Design and Implementation of Crawler based on Query Parameter and Categorization

Kavita Goel[1], Dr. Jay Shankar Prasad[2], Dr. Saba Hilal[3]
[1]Research Scholar, Dept of Computer Science, MVN University, Palwal, India
[2]Dept of Computer Science & Engg., MVN University, Palwal, India
[3]http://sabahilal.blogspot.in/

***Abstract* -** Web is the ocean of information and it is common to get knowledge from the web. Finding useful and relevant information from the web is always challenging. A User gives query through a search engine which uses a crawler to fetch desired results. Crawler creates the database for the search engine by extracting links from the web. Finding useful information from the web have challenges of page refreshment, aging problem, crawling multimedia content, and duplicate links. Removing duplicate URL is important as different URL with same content increases crawling time and space to store links in the database. These are some hurdles in getting desired and relevant results from the web crawler. This work studies previous approaches of designing and optimizing crawler's performance and designs new crawler based on the idea of removing duplicate URLs in crawling. The concept is implemented using URL normalization and categorization. Removing duplicate URLs during crawling will improve crawler's performance and hence whole searching process.

***Keywords* -** *Web crawler, URL normalization , categorization.*

I. INTRODUCTION

In today's era of Information, everyone is desired to get information. The requirement of information differs from person to person but the source of information is common to all i.e. World Wide Web. It is the deep sea of information which contains information from all the domains but the information is spread across the sea. In order to get the preferred information some tools are required and Search engine is the most commonly used tool. It is an interface used by the end user to get the desired result. Search engine uses software called crawler or bot to get particular information from web. Crawler [7] crawl the web in order to give required results to search engine which in turn are served to end users. End-user demands that results are relevant and it should not contain duplicate URLs. A large number of URLs on the web are duplicates and it affects all functions of search engine including crawling, indexing and serving [6]. De-duping of URL is a big challenge [6]. Although many researchers have work in this field and results are better. But only few researchers have work in the area of removing duplicity at crawling level. In [1], multiple alignment sequence algorithm with URL normalization in order to optimize the links has used. In [3], a method based on decision tree and machine algorithm to generate rules which in turn are used with URL normalization for deduplication of web page is used. It defines general set of rules for tokenization of URLs which are used to extract tokens for normalization of URL. In [2], a new algorithm for Naïve Baise Classifier which is used for classification of web crawler is merged with URL scoring algorithm to improve the feat of system. In [5], certain rules are defined to mine DUST from crawl logs without checking content of web page. In [6],a framework is designed to observe the common patterns of URLs on web. These rewrite rules remove duplicate URLs which will occur first time for crawling without retrieving the content.

This paper design crawler based on URL normalization and categorization to optimize crawler's performance and search results.

The paper has been organized as follows. Section two Discuss previous works to Design and optimize crawler's performance. Section three discuss Design and Flowchart of New crawler. Section Four gives the implementation of crawler with an illustration. Section five concludes.

II. RELATED WORKS

Various soft computing approaches have been proposed to optimize the focused crawler. In [8] Banu Wirawan Yohanes et al. proposes genetic algorithm to improve the performance of focused web crawler. The Genetic algorithm works in four steps. The first step is initialization in which various parameters of the genetic algorithm are initialized. Seed set is given to the Crawler; it fetches the page from URL and stores it. In the second phase, Jaccard similarity is used to measure the fitness of a web page. In the third phase, out links of the web page are extracted and most suitable links are put into the crawling queue. In last phase it applies mutation based search, some keywords are selected and run a query on search engines. The filtering rate of GA crawler is better than BFS.

In [9], Sotiris Batsakis et al. uses HMM (hidden Markov Model) through the combination of web page content and link anchor.HMM is a learning crawler and it works on user inputs to indicate whether the downloaded page is relevant or not. It records the sequence and it is used by the crawler to find relevant pages. It compares some crawlers with 10 selected topics and HMM model improves the performance of resulting focused web crawler and gives better results. They observe crawling pattern and try to find relevant pages through other non-relevant pages which increase the chances of failure.

In [10], Brieshwar Devarish et al. used semantic content to assign a number to web pages. It partitions the web page into content block based on hierarchical structure.

