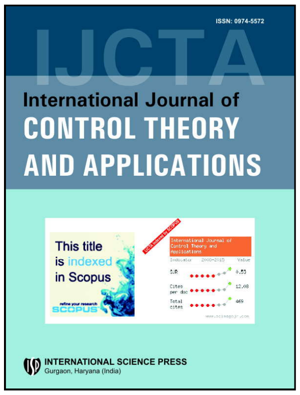

International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Volume 10 • Number 14 • 2017

Change detection of web page in Focused Crawling system

Naresh Kumar[1], Devvrat Tyagi[2], Shivank Awasthi[2] and Jyoti Mor[3]

[1] *Assistant Professor, Department of Computer Science and Engineering Maharaja Surajmal Institute of Technology, New Delhi, India*

[2] *Student, Bachelor of Technology Department of Computer Science and Engineering* Maharaja Surajmal Institute of Technology, New Delhi, India

[3] *Assistant Professor, School of Engineering and Technology, Ansal University ,Gurgaon, India*

Abstract: Ample amount of new information is updated each day on the web which results in creating new web pages or updating existing web pages. Humongous size of the web makes it difficult for the web crawler to detect these changes on regular basis. But for the better experience of the user collected URLs must be updated on regular basis. So, a separate module should be incorporated in the crawling system to detect these changes and this module is named as freshness checker module. In this paper, freshness checker module can detect the structural change, content change and image change in the web page. This module helps in maintaining the freshness of repository and also reducing the consumption of bandwidth.

Keywords: Web Crawlers, Page Revisit policy, Focused Crawlers, Web Page Structure.

1. INTRODUCTION

Web has become one of the basic necessities in human life since its advent. Every human in present era uses the web daily in direct or indirect form. Due to this the size of the web is increasing day by day. So for the better end user experience, search engines are using different types of web crawler at different geographical locations to maximize the coverage area and the amount of information on the web.

Web crawler is a program in search engines which collects the web pages from the World Wide Web. The process of crawling begins with the set of seed URLs having large number of outgoing links. These outgoing links are downloaded and URLs are extracted from them to be traversed further. The downloaded URLs are stored in repository .So there is need arises to keep the repository updated. This problem comes with the idea of page revisit policy to be linked with the crawling system. Page revisit policy is used to keep the freshness of the repository as high as possible.

In this paper a module called freshness checker is used as the revisit module with the focused crawling system. This module is used to detect the changes in the web page. This module can detect page structural change, page content change and image change.

2. PROBLEM STATEMENT

How should crawler update the stored web pages in database? Which revisit policy should be planned for efficient crawling system?

Web crawling is a continuous process and it takes several days or even months to crawl the fraction of the web. But web is very dynamic in nature and it keeps on changing. Some web pages change very frequently within days while some are kept unchanged for several days. So to keep the collection of downloaded webpage updated they should be revisited regularly. It is very hard to synchronize page revisit policy [1] with the crawling process as the size of the web is very large.

According to the studies, for a short span of time the ratio of change in web content with the total web content is significantly smaller. In [2] authors from their study of 720,000 pages of dataset concluded that around 70 percent of the web remains unchanged for 30 days but some of the domains like .com changes very frequently. These studies [1] [2] show that page revisit policy should be based on domain of the web page. But there are chances that page does not change when it is revisited based on the type of domain.

On the basis of such considerations, the algorithm uses a different color image multiplied by the weighting coefficients of different ways to solve the visual distortion, and by embedding the watermark, wavelet coefficients of many ways, enhance the robustness of the watermark.

3. RELATED WORK

So for the efficient web crawler, need arises for a page revisit policy to be incorporated in crawling system. Some of the proposed page revisit policies are as follows:

- Revisit policy in [3] is based on the frequency of change in content. Refresh rate of web page is computed dynamically according to which URLs are distributed among several URL queues. URLs from the queues are selected at every $n^2 * 20\ \mu$ sec where n (1…..n) is number assigned to the queue based on the refreshing time of URL.
- In [4] algorithm is proposed in which page score is used to depict which URL has to visit first. Page score of the URL is calculated as the ratio of page rank with the age of the page where age of page is the difference in the present time and last modified time. Pareto's principle of 80-20 is applied in which top 20% URLs having high page score are kept in B collection and the frequency of revisiting these pages are high than that of A collection URLs.
- Authors in [5] suggest a technique to maintain the freshness of crawled pages stored in the repository. The proposed technique encompasses compression techniques to store crawled pages. The compression utility has been found by surveying different techniques of compression for the best compression ratio, its compression efficiency is about 74.5%.
- A page update policy has been proposed in [6] that minimizes the carbon footprints and energy consumption of servers. This is achieved by lowering the number of requests to web servers. The approach calculates the optimal policy for crawling on the basis of values of page staleness and greenness indicators.
- In [7] web page changes are broadly classified in four major categories namely 1.Content /Semantic changes 2. Presentation/Cosmetic changes 3.Structural changes 4.Behavioral changes. The node of HTML tree structure of web page should contain ID, CHILD, PARENT, LEVEL, CONTENT VALUE information. Structural change is depicted when any new tag is created or deleted then the initial set and later set have different values. Content change is recognized by comparing the product of R.M.S value and ASCII value of the characters in text.

