A Query based Approach to Reduce the Web Crawler Traffic using HTTP Get Request and Dynamic Web Page

Shekhar Mishra
RITS Bhopal (MP)
India

Anurag Jain
RITS Bhopal (MP)
India

Dr. A.K. Sachan
RITS Bhopal (MP)
India

ABSTRACT

The functions of Web crawler download information from web for search engine. Web pages changed without any notice. Web crawler has to revisit web site to download updated and new web pages. It is estimated 40% of current web traffic is generated by web crawler. This paper proposes query based approach to inform updates on web site to web crawler using Dynamic web page and HTTP GET Request. Dynamic web page generates HTML based response having list of updates on web site after crawler last visit. Web crawler only visits updated web pages instead of visiting full web sites for updates. Proposed scheme is tested & results show that it is very promising.

General Terms

HTTP GET request query based approach to reduce Web Crawling Traffic.

Keywords

Web Search Engine, Web, Web Crawler. Web Crawling Traffic. HTTP GET request query.

I. INTRODUCTION

Today Web is main source of information. Web is network of interconnected Hyperlinks on Internet. Web is accessed with the help of web browser [2].

To reach particular page most of the user use search engines. Web Crawler is a main component of Search Engine. Web Crawler automatically browses Web and downloads information for Search engine [1]. Search engine needs updated information to generate results for user queries. Web crawler revisit web sites for search of updates.

Yuan, et al. perform analysis to web log of University of Alberta web site in 2002. They find that maximum 40.6% of total web traffic is generated by web crawler [15].

In 2010 Sun, et al analysis log file of many different web sites. They find average 50% of web request is generated by web crawler [13].

Authors propose approach to use a dynamic web page. This dynamic web support query. Web crawler send HTTP GET request with parameter. Dynamic web page checks parameter and generate response to web crawler having list URLs of updated web pages.

In this paper section two is about related work done in this research area. Section three has approach used with figure. Simulator used for experiment is in section four. Section five has experiment with tables. Data analysis with graph is in section six. Conclusion and references are at last

II. RELATED WORK

To reduce the crawling traffic and to find updates effectively, research is being conducted in different areas.

One of the proposed approach is to place web crawler in different geographical areas. Web crawler downloads web pages within its geographical area [7].

Other solution is placing active routers in key place in network. Active routers record underling traffic for indexing [15].

Another proposed scheme is to place mobile crawler at web server. Crawler check updates in web site and send updates for indexing to search engine [6, 11].

Unethical crawler cause problems to network, web servers. Causing denial of service, copying user private information and downloading copyright data [14].

Research is in progress on Crawler ethics. Researchers have derived the formula to calculate the ethicality of web crawler [13]

III. APPROACH USED

Authors propose the use of dynamic web page to inform the web crawler about the new pages and updates on web site. Dynamic web page accepts one parameter LAST_VISIT. This parameter indicates time and date of last visit of web crawler to website. Parameter value is passed by HTTP GET request.

Dynamic web page accepts the HTTP GET request with parameter LAST_VISIT and generates response. Response is HTML based. Response is having list of URLs of pages updated after crawler last visit. Dynamic web page use "List "data Structure to generate results

Crawler passes the last visit parameter to dynamic web page with HTTP GET request. Web crawler receives the HTML based response having list of all updated pages URLs. Crawler only visits the URLs list instead of visiting full web sites for updates. Figure 1 shows the working of proposed scheme.

List: it is an array type of Data Structure, maintains the page URLs and time of its update. List is updated by web crawler at web server.

Algorithm for crawler

1. Web crawler sends the HTTP GET request with LAST_VISIT parameter value to the dynamic page. Value of LAST_VISIT is time and date of last visit of web crawler to web site.
2. Crawler receives updated page URLs in HTML form.
3. Web crawler visit updated URLs.

Algorithm for dynamic web pages

1. Dynamic web page Accepts HTTP GET request with LAST_VISIT parameter.
2. Dynamic web page generates the HTML based response having URLs of pages updated after crawler last visit. With the help of "LIST" data structure.

Advantages

1. Proposed scheme can be implemented on existing system.
2. Existing web crawler can easily send the query to web server with request URLs.
3. Different web crawler visit web site at different frequency. Dynamic web page every time produces different response for web crawler.
4. This scheme guarantees that crawler only visit update pages
5. When there is no update crawler will only visit the Dynamic update pages and leave the web site.
6. Crawling is only visiting to update pages. So crawler will work fast.