In [11], Junghoo Cho, Hector Garcia-Molina defines a parallel crawler as a crawler which uses multiple processes in parallel to improve the basic crawling process. This paper focuses on various issues of parallel crawling by proposing architecture and then proposes a metrics to evaluate a parallel crawler.

It implements crawlers in different modes. Firewall mode provides good coverage but it will not work for more than 4 processes at the same time. Crawler- based on exchange mode requires less bandwidth and maximize the quality of downloaded pages. But retrieving and comparing signatures of web documents was proposed knows as Spotsigs. In [4], exchange mode (replication) does not work well for more than 10,000 URLs.

In [12], Mauricio Marin et al. focused on efficient Management of URLs and synchronization issues. The set of new URLs are organized on the basis of priority measure. The URLs with high priority are downloaded first.

For efficient management of URLs, two priority Queues are used. One is based on complete binary tree and it achieves optimal performance in multi-core processor and the second one is suitable for effective utilization of secondary memory.

In [14], Vladislav Shkapenyuk, Torsten Suel describes the implementation of the optimized system on a network of workstations. It presents the architecture to solve the issues of scalability and to achieve higher efficiency. To achieve this, it uses breadth - first crawler. Although it is a scalable structure still it requires detail study of scalability and behaviour of its component.

In [15], Xiaochen Zhang et al. optimize distributed crawler under Hadoop platform. The working efficiency is optimized by increasing the number of Initial URLs, by improving the fetching process and by improving certain parameters.

Performance optimization is done through parameters modification and it is a feasible scheme. But it does not improve the efficiency of fetching and indexing process.

In [16], Kevin S. McCurley defines incremental crawling as the process of revisiting and prioritizing URLs. The main issue in incremental crawling is defining metrics for performance, for maintaining the quality of database and resources required to build and maintain the database. Incremental crawling depends upon polling, so it is inefficient. Notification and invalidation is an efficient approach but there is some economic expense to implement them.

In [17], Kevin McCurley, Jenny Edwards describes an adaptive approach for optimizing the performance of scalable and incremental web crawler. It optimizes the performance by identifying obsolete pages. In adaptive approach, it used information in the metadata of the page and when that page is crawled, it records whether a page is changed since the last crawl or not. The frequently changed pages are stored in the separate basket. It is two- stage adaptive models, within a crawler cycle it coordinates the management of URL and between cycle data necessary for the optimization is updated for the future cycle. In future, it can be implemented at large scale.

In [18], M.E.ElAraby et al. improve crawler efficiency in collecting web pages by using available resources. It uses Alchemi tool to implement crawler with grid computing. Using grid computing increases crawler performance. The architecture is based on parallel and distributed computing where tasks are divided into subtask in order to achieve maximum resource utilization and save time. Crawler performance can be enhanced by increased by increasing the number of executor nodes. In future crawler will be implemented which will update collected web pages.

In [19], Fengyun Cao et al. mentioned that speed and quality of webs pages are two important factors in optimizing crawler performance. Work is done by using scheduling algorithm to balance both the factor and optimize global crawl efficiency. It defines crawl-ability as the new ranking metric which combines performance and quality factors into scheduling decision-making.

Implementation is done with a real web crawler and tests four scheduling algorithm. Crawl-ability scheduling algorithm calculates aggregation of page quality based using algebraic summation. But it does not reflect the total value of the set of web pages where page qualities are dependent on each other.

III. DESIGN OF CRAWLER

In this paper, a crawler is designed which is based on the URL normalization and Categorization.URL normalization is applied in the query parameter of URL and categorization is applied at the level of crawling and searching. Categorization during crawling crawls only in particular category which helps to provide only relevant results and normalization reduces duplicate parameters. The proposed crawler shown in the "Fig.1." given below is basically working on reducing duplicate URLs in the crawl process.