- Authors in [8] proposed parallel domain focused crawler in which crawler is sent to the remote site to check for any change in web page. Crawler hand compares new ASCII count of web page with the old one stored in database. If the compared value is same as of stored value then it rejects the new page otherwise accepts.
- In [9][10] [11]algorithm for web page change detection has been proposed based on structural change, content change and image change. Page structural change is depicted by the value stored in two strings. String 1 store characters at first position in the tag and String 2 stores last character of the tag. If the compared values are same then process proceeds further to find the changes in text by comparing the text code which is computed by the finding the sum of ASCII code of characters. Method for detection of change in image of web page is also proposed. In proposed method image is scaled at standard size of n*n and converted to two tone image. After several computations ival of an image is calculated and stored.

This literature survey draws the inference that:

- In [7] [8] there is no method proposed to recognize the change in image while revisiting the web page. Proposed approach fails in case when the image URL link is kept same and the image changes. This change cannot be encountered by the proposed technique.
- In [9] [10] [11] image change detection method is proposed but the time intricacy is high. The image change detection technique incorporates five steps and several iterations to reduce the image to a single value. In case a web page contains more than one image than the complexity increases.
- From HTML5 onwards the change in the style and structure of the web page is done in CSS files linking to the HTML page. Proposed technique of page structural changes in [7][9][10][11] fails to detect changes in the linked CSS files as they only check for the change in HTML page.

4. PROPOSED APPROACH

As the size of the web is increasing enormously focused crawling system is gaining popularity. In focused crawling system [12] [13]only relevant pages specific to certain topics are crawled which helps in increasing crawler efficiency. In this paper, proposed revisit policy is linked with the focused crawling system.

4.1. Architecture

The architecture shown in fig. 1 can be categorizes in two levels:

1. Focused crawler Mechanism (1st level)
2. Freshness Checker Mechanism (2nd level)

4.2. Focused Crawler Mechanism–

This is 1st level as shown in figure1. This is the basic level in which crawling of the web is done and the collected web pages are kept in the repository. Working of the Focused Crawler can be discussed as:

- The crawling process starts with the set of seed URLs which are transferred in Initial Set and later sent to the Trainer.
- It then uses Topic Taxonomy to train the Classifier about which page is to be accepted or rejected.
- Downloader fetches URLs from the URL Queue and downloads the web pages.
- Link Extractor extracts all links from the downloaded pages. These extracted links are sent to the classifier.

