ISSN 2278-3091
Volume 8, No.2, March - April 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse04822019.pdf
https://doi.org/10.30534/ijatcse/2019/04822019

An Improved Crawler Based on Efficient Ranking Algorithm

Jyoti Mor[1], Naresh Kumar[2], Dinesh Rai[3]
[1]Ph.D. Research Scholar, School of Engineering & Technology, Ansal University, Gurugram, India
[2]Associate Professor, Department of Computer Science & Engineering, MSIT, Janakpuri, New Delhi, India
[3]Associate Professor, School of Engineering & Technology, Ansal University, Gurugram, India

ABSTRACT

With the increase in number of pages being published every day, there is a need to design an efficient crawler mechanism which can result in appropriate and efficient search results for every query. Everyday people face the problem of inappropriate or incorrect answer among search results. So, there is strong need of enhance methods to provide precise search results for the user in acceptable time frame. So this paper proposes an effective approach of building a crawler considering factors of URL ranking, load on the network and number of pages retrieved. The main focus of the paper is on designing of a crawler to improve the effective ranking of URLs using a focused crawler.

Key words: Page Rank, WWW, Web Crawler, Indexer, Scheduler, URL Extractor.

1. INTRODUCTION

The WWW contains enormous amount of information in terms of web pages (WP) and this information is growing exponentially day by day. These WP are searched with the help of Search Engine (SE). SE have automated programs called crawlers (Web spider or Web Robot), that downloads the pages automatically from WWW. Due to the growing size, crawlers are also downloading irrelevant data in large amount making them inefficient. In Figure 1 the crawler downloads the URLs from WWW, store it in URL Queue. After elimination of duplicate URL it stores the filtered URL into new URL Queue. The filtered queue is indexed by Indexer and forwarded to the repository for future use. Then ordering of URLs has been done and forwarded to the crawler to crawl.

Types of Crawlers: - Different types of crawlers are as follows-

- Incremental Web Crawler [1]- This type of crawler updates only those downloaded pages that are modified instead of crawling the entire web again from the scratch. This improves efficiency of the crawler in terms of storage.
- Parallel Crawler [1]-This type of crawler consists of number of crawlers working parallelly but on same network. This improves efficiency of the crawler in terms of speed.
- Distributed Crawler [1] - This type of crawler also consists of a number of crawlers working parallelly and on network of workstations independently. This improves efficiency of the crawler in terms of processing speed and if one of the crawlers is failed then it didn't impact other crawlers.

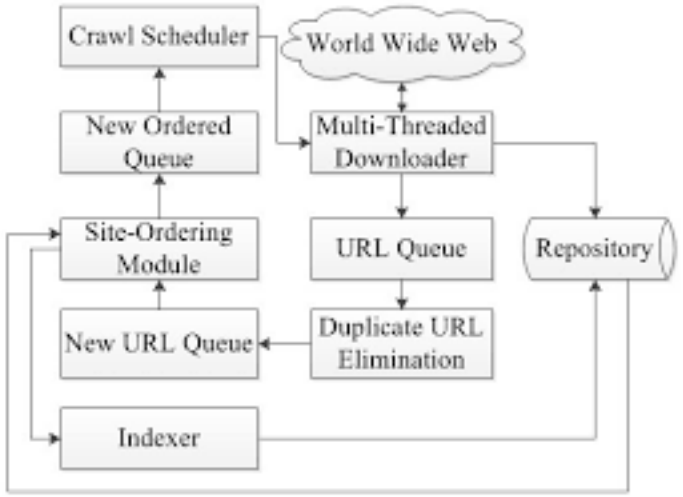

Figure 1: Architecture of the crawler.

- Focused Crawler [1]- This type of crawler retrieves, download, indexes and manages WP for a particular set of topics which define a limited section of web.
- Breadth First Crawler- This type of crawler initiates the crawling from a set of URLs then it uses breadth first technique [2] to further crawl the rest of the URLs.
- Mobile Crawler [3]- This type of crawler is sent to remote sites to filter out the unwanted data locally and sending only relevant data to the SE end. This reduces load on network.
- Ontology based Crawler [3]- This type of crawler work as per the ontology i.e. crawling of pages related to the topic.

