Web Crawling Algorithms – A Comparative Study

Dr.S. Vijayarani[1], Ms. E Suganya[2]
[1, 2] Department of Computer Science
[1, 2] Bharathiar University,Coimbatore

Abstract- *The web contains a lot of information and it keeps on increasing every day. Thus, due to the availability of abundant data on the web, searching for some particular data in this collection has become very difficult. Emphasis is given to the relevance and robustness of data by the on-going researches. Even though only relevant pages are to be considered for any search query, but still huge data needs to be explored. Another important thing is to keep in mind is that usually one's needs may not be desirable for others. Hence, crawling algorithms are essential in selecting the pages that satisfy the user's need. The web crawler is an essential component of search engines, data mining and other internet applications. This paper reviews the researches on web crawling algorithms used for searching.*

Keywords- Web Crawling Algorithms, Breadth First Search, Depth First Search, Best First Search, Shark Search, Page Rank Algorithm, Online Page Importance Calculation Algorithm, HITS Algorithm

I. INTRODUCTION

A web crawler is a computer program that browses the World Wide Web in sequencing and automated manner. A crawler also referred as spider which can be used for accessing the web pages from the web server as per user pass queries commonly for search engine. A web crawler also used sitemap protocol for crawling web pages. Crawling the web is not a programming task, but an algorithm design and system design challenge because the web content is very large. The web crawling processes starts from a URL but the starting URL will not reach all the web pages [12]. The basic web crawling algorithms fetches a web page and parse it to extract all linked URLs and then extracted the relevant web pages. Again, it performs the same process until complete the task. The size of the web is large; web search engine is not possible to cover all the websites in World Wide Web [7]. There should be high chances of the relevant pages to be in the first few download, as the web crawler always downloads web pages in a fraction of a second.

The main objective of this work is to compare five algorithms such as Breadth First Search, Depth First Search, Best First Search, Shark Search, Page Rank Algorithm, Online Page Importance Calculation (OPIC) algorithm and HITS algorithms are used and this performance is compared using performance factor they are precision, recall, f-score and accuracy.

The remaining portion of the paper has five sections. Section 2 describes the methodology of this analysis work. Section 3 gives an overview of the web crawling algorithms used in this work. Section 4 discussed the results of the experiments. The conclusion is given in section 5.

II. METHODOLOGY

Figure 1 shows the system architecture of the proposed work.

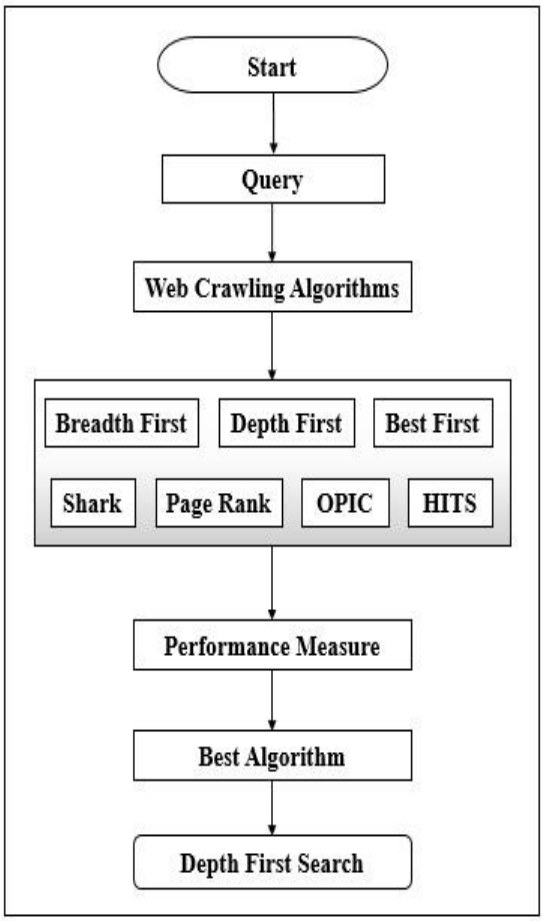

Figure1. System Architecture

Crawlers have bots that fetches new and recently changed websites, and indexes them [5] [16]. By this process billions of websites are crawled and indexed using algorithms depending on a number of factors. Several commercial search engines change the issues often to improve the search engines process. It starts with a set of URLs from the previous crawl, visits each of these websites, detects links and adds it to the list of links to crawl. It also notes whether there is any new website or website that has been recently updated, websites that are no more in use and accordingly index is updated. When a user initiates a search, the key words are extracted and searches the

index for the websites which are most relevant [10]. Relevancy is determined by a number of factors and also it differs for the different search engines. Different researchers used different strategies such as bread first, depth first, best first, shark, page rank, online page importance calculation, and HITS for selecting the websites to be downloaded.

