ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Website Health Analyser to filter obsolete links to avoid Link Rot

Jay Doshi and Juhi Duseja

Final Year Students, Department of Computer Science & Information Technology, VJTI, Mumbai, India

ABSTRACT: The purpose of this project is to crawl a website to find out the status of each of the links contained in its pages. Each website hosted on the Internet contains several links and each of these links reference a certain web page. Over the time, these web pages are either removed or are no longer available due to several reasons, but the links referencing them still exist. This causes those links to be obsolete and it becomes the responsibility of the website developers to periodically check all the links in all the web pages in their website. Moreover, having obsolete or wrongly referenced links also harms the SEO ranking of the website in various Search Engines and in certain cases lead to a loss in revenue. This paper also discusses the necessary features a successful web crawler must possess.

KEYWORDS: Large Scale Distributed Crawling, Website Crawling, Broken links.

I. INTRODUCTION

The algorithms used by search engines such as Google, Bing, DuckDuckGo all have one thing in common and that is the way in which they analyse the credibility of a website. For instance, Google's PageRank algorithm works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more and high quality links from other websites. So wrongly referenced or broken links cause a website's importance to decrease and thus slipping low. Hence, analysing the health of links, i.e. to check if they are correctly referenced is of high value. Broken links also induce falsely influenced PageRank rankings which can mislead the crawler and in turn the search engines [13]. This ultimately leads to loss of users and they cannot obtain the requested content.

This paper analyses the approaches used to identify and eliminate obsolete and wrongly referenced links. Furthermore, it also suggests the necessary features any web crawler must possess based on their performance on a large number of websites.

II. RELATED WORK

In the year 1998, it was estimated that over 600 GB (out of the then 1.5 Tb Internet) of web pages were changing each month. This number has grown to several petabytes since the size of the internet has also significantly increased. As of 2014, Google has indexed 200 Terabytes (TB) of data. To put that into perspective 1 TB is equivalent to 1024 Gigabytes (GB). However, Google's 200 TB is just an estimated 0.004 percent of the total Internet. So keeping track of the changes in the web pages referenced by the links in any website becomes important.

Any link/URL can be present in the <a> tag of the HTML. It can be either a hyperlink, a button or even an image. Such a URL, when requested returns an HTTP status code. This HTTP status code enables the browsers to categorize the page as relevant or irrelevant. For example, on querying https://google.com, a status code of 200 is returned by the Web server (running google.com). Furthermore, a Web server returns a status code 400 when it thinks that the data stream sent by the client (i.e. your Web browser) was 'malformed' i.e. did not respect the HTTP protocol completely. So the Web server was unable to understand the request and process it. Section V offers an exhaustive list of HTTP status codes used for interaction between browsers and servers.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

III. PROPOSED ALGORITHM

A. *Breadth First Search:*

The breadth-first-search (BFS) algorithm starts at a vertex (root) s and visits, first the neighbors of s, then the neighbors of the neighbors of s, then the neighbors of the neighbors of the neighbors of s, and so on.
This algorithm is a generalization of the breadth-first traversal algorithm for binary trees, and is very similar; it uses a queue, q which initially contains only s. It then repeatedly extracts an element from Q and adds its neighbors to Q, provided that these neighbors have never been in Q before. The only major difference between the breadth-first-search Algorithm for graphs and the one for trees is that the algorithm for graphs has to ensure that it does not add the same vertex to Q more than once.

B. *Depth First Search:*

The depth-first-search fully explores one subtree before returning to the current node and then exploring the other subtree. Another way to think of depth-first-search is by saying that it is similar to breadth-first search except that it uses a stack instead of a queue.

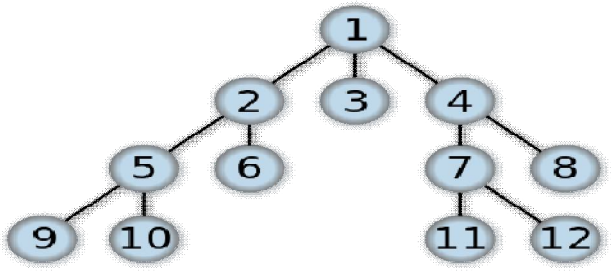

Fig. 1. Graph with nodes representing pages and branches representing links

Traversal by BFS: 1->2->3->4->5->6->7->8->9->10->11->12

Traversal by DFS: 1->2->5->9->10->6->3->4->7->11->12->8

Setting the depth limit to limit the scope of the web-crawler:
A few websites are too huge to scan completely, for that we need to set a max limit in the depth. For example if root -> page1 -> page2 is the relation between these pages and if we set a depth of 2, the link page2 (i.e. the children of page2) must not be processed.