2. RELATED WORK

Ordering of URLs only on the basis of keyword doesn't fetch

good results, so authors of [4] proposed the method to improve crawling by using content and structure similarity score. They developed the algorithm for the site ordering module and used SimRank [5] algorithm for calculating the structural and content similarity for the keyword. The proposed work has been tested by using the keyword "student" for similar URLs of Educational Websites and set the crawl limit from 5 to 10. The results were compared on the basis of crawling time, ordering time, precision and similarity score which helped in to increase the number of relevant pages. The main problem faced by authors was effective identification of topic relevant pages and priority of downloading. To overcome these issues authors of [6] implement a Focused Crawler that identified the topical content of WPs before processing and decided the order of priority. The Passage extraction algorithm along with lexical chaining approach had been used by the authors which improved crawler effectiveness in performing topical crawls. It also tested accuracy in building priority cues that are topic specific and resources affordable.

The calculated importance of each page is temporary and detecting the authoritative recently uploaded web pages was an issue with link-based metrics. So, to resolve this, authors of [7] proposed an architecture for a focused-based parallel Web crawler in which combination of click-stream analysis and text analysis were used to prioritize the crawling frontier. They used click-stream based prioritizing algorithm with which importance of a web page was checked by calculating the no of clicks and amount of time spent by the user on a WP. So, the proposed architecture of crawler helped authors to calculate the relevancy of a WP besides it also resulted in minimizing the overhead communication between parallel agents.

It was observed that Web searchers are unable to find useful results in the top listed URLs. So, the author of [8] developed a method to Identify WP for particular domain and discard pages that are not related to the domain of interest. Authors used relevance calculation algorithm to calculate the relevance score of a WP. For the testing of proposed work authors used educational URLs as input, assigned weights to ontological terms related to college and calculate the relevancy. The result shows web pages having relevance limit greater than four are downloaded as domain specific pages. Model supports multiple domains by using multiple ontologies hence resulting in faster search.

Authors of [9] faces the problem of how to update the pages of websites which undergo changes on regular basis, so authors proposed the freshness checker mechanism method to tackle with this. Authors takes set of URLs as input and process it through various metrics (like structure checker, image checker and content checker) and check whether the page is updated or not. Results proved that out of one hundred samples only twelve samples had been found changed and need updation.

As the web is growing, problem of web or network traffic also come in existence so, the authors of [10] proposed Last_visit and HTTP Get request header method to reduce web traffic. They give various pages of a general Website as input and downloaded relevant pages which needs to be updated as output and hence reduces the network traffic. Results show that this algorithm is 2.6% better than normal web crawling.

Authors of [11] explain various design issues of WC like "how to get relevant pages of a search query", "how to refresh pages" and "how should Crawler get time sensitive information. The highlights of the paper were the Last-modified header to check the freshness and Chronica's General Search interface to get time sensitive information. Results showed the URL with number of hits counts and past result of the search query efficiently.

Due to regular growth of WWW ordering of URLs becomes a problem. For this, authors of [12] proposed learning and non-learning methods to solve this issue. They take two datasets DS1 and DS2 (Web logs of Information and Telecommunication Technology Centre website) and DS2(Web logs of CiteSeer website) and then calculated an accuracy of 64.3% and 44.2% with dataset DS1 and dataset DS2 respectively. Results proved that learning method is better than non-learning method as accuracy of non-learning methods is less as compared to learning method.

Determining the content relevancy of WP, page popularity among the servers and time frequency of updation of pages caused the authors of [13] to propose a method for it. So, they used URL Ordering mechanism considering the mining of web contents, usage of pages and structure of the WP. The proposed method was tested on a data set of top 100 URLs returned from the web site (www.amu.ac.in) and counted the uniqueness of URLs. It resulted in reducing the number of duplicate URLs visits.

Maximizing the number of relevant pages and minimize the irrelevant ones from loading on query led the author of [14] to propose a method to prioritize the URLs in URL Queue. The

Author used the domain dependent Ontology with 2 cycles: Ontological Cycle and Internet Cycle. The results of the two cycles are mixed to determine the relevancy of pages. The proposed mechanism was tested on a group of five URLs of educational web sites. The results showed that implemented ontological - based focused crawler has the maximum harvest rate of 48% as compared to standard crawler.

