Volume 3, No. 11, November 2012
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

TEXT CONTENT BASED WEB PAGE REFRESH POLICY

Vidushi Singhal[1] and Sachin Sharma*[2]
Department of Computer Applications, Manav Rachna International University, Faridabad
Vidushi.fit@mrei.ac.in, sachin.fbc@mriu.edu.in

Abstract: Internet is actively used for the exchange of information. Working of WWW starts with the entering of a URL in address bar or by clicking on a link. It navigates us to a given location. The data is stored in the web servers which serve as the backbone of the World Wide Web. People upload the web pages and updating the new web pages very frequently. There is a frequent change in the content of the web page. A crawler recursively keeps on adding new URLs to the database repository of a search engine. It might possible that after downloading a particular web page, the local copy of the page residing in the web repository of the web pages becomes obsolete as compared to the copy presented on the web. Users are often not only interested in the current web page contents but also in changes in it. Hence it becomes necessary to develop an efficient system which could detect these changes efficiently and in the minimum browsing time. Therefore there should be some provision to update the database at a regular interval. Once we decide to update a page it should be ensured that minimum resources are used. Various tools and services are also available which can be used to detect these changes. In this paper, by taking the advantage of previous work a new approach is discussed to derive certain parameters, which can help in deriving the fact that whether the data has changed or not.

Keywords: Search Engine, Search Engine Optimization, Web Repository, Parser, Web Scheduler, Web World

INTRODUCTION

Definition:

A Computer Program that is used to retrieve information as per user requirement is known as search engine. A search engine is a tool which is used to search data on internet. Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine that will index the downloaded pages to provide fast searches. The job of a web browser is to make request for the web page to the web server and the server serves it with required web page. It then displays the web page as rendered by HTML, XML and other web languages used by the page. A Computer Program that is used to retrieve information as per user requirement is known as search engine. Crawler-based search engines use high level pieces of software called spiders or robots to search and index web pages. These spiders works constantly, crawl around the web, locate web pages, and take snapshots of those pages to be cached or stored on the search engine's servers. They are so designed that they can follow links from one page to another and from one site to another.

A crawler based search engine is dependent on three activities (1) Crawlers which are used to move the control to a Website (2) Index; it is the part of the search engine which is used to create the index of all links searched from the crawlers. All the information collected through the crawler is firstly stored in a central repository & then its index is formed. The purpose of index is to provide user with relevant data. (3) Search engine software is used to check the sites for updating with a defined frequency. Search engine indexing collects, parses, and stores information to facilitate fast and accurate information retrieval. The name for the process in the context of search engines designed to find web pages on the Internet is Web indexing. Web Indexing is done on the data stored in database. The whole data is indexed in a particular order as storing the data as per the alphabetical order. Query component reads the user query & locate the data from data repository. After locating the data from the query component it should be available to the user. But the data retrieved from it is too large & most of the data is irrelevant for the user. So it is necessary to discard some data & provide the user with relevant data only. This is done by page ranking. Pages that we believe are important receive a higher Page Rank and they are more probability to appear at the top of the search results. Search Engine Optimization is used to improve the visibility of a website in the list of search engine. It is done by improving the rank of a web site. Higher the rank means higher the visibility. Higher the visibility means we will find that page in the initial links of the pages search by the crawler. Search Engine Optimization revolves around two factors. From which one is on site optimization which says that everything is controlled on our website for example title tag, links, structure etc... And another is off site optimization which include everything else means links from the other site.

A White hat SEO is called an "ethical stance" that opposes the abuse of electronic media. Make the right thing as Optimize the text. Implement accessibility standards etc. A Black hat SEO is the opposite of White Hat SEO, this attempt to run an intrusion on these, is considered an ethical SEO strapped. We can use low quality links or unwanted as pornographic sites, pharmacists, etc. Those who practice Black Hat SEO, inflate page rank placing a link on a page with high page rank, but if this is removed from the page of the page fall it reduces it rank. Gray hat SEO refers to those practices that are between both black hat & white hat SEO. Using both methods, as seen as more beneficial for the page. From the beginning, a key motivation for designing Web crawlers has been to retrieve web pages and add them a local repository to form a database. This repository is used as database by several applications as Web Search Engine. In its simplest form a search engine begins from a seed URL and then uses the external links present in this page to crawl the other page. To use the external link, they are stored in Frontier and to search for the new links a parser is used.