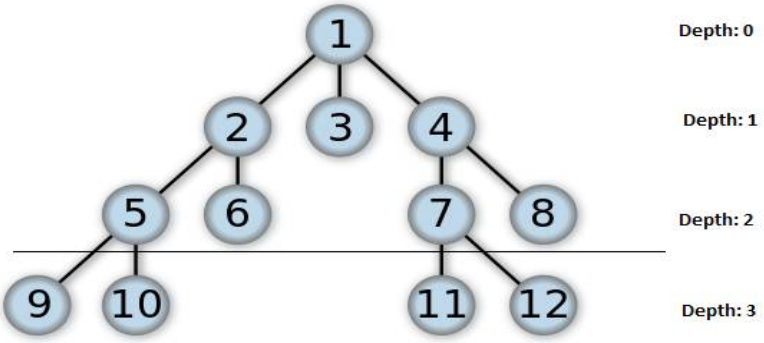

Fig.2. Graph with nodes representing pages and branches representing links and depth

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Let 'd' be the depth of the tree at any point. If d = 2, the search algorithm will scan the links up to a diameter of 2 and backtrack.

For example, using Depth Limited Search on the above tree, i.e. using DFS with a max depth limit would produce the result as: 1->2->5->6->3->4->7->8. Thus, the nodes 9,10,11,12 have been ignored as they fall out of our scope. This approach is very similar to Depth Limited search [6].

Algorithm	Completeness	Time Complexity	Space Complexity
Breadth First Search	Yes	b raise to d+1	b raise to d+1
Depth First	No	b raise to m	b*d
Depth Limited	No	b raised to l	b*l

b = Branching factor of any search tree,
d = Depth of the search tree,
l = max depth limit of tree to control scope.

Using the above mentioned graph traversal algorithms, we can achieve our result of crawling through a website, by taking the home page (root) as the input. Using the HTTP status codes (briefly discussed in the next section) we can check the health of the link and find out which are broken or damaged.

IV. PSEUDO CODE

We use 2 data structures, a queue 'Q' (for using BFS) and a list crawledList which stores all processed links and is initialized as empty

Step 1: We take the root and the max depth limit as input.

Step 2: Check if Q is empty, if yes then terminate, else, proceed further.

Step 3: Remove an element from Q, store it to L

Step 4: Check if L is in crawledList, if yes then go back to step 2, else proceed.

Step 5: Check the HTTP response status of L and append to crawledList.

Step 6: Crawl and find all links in L and add them to Q

Step 7: go to step 2.

Step 8: End.

This flowchart gives an overview of the technique used in crawling through a large number of pages.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