III. WEB CRAWLING ALGORITHMS

A. Breadth First Search

Breadth First Search is an algorithm for traversing or searching tree or graph data structures. It works on a level by level, i.e. algorithm starts at the root URL and searches all the neighbors URL at the same level [7]. If the desired URL is found, then the search terminates. If it is not, then search proceeds down to the next level and repeat the processes until the goal is reached. It uses the boundary as a FIFO queue, crawling links in the order in which they are encountered [7]. The Breadth First Search algorithm is generally used where the objective lies in the depthless parts in a deeper tree.

The time complexity of breadth first search can be expressed as O (|V|+|E|), since every vertex and every edge will be explored in the worst case.

Where |V| is the number of vertices and |E| is the number of edges in the graph

```
BreadthFirst ( StartingUrls)
{
for (i=0;i<=StartingUrl;i++)
ENQUEUE(Boundary, url);
   do
   {
   url=Dequeue(Boundary);
   Page=Fetch(Url);
   Visited=Visited+1;
   Enqueue(Boundary,ExtractLinks(Page));
   }
while (Visited < MaxPages && Boundary != Null);
 }
```

B. Depth First Search

Depth First Search is an algorithm for traversing or searching tree or graph data structures [8]. It is a powerful technique of systematically traverse through the search by starting at the root node and traverse deeper through the child node. If there is more than one child, then priority is given to the left most child and traverse deep until no more child is available. It is backtracked to the next unvisited node and then continues in a similar manner [8]. This algorithm makes sure that all the edges are visited once. It is well suited for search problems, but when the branches are large then this algorithm takes might end up in an infinite loop [8].

```
Algorithm DFS(graph G, Vertex v)
//Recursive algorithm
for all edges e in G.incidentEdges(v)
do
if edge e is unexplored then
w = G.opposite(v, e)
if vertex w is unexplored then
label e as discovery edge
recursively call DFS(G, w)
else
label e as back edge
```

C. Best First Search

The A* search algorithm is an example of best-first search, as is B*. Best-first algorithms are often used for path finding in combinatorial search. Best First search is a search algorithm which travels a graph by expanding the most promising node chosen according to a specified rule. The basic idea is that given a boundary of URLs, the best URL according to some estimation criterion like precision, recall, accuracy and F-Score. In this algorithm, the URL selection process is guided by the lexical similarity between the topic's keywords and the source page of the URL [11]. Thus the similarity between a page p and the topic keywords is used to estimate the relevance of all the outgoing links of p.

```
BestFirst ( StartingUrls)
{
for (i=0;i<=StartingUrl;i++)
ENQUEUE(Boundary, url,MaxScore);
do
{
url=Dequeue(Boundary);
Page=Fetch(Url);
Score=GetTopicScore(Page);
Visited=Visited+1;
Enqueue(Boundary,ExtractLinks(Page),Sc
ore);
}
while (Visited < MaxPages && Boundary != Null);
}
```

D. Shark search

Shark-Search is a more aggressive version of Fish-Search. In Fish-Search, the crawlers search more extensively in areas of the web in which relevant pages have been found. At the same time, the algorithm discontinues searches in regions that do not return relevant pages. Shark-Search offers two main improvements over Fish-Search. It uses a continuous valued function for measuring relevance as opposed to the binary relevance function in Fish-Search [3]. In addition, Shark-Search has a more refined notion of probable scores for the links in the crawl boundary.