Web scheduler fetches a link from the frontier for the user. The process repeats with the new links present in the frontier so that it can offer more external links to follow, until a sufficient number of pages are identified or some higher level objective is reached. Behind this simple description lies a host of issues related to network connections which user never bothers about. Thus a crawler recursively keeps on adding new URLs to the database repository of a search engine. Several major problems affect non-cooperative web crawlers on the web. The first problem is that web crawlers do not maintain a high-degree of freshness. The second is that multiple crawlers can redundantly crawl the same regions of the web. The third is that with the proliferation of web crawlers comes increased contention for shared network resources. Here in this paper we are discussing about the freshness problem. It might possible that after downloading a particular web page, the local copy of the page residing in the web repository of the web pages becomes obsolete as compared to the copy presented on the web. Therefore there should be some provision to update the database at a regular interval. Once we decide to update a page it should be ensured that minimum resources are used. For that particular purpose we only refer to those pages which have been changed. Here in this paper we are discussing a fresh approach to update a page presented in web repository.

Related Work:

[1] Describes an efficient Web page change detection system based on three optimizations that were implemented on top of the Hungarian algorithm, which is employ to compare trees that correspond to HTML Web pages. The optimizations attempt to stop the comparator algorithm that employs this $O(n^3)$ algorithm before it completes all its iterations based on criteria having to do with properties of HTML and heuristics. Analysis and experimental results prove the effectiveness of these optimizations and their ability to render $O(n^2)$ performance, where n denotes the number of nodes in the tree. In this complete system was implemented and used to carry out the performance experiments. This system includes functionalities and interfaces for processing user requests, fetching Web pages from the Internet, allowing users to select zones in Web pages to monitor, and highlighting changes on the Web pages being monitored. [2] Describes web page change detection system based on Signature of Node corresponds to HTML pages. In this proposal first HTML document is filtered into XML structure document and then transforms XML pages to trees using DOM.

The node signature comparison algorithm is developed to compare the trees of old web page and modified web page to find the changes in the web page. This system highlights changes of content i.e. deletion and addition of text and attribute changes i.e. font change, caption change, color change etc. and highlight the changed part in red color and displays to the user. This algorithm gets result faster as it does not search the sub tree if that sub tree does not have any changes. Focused crawlers aim to search only the subset of the web related to a specific topic, and offer a potential solution to the problem. The major problem is how to retrieve the maximal set of relevant and quality pages. [3] Propose an architecture that concentrates more over page selection policy and page revisit policy. The three-step algorithm for page refreshment serves the purpose. The first layer contributes to decision of page relevance using two methods.

The second layer checks for whether the structure of a web page has been changed or not, the text content has been altered or whether an image is changed. Also a minor variation to the method of prioritizing URLs on the basis of forward link count has been discussed to accommodate the purpose of frequency of update. And finally, the third layer helps to update the URL repository. [4] Presents a new algorithm for the structural as well as content change detection. For better results tree has been designed for the corresponding web pages. Proposed change detection algorithm is based on assigning hash value to each leaf node and tag value to the non leaf nodes. Bottom up approach has been used for assignment.

The level of each node has been used to find hash values and modification in a node. To efficient retrieval and monitoring the changes made in web pages and compare the difference between refreshed page and old page efficiently that too in minimum browsing time, an effective monitoring system for the web page change detection based on user profile is needed. The web page change detection system can be implemented by using various Tools or Algorithms. [5] Shows a comparative study of various algorithms and tools in terms of their complexity, memory and tree structure. One of the algorithms introduces three running time optimizations that control the operations of the Hungarian algorithm. [6] Describes an efficient Web page detection approach based on restricting the similarity computations between two versions of a given Web page to the nodes with the same HTML tag type. Before performing the similarity computations, the HTML Web page is transformed into an XML-like structure in which a node corresponds to an open-closed HTML tag. Analytical expressions and supporting experimental results are used to quantify the improvements that are made when comparing the proposed approach to the traditional one, which computes the similarities across all nodes of both pages. It is shown that the improvements are highly dependent on the diversity of tags in the page. That is, the more diverse the page is the greater the improvements are, while the more uniform it is, the lesser they are.[7] Discuses a fresh approach for parallel crawling the web using multiple machines and integrates the trivial issues of crawling also.