Qualitative information Retrieval from the large amount of data is difficult for the crawlers. So, the authors of [15] proposed Query Based Crawler where set of keywords to shoot search interface (like radio button, check boxes, etc.) were used to generate URL. If the search interface not found then they used Google Search API to generate results. The proposed crawler was implemented in python programming language. They also utilized libraries like Beautiful Soup, Selenium Client API and Web Driver, Google Search API for implementation. The proposed work was tested to find the Indian origin academicians which are working outside India. They took Indian names as keywords and run on twenty-five foreign universities websites to find the Indian academicians. The results obtained showed that URL with highest fitness value had been preferred first over others.

Different types of crawler aren't able to filter out the irrelevant URLs, so authors of [16] proposed the analysis of different crawling technologies according to the specific need of user to tackle the enormous growth of web. The authors described CATCH crawler, Internet Forum Crawler, Wrapper Model of crawler according to the different need of users. They also stated the challenges that are encountered in crawling using the crawling techniques such as Page Update Policy, Obsolete Algorithms, Loss of Data during compression or some crawlers that are limited to specific domain. Authors also specified some feasible solutions that if included in the existing crawlers will improve the specific domain crawlers.

The Authors of [17] concluded that comparing the text / keyword for a direct match of a query in the webpage does not provide best results in terms of relevancy. They proposed a method to categorize the WPs based on HTML tags, images, hyperlinks, anchor text and the result is sent to classifiers. The classifiers classify the WP by selecting the conceptual knowledge of the keywords and classifying them into different categories. The experiment was carried out on four different categories: Cricket, Hockey, Football and Basketball. The complete data was divided into 70c/o training set and 30c/o testing set. The 90c/o of the WPs were classified accurately with respect to TF-IDF approach.

3. CHALLENGES

Some of the challenges available with existing Focused Crawler are explained below:

- The authors of [4] didn't propose any methodology to tackle the problem of frequently modified web pages.
- The authors of [6] only considered 10% of WP and did not considered the rest 90% of WP for matching the pattern of a WP. Moreover they also considered small amount of training set for matching of pattern of a WP to be downloaded.
- Authors of [7] did not covered the content of authorized pages i.e. the pages from semantic web only were covered by the crawler and crawler was not able to cover the authorized pages present on dark net or dark web.
- No method was proposed by author of [8] to calculate the value relevance limit and tolerance limit which were considered in the data set in comparison with the calculated value.
- The threshold value between an updated WP from a non-updated WP was not elaborated in [9].
- Forcefully updated the Last_modification date of the WP while loading the results of query and also did not define any threshold after which crawled pages needs to be refreshed [10].
- Authors of [12] did not suggest any method to order the URLs shared through RSS feeds, text messages or shortened URLs.
- The authors of [13] did not work on improving the ordering of newly updated pages in URL queue.
- While maximizing the relevant pages in search results, author of [14] didn't consider the speed of crawling.
- The Authors of [17] does not test their proposed method on multi-leveled WP. HTML pages can be in unstructured format [18]. HTML page structure may fails to detect changes in CSS documents [19].

4. PROPOSED ARCHITECTURE

Figure 2 shows the proposed architecture of Parallel Focused Crawler. It's major components are - Controller, URL Extractor, Page Score Calculator, Page Score Comparator and Scheduler. The detail working of the proposed crawler is as explained below:-

1. User enters the keyword into search keyword column.
2. The correspondingly successive hyperlinks have been fetched.
3. All the URLs fetched are categorized with the help of URL extractor in three domains that are .com, .in and .org domain.

4. Content Fetcher module (Available in Figure 3) helps in fetching the content of every URL obtained in the list.
5. Determining the occurrence of the input keyword in every URL of the list is done by Occurrence count module and then page score is calculated for a particular URL.
6. Controller manages all the calling and handling of functions.
7. Page score of every URL is compared with each other with the help of Page Score Comparator which describe that which URL should be given the most importance.
8. All the crawled data is collected in database for faster retrieval in future.
9. Scheduler keeps the track of updation of URLs and divide the URLs in three categories of frequently (updated every hour), frequent (updated in 24 hours), static (updated in 15-30 days).
10. Pages are now sorted according to page score in terms of relevancy.