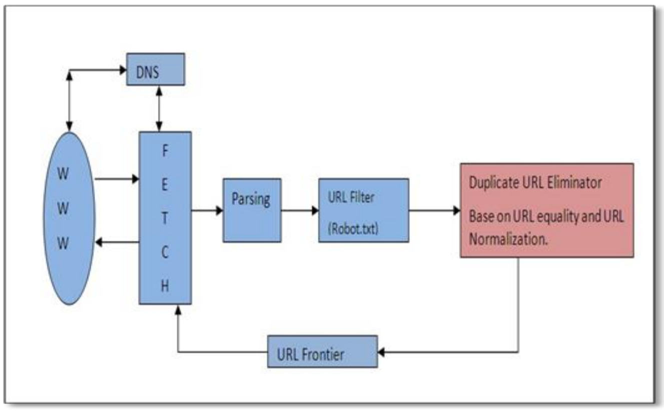

Fig1: Proposed architecture of Web crawler

The above architecture of proposed crawler is based on basically duplicity removal. The general process starts with seed URL .Crawler gets the seed URL and starts the process of crawl a URL after fetching a URL and parsing a web page, crawler gets the next URL to be visited. Here is the role of

URL normalization begins. Instead of adding each URL to the list of unvisited URLs i.e. to the URL frontier a duplicity detection method has been implemented in the architecture of crawler. The module is URL eliminator which checks next URL to be crawled for duplicity. If the URL passes the test it will be added to the list of unvisited URL otherwise, it will be rejected.

URL similarity test: When crawler gets new URL for crawling it first checks its equality. If URL is equal to already crawled URL then it will not be crawled again, if URL is not equal to previous URL then domain similarity and query parameter will be applied on the URLs. Domain similarity is the criteria of URL normalization which states that if

two URLs have same domain and different query string then they may lead to same page. In order to detect these URLs, If two URLs have same domain then rules of URL normalization are applied to Query string and new generated string will be compared with already existing string. If after applying rules, two strings matches then URLs will be treated same and one URL will be deleted.

FETCH: The fetch module is initiated by seed URL. Crawler is given seed URL or initial URL. Crawler fetches the web page correspond to seed URL and extracts hyper links from web page. These hyperlinks are link next to be visited.

DNS: (Domain Name Server): Domain name server converts domain name like .com .org into its IP address. Domain name are identified by human being but Computer works on the binary values. So DNS server converts particular Domain name in IP address so that it is recognizable by computer.

Parsing: It is the process of separating text and hyperlinks. After web page is fetched parsing is done in order to separate text from web links. Pages are stored separately and links are stored at one place so that they can be crawled in next cycle.

URL filters: There are set of rules that define web pages not to be crawled which are stored in a file called robot.txt. After fetching the page, URLs are checked in Robot.txt file.

URL Eliminator: Before adding URL to the URL frontier URL eliminator scans the URL on the basis of URL equality and URL normalization.URL equality checks that whether URL requested is equal to already crawled URL. If two URLs are equal then it is not added second time. After applying URL equality it applies URL normalization which states that syntactically different URLs are actually same or not.

URL Frontier: If two URLs are different in all aspects then it is added to the URL frontier.URL frontier contains links which are to be visited next.

WWW is World Wide Web of documents from where documents will be fetched. Fetch module will extract web page from the web and after that parsing will be done. Parsing involves separation of text from URLs. After the web page is extracted text and Links are separated. The Text is stored separately and links are processed further for crawling. After that links are passed through URL filter for checking robot.txt file. After that URL will be checked for duplicity. Duplicity will be checked on the basis of URL normalization and categorization. If URL passes the test of duplicity then it will be added to the URL frontier for crawling in the next cycle. The Same cycle will repeat for all the URLs in the Frontier.

Process flow of Crawler is shown in the “Fig.2.” given below:

A. Process Flow of Crawler

Step1: The process of Crawling will begin with user input. The User will enter seed URL and desired category in which URL is to be crawl. In proposed crawler, we have defined 10 categories for an experiment. However, it is not limited to 10 and number of categories can be added as per requirement.

Step2: After getting seed URL and category to crawl, the crawler will check that whether seed URL retrieves a web page or not. If it is not a web page, it will return an error message and stop the processing.

Step3: If the seed URL results in a valid webpage then Crawler will execute URL normalization process.