A major part of the web is dynamic and hence, a need arises to constantly update the changed web pages. In this paper a three-step algorithm for page refreshment is used. This checks for whether the structure of a web page has been changed or not, the text content has been altered or whether an image is changed. [8] Proposed an algorithm uses to discover and detect changes to the Web pages efficiently. A solution for searching new information from the web page by tracking the changes in Web document's structure has been discussed. In the methodology section, the author present the algorithm and technique useful for detecting web pages that are changed, extracting changes from different versions of a Web page, and evaluating the significance of web changes. Our algorithm for extracting web changes consists of three steps: document tree construction,

document tree encoding and tree matching for the detection of two types of changes basically - structural changes and content changes. In [10] author is proposing two methods for refreshing the page by comparing the page structure. First method compares the page structure with the help of tags used in it. And second method creates a document tree compare structures of pages.

EXISTING METHOD

Existing method is using client server architecture. Server gives instruction to the client crawlers and client crawlers follows the server instruction to download the data. In the existing method [7] author gives a formula to find the fact that whether we need to update the repository or the page presented on fresh link & page in the web repository are same. To calculate the freshness of a page it can be compare in three steps: (1) comparison of page structure (2) comparing of page contents (3) comparing of images. In the existing method following formula is used to compare the text contents of web pages:

$$\frac{\sum (\text{frequency}) * \text{ASCII}_{\text{symbol}}}{\text{Distinct Character Count}}$$

Author assigns a code to all text content appearing in a web page. At the time of page updating, comparison will be made to the pages and if any change in those web pages is detected for the actual copy on the web as compared to the local copy, those pages will be refreshed or re-crawled. In the existing method user is applying the formula to whole page to find the difference between two pages. And ASCII code is using to code the keywords. In this proposed method we are using UNICODE rather than ASCII code and rather than comparing the whole page we are comparing both the pages till the first unmatched value.

PROPOSED METHOD

Working of a Basic Web Crawler:

Web crawlers retrieve web pages from seed URL and add them a local repository to form a database. This repository is used as database by several applications; a search engine begins its process from a seed URL and then uses the external links present in this page to crawl the other page. To use the external link, they are stored in Frontier and to search for the new links a parser is used. Web scheduler fetches a link from the frontier for the user. The processing of a basic crawler is shown in fig 1. The process repeats with the new links present in the frontier so that it can offer more external links to follow, until a sufficient number of pages are identified or some higher level objective is reached. Behind this simple description lies a host of issues related to network connections which user never bothers about. The algorithm of the typical web crawler is as given below:

a. A crawler starts it's searching from a seed URL provided by the user.
b. Downloader downloads the page and parser parse the web page to find the new links.
c. All these new links are stored in a queue.
d. Web scheduler retrieves a link from queue Frontier.
e. For each retrieved link it repeats the steps 2 to 4 until a sufficient number of pages are not downloaded.
f. These downloaded pages are stored in web repository.

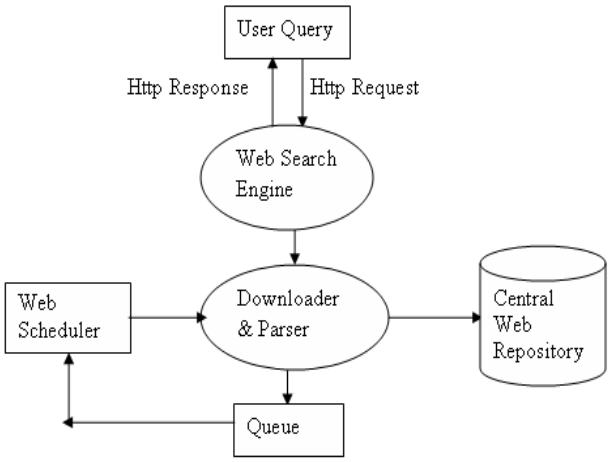

Figure: 1.1 (Working of a basic web crawler)