Limitation

Both web crawler and web site should follow rules.

Difference between proposed approach with other protocols

Sitemaps protocol: this protocol tells the web crawler about the Site map of web site. It is based on XML [5]. Proposed scheme based on HTML. Main aim of proposed scheme is to tell web crawler about updates.

Robots Exclusion Protocol: this protocol is used to tell web crawler as to which page to visit and which page should be omitted. Protocol does not provide updates [3, 4].

Static HTML Update File: different crawler visit web site at different frequency. It does not support query based approach. Proposed approach makes sure that crawler only get the list of updates after crawler last visit [9].

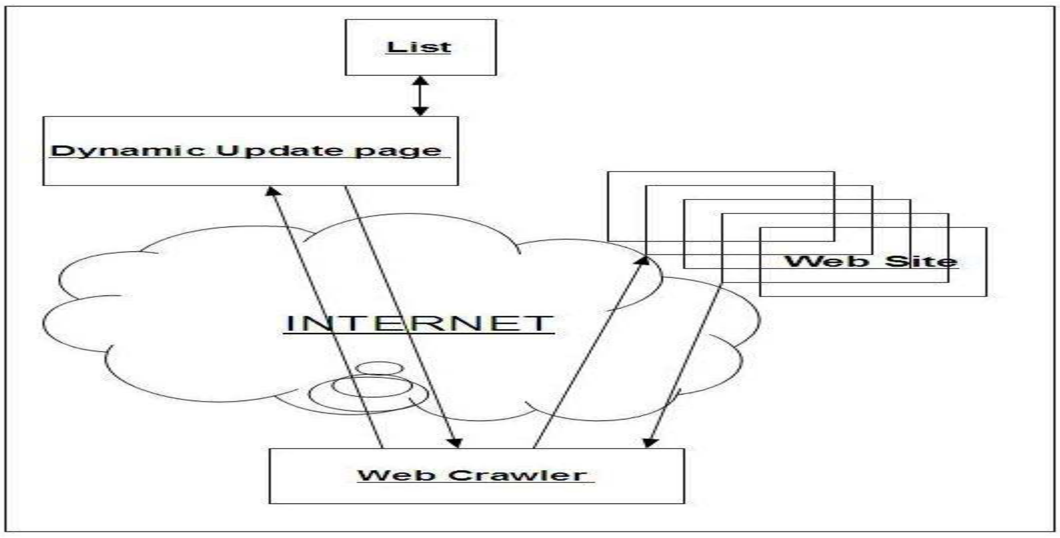


Figure 1 Proposed approach

IV. SIMULATOR

In experiment we use web site of 21 web pages. General structure of web site is in Figure 2. Web site deployed on Apache Tomcat 6.0.18 J2ee Web server. Dynamic web page and list data structure is coded in JAVA and JSP. Web crawler is coded in java. LAST_VISIT parameter passed is millisecond time of system, return by java function (System.currentTimeMillis()).same millisecond time is maintained in "list"data structure. First we perform crawling on web site using old approach. Then we perform crawling using proposed approach.