During URL normalization first we saved the name of website and website link into T_WEBSITE table with a unique website id (WSID) And entire URL will be temporarily stored in the table T_TEMPWEBPAGE for Further use with uniqe URLID value.

a) Next step in URL normalization process is to remove Http and https from URL as both schemes direct to the same web page.

b) Next, it will check that whether URL consists of index.html, default.html, and start.html. If these extensions are present then they will be removed from the URL as all these extensions lead to the same web page. In the above example, there is no such extension so URL will remain same.

c) After removing extensions it will remove port value 80,443. Value 80 is used with Http protocol and 443 is used with https. Both represent the same webpage so it will be removed. Next, it will remove .com from URL

d) Next, it will remove ‘..’ and ‘.’ From URL Double slash will be converted to single slash

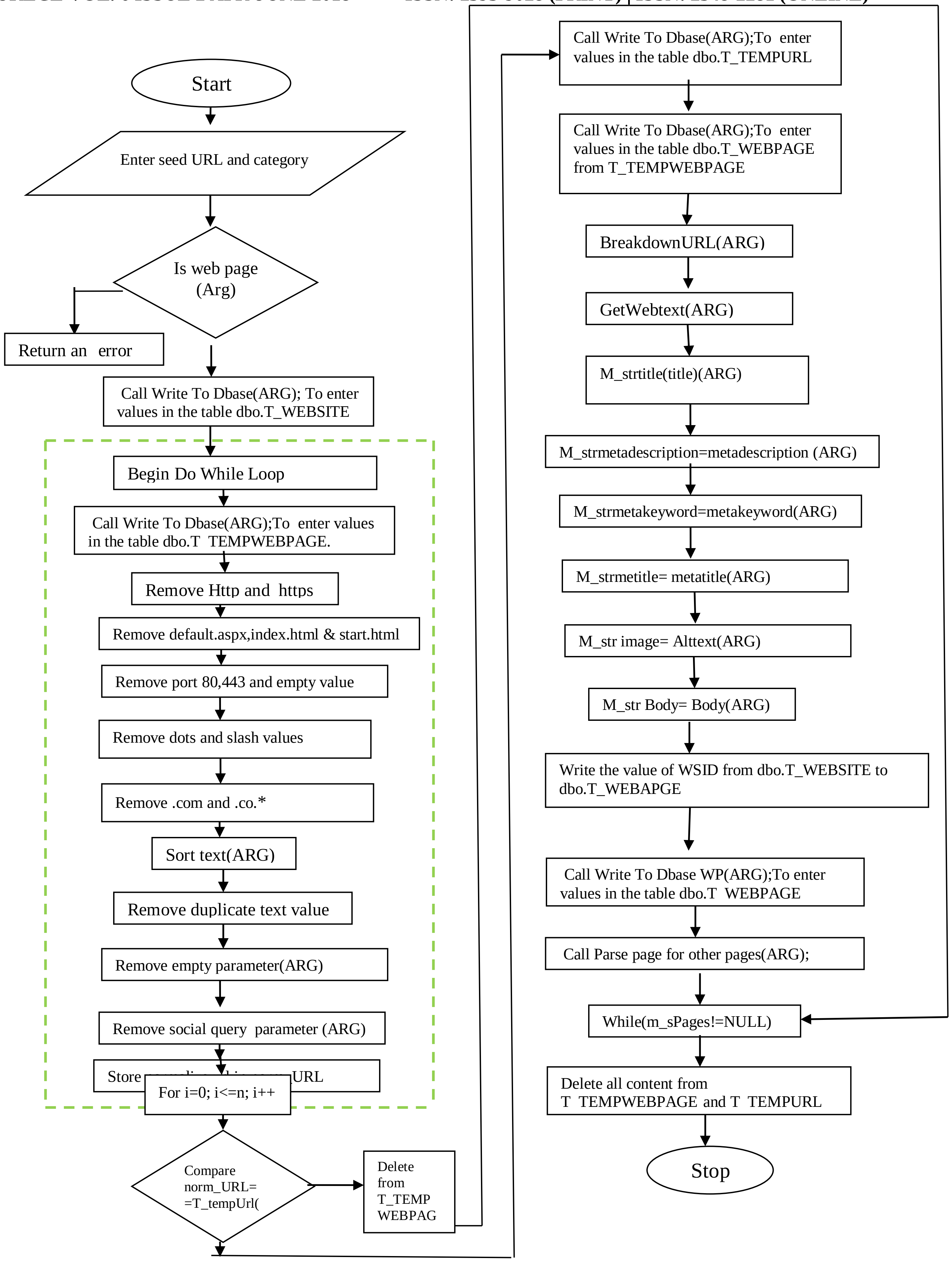
Start
Enter seed URL and category
Is web page (Arg)
Return an error
Call Write To Dbase(ARG); To enter values in the table dbo.T_WEBSITE
Begin Do While Loop
Call Write To Dbase(ARG);To enter values in the table dbo.T TEMPWEBPAGE.
Remove Http and https
Remove default.aspx,index.html & start.html
Remove port 80,443 and empty value
Remove dots and slash values
Remove .com and .co.*
Sort text(ARG)
Remove duplicate text value
Remove empty parameter(ARG)
Remove social query parameter (ARG)
Store URL
For i=0; i<=n; i++
Compare norm_URL= =T_tempUrl(
Delete from T_TEMP WEBPAG
Call Write To Dbase(ARG);To enter values in the table dbo.T_TEMPURL
Call Write To Dbase(ARG);To enter values in the table dbo.T_WEBPAGE from T_TEMPWEBPAGE
BreakdownURL(ARG)
GetWebtext(ARG)
M_strtitle(title)(ARG)
M_strmetadescription=metadescription (ARG)
M_strmetakeyword=metakeyword(ARG)
M_strmetitle= metatitle(ARG)
M_str image= Alttext(ARG)
M_str Body= Body(ARG)
Write the value of WSID from dbo.T_WEBSITE to dbo.T_WEBAPGE
Call Write To Dbase WP(ARG);To enter values in the table dbo.T WEBPAGE
Call Parse page for other pages(ARG);
While(m_sPages!=NULL)
Delete all content from T TEMPWEBPAGE and T TEMPURL
Stop