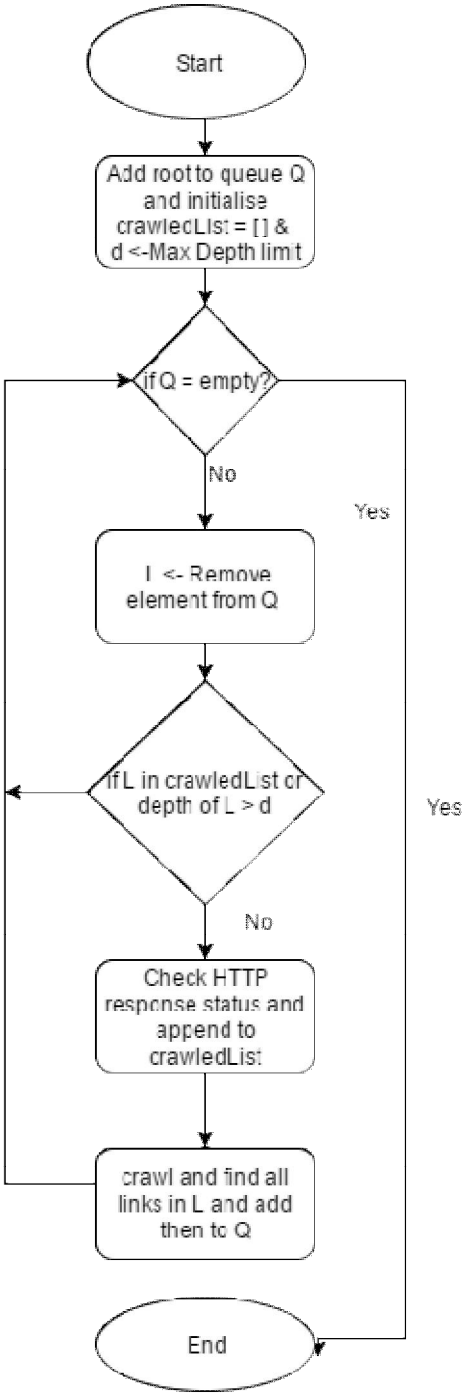

Fig. 3. Flowchart showing program flow in Website Health Analyser

Some web pages have out-links (To Web page W, an out link is a URL appearing in W which points to a Web page of another domain.) in their pages. These links are also added to the queue for processing. But since they belong to a different domain, they must be ignored as they would change the scope of the scan.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

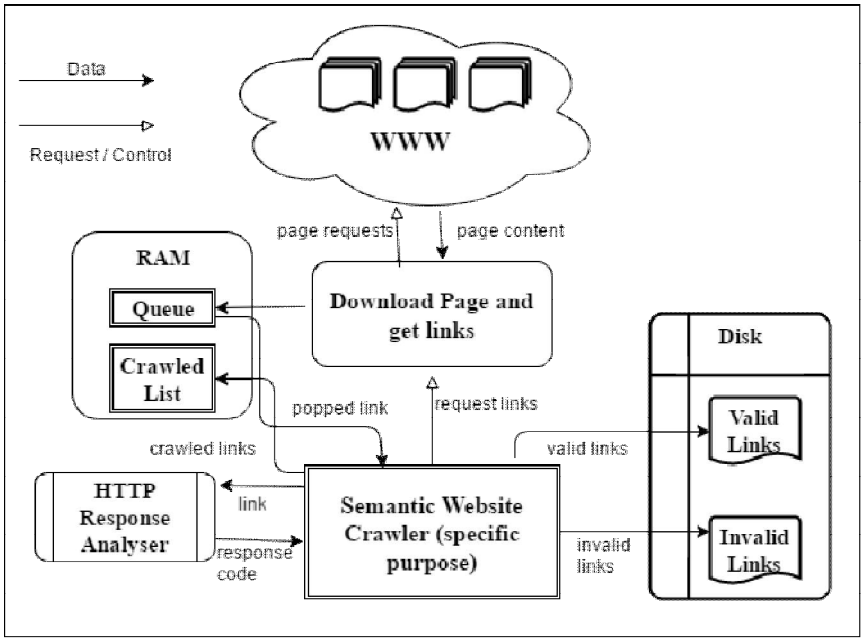

Fig. 4. Flow of data and control in Website Health Analyser

To understand the behavior of the crawler from a bigger perspective, consider the above figure, it shows the flow of data in the entire system.

V. SIMULATION RESULTS

The simulation works as follows. Crawling begins from the input link 'l' whose status is checked first. The entire website can be represented as a tree with variable branching factor 'b'. The root of the tree is l, and the links (children) in l are fetched to be crawled and checked for their health. The number of children in each page equals its branching factor as each of those links will be explored further.

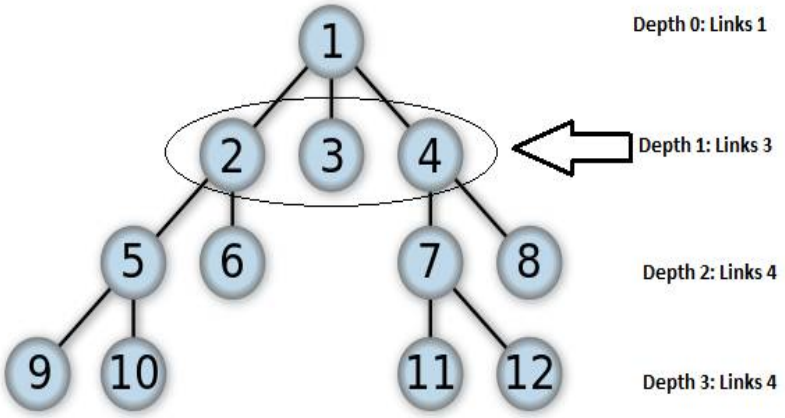

Fig. 3. Graph with nodes representing pages and branches representing links and depth

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

For example, in the above figure, the right side column denotes the depth d of the links along with its count l. For d = 1, we have l = 3. Similarly, for d = 2, we have l = 4.