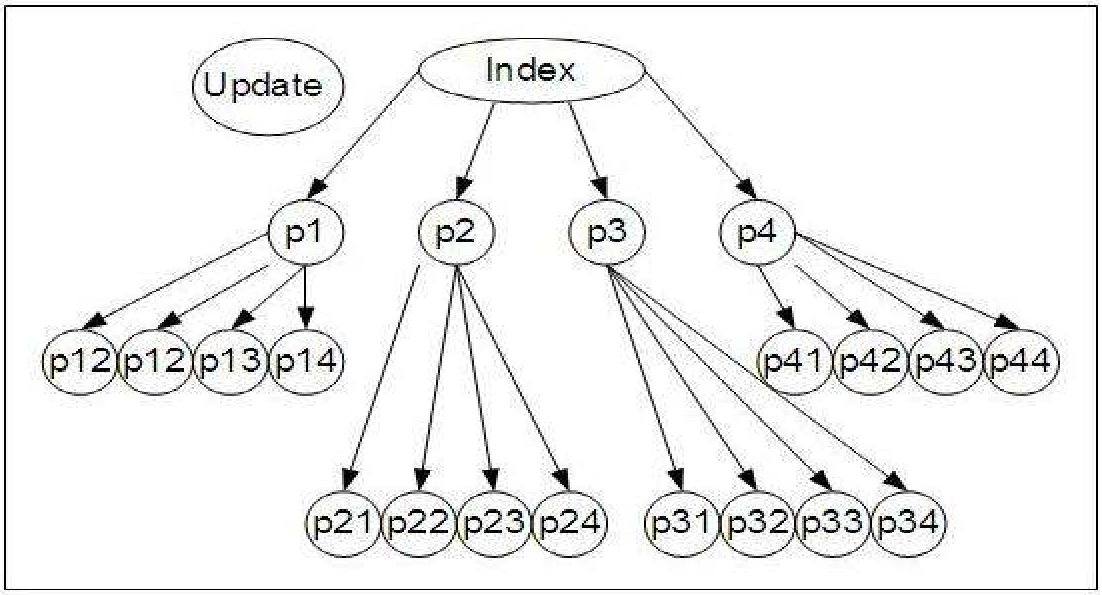


Figure 2 structure of website used for experiment

V. EXPERIMENT

We perform the web crawling on Experimental website. The results obtained shown in Table1.

Table 1

Index	URL	Start Time of Crawler (Time in milliseconds)	End Time of Crawler (Time in Milliseconds)	Time to reach this URL(Time in milliseconds)
1	http://localhost:8084/CrawlerTest1/index.jsp	1292068731963	1292068732197	234
2	http://localhost:8084/CrawlerTest1/p2.jsp	1292068731963	1292068732353	390
3	http://localhost:8084/CrawlerTest1/p3.jsp	1292068731963	1292068732509	546
4	http://localhost:8084/CrawlerTest1/p4.jsp	1292068731963	1292068732680	717
5	http://localhost:8084/CrawlerTest1/p1.jsp	1292068731963	1292068732835	872
6	http://localhost:8084/CrawlerTest1/p21.jsp	1292068731963	1292068733022	1059
7	http://localhost:8084/CrawlerTest1/p24.jsp	1292068731963	1292068733193	1230
8	http://localhost:8084/CrawlerTest1/p22.jsp	1292068731963	1292068733348	1385
9	http://localhost:8084/CrawlerTest1/p23.jsp	1292068731963	1292068733550	1587
10	http://localhost:8084/CrawlerTest1/p32.jsp	1292068731963	1292068733706	1743
11	http://localhost:8084/CrawlerTest1/p33.jsp	1292068731963	1292068733892	1929
12	http://localhost:8084/CrawlerTest1/p31.jsp	1292068731963	1292068734079	2116
13	http://localhost:8084/CrawlerTest1/p34.jsp	1292068731963	1292068734234	2271
14	http://localhost:8084/CrawlerTest1/p41.jsp	1292068731963	1292068734421	2458
15	http://localhost:8084/CrawlerTest1/p44.jsp	1292068731963	1292068734686	2723
16	http://localhost:8084/CrawlerTest1/p43.jsp	1292068731963	1292068734811	2848
17	http://localhost:8084/CrawlerTest1/p42.jsp	1292068731963	1292068735095	3132
18	http://localhost:8084/CrawlerTest1/p11.jsp	1292068731963	1292068735188	3225
19	http://localhost:8084/CrawlerTest1/p12.jsp	1292068731963	1292068735437	3474
20	http://localhost:8084/CrawlerTest1/p13.jsp	1292068731963	1292068735655	3692
21	http://localhost:8084/CrawlerTest1/p14.jsp	1292068731963	1292068735831	3868

To test proposed approach we direct the web crawler to dynamic web page update.jsp and set the last visit time at URL and perform crawling.

Experiment 1: we update time and URLs of pages P1, P23 and P34 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. Performed crawling, results obtained are shown in table 2.

Table 2

Index	URL	Start Time of Crawler (Time in Milliseconds)	End Time of Crawler (Time in Milliseconds)	Total Time to Download this URL(Time in milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit=1292236840408	1292236868332	1292236868551	219
2	http://localhost:8084/CrawlerTest1/p1.jsp	1292236868332	1292236868769	437
3	http://localhost:8084/CrawlerTest1/p23.jsp	1292236868332	1292068732509	546
4	http://localhost:8084/CrawlerTest1/p34.jsp	1292236868332	1292236869099	767

Experiment 2: we update time and URLs of pages P1, index and P32 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. Performed crawling, results obtained are shown in table 3.