One immediate improvement is to use, instead of the binary (relevant/irrelevant) evaluation of document relevance similarity engine in order to evaluate the relevance of documents to a given query. Such an engine analyzes two documents dynamically and returns a "fuzzy" score, i.e., a score between 0 and 1. Here 0 for no similarity whatever, 1 for perfect "conceptual" match rather than a binary value. A straightforward method for building such an engine is to apply the usual vector space model [18]. The similarity algorithm can be seen as orthogonal to the fish-search algorithm. An engine is available and that for any pair query, document, (q, d), it returns a similarity score sim (q, d) between 0 and 1. Here q is represents a query and d represents document.

```
Shark (topic, startingUrls) {
Foreach link (startingUrls) {
Set_depth (link, d);
Enqueue(Boundary,links);
}
While (visted<MaxPages) {
Link=DequeueTopLink(Boundary);
Doc=fetch(Link);
DocScore=sim(topic,doc);
If (depth(link)>0) {
Foreach outlink (extractLink(doc)) {
```

```
Score = (1-r) * neighborhoodScore(outlink)
+ r * inheritedScore(outlink);
If DocScore > 0) {
setDepth (outlink, d); }
else {
setdepth (outlinkl, depth(link) – 1);
}
Enqueue (Boundary,outlink,score);
}
If (Boundary > MaxBuffer) {
dequeueBottomLink(Boundary);
}
}
}
}
```

E. Page Rank Algorithm

Page rank algorithm determines the importance of the web pages in any web site by counting citations or backlinks to a given page [14]. For example, if page P1 has a link to page P2, then, P2's content is probably appealing for P1's creator. The page rank of a provided web page is calculated as Relatedness between the webpages are taken into account by the Page Rank algorithm [1]. The web page whose number of

input link is high is considered of more importance relative to other web page, i.e. interest degree of the page to another [2]. When the number of input link is increased, then interest degree of a page also increases. Therefore, the total weighted sum of input links defines the page rank of a web page.

PR(A) = (1-d) + d(PR(T1)/C(T1) + ….. +PR(Tn)/C(Tn))

Where, PR (A) - Page Rank of a given Page
d - Dumping Factor
Ti - links

```
PageRank (Topic, StartingUrls[], frequency)
{
for (i=0;i<=StartingUrl;i++)
ENQUEUE(Frontier, Link);
do
{ if (multiplies(visited,frequency))
{ recomputed_scores_pr;
}
}
while (visited < MaxPages);
Link = DequeueTopLink(frontier);
Document=Fetch(Link);
```

F. Online Page Importance Calculation (OPIC) Algorithm

Online Page Importance Computation (OPIC) is to discover that importance of any page on web site, i.e. each page has a cash value that is allocated equally to all output links, initially all pages have the same cash equal to 1/n. This algorithm is similar to the Page Rank algorithm while it is done in one step. The crawler will download web pages with higher cashes in each stage and cash will be distributed between the pages it points when a page is downloaded. In this method, when this algorithm performs each page will be downloaded many times that will increase the web crawling time also.

```
OPIC(starting_url)
K = 100
enqueue(url_queue,starting_url)
while (not empty(url_queue))
url = dequeue(url_queue)
 crawl_page(url) for each child u of url
if (u 62 url_queue and (u) 62 crawled_pages)
enqueue(url_queue,u)
if (crawled_pages.count() %k == 0)
reorder_queue(url_queue)
End while
```

G. HITS

HITS (Hyperlink-Induced Topic Search) algorithm, retrieves a set of results for a search query and calculate the authority and hub score within that set of results. Counting the number of links to a page can give us a general estimate of its prominence on the Web, but a page with very few incoming links may also be prominent, if two of these links come from the home pages of sites like Yahoo!, Google, or MSN. Because these sites are of very high importance but are also search engines, a page can be ranked much higher than its actual relevance. The computation is performed only on this result set, not across all web pages. Authority and hub values are defined in terms of one another in a mutual recursion. An authority value is computed as the sum of the scaled hub values that point to that page. A hub value is the sum of the scaled authority values of the pages it points to. Some implementations also consider the relevance of the linked pages.

$$\text{auth}(p) = \sum_{i=1}^{n} hub(i)$$

Where n is the total number of pages connected to p and i is a page connected to p. That is, the Authority score of a page is the sum of all the Hub scores of pages that point to it.

$$\text{hub}(p) = \sum_{i=1}^{n} auth(i)$$

Where n is the total number of pages p connects to and i is a page which p connects to. Thus a page's Hub score is the sum of the Authority scores of all its linking pages

```
Hits ( Pages) {
G= SetofPages
For(p=0;p<G;p++)
{
p.auth = 1
p.hub = 1
HubsAndAuthorities(G){
for (i=0;i<k;i++)
for (p=0;p<G;p++)
for (q=0;q< p.incomingNeighbors; q++)
p.incomingNeighbors

p.auth += q.hub
for (p=0;p<G;p++)
for (page=0;PAGE<R;PAGE++)
p.hub += r.auth
} } }
```

IV. EXPERIMENTAL RESULT

In order to perform the experiments, the real time dataset is collected from different web pages. This dataset contains seven arts and science/college faculty information. It has 1240 instances and 8 attributes namely university name, department name, faculty name, qualification, designation, phone number, address, email id and URL. Table 1 and figure 2 shows how many number of pages visited by using web crawling algorithms. By the observation depth first search algorithm gives the better result than other algorithms.