The depth of the tree simply denotes the distance at which a particular node is from the root. In this case, it denotes the distance of the page or link from the homepage.

On crawling a few websites, and recording the varied l at different depths d gives us the following results.On analyzing websites such as Wikipedia.org and Amazon.in, we achieve the following results.

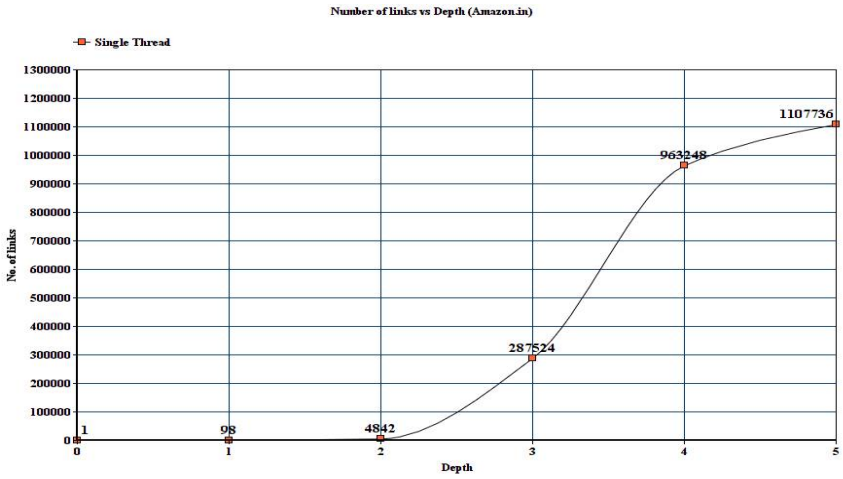

Fig. 6. Graph of Number of Links VS Depth for Amazon.in

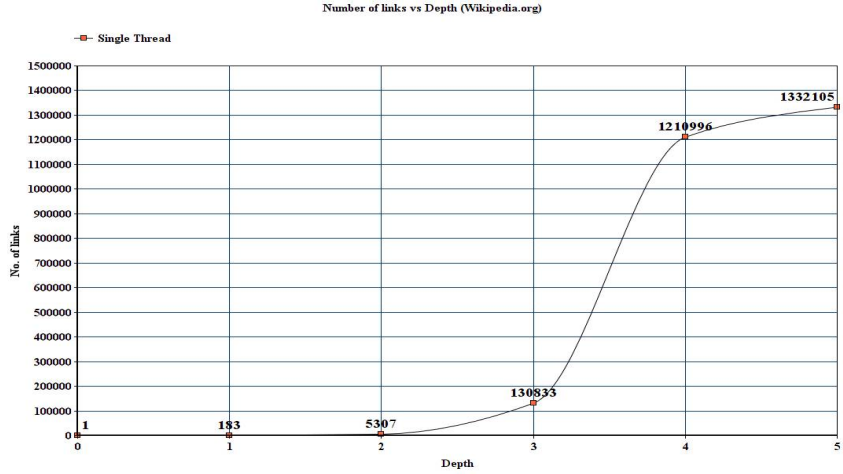

Fig. 7. Graph of Number of Links VS Depth for Wikipedia.org

As we can see from both the figures, there is a steep increase in the no. of links as the depth exceeds a certain value. But as d increases further, we can see that the increase in l becomes less steep.