Table 3

Index	URL	Start Time of Crawler (Time in Milliseconds)	End Time of Crawler (Time in Milliseconds)	Total Time to Download this URL(Time in milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit=1292236840408	1292259222088	1292259222463	375
2	http://localhost:8084/CrawlerTest1/p32.jsp	1292259222088	1292259222634	546
3	http://localhost:8084/CrawlerTest1/p1.jsp	1292259222088	1292259222761	673
4	http://localhost:8084/CrawlerTest1/index.jsp	1292259222088	1292259222948	860

Experiment 3: we update time and URLs of pages P1, P2 and P3 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. Performed crawling, results obtained are shown in table 4.

Table 4

Index	URL	Start Time of Crawler (Time in milliseconds)	End Time of Crawler (Time in Milliseconds)	Time to reach this URL(Time in Milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit=1292272248608	1292272266330	1292272266517	187
2	http://localhost:8084/CrawlerTest1/p2.jsp	1292272266330	1292272266642	312
3	http://localhost:8084/CrawlerTest1/p3.jsp	1292272266330	1292272266845	515
4	http://localhost:8084/CrawlerTest1/p1.jsp	1292272266330	1292272267119	789

Experiment 4: we update time and URLs of pages P22 and P11 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. . Performed crawling, results obtained are shown in table 5.

Table 5

Index	URL	Start Time of Crawler (Time in milliseconds)	End Time of Crawler (Time in Milliseconds)	Total Time to Download this URL(Time in milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit =1292307271473	1292307297416	1292307297665	249
2	http://localhost:8084/CrawlerTest1/p11.jsp	1292307297416	1292307297837	421
3	http://localhost:8084/CrawlerTest1/p22.jsp	1292307297416	1292307297992	576

Experiment 5: we update time and URLs of page P11 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. . Performed crawling, results obtained are shown in table 6.

Table 6

Index	URL	Start Time of Crawler (Time in Milliseconds)	End Time of Crawler (Time in Milliseconds)	Total Time to Download this URL(Time in Milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit =1292307947094	1292307968746	1292307968980	234
2	http://localhost:8084/CrawlerTest1/p11.jsp	1292307968746	1292307969168	422

Experiment 6: we update time and URLs of pages P33 and P42 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. . Performed crawling, results obtained are shown in table 7.

Table 7

Index	URL	Start Time of Crawler (Time in Milliseconds)	End Time of Crawler (Time in Milliseconds)	Time to reach this URL(Time in Milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit =1292309416411	1292309445240	1292309445474	234
2	http://localhost:8084/CrawlerTest1/p33.jsp	1292309445240	1292309445661	421
3	http://localhost:8084/CrawlerTest1/p42.jsp	1292309445240	1292309445787	547

Experiment 7: we update time and URLs of pages P21 and P41 in "List" data structure. At web crawler we set the LAST_VISIT time before time of pages in the LIST. . Performed crawling, results obtained are shown in table 8.

Table 8

Index	URL	Start Time of Crawler (Time in milliseconds)	End Time of Crawler (Time in Milliseconds)	Total Time to Download this URL(Time in milliseconds)
1	http://localhost:8084/CrawlerTest1/update.jsp?LastVisit=1292308646738	1292308662962	1292308663196	234
2	http://localhost:8084/CrawlerTest1/p21.jsp	1292308662962	1292308663321	359
3	http://localhost:8084/CrawlerTest1/p41.jsp	1292308662962	1292308663507	545