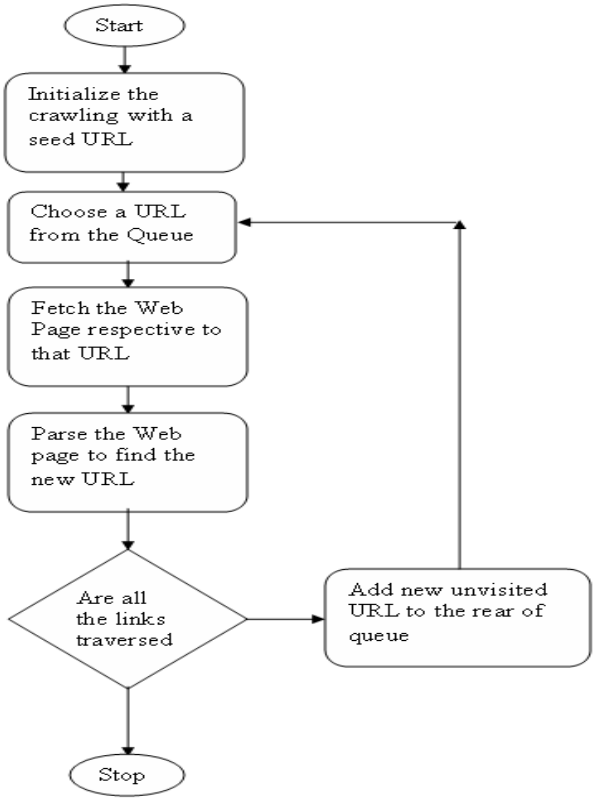

Figure 1.2 (Flow chart of a web crawler)

Freshness Problem:

About 60% of the data on the net is dynamic [9] in nature it means that the data keeps on changing every moment. Assume at time t1 a set of web objects are considered such that state of all those web objects is captured through an event. Between time t1 and t2 some subset of these objects may have changed due to some internal or external forces. The set of objects that have changed between time slot t1 and t2 are known as web event. And the fraction of web objects that have not changed between time t1 and t2 defines the freshness of the page. Crawlers that try to keep a fresh copy of web page must visit the web page again and during the course of time. If a page does not change between time t1 and t2 we said that crawlers have wasted the resource otherwise it returns a page that need to be changed with the updated copy of it.

Relevancy Check:

Before downloading a web page server must check for its relevance because an irrelevant page just wastes the resources. To check the relevance server parse the whole page contents, collects the keywords and match these keywords with the keywords of actual page presented in web repository. If a web page found to be an appropriate candidate for downloading, it is sent to the client for actual downloading but to maintain the eminence of crawling only important pages should be downloaded. To define the importance of a web page server maintains a priority value for each URL. A page with highest priority is treated as the most important page. To calculate the priority following formula [27] is used.

Priority Value = Aging Factor * (Forward Link – Backward Link)

To calculate the value of Forward Link server just parse the complete page and count the number of links presented on it and to calculate Backward Link server uses the existing database by the client crawler and calculates number of pages refer to this page. Initially Forward Link holds more than Backward Link because the database would be in growing stage. But as the time passes the value of Backward Link also increases. Server sorts the list of URL collected from the client machine as per the descending order of their priority value. It then sends the sorted list of URL to client machine for actual downloading.