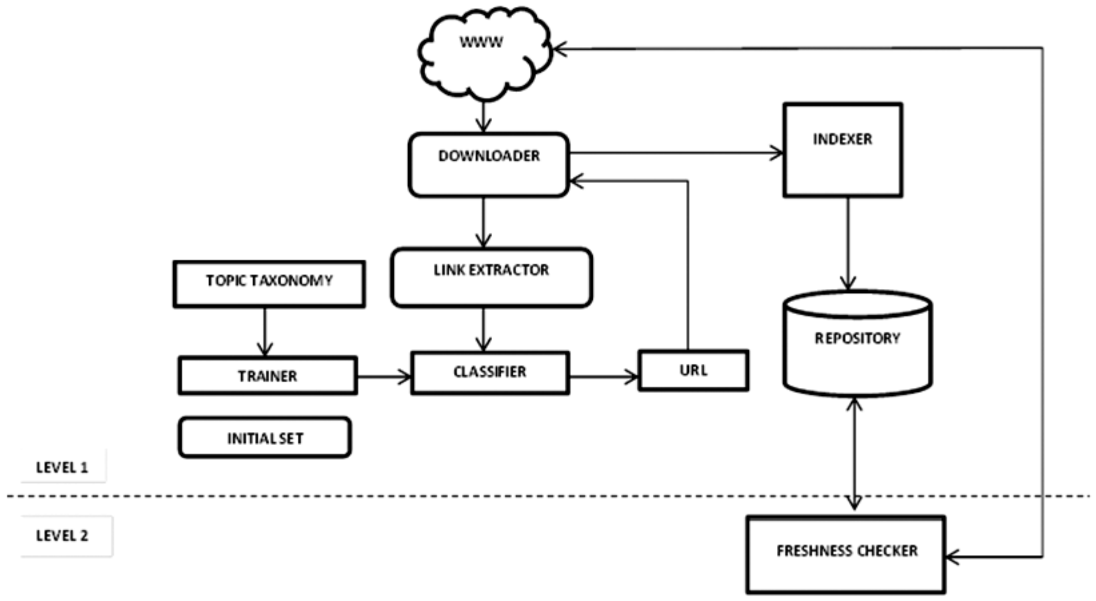

Figure 1: General Architecture

- Downloaded pages are sent to the Indexer and later indexer indexes these web pages in such a way that it facilitates future searches and store them in the repository in compressed form.

4.3. Freshness Checker Mechanism–

After crawling process is started there are some collected web pages in the repository then it is the responsibility of the Freshness Checker module to keep the repository update. After a suitable time intervals this module become active. This module majorly can detect three types of changes in web page which are:

1. Page Structure Change- In this change, styling and structure of the pages changes by making changes in tags of the linked CSS files. Page Structure of the web page gets changed when any new tag is created or any existing tag is deleted.
2. Page Content Change- The content of the web page changes due to the updating of text. If there can be slight modification inside the tags that can also results in the change of the content of web page.
3. Image change- In the HTML page when the image URL is not changed but the image on that URL gets changed.

4.4. Internal Architecture of Freshness Checker

The internal architecture of Freshness Checker shows that pages from repository are fetched for checking their freshness on three criterias namely, change in structure of web page, change in images of web page or change in content of web page. If any of these criterias is satisfied, then page is again refreshed in the repository. This way the bandwidth of system is preserved as the pages which have not undergone any type of change are not downloaded.

4.5. Freshness Checker Metrics

4.5.1. Page Structure change Metric (PSM)

From HTML5 onwards styling in web page is done through CSS files. So any change in the styling of the web page can be done via linked CSS files. The sub-module Structure Checker as shown in fig 2 of module Freshness