Table1. Number of Relevant Pages Visited

Web Crawling Algorithms	No. of Pages Visited	No. of Relevant Pages Visited
Breadth First	100	32
Depth First	**100**	**39**
Best First	100	33
Shark Search	100	27
Page Rank	100	35
OPIC	100	32
HITS	100	34

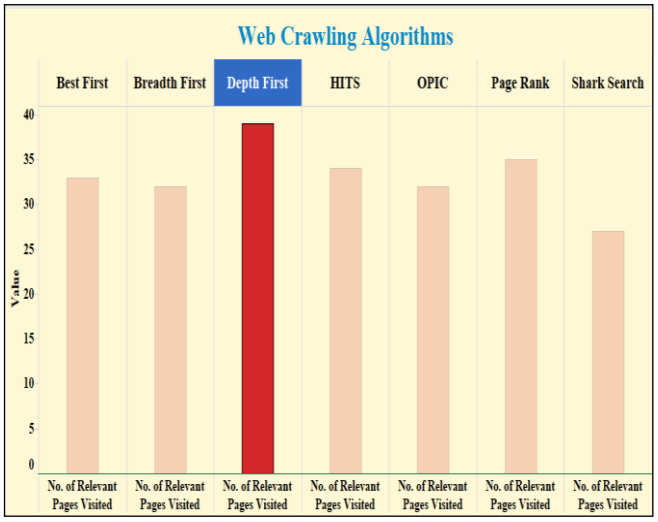

Figure2. Number of Relevant Pages Visited

Based on the data set experiments with the following performance measures like precision, recall, accuracy and F-Score are taken into account for their assessment. Precision is defined as the proportion of documents retrieved that are relevant to the user's query.

Precision (P) = TP/ (TP+FP)

Recall is defined as the ratio of relevant documents found in the search result to the total of all relevant documents.

Recall(R) = TP/TP+FN

F-Measure is a way of combining recall and precision scores into a single measure of performance.

F-Measure = 2*(P*R)/ P+R

Accuracy is defined as the proportion of true results of both TP and TN in the population.

Accuracy= (TP+TN)/ (TP+FP+FN+TN)

Table 2 and figure 3 shows the accuracy measure for web crawling algorithms such as breadth first search, depth first search, best first search, shark search and page rank algorithm.

Table2. Accuracy measure for Web Crawling Algorithms

Web Crawling Algorithms	Precision	Recall	F-Measure	Accuracy (%)
Breadth First	28.8	40.4	33.61	91
Depth First	**34.2**	**46.6**	**37.64**	**96**
Best First	30.3	42.23	32.66	92
Shark Search	27.05	41.13	35.6	89
Page Rank	31.45	43.09	34.78	94
OPIC	32.51	41.12	34.62	93
HITS	31.8	43.6	33.52	92

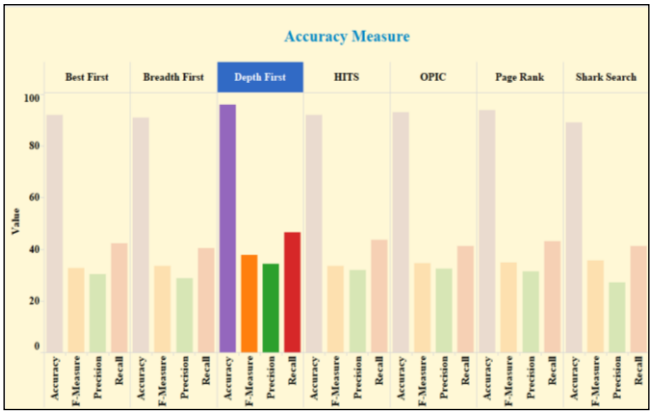

Figure3. Accuracy Measure for Web Crawling Algorithms

From this it is observed that depth first search algorithm gives the better results than other web crawling algorithms.