Fig.2: Process Flow of Crawler

After applying rules of URL normalization on different parts of URL, next step is to normalize the query string of URL. Below functions will be used to normalize query string :

e) Sort Text (ARG). There can be more than one query parameter in a URL. In order to normalize a URL it is necessary to arrange the query parameter in a particular order. This arrangement is achieved by sorting the parameters. Parameters are sorted alphabetically and URL is reassembled. For sorting the query parameter, Selection sort is applied

f) Remove duplicate text Value (ARG): After sorting query parameter, the duplicate parameter will be removed from the URL. If the value of one key/pair is equivalent to other key/pair then one will be removed.

g) Remove empty Query Parameter (ARG): Once duplicate parameters are removed, the next step is to remove empty parameters and from URL. It will remove BLANK values but 0 will be considered as a value.

h) Remove Social Query parameter: After sorting, removing duplicate values, removing an empty parameter, the social parameter will be removed.

After applying steps URL will be normalized and it will be stored in a string variable called norm_URL.

Step4**:** T_TEMPURL table in the database is used to store all unique normalized URL of a particular website (WSID). To check the uniqueness norm_URL value is compared with all others URLs stored in T_TEMPURL table. Since the two URLs are not same it will go to step 5. If two URLs are same then the URL will be removed from T_TEMPWEBPAGE, as it is a duplicate URL and control will be transferred to get the new page from the website for crawling.

Step5: If two URLs are not same then it will be stored in table T_TEMPURL.

Step6: The value of Original URL that was saved in T_TEMPWEBPAGE will be copied in T_WEBPAGE table. T_WEBPAGE table stores different information in different fields like URL will be stored in the field WPURL of T_WEBPAGE table.

Step7: URL will be broken into different parts using Break down URL (ARG) function. This function will accept URL as an argument and it will break the URL into Parts, It will separate, scheme, host, path and query string. Host part determines the web address of the page to be open.

Step8: Get Web text (ARG): After separating different parts of the URL, Crawler will retrieve text from the URL.