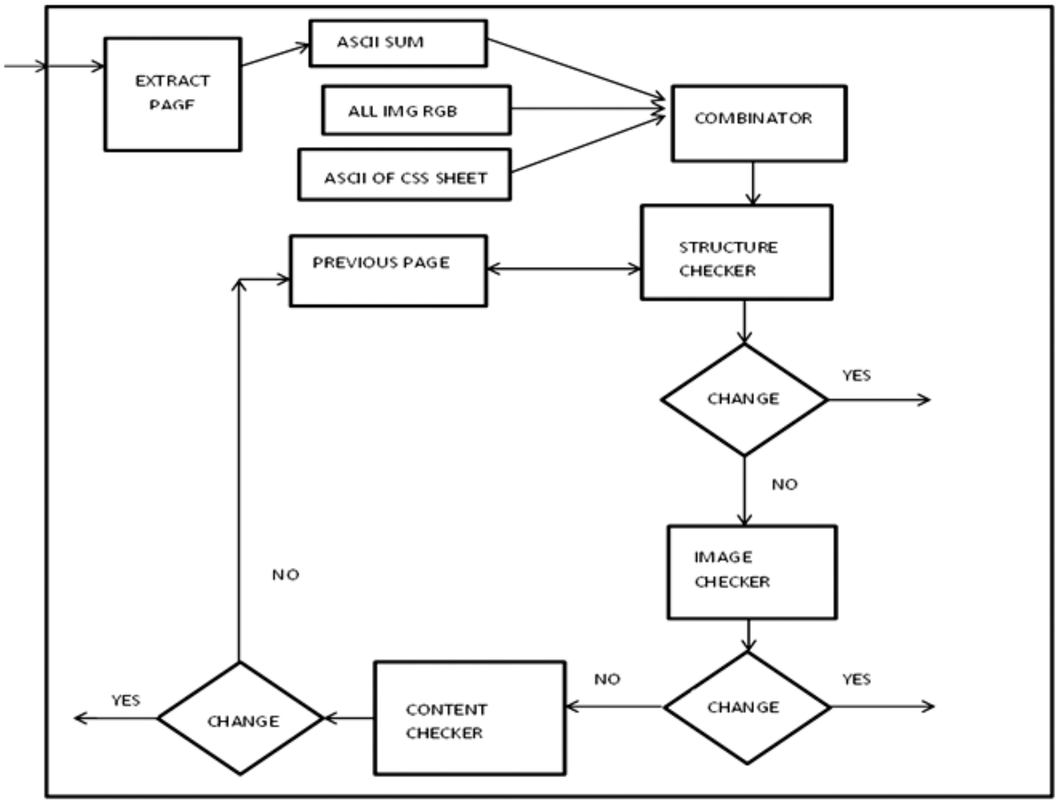

Figure 2: Internal Architecture of Freshness Checker

Checker detects structural and styling change. It uses an approach that detects the CSS files linked with the HTML pages. These CSS files are then checked by comparing the initial value of the PSM with the later value. PSM can be defined as the sum of the ASCII values of the whole content in CSS files including tags and attributes. PSM can be calculated as:

$$PSM = \sum ASCII\ Values\ \left(CSS\ Files\right) \tag{1}$$

This way even a minute change in structure is detected and the freshness of repository is updated. The sub-module extracts all CSS files while crawling the web page.

4.5.2. Page Content Change Metrics (PCM)

The sub-module Content Checker as shown in fig 2 of module Freshness Checker detects any change in the text of the web page. Change in text of the web page can be detected by comparing the initial value of the PCM with the later value. PCM can be defined as the sum of the ASCII value of the character in the HTML page. PCM can be calculated as:

$$PSM = \sum ASCII\ Values\ \left(CSS\ Files\right) \tag{2}$$

4.5.3. Image Change Metrics (IM)

Most of the web pages contain information in the form of images. So change in the image can entirely change information of the web page. Existing techniques to detect change in images have very high time intricacy. So there a need for new method to detect image change.

Each image is made up of pixels and each pixel consists of RGB value i.e. Red , Green and Blue color component value. Calculation of RGB value of whole image is very complex. So the proposed approach uses the RGB value of the dominant color in the image. Change in the image is calculated by comparing the initial value of IM with the later value. IM can be defined as the average of dominant color RGB values of number of images present in the web page. IM can be calculated as:

$$IM = \frac{\sum Dominant\ color\ RGB\ value\ of\ images}{number\ of\ images} \tag{3}$$

4.5.4. Page Revisit Id (PRI)

After computing the three metrics PSM, PCM and IM of web page PRI of each web page is computed and each downloaded page is tag with this PRI. PRI is calculated as:

$$PRI = PSM.PCM.IM \tag{4}$$

Proposed Algorithm For Freshness Checker Module:

```
START
Step1:  Collect the URLs to be revisited from the repository.
Step2:  Take one URL at a time
Step3:  Initialize the Old PSM=PRI(PSM),
        Old PCM =PRI(PCM)and
        Old IM =PRI(IM)
Step4:  Calculate current PSM :
        If(Current PSM==Old PSM)
            Then Go to Step5
        Else
            Update the page
        Go to Step2
Step5:  Calculate current PCM:
        If(Current PCM==Old PCM)
            Then Go to Step6
        Else
            Update the page
        Go to Step2
Step6:  Calculate Current IM:
        If(Current IM==Old IM)
            Then keep the old page
            END
        Else
            Update the page
            Go to Step2
```

4. EXPERIMENTAL RESULTS

The authors have tested the proposed approach on a given set of one hundred web pages. These samples were analyzed for any change in their content, images and page structure. The testing was done for a period of one month. The results proved that out of one hundred samples only twelve samples had a change in them. The variation of pages is shown in fig. 3

Hence, the proposed system resulted in reduction of 84% of crawling bandwidth.