V. CONCLUSION

In this paper, an attempt has been made to improve the efficiency of the Web Crawler for improving the web searching with the help of comparing certain features of several algorithms such as breadth first, depth first search, best-first, shark search, page rank, online page importance calculation (OPIC) and HITS. For this, various performance parameters such as precision, recall, F- score and accuracy are taken into consideration. Based on the output parameters, it is observed that depth first search algorithm outperforms over other algorithms by various performance measures. This result leads to the conclusion that the depth first search algorithm improves the performance of web crawler for quick information retrieval.

REFERENCES

[1] Sonali Gupta, Komal Kumar Bhatia, "A Comparative Study of Hidden Web Crawlers", International Journal of Computer Trends and Technology (IJCTT), volume 12. Number 3, June 2014.

[2] Sushil Kumar, Dr. Anuj Kumar, "A Comparative Study of Scheduling Algorithms for Web Crawling Using Vb.Net Technology" International Journal of Research in Science And Technology, (IJRST) 2011, Vol. No. 1, Issue No. II, July-Sept ISSN: 2249-0604.

[3] S. Jaiganesh , P. Babu, K. NimmatiSatheesh, "A Comparative Study Of Various Web Search Algorithms For The Improvement Of Web Crawler", International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 4, April – 2013, ISSN: 2278-0181.

[4] Apoorv Vikram Singh , Vikas , Achyut Mishra, "A Review of Web Crawler Algorithms", International Journal of Computer Science and Information Technologies, Vol. 5 (5) , 2014

[5] Pavalam S M, S V Kashmir Raja, Felix K Akorli and Jawahar M, "A Survey of Web Crawler Algorithms", International Journal of Computer Science Issues, Vol. 8, Issue 6, No 1, November 2011, ISSN (Online): 1694-0814.

[6] Rashmi Janbandhu, Prashant Dahiwale, M.M.Raghuwanshi, "Analysis of Web Crawling Algorithms", International Journal on Recent and Innovation Trends in Computing and Communication, Volume: 2 Issue: 3. ISSN: 2321-8169, 488 – 492.

[7] Rahul kumar, Anurag Jain and Chetan Agrawal, "Survey of Web Crawling Algorithms", Advances in Vision Computing: An International Journal (AVC) Vol.1, No.2/3, September 2014.

[8] Dr. Ammar Falih Mahdi, Rana Khudhair Abbas Ahmed, "A New Technique for Web Crawling In Multimedia Web Sites", International Journal of Computational Engineering Research, Vol 04, Issue2.

[9] Dr.P.Jaganathan, S.Jaiganesh, P.Babu, "Improvement of performance of web crawlers for efficient web searching and crawling", International Journal of Computer Science & Engineering Technology (IJCSET).

[10] Aggarwal, C., Al-Garawi, F., and Yu, P. 2001. Intelligent crawling on the World Wide Web with arbitrary predicates. In Proceedings of the 10th International World Wide Web Conference. 96–105.

[11] Brin, S., Page, L., Motwami, R., Winograd, T. "The pagerank citation ranking: bringing order to the web". Technical report, Standford digital library technologies project, Stanford University, Stanford, CA, USA, 1998

[12] Chakrabarti, S., Van Den Berg, M., and Dom, B. 1999. "Focused crawling: A new approach to topic specific Web resource discovery". Comput. Netw. 31, 11–16, 1623–1640.

[13] Cho, J.,Garcia-Molina, H., and Page, L. 1998. "Efficient crawling through URL ordering". Comput. Netw. 30, 1–7, 161–172.

[14] Filippo Menczer , Indiana University ,Gautam Pant, University of Utah and Padmini Srinivasan, University of Iowa , November 2004. "Topical Web Crawlers:

Evaluating Adaptive Algorithms". ACM Transactions on Internet Technology, Vol. 4, No. 4

[15] Marc Nojark, "Web Crawler Architecture" retrieved from

[16] http://research.microsoft.com/pubs/102936/EDSWebCrawlerArchitecture.pdf

[17] Carlos Castillo, Mauricio Marin, Andrea Rodriguez, "Scheduling Algorithms for Web Crawling in the proceedings of Web Media and LA-Web", 2004

[18] Cho, J.,Garcia-Molina, H., and Page, L. 1998. Efficient crawling through URL ordering. Comput. Netw. 30, 1–7, 161–172

[19] G. Salton and M.J. McGill, "Introduction to Modern Information Retrieval". Computer Series. McGraw-Hill, New York, 1983.