Step9: Title of the webpage will be stored in the variable called M_strtitle.

Step10: metaTitle of the Web page will be stored in a variable called M_strmetatitle.

Step11: metaKeyword of the Web page will be stored in a variable M_strmetakeyword**.**

Step12: metadescription of the Web page will be stored in variable M_strmetadescription.

Step13: Body text of Web page will be stored in the variable M_str body.
Step 14: Information about images will be stored in M_str image.

Step 15:After storing information in different variables it will be stored in table dbo.T_Webpage. The Title will be stored in field WPTITLE. Meta Title will be stored in field WPMETATITLE. Meta keyword will be stored in table WPMETAKEYWORD. Meta description will be stored in WPMETADESCRIPTION. The Body text of Web page will be stored in field WPHTMLWITHOUTTAGS. Website Id will be obtained from field WSID of T_WEBSITE and it will be stored in WSID of T_WEBPAGE.WSID is the connecting key between two tables (T_WEBSITE and T_WEBPAGE).

Step16: After storing page in table T_WEBPAGE , the crawler will get next link of the website to crawl and the process will continue until it crawls all the links of the website.

Step17: After crawling is completed for all the pages of the website, it will remove data from temporary table T_TEMPWEBPAGE and T_TEMPURL. This is how we are removing duplicity in a particular website based on query string parameter during crawling.

IV. IMPLEMENTATION OF CRAWLER

A. Frontend of Crawler

There are two parts of the crawler: Frontend of crawler and Backend of the crawler. The Frontend of the crawler is implemented using dot net framework and Visual Studio 2012.

B. Backend of Crawler

The Backend of the crawler is implemented by created tables in SQL Server management studio 2008 R2.
Five tables (T_WEBSITE, T_WEBPAGE, T_CATEGORY, T_TEMPWEBPAGE, T_TEMPURL) are used to implement and store URLs crawled by the crawler. Brief descriptions of the tables are summarized below:

1) T_WEBSITE

This table will be used to store information about Website name. Three fields are used in the table to store information: WSID, WSNAME, and WSLINK.

WSID: This is a unique field and contains the order of website in which it will be crawled.

WSNAME: It will contain the name of the Website.
WSLINK: It will contain URL of Website.

2) T_WEBPAGE

This table holds all the information of Web page of Website. Eleven fields are used in the table to store information. WPID, WSID, WPTITLE, WPMETATITLE, WPMETAKEYWORD, WPMETADESCIPTION, WPCATERORY, WPEXTENSION, WPURL, WPFULL HTML, WPHTMLWITHOUT TAGS.
WPID: This field will contain the ID of the page that will be crawled.
WSID: This field will contain the ID of the website and it will retrieve a value from T_WEBSITE table. This is the connecting key between the two tables.

WPTITLE: It will contain Title of a webpage.
WPMETATITLE: It will contain meta title of the web page.

WPMETAKEYWORD: All the meta keywords of a page will be extracted and will be stored in this field.

WPMETADESCRIPTION: Meta description of a page will be stored in this field.
WPCATEGORY: It will retrieve category number from CTID of dbo.T_CATEGORY table. CTID is the connecting key between two tables.
WPEXTENSION: It will contain extension of page i.e. .com, org.

WP URL: It will contain web page of a URL.

WP FULL HTML: It will contain full HTML of a page.

WPHTMLWITHOUTTAGS: It will contain full HTML.

3) T_CATEGORY

This table contains information about category in which crawler will crawl a URL. Whole information will store in two fields i.e. CTID and CTNAME.
CTID: It will contain ID number of category and
CTNAME: It will contain the name of the category.

4) T_TEMPWEBPAGE

This table will contain URL temporarily. Here URL will be stored for processing. After processing either URL will be deleted from the table or it will be moved to T_WEBPAGE table. This table will contain WSID, WPID , and WPURL.
WSID: It will retrieve the value from T_WEBSITE and it will contain website Id.
WPID: It will store the ID of page that will be crawled.
WPTURL: This field will contain actual URL to be crawled.