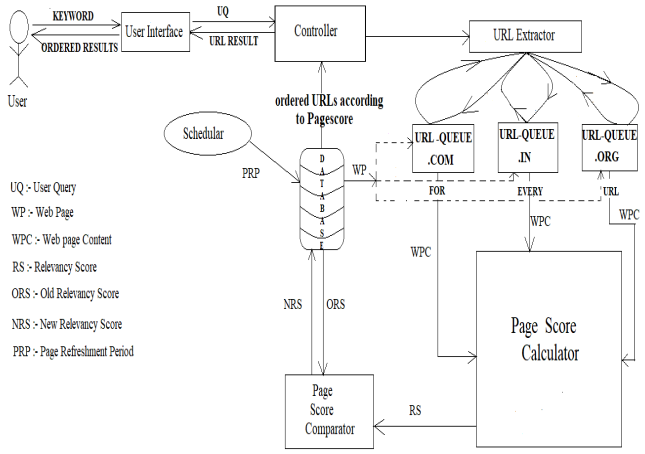

Figure 2: Architecture of Focused Web Crawler

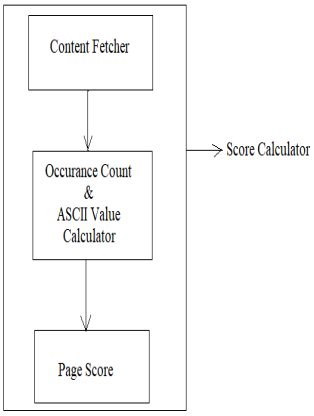

Figure 3: Internal Structural of Page Score Calculator

The step by step working is also described with the help of an algorithm as explained below:

Algorithm: - Page ranking through page score
Input:-Query
Output: - Creation of the Database
Step 1: - Start
Step 2: -Query fired by the User to the interface
Step 3: - Now the crawler crawl for the URLs from the WWW.
Step 4: - After finding the URLs, Crawler read the URLs from the crawl set.
Step 5: - Distribute the URLs according to its domain and saves its time and date of Visiting.
Step 6: - Now, compare the page score of URLs from previous page score
Page Score = d*Kc+ K*tl +m*(ac/wc)

If change in page score is greater than 10%, then update the page score and save it.
If change in page score is less than 10%, then do not crawl and do not save it.
Step 7: - Rank or order the URLs according to the Page Score in Ascending or Descending.
Step 8: - Save the updated page score for the respective URLs in Database.
Step 9: - End.

5. EXPERIMENTAL SETUP AND RESULT DISCUSSION

The proposed architecture has been implemented in Java Language Microsoft SQL Server 2017 was used for Database with IDE eclipse. The experimental setup with the changes in existing system can be described as follows:

(i) In existing crawler some overhead modules were required for extraction of WPs which are not available here. Here we removed the extra module for extraction of WP into database by retrieving only the required parameters for page score calculation. This reduced the overhead of extra module, thereby improving the process speed.

(ii) The WPs have been divided into three categories i.e. Frequent, Static and frequently updated WPs. A Frequent WP is a WP which gets updated within a week. It is not updated daily. Static WPs are those which is hardly updated. Frequently updated WPs are those which is updated every now and then. The size of Frequently WP taken into experiment is 147kb and 54kb. The size of Frequent and Static WPs are 82kb links and 95kb links respectively.

(iii) Number of days of observation
The experiment was carried out for 15 days in order to observe the changes into a WP. For a Frequent website, noticeable

changes were visible. In Frequently updated WP, major changes were noticed whereas in static WPs, no changes were seen.

(iv) Average Number of pages retrieved by taking two seed URLs for Frequently updated WP, one seed URL for Frequent WP and 1 for Static WPs. In frequently updated WP, 147 and 54 WP were retrieved. In Frequent webpage, 82 WP were retrieved and in static WP category, 95 WP were retrieved. On an average, 95 URLs were retrieved for every seed URL. The same is shown in Figure 4 and Figure 5 respectively.