Page Refresh Algorithm:

Several major problems affect non cooperative web crawlers on the web. A major problem is that web crawlers do not maintain a high degree of freshness. We consider a database fresh when the database has more up to date elements. For instance, if database X has 20 up-to-date elements out of 25 elements and database Y has 15 up-to-date elements out of 25 elements, we consider X to be fresher than Y. In this paper, by taking the advantage of previous work a new approach is discussed to derive certain parameters, which can help in deriving the fact that the whether the page is fresh or same as the page present in web repository. These parameters are found out at the time of page parsing. Whenever a client fetches a URL which is already present in the data repository, it compares the new web page with the existing one with the help of matching parameters. If matching parameters shows differences between two copies it update the repository with the new copy otherwise it discard the new copy & continue with the older one. According to the previous approach the whole page is parsed & the formula is applied on the whole page. But as per our approach rather than comparing whole page we just compare the documents till the first unmatched value. Because if a single value is found to be different, page is declared to update & the older copy will be replaced by the fresh copy.

Secondly, author proposes the ASCII encoding technique in the formula. ASCII is a seven bit encoding technique which supports only 128 characters frequently used in American English like language. Here we are using UNICODE which has developed a coding system that includes every written alphabet in existence. In our approach we assign a code to the text content appeared in a web page. We did the process of assigning code word by word. Whenever we found the difference between two words we came to the conclusion that page has been updated otherwise comparison is made till the end of the web page. The formula for text coding is as follows:

$$\frac{\sum (\text{Character frequency}) * \text{UNICODE}_{\text{Symbol}}}{\text{Distinct Symbol Count}}$$

Where

Frequency is the sum of the occurrence of a character in a web page.

UNICODE symbol defines the code of a character.

Distinct Symbol Count is the count of different symbols appearing in the page.

For example consider the following text content

> Internet is actively used for exchange of information. People upload the web pages and updating the new web page very frequently. There is a frequent change in the content of the web page hence it become necessary to develop an efficient system which could detect these changes efficiently and in the minimum browsing time. So as to achieve this we compare the old web page and the new web page. Changes in a web page can be detected with the use of various algorithms. Various tools and services are also available which can be used to detect these changes.

Figure 1.3 (Initial Web Page)

Even single changes in above text lead to a significant changes in the result. Consider the next paragraph which has been changed from the previous one. Words in bold are changed from the above paragraph.

> Internet is actively used for exchange of "**data and** information". People upload the **new** web pages and update the older web page very frequently. There is a **rapid** change in the content of the web page hence it become **compulsory** to develop an efficient system which could detect these changes efficiently and in the minimum browsing time. **To detect the changes older copy from web repository is compared with the fresh copy presented on fresh link** Changes in a web page can be detected with the use of various algorithms. Various tools and services are also available which can be used to detect these changes."

Table 1.1 (Word by word comparison)

Initial Web Page (Fig 1.3)					Web Page After Changes (Fig 1.4)				
Word: Internet					**Word: Internet**				
Character	Character Frequency	Unicode Symbol	Distinct Character Count	Result from Formula	Character	Character Frequency	Unicode Symbol	Distinct Character Count	Result from Formula
I	1	73	5	168.2	I	1	73	5	168.2
n	2	110			n	2	110		
t	2	116			t	2	116		
e	2	101			e	2	101		
r	1	114			r	1	114		
Word: is					**Word: is**				
i	1	105	2	120	i	1	105	2	120
s	1	115			s	1	115		
Word: actively					**Word: actively**				
a	1	97	8	108.1	a	1	97	8	108.1
c	1	99			c	1	99		
t	1	116			t	1	116		
i	1	105			i	1	105		
v	1	118			v	1	118		
e	1	101			e	1	101		
l	1	108			l	1	108		
y	1	121			y	1	121		
Word : used					**Word : used**				
u	1	117	4	108.3	u	1	117	4	108.3
s	1	115			s	1	115		
e	1	101			e	1	101		
d	1	100			d	1	100		
Word: for					**Word: for**				
f	1	102	3	109	f	1	102	3	109
o	1	111			o	1	111		
r	1	114			r	1	114		
Word: exchange					**Word: exchange**				
e	2	101	7	119.3	e	2	101	7	119.3
x	1	120			x	1	120		
c	1	99			c	1	99		
h	1	104			h	1	104		
a	1	97			a	1	97		
n	1	110			n	1	110		
g	1	103			g	1	103		
Word: of					**Word: of**				
o	1	111	2	106.5	o	1	111	2	106.5
f	1	102			f	1	102		
Word: information					**Word: information**				
i	2	105	8	148.8	i	2	105	8	148.8
n	2	110			n	2	110		
f	1	102			f	1	102		
o	2	111			o	2	111		
r	1	114			r	1	114		
m	1	109			m	1	109		
a	1	97			a	1	97		
t	1	116			t	1	116		