5) T_TEMPURL

This table will store normalize form of URL at every stage. After applying the entire rules table will store final URL which will be used for comparison with other URLs. This table will store information in two fields i.e. WPID and Norm_URL.

WPID: It will retrieve the value from WPID of T_TEMPWEBPAGE.
Norm_URL: It will contain entire normalized URL of the website for comparison.

C. Illustration of the crawling process

A User wants to crawl educational website http://www.abc.com then it will be treated as seed URL.URL will be entered in the field website and category will be selected from the drop-down website
Category in the interface of the crawler. After that User will click crawl button and crawling will start.
The Crawler will start its process by entering values in T_WEBSITE table in the database for seed URL http:.//www.abc.com.
WSID: So a unique value will be assigned to this field based on the previous values.
WSNAME: It will contain hostname of website i.e. abc.
WSLINK: It will contain actual link i.e. http://www.abc.com.
Suppose first URL encounter is http://**www.abc.com:80/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?**
After this value will be entered in T_TEMPWEBPAGE table.

WSID: It will store same value as in dbo.T_WEBSITE.

WPID: It will contain page id of URL retrieve from seed URL. Here URL is http://**www.abc.com:80/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?** and page id of URL is supposed to be 12. So, it will hold the value 12.

WPURL: It will contain actual URL: http://**www.abc.com:80/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?**

a) From the above URL http will be removed and resulting URL will be www.abc.com:80/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?

b) After removing 80 from above URL, resulting URL will be www.abc.com/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?

c) After removing index.html and.com resulting URL will be **www.abc/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?**

d) After removing '.' And '..' resulting URL will be:
www.abc/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?
e) After applying sort function resulting URL will be:
www.abc/a/b/c?d=1&d=2&e=3&f=4&f=4&g=0&h=?.

f) Remove duplicate values and URL will be:
www.abc/a/b/c?d=1&d=2&e=3&f=4&g=0&h=?.

g) Remove empty parameter and URL will be:

www.abc/a/b/c?d=1&d=2&e=3&f=4&g=0.

This normalized URL will be compared with other URLs saved in TEMP_URL table. If the URL is unique it will be saved otherwise it will be deleted from TEMP_WEBPAGE. Suppose it is a unique URL then entries in all the table will be at this particular time will be:

T_WEBSITE

WSID	WSNAME	WSLINK
1	abc	http://www.abc.com

T_TEMPWEBPAGE

WSID	WPID	WPTURL
1	12	http://www.abc.com:80/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?

T_TEMPURL

WPID	NORMURL
12	www.abc/a/b/c?d=1&d=2&e=3&f=4&g=0.

T_WEBPAGE

WPID	WSID	WP CATEGORY	WP TITLE	WP META TITLE	WP META KEYWORD	WP META DESCRIPTION	WP EXTENSION	WP URL	WP FULL HTML	WP HTML Without tags
12	1	1	abc.com		abc education school college	No.1 college	.com	http://www.abc.com:80/../a/b/./c?d=1&e=3&f=4&D=2&f=4&g=0&h=?		

V. CONCLUSION AND FUTURE WORK

The crawler is designed to perform the categorization in crawling and URL Normalization on query parameter to remove duplicates URLs during crawling. The crawler is designed with the idea of removing duplicate URLs during crawling to improve the performance of crawler. In future Crawler can be extended to include automated categorization during crawling.

The proposed crawler compares a particular website only once and it allows re-enter the website again. For e.g. Suppose crawler receives the seed URL http://www.abc.com. For the first time of crawling, the crawler will crawl all the URLs of the website and will not enter duplicate URLs. But after some time if user re-enters the same website, it will again enter it into the database without checking that URLs are already entered. In future mechanism can be devised to check, if user enters the same URL then, crawler renter only those pages which are updated and ignore the rest. Also, it should keep the only updated page and remove the previous one.

VI. REFERENCES

[1]. Rodrigues, K , Cristo,M, de Moura, E. S. and Silva da, A(2015), "Removing DUST Using Multiple Alignment of Sequences," IEEE Transactions on Knowledge and Data Engineering, volume. 27,Issue 8, pp. 2261-2274.