FREQUENTLY Updated URLs

Size of Dataset: 147 links
Query keyword: Cricket
FREQUENTLY Updated URLs
Parent URL:http://www.espncricinfo.com/scores

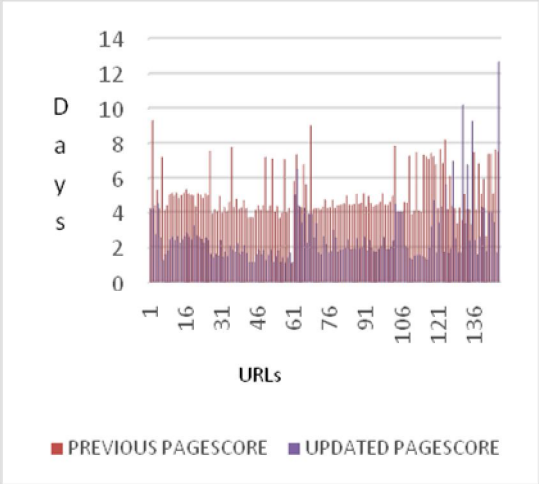

Figure 4: Graph for Frequently Updated URLs

FREQUENT Updated URLs

Size of Dataset: 82 links
Query keyword: Brexit
Parent URL:https://www.bbc.com/news

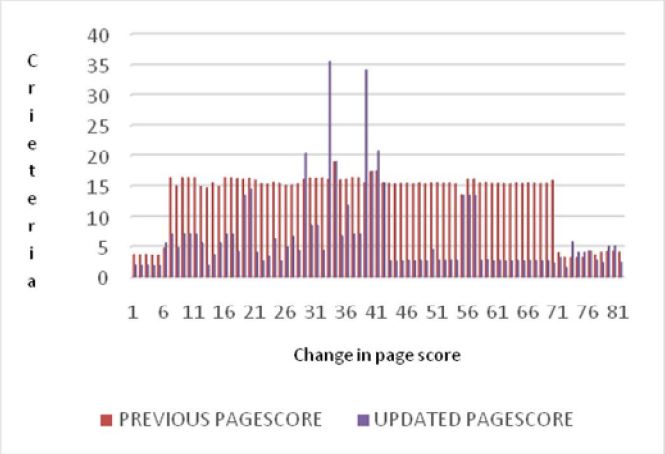

Figure 5: Graph for Frequent updated URLs

(v) Webpage change behavior:- The ASCII count, keyword occurrence and other factors effecting the page score were observed to be changing majorly in Frequently updated WP, followed by Frequent WP and minimum changes in Static URL. This is graphically shown in Figure 6.

(vi) Load on network:- In order to reduce the load on network, the work was divided into manageable and synchronous steps one after the other to reduce the load on network. Instead of retrieving the complete WP from the server, we retrieved only the count of parameters essential for the calculation of page score. This in turn reduced the load on network significantly.

STATIC URLs

Size of Dataset: 258 links
Query keyword: android
Parent URL: https://www.android.com/