Word: People					Word : data				
P	1	80	5	122.6	d	1	100	3	136.7
e	2	101			a	2	97		
o	1	111			t	1	116		
p	1	112			-	-	-	-	-
l	1	108			-	-	-	-	-

The table 1.1 shows the word by word comparison from both the fig 1.3 and fig 1.4. As soon as we find the first difference we declare that page needs to re-crawl. This approach gives us following benefits:

a. It works for all languages & all type of special characters.
b. It gives unique code for each & every content present on web page.
c. The formula is designed in such a way that even a single change in the web page can be caught easily.
d. Comparison is made till the first unmatched value. It saves the resources in terms of time, memory etc.
e. Only the text of that page is compared, there is no need to create indexed structure, hence it saves ample amount of memory.
f. UNICODE values have been used in the formula because in UNICODE each symbol has a distinct representation so it does not lead to any type of ambiguity.

CONCLUSION & FUTURE SCOPE

Size of web is increasingly rapidly. Crawler forms the back-bone of applications that facilitate Web Information Retrieval. It is a challenge for the web crawlers to keep the user with the updated information. To check the freshness of a web page we need to compare the page with its copy presented on link. In this paper we have discussed a fresh method to compare the pages. There can be three steps to compare these two pages. We can compare the structure of page, contents of the page and images of page. Here in this paper, we are comparing the text content to find the differences among pages. UNICODE is used to code the text content. Unicode has developed a system in which each & every symbol can be coded. So this approach can be used for multilingual sites.

Though the implementation of proposed method gives the benefits in terms of time, storage etc. still several interesting issues have been identified which needs to be addressed in future. For example we can work on images to detect the changes. We can more efficient ways to compare two pages which cost minimum in terms of memory, bandwidth etc.

BIBLOGRAPHY

[1]. Imad Khoury, Rami M. El-Mawas, Oussama El-Rawas, Elias F. Mounayar, Hassan Artail, "An Efficient Web Page Change Detection System Based on an Optimized Hungarian Algorithm", IEEE, Vol. 19, No. 5, May 2007

[2]. H.P. Khandagale, P. P. Halkarnikar, "A Novel Approach for Web Page Change Detection System", Vol. 2, No. 3, June 2010

[3]. Swati Mali, B.B. Meshram, "Focused Web Crawler with Page Change Detection Policy", Proceedings published by International Journal of Computer Applications, IJCA, 2011

[4]. Srishti Goel, Rinkle Rani Aggarwal, "An Efficient Algorithm for Web Page Change Detection", International Journal of Computer Applications, Vol 48, No.10, June 2012

[5]. Srishti Goel, Rinkle Rani Aggarwal, "Comparative Analysis of Webpage Change Detection Algorithms", IJREAS, Vol 2, No. 2, Feb 2012

[6]. Hassan artail, michel abi aad ," An enhanced Web page change detection approach based on limiting similarity computations to elements of same type", Journal of Intelligent Information Systems, Vol 32, No 1, February 2009

[7]. Divakar Yadav, A. K. sharma, J. P Yadav," Parallel crawler architecture and web page change detection" WSEAS, Vol 7, No 7, July 2008.

[8]. Divakar Yadav, A. K. sharma, J. P Yadav," Change Detection in Web Pages ", ICIT, 10th International Conference, Dec 2007

[9]. Michael K. Bergman," White Paper: The Deep Web: Surfacing Hidden Value Availability", Journal of electronic publishing, Vol. 7, No. 1, August 2001.

[10]. Vidushi Singhal, Sachin Sharma, "Crawling the web surface databases", IJCA, Vol 52, No 19, August 2012