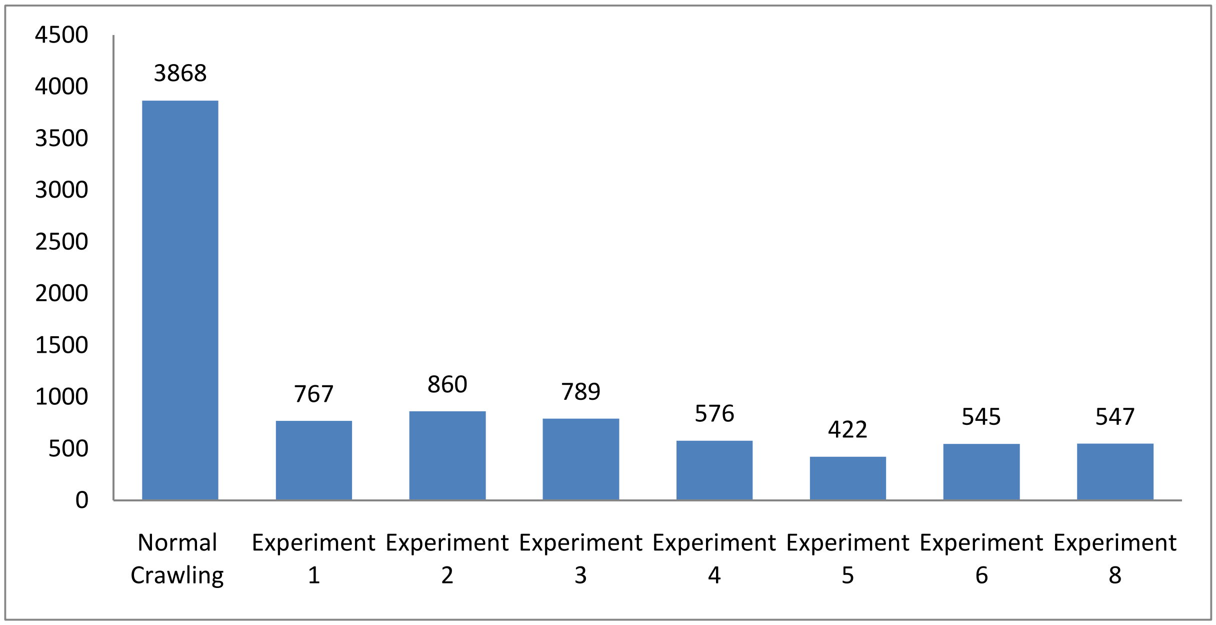
VI. DATA ANALYSIS

In normal crawling crawler visit every web page to know all updates in web site. In normal crawling it visits a total of 21 pages. Crawler takes 3868 milliseconds to visit complete site. In proposed approach crawler visits Dynamic update page and updated web pages only. Crawler take about 800 milliseconds when there are 3 updates, about 550 milliseconds when there are two update. About 420 milliseconds when there is one update.

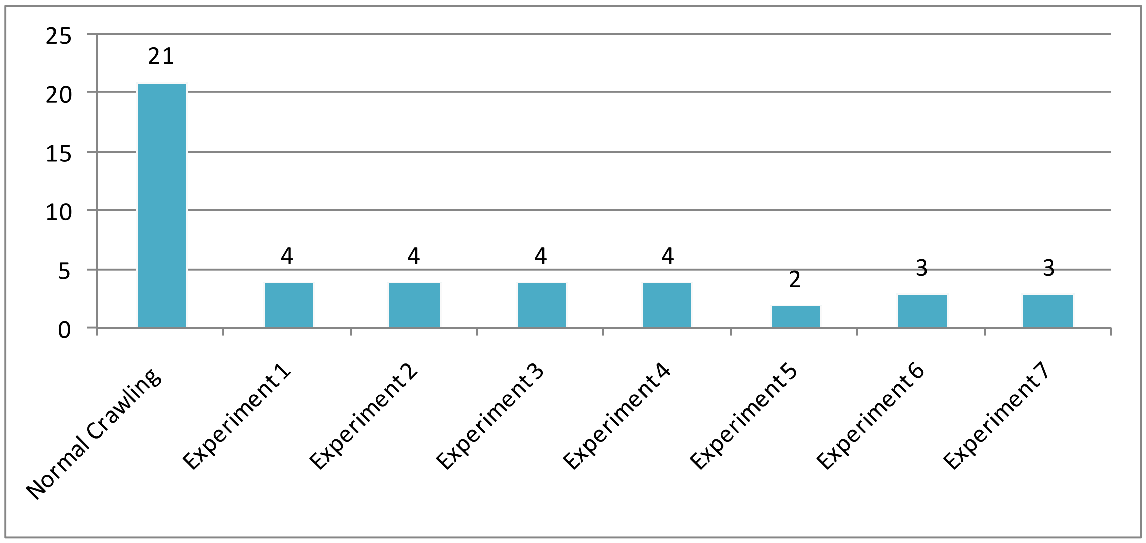
When there are three updates in experimental web site proposed sachem is 4.83 time faster than old approach. With two updates proposed scheme is 7.03 times faster than old scheme. If there is only one update scheme is 9.2 times faster than old approach.