Also, let dsi-j be the change in the slope from depth I to j. From the figure, in Amazon.in, **ds3-4 > ds4-5**, in Wikipedia.org also **ds3-4 > ds4-5**. So, we can say that increase in the number of children produced by a tree is less as d increases. This is equivalent to saying, that initially several new (not already crawled) links are found that in the parent

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

page, which form its children, but as d increases, most of the links have been visited and the only left are the new ones belonging to the body of the page (link) in consideration. For example, when the homepage of Wikipedia.org, i.e. the root page in crawled, all the links in it, are new. These contain the ones belonging to sections of the homepage such as the header, sidebar,

footer, social icon links etc. But as we move to any of the child of this homepage, the links in the sections such as the header, sidebar, footer, social icons etc are already in the toCrawl data structure(provided that these sections remain unchanged, if not, then they are termed as new), so it makes no sense to add them in it again. The only links added in the child page are the new ones. Thus, even though the number of links in a page independently are high, several are not considered or added to the data structure as they are already present in it. This significantly reduces the computation time for the website developer as well as the search engine crawlers.

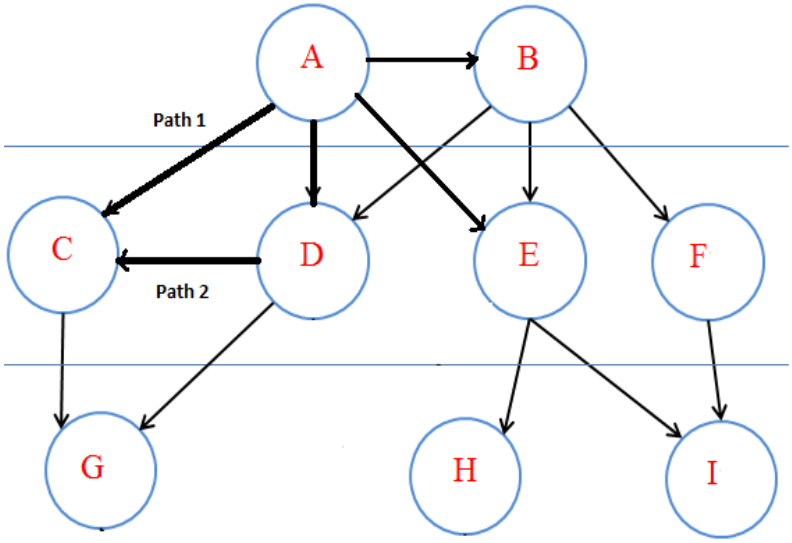

Fig. 8. Graph representing multiple path to a node

Consider node A to be root, and Q as the queue (data structure). As the execution starts the nodes C and D are added to Q, thus Q = [D, C]. If D is crawled then, it would again try to push node C into Q, but this is redundant as Q already has C. Thus we can say, as the depth increases the number of links considered as new and added to the data structure decrease.

But then it becomes important to understand the behaviour after the point where dsi-j is less. Websites such as Amazon, Wikipedia etc contain more than a million links. So, analysing them after a certain depth d, ishighly time consuming. For example, to analyse Amazon.in up to d = 5 consumes more than 12 hours when executed as a single thread. So let us, for the purpose of analysing the behaviour after the dip in the number of new links encountered, crawl a website with much fewer links.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

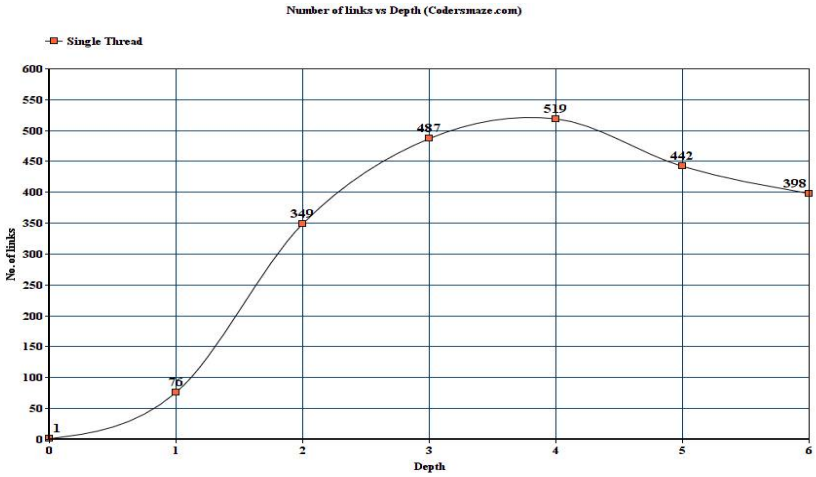

Fig. 9. Graph denoting number of nodes vs Depth for Codersmaze.com

The above figure shows that from d = 4 to d = 6 there is a gradual decrease in the number of newly encountered links. Thus we can see that after a certain depth, the number of new links (not crawled previously or not added in the Q data structure) decrease.

