(X)$	0.00	0.00	0.00	0.00	0.01	0.01	0.02	0.03	…	0.01	0.01	0.00	0.00

Initial Guess λ=10,15,and 20 are considered for determining depth at each vertex and Probability Mass Function is called as in the Algorithm 2 and Algorithm 3.

4. PERFORMANCE EVALUATION

4.1 Graphical Output

Existing crawler uses fixed defined depth in collecting adequate documents to perform mining on huge web data. Mining results can be better if more and more relevant pages are available in virtual web. With an objective to enhance the existing crawler, a probability mass function is included in the crawler and results have been proved to be worth considering as an upgraded version.

The new probability function enabled crawler goes further deep in to the virtual web for a mean or λ=10 with a generated random depth of 19 as in Table 2. Similarly from Figure 3, for λ=15, generated random number is 25 as given in Table 3 and for λ=20, generated random number is 32 as in Table 4. It is evident from the graph that function enabled crawler is advantageous than existing fixed depth crawler in web extraction task.

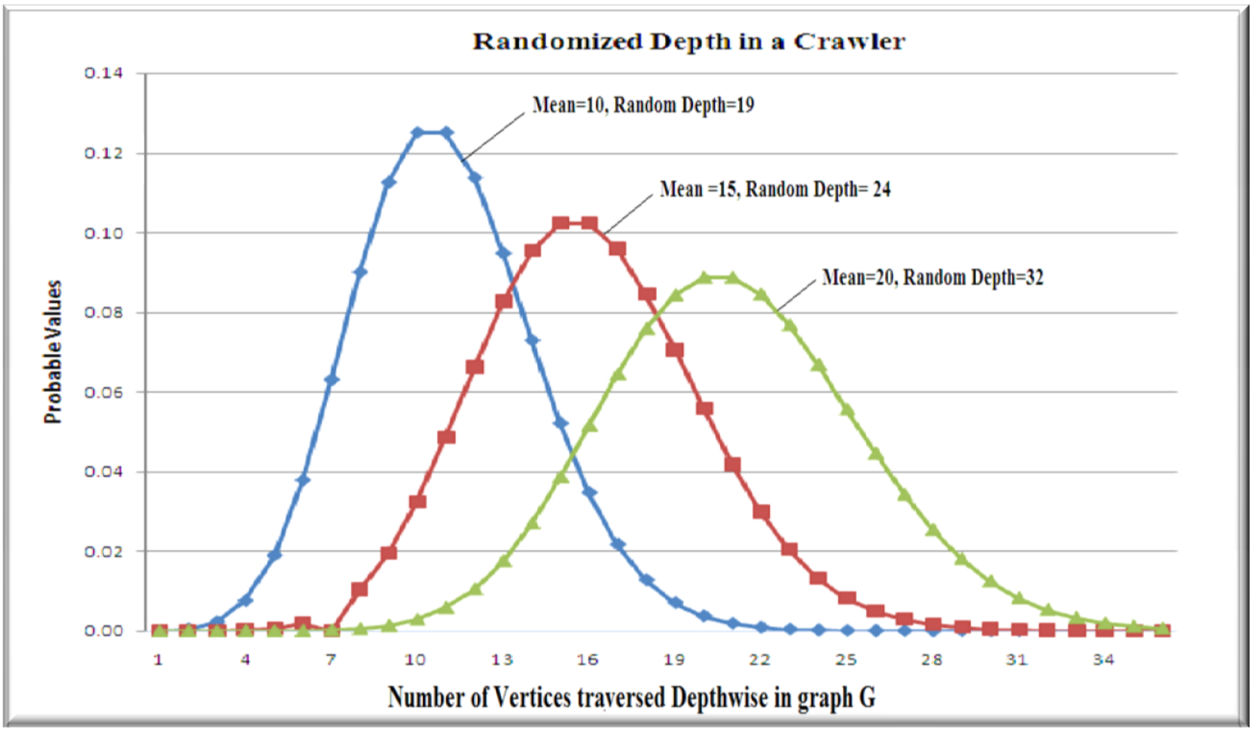

Figure 3: Generated Random depth for λ=10,15, and 20

5. CONCLUSION

New function enabled crawler traverses across unlimited breadth and random depth to fetch desired documents for further searching and extraction. Since the depth of the web is infinite and it is not possible for a user to determine depth in advance. Moreover, it takes several months of indefinite time to halt. For the experimental purpose, algorithm chosen a depth range from 0 to 35 and it is proved in Figure 3 showing generated random numbers as it is progressing which is much better than existing one. This crawler can also make dynamic generation of depth by providing interactive mean values to the crawler at runtime.