[2]. Theobald, M , Siddharth, J. and Paepcke, A (2008), "Spotsigs: Robust and efficient near duplicate detection in large web collections", Proceedings of 31st Annual International ACM SIGIR Conference on Research and development in information retrieval, Singapore, pp. 563–570.

[3]. Agarwal, A, Koppula, H.S, Leela, K.P, Chitrapura K.P, Garg S., GM P.K., Hatty C, Roy, A and Sasturkar, A(2009) "Url normalization for de-duplication of web pages", Proceedings of the 18th ACM conference on Information and knowledge management, Hongkong: China, pp.1987–1990.

[4]. Taylan, D, Poyraz, M, Akyokuş, S and. Ganiz, M. C. (2011), "Intelligent focused crawler: Learning which links to crawl," International Symposium on Innovations in Intelligent Systems and Applications, Istanbul, pp. 504-508.

[5]. Bar-Yossef , Z. , Keidar I. and Schonfeld, U.(2009), "Do not crawl in the dust: Different urls with similar text", ACM Transactions on the Web, volume. 3.

[6]. Dasgupta, A, Kumar, .R and Sasturkar, A.(2008), "De-duping urls via rewrite rules", Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, USA: Las Vegas, pp.186–194.

[7]. Olston, C and Najrok,M (2010), "Web Crawling", Foundations and Trends in Information Retrieval, volume .4, pp.175-246.

[8]. Yohanes, BW, Handoko,H and Wardana, HK (2011) , "Focused Crawler Optimization Using Genetic Algorithm", TELKOMNIKA, volume. 9, pp. 403-410.

[9]. Batsakis, S, Petrakis, E G.M. and Milios, E (2009), "Improving the Performance of Focused Web Crawlers", Data and Knowledge Engineering, volume 68, pp. 1001-1003.

[10]. Ganguly, B and Raich, D(2014), "Performance Optimization of Focused Web Crawling Using Content Block Segmentation", International Conference on Electronic Systems, Signal Processing and Computing Technologies, Nagpur, pp. 365-370.

[11]. Cho, J and Garcia-Molina, H(2002), "Parallel Crawlers", Proceedings of the 11th International Conference on World Wide Web, USA: Hawaii, pp 124-135.

[12]. Marin, M. Paredes, R and Bonacic , C(2008) , "High-Performance Priority Queues for Parallel Crawlers", Proceedings of the 10th ACM workshop on Web information and data management, USA: Napa Valley, California, pp. 47-54.

[13]. Olston, C and Najrok, M (2010), "Web Crawling", Foundations and Trends in Information Retrieval, volume .4, pp.175-246.

[14]. Shkapenyuk, V and Suel, T (2002),"Design and implementation of a high-performance distributed Web crawler," in Proceedings of the 18th International Conference on Data Engineering, San Jose, CA, pp. 357-368.
[15]. Xiaochen , Z and Xian , M (2015), "Optimization of Distributed Crawler under Hadoop", MATEC web of conferences: International Conference on Engineering Technology and Application.
[16]. McCurley, KS., "Incremental Crawling", Encyclopedia of Database Systems, Liu, Ling, Ozsu M. Tamer, Ed. Boston (MA): Springer, pp.1417-1421.
[17]. Edwards, J., McCurley, K. and Tomlin, J .A(2001), "An Adaptive Model for Optimizing Performance of an Incremental Web Crawler", Proceedings of the 10th international conference on World Wide Web, Hong Kong, pp.106-113.
[18]. Elaraby, M.E., Sakre , M. , Rashad, M.Z. and Nomir, O(2012), "Crawler Architecture using Grid Computing", International Journal of Computer Science & Information Technology, volume. 4, Issue 3.
[19]. Cao, F., Jiang, D. and Singh, JP(2003), "Scheduling Web Crawl for Better Performance and Quality". Tech Rep., TR-682-03.

Kavita Goel, MCA, Pursuing P.hd. Received the Bachelor Degree in computer application from MDU Rohtak in 2005 and received the Post graduation degree in computer application from MDU Rohtak in 2008.Teaching experience of 2.5 years in GSMVNIET and presently working in DAV Institute of Management since 2015.