Obviously, the depths at which this behavior is observed differs w.r.t to the size and the layout of the website, but the general behavior remains the

A. *STATUS CODES:*

The Status-Code[12] element in a server response, is a 3-digit integer where the first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role.
Using these response status codes, we can check for the health of the link.

Value	Description	Reference
100	Continue	[RFC7231, Section 6.2.1]
101	Switching Protocols	[RFC7231, Section 6.2.2]
102	Processing	[RFC2518]
103-199	Unassigned	
200	OK	[RFC7231, Section 6.3.1]
201	Created	[RFC7231, Section 6.3.2]
202	Accepted	[RFC7231, Section 6.3.3]
203	Non-Authoritative Information	[RFC7231, Section 6.3.4]
204	No Content	[RFC7231, Section 6.3.5]
205	Reset Content	[RFC7231, Section 6.3.6]
206	Partial Content	[RFC7233, Section 4.1]
207	Multi-Status	[RFC4918]
208	Already Reported	[RFC5842]

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

209-225	Unassigned	
226	IM Used	[RFC3229]
227-299	Unassigned	
300	Multiple Choices	[RFC7231, Section 6.4.1]
301	Moved Permanently	[RFC7231, Section 6.4.2]
302	Found	[RFC7231, Section 6.4.3]
303	See Other	[RFC7231, Section 6.4.4]
304	Not Modified	[RFC7232, Section 4.1]
305	Use Proxy	[RFC7231, Section 6.4.5]
306	(Unused)	[RFC7231, Section 6.4.6]
307	Temporary Redirect	[RFC7231, Section 6.4.7]
308	Permanent Redirect	[RFC7538]
309-399	Unassigned	
400	Bad Request	[RFC7231, Section 6.5.1]
401	Unauthorized	[RFC7235, Section 3.1]
402	Payment Required	[RFC7231, Section 6.5.2]
403	Forbidden	[RFC7231, Section 6.5.3]
404	Not Found	[RFC7231, Section 6.5.4]
405	Method Not Allowed	[RFC7231, Section 6.5.5]
406	Not Acceptable	[RFC7231, Section 6.5.6]
407	Proxy Authentication Required	[RFC7235, Section 3.2]
408	Request Timeout	[RFC7231, Section 6.5.7]
409	Conflict	[RFC7231, Section 6.5.8]
410	Gone	[RFC7231, Section 6.5.9]
411	Length Required	[RFC7231, Section 6.5.10]
412	Precondition Failed	[RFC7232, Section 4.2]
413	Payload Too Large	[RFC7231, Section 6.5.11]
414	URI Too Long	[RFC7231, Section 6.5.12]
415	Unsupported Media Type	[RFC7231, Section 6.5.13][RFC7694, Section 3]
416	Range Not Satisfiable	[RFC7233, Section 4.4]
417	Expectation Failed	[RFC7231, Section 6.5.14]
418-420	Unassigned	
421	Misdirected Request	[RFC7540, Section 9.1.2]
422	Unprocessable Entity	[RFC4918]
423	Locked	[RFC4918]
424	Failed Dependency	[RFC4918]

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

425	Unassigned	
426	Upgrade Required	[RFC7231, Section 6.5.15]
427	Unassigned	
428	Precondition Required	[RFC6585]
429	Too Many Requests	[RFC6585]
430	Unassigned	
431	Request Header Fields Too Large	[RFC6585]
432-450	Unassigned	
451	Unavailable For Legal Reasons	[RFC7725]
452-499	Unassigned	
500	Internal Server Error	[RFC7231, Section 6.6.1]
501	Not Implemented	[RFC7231, Section 6.6.2]
502	Bad Gateway	[RFC7231, Section 6.6.3]
503	Service Unavailable	[RFC7231, Section 6.6.4]
504	Gateway Timeout	[RFC7231, Section 6.6.5]
505	HTTP Version Not Supported	[RFC7231, Section 6.6.6]
506	Variant Also Negotiates	[RFC2295]
507	Insufficient Storage	[RFC4918]
508	Loop Detected	[RFC5842]
509	Unassigned	
510	Not Extended	[RFC2774]
511	Network Authentication Required	[RFC6585]
512-599	Unassigned	