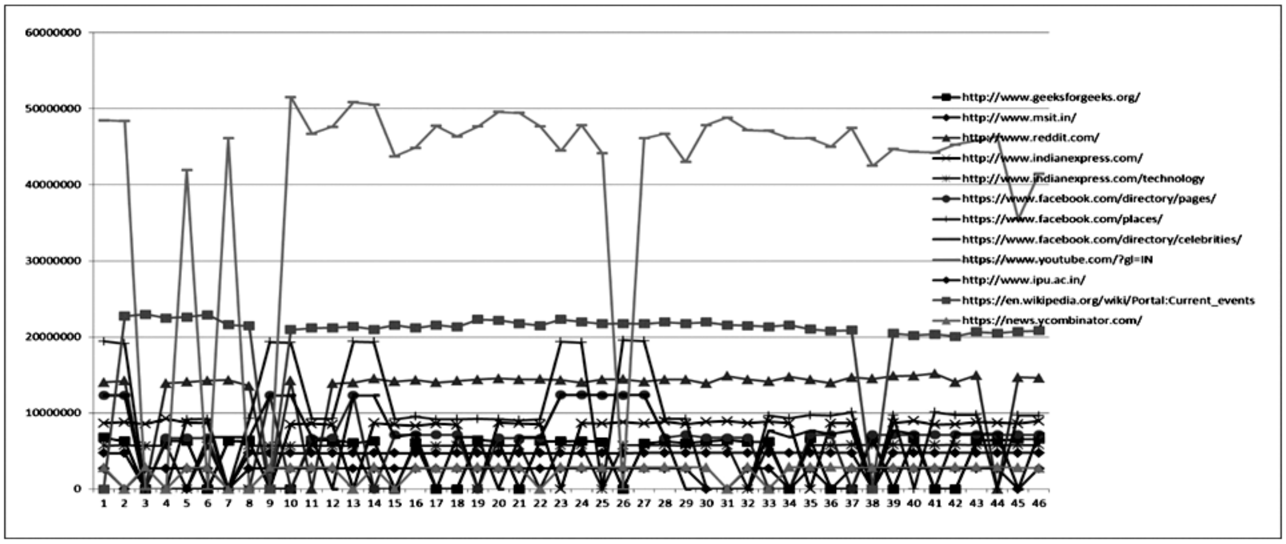

Figure 3: Experimental Results

REFERENCES

[1] J.Cho and H.Garcia-Molina, "Synchronizing a database to improve freshness," In proceedings of 2000 ACM International Conference on Management of Data(SIGMOD),1999.

[2] J.Cho and H.Garcia-Molina. "The evolution of the web and implications for an incremental crawler," In proceedings of 26 International Conference on Very Large Databases(VLDB).pages 136-147.2000.

[3] N. Singhal, A., Dixit and A.K., Sharma, "Design of Priority Based Frequency Regulated Incremental Crawler," In 2010 International Journal of Computer Applications(0975-8887) Volume 1-No 1.

[4] V. Sharma, M. Kumar and R.Vig. " A Hybrid Revisit Policy For Web Search," In Journal of Advances In Information Technology, Vol 3,No. 1,February 2012. DOI: 10.4304/jait.3.1.36-47.

[5] S. Tuteja, N. Kumar and R. Nath, "Reduction of Load on Network using Compression Technique with Web Crawler," In Advances in Computer Science and Information Technology (ACSIT) , print ISSN: 2393-9907; Online ISSN: 2393-9915, Volume 2, Number 3; January-March, 2015 pp.200-203.

[6] V. Hadri, B.B., Cambazoglu and I. Kostopoulos, "Optimal Web Page Download Scheduling policies for Green Web Crawling," In IEEE Journal on Selected Areas in Communications , Volume 34,No. 5,May 2016. DOI: 10.1109/JSAC.2016.2520246.

[7] D. Yadav,D.,Sharma,A.k.,and Gupta,J.P., "Change Detection in Web Pages," In 10 International Conference on Information Technology, 0-7695-3068-0/07 in IEEE,DOI 10.1109/ICIT.2007.37.

[8] R. Nath and N. Kumar, "A Novel Architecture for Parallel Domain Focused Crawler," In International Journal of Engineering Research and Applications (IJERA) , ISSN: 2248-9622, Vol 2, Issue 5,Sept-Oct,2012,pp.266-269.

[9] S. Mali and B.B. Meshram,"Focused Web Crawler With page Change Detection Policy," In 2 International Conference and workshops on emerging Trends in Technology (ICWET)2011, Proceedings published by International Journal of Computer Applications(IJCA).

[10] D. Yadav, A.K. Sharma ,J.P. Gupta, N. Garg, and A. Mahajan, " Architecture for Parallel Crawling and Algorithm for Change Detection in Web Pages," In 10th International Conference on Information Technology,0-7695-3068-0/07 in IEEE,DOI 10.1109/ICIT.2007.64.

[11] D. Yadav, A.K.Sharma and J.P. Gupta," Parallel crawler Architecture and Web Page Change Detection," In WSEAS TRANSACTION on Computers, Issue 7,Volume 7,July 20008.,ISSN:1109-2750.

[12] N. Chauhan and A.K. Sharma,"Design of an Agent Based Context Driven Focused Crawler," In BVICAM's International Journal of Information Technology, Vol.1 No.1 ISSN 0973-5658, Bharati Vidyapeeth's Institute of Computer Applications and Management (BVICAM), New Delhi(2009

[13] Q. Gao, B., Xiao, Z., Lin, X., Chen and B. Zhou,"A High-Precision Forum Crawler Based on Vertical Crawling," In IC-NIDC, IEEE(2009) 978-1-4244-4900-2/09.