6. FUTURE WORK

Normally, general purpose crawler inserts more irrelevant pages as garbage in to database. Minimizing number of irrelevant and noisy pages while crawling is an interesting problem. Devising an intelligent crawler with a hybrid searching methods would be rewarding. Accessing hidden web pages and crawling deep web would be futuristic challenge in this area.

7. REFERENCES

[1] Beaza-Yates R and Castillo C, “Crawling the Infinite Web,” Journal of Web Engineering, vol. 6, no. 1, pp 49–72, 2007

[2] Ben Coppin, "Artificial Intelligence Illuminated", Jones and Barlett Publishers, 2004, Pg 77

[3] Brin S and Page L, ‘’The Anatomy of a Large-Scale Hyper textual Web Search Engine”, In Proceedings of 7th International World Wide Web Conference, April 14-18, 1998, Brisbane, Australia

[4] Broder A Kumar R, Maghoul F, Raghavan

P,Rajagopalan R, Stata A, Tomkins and Wiener J, "Graph Structure in the Web: Experiments and Models", In Proceedings of the Ninth Conference on World Wide Web, pages 309-320,Amsterdam, Netherlands, May 2000

[5] Burner M, " Crawling Towards Eternity - Building an Archive of the World Wide Web", Web Techniques, 2(5), May 1997

[6] Chakrabarti S,"Mining the Web", Morgan Kaufmann Publishers, 2003

[7] Cho J, Garc H,"Efficient Crawling through URL ordering", In Proceedings of the seventh conference on World Wide Web", Brisbane, Australia, April 1998

[8] Cho J, Shivakumar N, and Garcia-Molina H, " Finding Replicated Web Collections", In ACM SIGMOD, pages 355-366,1999

[9] Danushka Bollegala, Yutaka Matsuo, Mitsuru Ishizuka, "Automatic Discovery of Personal Name Aliases from the Web", IEEE Transactions On Knowledge and Data Engineering, 2011, pp 831-844

[10] Edward R W, Coffman Z Liu," Optimal Robot Scheduling for Web Search Engines", Journal of Scheduling, 1998

[11] Guha R and Garg, "Disambiguating people in search", Technical Report, Stanford University, 2004

[12] Junghoo Cho and Hector Garcia-Molina, "Effective Page Refresh Policies for Web Crawlers", ACM Transactions On Database Systems, 2003

[13] Lawrence S, Giles C L," Searching the World Wide Web Science", 1998

[14] Menczer, Filippo Gautam Pant and Padmini Srinivasan, "Topical Web Crawlers: Evaluating Adaptive Algorithms", ACM Transactions on Internet Technology (TOIT), vol. 4, no. 4, pp 378-419, 2004

[15] Miller G A,"WordNet: A Lexical Database for English," Communications of the ACM (Vol. 38, No. 11), 1995, pp 39-41

[16] Najork M and Wiener J L," Breadth-First Crawling Yields High-Quality Pages", In Proceedings of the Tenth Conference on World Wide Web, pp 114-118, Hong Kong, May 2001, Elsevier

[17] Narasingh Deo,"Graph theory with Applications to Engineering and Computer Science", PHI, 2004 Pg 301

[18] Pavalam S M, Jawahar M, Felix K Akorli, Kashmir raja S V," Web Crawler in Mobile Systems", In the proceedings of International Conference on Machine Learning (ICMLC 2011)

[19] Sandeep Sharma and Ravinder Kumar,"Web-Crawlers and Recent Crawling Approaches", In Proceedings of the International Conference on Challenges and Development (IT-ICCDIT-2008), PCTE, Ludhiana (Punjab), May 30th, 2008

[20] Steven S. Skiena, "The Algorithm Design Manual", Second Edition, Springer Verlag London Limited,2008 pg 162

[21] Tan P N and Kumar V," Discovery of Web Robots Session based on their Navigational Patterns", Data Mining and Knowledge discovery, 2002

[22] Thangaraj M and Sivagaminathan P G, "A Web Robot for Extracting Personal Name Aliases", International Journal of Applied Engineering Research, ISSN 0973-4562 Volume 10, Number 14 pp , 2015, pp 34954-34961

[23] Yang sun, Isaac G, Councill C, Lee Giles, " The Ethicality of Web Crawlers", 2010

[24] Breadth First Search, Accessed June 1, 2014, en.wikipedia.org/wiki/Breadth-First_Search

[25] https://en.wikipedia.org/wiki/Web_crawler

[26] http://www.slideshare.net/sanchitsaini/working-with-websphinx-web-crawler-9506067

[27] https://en.wikipedia.org/wiki/Poisson_distribution

[28] https://en.wikipedia.org/wiki/Probability_mass_function