B. ***Necessary features of a crawler***

There are a few necessary features that any web crawler must have to efficiently function and gather as much data as possible in less time:

- **The crawler should not overload the originating servers**: Many servers may have hostile approach to crawlers. If any site is hammered too hard, it'll blacklist the IP address(s) of the crawler. Once blacklisted, the crawler will be throttled to 1 query per minute or less, effectively making it extremely difficult and time consuming to crawl the site.
- **Avoid Circular references**: The crawler must have a way to notice if it has already seen a given page during the run. Many sites have multiple links to the same content; also, it must avoid getting caught in a chain of circular references this way.
- **It should strip the parameters**: They can have a mild or profound effect on page content; in extreme cases, they can be the basis of a spider trap. The Crawler should be able to detect infinite loop traps (Spider traps) with eats up the crawler's efficiency [10].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

- **JavaScript based Page Rendering**: With the growing number of Single Page Webapplications using JavaScript Frameworks, the crawler must be able to handle pages being rendered dynamically and ensure that all its JavaScript is executed before parsing the page. There are a few crawlers with depends on Selenium to this [4].
- **Distributed Crawling**: When crawling huge websites, if your crawler can be distributed in multiple computers (with different IP Address) it's crawling speed can be increased. But we must keep in mind that excessive distribution ends up being a DDoS on the target website [8].

C. *Optimizations and why are they necessary*

As we can see from the above figures, the time taken for the crawler to scan the websites at higher depths is very high. This is a serious drawback. The reason for it is, the crawler works as a single process running a single thread.

Now, if an application runs on a quad-core machine, it has only on one thread and cannot take advantage of the complete CPU. So to fully utilize CPU it is more productive to divide task on multiple threads [5].

So, to optimize this i.e. decrease the crawling time, we can make use of 'N' number of threads, where N depends on the processing power 'P'. Higher the available memory and processing power, enables the use of higher N. This in turn makes the execution faster [11].

Multiple threads can exist within one process, executing concurrently and sharing resources such as memory needed during the execution.

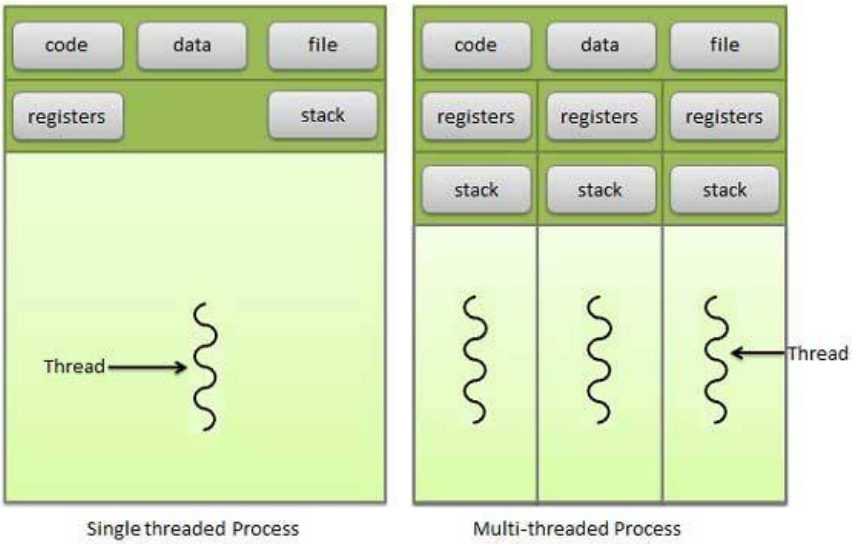

Fig.10. Structure of memory for a single thread and a multi thread process

To check this theory, the website Wikipedia.org was crawled. Let 'N' be the number of threads used, 'ls' be the number of links scanned in a single thread environment and 'lm' be the number of links scanned in a multi-thread environment. The number of links generated in 1, 2 and 3 mins each are plotted, in both single and multithreaded environments.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