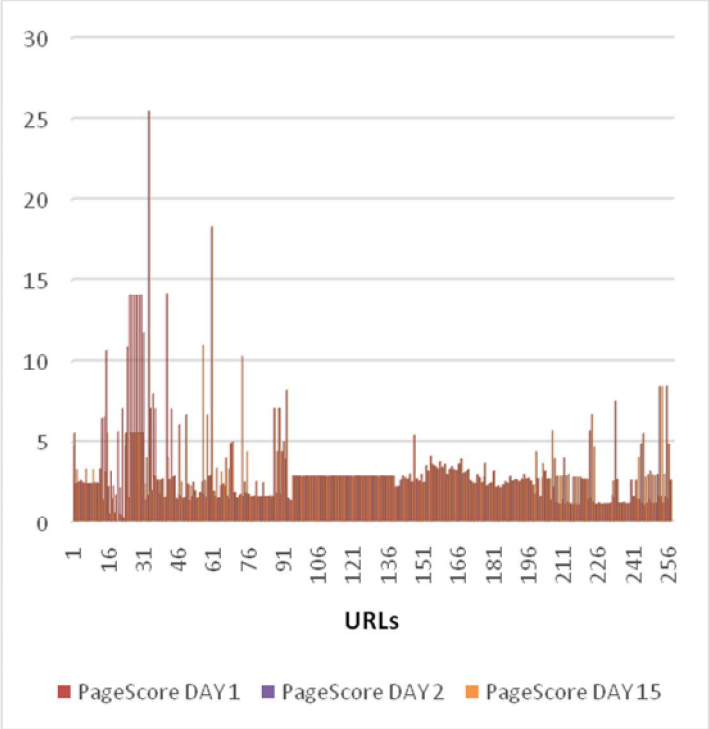

Figure 6: Graph for Static URLs

6. COMPARISON WITH EXISTING SYSTEMS

The Table 1 compares and contrast different crawlers where authors have shown that proposed and implemented crawler is superior than the existing crawlers. The parameters for comparison are Frequency of Revisit (Freshness), Speed of Crawling Categorization of updated and non-updated pages, Ranking through page score, Efficiency, Multi-threaded and Optimized Bandwidth Utilization.

7. CONCLUSION

The proposed & implemented crawler separates the URLs based on their frequency of page updation. Authors categorized URLs as: Static, Frequent and frequently updated URLs. A Frequently Updated URL is a URL which is updated every now and then. It most consists of sites like sports news in which score is updated every time and demonstrate how the implemented crawler responds to such URLs. The datasets are New Cricket dataset and World News dataset. Some cases were observed with no change in its page score on the revisit of crawler. For the Static URL, the dataset named android dataset with the parent URL as https://www.android.com/ had been chosen. The page score is calculated for every child URL and their respective ranking is obtained.

A Frequent Updated URL is a URL which is updated at regular interval ranging from a day to a week. It was tested on query BBC News with its parent URL as https://www.bbc.com/news. The Queried Keyword for its child URLs is Brexit. At the end the results were compared with some existing crawler which proves that proposed crawler have reduce the load on the network and rank the pages efficiently.

Table 1: Comparison between different crawlers

Name	Freshness	Speed of Crawling	Categorization of updated and non-updated pages	Ranking through Page score	Efficiency	Multithreading	Optimum Bandwidth
Incremental Web Crawler	Yes	Yes, since crawls only modified pages	Yes	No	Yes, in terms of storage	No	Yes, but it is missing the crawling of unmodified pages
Parallel Web Crawler	No	Yes	No	No	Yes, in terms of speed	Yes	No, due to redundant crawling
Distributed Crawler	No	Yes	No	No	Yes, in terms of processing speed	Yes	No, due to redundant crawling
Breadth First Crawler	No	Yes	No	No	Yes	No	Yes, never crawls unwanted URLs
Mobile Crawler	No	Yes	Yes	No	Yes	No	Yes, never crawls unwanted URLs
Ontology based Crawler	No	No, since it matches every word in WP with the ontological word of keyword	No	No	Yes	No	Yes, never crawls unwanted URLs
Proposed Focused Web Crawler	Yes	Yes	Yes	Yes	Yes	Yes	Yes

REFERENCES

[1] M. S. Ahuja, J. S. Bal and Varnica, “Web Crawler: Extracting the Web Data”, International Journal of Computer Trends and Technology (IJCTT), ISSN: 2231-2803, Volume 13, number 3, pp 132-136, 2014.
https://doi.org/10.14445/22312803/IJCTT-V13P128

[2] V. Shkapenyuk and T. Suel, “Design and Implementation of a high performance distributed web crawler”, In Proc. 18th International Conference on Data Engineering, doi: 10.1109/ICDE.2002.994750, pp. 357–368, 2002.