Graph 1 shows time taken by web crawler to download updates.

In normal crawling crawler visits 21 pages to find updates. But number of page visit is very small in proposed approach. When there is one update crawler only visit 2 pages. When there are 2 updates crawler only visits 3 pages. If there are 3 updates in web site crawler visit 4 pages. Graph 2 shows number of page visits by crawler to reach update.



Graph 1. Time take by crawler to download updated web page in Experiments (Time in Milliseconds*)*



Graph 2 Shows number of page visited by crawler to find updates.

VII. CONCLUSION

With this approach crawler can ask the web server what is new on server. It is very easy to implement on today's infrastructure. Crawler can easily send the queries with request URLs. Many different types of queries can be implemented with this scheme. Other metadata of web pages sharing can also be implemented with this idea.

REFERENCES

[1] "Web crawler", From Wikipedia, http://en.wikipedia.org/wiki/Web_crawler
[2] "World Wide Web", From Wikipedia, http://en.wikipedia.org/wiki/World_Wide_Web
[3] "Robots Exclusion Protocol", http://www.robotstxt.org/robotstxt.html
[4] "Robots exclusion standard", Wikipedia http://en.wikipedia.org/wiki/Robots_exclusion_standard
[5] "Sitemaps", Wikipedia, http://en.wikipedia.org/wiki/Sitemaps
[6] Bal.S and Nath.R,"Filtering the web pages that are not modified at remote site without downloading using mobile crawler". Information Technology journal 9(2)2010 ISSN 1812-5638, Asian Network for Sciencetific information. (pp: 376-380)
[7] Cambazoglu, B.B.; Junqueira, F.; Plachouras, V.; Telloli, L., "On the feasibility of geographically distributed web crawling." (ISBN: 978-963-9799-28-8) In the proceedings of Third International ICST Conference on Scalable Information Systems, ICST, Vico Equense, Italy (2008)
[8] Chandramouli A and Gauch. S. "A Co-operative Web Services Paradigm for Supporting Crawlers", In the proceedings of Computer-Assisted Information Retrieval (Recherche d'Information et ses Applications) - RIAO 2007, 8th International Conference, Carnegie Mellon University, Pittsburgh, PA, USA, May 30 - June 1, 2007.
[9] Mishra.S, Jain.A and Sachan A.K,"Smart Approach to Reduce the Web Crawling Traffic of Existing System using HTML based Update File at Web Server", International Journal of Computer Applications 11(7), December 2010(pp: 34–38)
[10] McCurley S. Kevin "Incremental Crawling" Google Research http://static.googleusercontent.com/external_content/untrusted_dlcp/www.google.com/en//research/pubs/archive/34403.pdf
[11] Pahal N, Kumar S, Bhardwaj A and Chauhan N," Security Mobile Agent Based Crawler = (SMABC)". International Journal of Computer Applications 1(14), February 2010. (pp: 5–11)
[12] Sharma A.K, Dixit. A and Singhal N. "Design of a Priority Based Frequency Regulated Incremental Crawler" 2010 International Journal of Computer Applications (ISSN: 0975 – 8887) Volume 1 – No. 1. (pp: 42-47)
[13] Sun. Y, Councill G. Isaac and Giles C. Lee, "The Ethicality of Web Crawlers", in the proceedings of 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Toronto Canada august 2010. (pp: 668-675)
[14] Thelwall. M and Stuart. D, "Web crawling ethics revisited: Cost, privacy and denial of service". Journal of the American Society for Information Science and Technology. 2006. Volume 57, Issue 13 November 2006. (pp: 1771 - 1779)
[15] Yuan, X.M. and J. Harms, "An efficient scheme to remove crawler traffic from the internet." Proceedings of the 11th International Conference on Computer Communications and Networks, Oct 2002. 14-16, IEEE CS Press, (pp: 90-95).