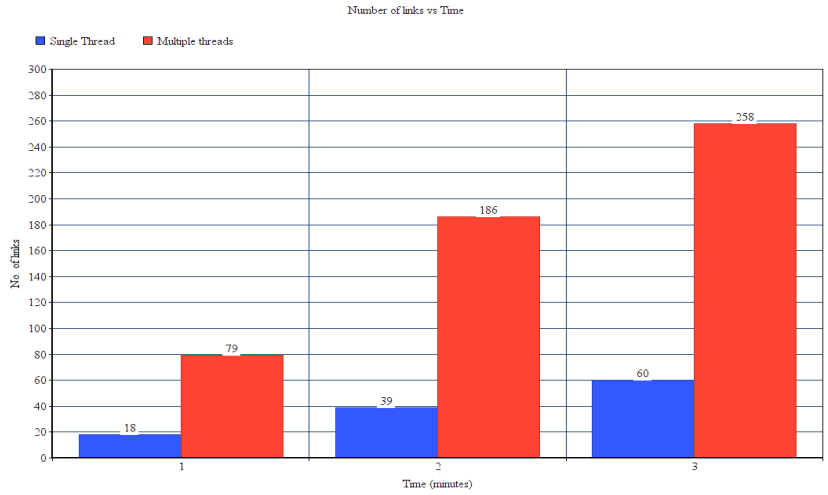

Fig. 11. Comparison between single thread and multithread

If 5 threads are used, i.e. n = 5, we can see that,
lm>>ls.
Of course,**lm ≠ n*ls**, but **lm ≤ n*ls**,
This is mainly because the CPU does not work in an ideal scenario where all threads waste no time. Some fraction of time is also spent in time-slicing, and securing the critical sections to enforce concurrent execution.
Thus we can see that increasing the number of threads, or in other words, adding a mechanism of concurrent execution, we can achieve better results.

VI. CONCLUSION AND FUTURE WORK

The techniques described in this article ensure that the users do not have to click on unnecessary and broken links by eliminating link-rot [9], thereby improving user experience. Also the SEO ranking of a website can be effectively maintained as non-functional links traced and eliminated using their HTTP status codes.

REFERENCES

1. P. Ravi Kumar, Ashutosh Kumar Singh, Anand Mohan. "Efficient methodologies to optimize Website for link structure based search engines".
2. Shutong Cheng, Congfu Xu, Hongwei Dan. "Website Structure Optimization System Model and Algorithms".
3. AtsuyukiMorishima, AkiyoshiNakamizo, Toshinari Iida, Shigeo Sugimoto, Hiroyuki Kitagawa. "A Tool for the Automatic Correction of Broken Web Links".
4. http://openmymind.net/2012/5/30/Client-Side-vs-Server-Side-Rendering. "Java Script Based Page rendering".
5. G. Havas, S.A.M. Makki. "Distributed Algorithms for Constructing a Depth-First-Search Tree".
6. Paolo Sipala, Harold S. Stone. "The average complexity of depth-first search with backtracking and cutoff"
7. JieXia ,Wanggen Wan ,Renzhong Liu ,Guodong Chen ,Qing Feng. "Distributed web crawling: A framework for crawling of micro-blog data"
8. G. Geetha, Sawroop Kaur Bal. "Smart distributed web crawler".
9. https://en.wikipedia.org/wiki/Link_rot. "Link Rot – Wikipedia".
10. https://en.wikipedia.org/wiki/Spider_trap. "Spider Trap - Wikipedia"
11. http://www.w3ii.com/en-US/operating_system/os_multi_threading.html "Operating System Multi-Threading"
12. https://en.wikipedia.org/wiki/List_of_HTTP_status_codes. "HTTP Status Codes".
13. https://en.wikipedia.org/wiki/PageRank. PageRank algorithm, Google.

BIOGRAPHY

Jay Doshiand **JuhiDuseja** areB.Tech. Students in the Department of Computer Science & Information Technology at VeermataJijabaiTechnological Institute(VJTI), Mumbai, India expected to graduate in the year 2016-2017.Their research interests are Cyber Security, Data Structures, Algorithms, Large Scale Parallel Computing, Data analysis etc.