[3] R. Nath and K. Chopra, "Web Crawlers: Taxonomy, Issues & Challenges", International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume 3, Issue 4, pp 944-948, April 2013

[4] M. Shoaib and A. K. Maurya, “URL Ordering based Performance Evaluation of Web Crawler”, IEEE International Conference on Advances in Engineering &

Technology Research (ICAETR – 2014), ISSN: 2347-9337, 2014.
https://doi.org/10.1109/ICAETR.2014.7012962
[5] G. Jeh, and J. Widom, "SimRank: a measure of structural-context similarity", Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data mining, doi:10.1145/775047.775126, pp. 538-543, 2002.
[6] E. K. L. Kapetanios, V. Sugumaran and M. Spiliopoulou, "An Ontology-Based Focused Crawler." In Natural Language and Information Systems, ISBN:978-3-540-69857-9, Volume 5039, 2008.
[7] F. Ahmadi, Abkenariandand A. Selamat, "Parallel Web Crawler Architecture for Clickstream Analysis", Communications in Computer and Information Science, Springer, Berlin, Heidelberg, ISBN:978-3-642-32825-1, Volume 295, 2011
[8] S. Sinha, R. Dattagupta and D. Mukhopadhyay, "A New Approach to Design a Domain Specific Web Search Crawler Using Multilevel Domain Classifier", In: Distributed Computing and Internet Technology. ICDCIT 2013, ISBN:978-3-642-36070-1, Volume 7753, 2013
[9] N. Kumar, D. Tyagi, S. Awasthi and J. Mor, "Change Detection Of Webpage In Focused Crawling System", International Journal of Computer Technology and Applications (IJCTT), ISSN: 0974-5572, pp 969-976,2016
[10] S. S. Vishwakarma, A. Jain and A. K. Sachan, "A Novel Web Crawler Algorithm on Query based Approach with Increases Efficiency", International Journal of computer Applications, ISSN:0975-8887, Volume 46, number 1, pp 34-37,2012
[11] Deepika and A. Dixit, "Web Crawler Design Issues", International Journal of Management, IT and Engineering (IJMEI), ISSN: 2249-0558, volume 2, Issue 8, pp 394-404, 2012
[12] A. Changramouli, S. Gauchand and J. Eno, "A Popularity-based URL Ordering Algorithm for crawlers", 3rd International Conference on Human System Interaction, ISSN: 2158-2246, pp 556-562, 2010
https://doi.org/10.1109/HSI.2010.5514512
[13] Sandhya, M. Q. Rafiq and O. Farooq, "Efficient Web Crawling With Proposed URL Ordering", 2011 International Conference on Multimedia, Signal Processing and Communication Technologies, ISBN: 978-1-4577-1107-7, pp 44-47, 2011.
https://doi.org/10.1109/MSPCT.2011.6150516
[14] D. Koundal, "Prioritizing the ordering of URL queue in focused crawler", Journal of AI and Data Mining (JAIDM), URL: http://jad.shahroodut.ac.ir/article_146_0.html, Volume 2, Number 1, pp 25-31, 2014.
[15] M. Kumar, A. Bindal, R. Gautam and R. Bhatia, "Keyword Query Based Focused Web Crawler", 6th International Conference on Smart Computing and Communications (ICSCC), DOI - https://doi.org/10.1016/j.procs.2017.12.075, Volume 125, pp 584-590, December-2017.
[16] N. Kumar, S. Awasthi and D. Tyagi, "Web Crawler Challenges and Their Solutions", International Journal of Scientific & Engineering Research, ISSN 2229-5518, Volume 7, Issue 12, pp 95-99, December-2016
[17] A. Qazia and R. H. Goudarb, "An Ontology-based Term Weighting Technique for Web Document Categorization", International Conference on Robotics and Smart Manufacturing (RoSMa2018), DOI-https://doi.org/10.1016/j.procs.2018.07.010, pp 75-81, 2018.
[18] I.S. Makki and F. Alqurashi, "An Adaptive Model for Knowledge Mining in Databases "EMO_MINE" for Tweets Emotions Classification", ISSN 2271-3091, Volume 7, No.3, May- June 2018.
https://doi.org/10.30534/ijatcse/2018/04732018
[19] N. Kumar, D. Tyagi, S. Awasthi and J. Mor, " Change Detection of Web Page in Focused Crawling System", International Journal of Control Theory and Applications, ISSN: 0974-5572, 9(41) pp. 969-976, 2016.