Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 6, June 2013, pg.294 – 299

SURVEY ARTICLE

A Survey on Web Page Change Detection System Using Different Approaches

Shobhna[1], Manoj Chaudhary[2]
[1]Department of Computer Engineering, Punjabi University, Yadavindra College of Engineering, Talwandi Sabo, Punjab, India
[2]Department of Computer Engineering, Punjabi University, Yadavindra College of Engineering, Talwandi Sabo, Punjab, India

[1] *bansalshobhna.17@gmail.com;* [2] *ermanojchaudhary@gmail.com*

Abstract— ***Due to limited network and computational resources, it is often difficult to monitor the sources constantly to check for changes and to download changed data items to the copies. The detection of changes across two versions of a page is accomplished by performing similarity computations after transforming the web page into an XML-like structure in which a node corresponds to an open–close HTML tag. Sub trees with common marks will be compared and the similarity parameters of the subtree elements will be calculated. Finding the changes between different versions of the web pages is the core operation of web page change detection system.***

Key Terms: - Change detection; tree similarity; node comparison; types of changes; HTML tags

I. Introduction

Recently, there has been a dramatic increase in the use of XML data to deliver information over the Web. In particular, personal Weblogs, news Web sites, and discussion forums are now delivering up-to-date postings to their subscribers. Clearly, having a central access point makes it significantly simpler to discover and access new content from the existing content. Changes occurring in web pages are best classified as content changes (e.g., deletions, and additions of text), layout change (e.g., changes in the position of elements in the page), and attributes change (e.g., changes in fonts and colors) **[1].** Hence, in addition to tracking content changes, any useful detection system should also be able to track changes in layout. Thus web page change detection system helps to reduce the browsing time of the user and allows the user to find the items in web pages which change frequently. There may be some situations where users are interested in tracking changes across multiple web pages. For example, a user want to know about the latest recruitment in particular field related to particular location in the recent time. In this case, the user wants to be notified of the changes related to various recruitment details present on different web pages. In general, the ability to specify changes to arbitrary documents and get notified in different ways will be useful for reducing the inefficient navigation. There are various types of properties should be considered for detecting the changes in different versions of the web page like speed, accuracy, complexity, storage space, effective version management etc. This paper explains the previous work related with detecting changes using different types of algorithms in summarized way.

II. Previous Work

In this section, a survey of existing Web change detection systems in the market as well as similar systems that have been proposed in the literature is presented.

Various change detection products are existed now. One of that products is *AIDE* [3] presents a set of tools (collectively called AT&T Internet Difference Engine) uses *HTML diff*, the differencing tool used to compute the changes between two pages which uses the weighted LCS. This approach may be expensive computationally as each sentence may need to be compared with all sentences in the document even if user is interested in change to a particular phrase. Another drawbacks of this system is that the user cannot specify customized changes (links, images, keywords) or composite change (links AND images) on a page. Changes to XML pages are not supported.

Second product is *WebCQ* [4][5], which offers personalized delivery of change notifications as well as summarization and prioritization of Web page changes. Notifications can be sent via e-mail to the user and they only describe flagged changes that correspond to raw text between a pair of tags. One drawback of *WebCQ* is that it detects changes only between the last two versions of a HTML page.

Third Product is *Copernic Tracker* [6] that can track changes in the text and images and monitors for the presence of specific text. Some drawback of this software are : *firstly* it does not allow for specifying how much emphasis to place on monitoring different aspects of the Web page, *secondly* it does not provide a utility for monitoring a specific region of the Web page. *Third* this product does not reveal performance data that discusses speed or accuracy.

Fourth one is *ChangeDetect*[7] which detects all the changes but it does not differentiate between specific changes like links, keywords and phrases. Another drawback of this product is that it does not specify composite changes on a page.

Fifth product is *Website Watcher*[8] which includes the ability to monitor password protected Web pages. The drawback of this system is that it offers limited freedom for selecting a zone to monitor and lacks a proper user interface to show the changes. Another demerit is that this system does not provide objective performance data other than subjective user reviews. So it is complicated for users to understand the performance of existing system.

Sixth product is *WYSIGOT* [9] which is a commercial application that detects changes between HTML pages. This system has to be installed on the local machine and the granularity of change detection is at page level.

Seventh product is NetMind [20] which detects changes to links, images, keywords and phrases in an HTML page. The way of notification to web users about the change is through e-mail or mobile phone. There is no support for composite changes (for example if both links AND images changes) on a page. There is no provision for the user to come back later and view the last changes that have been detected. Since the implementation is hidden behind a CGI interface, how changes are detected is not known. Change detection to XML pages is not supported.

Several research papers were found that tackle the design of efficient s for detecting changes in Web pages.

BIODIFF [10] algorithm covers some limitation of X-Diff algorithm. Unlike X-DIFF algorithm, BIODIFF designed for genomic and proteomic data. It outperforms app. 1.5 – 6 times faster than X-DIFF for different datasets. But one limitation of this algorithm is that If a database has more nodes that require min-cost max-flow matching, the improvement of Bio Diff is less as compared to X DiFF. Another limitation is that it takes more time to assign different matching types to the nodes in XML tree.

XML TREE DIFF [11] algorithm presents support for change control in the context of the Xyleme project that is investigating dynamic warehouses capable of storing massive volume of XML data. This algorithm is efficient in speed and memory space. It uses operations such as change node, delete node and insert node. Delta is constructed to find the matching of nodes between two trees. The use of XML specificities in algorithm leads to significant improvements. Drawback of this algorithm is that there is some loss of quality. Another drawback is that there is need of gathering more statistics about the size of deltas and in particular for real web data.

CH-DIFF and *CX-DIFF* [12] s are developed by the Webvigil [12], a system that automates the change detection and timely notification of HTML/XML pages based on user specified changes of interest. *CH-DIFF* detects changes to various components such as links, images, keywords, phrases and any change using Longest Common Subsequence (LCS) . But using LCS will be computationally expensive. The improves up by introducing the concept of window-based change detection. *CX-DIFF* algo consists of steps like object extraction and signature computation, filtering of unique inserts/deletes and finding the common order subsequence between the leaf nodes of the given trees. Assorted and Linked Monitoring is also introduced in the paper. Immediate, Best-effort, Interval based, Interactive notifications are provided to the user. Drawback of this paper is that expensive computation and sentinels or user requests can be overloaded on the single server.

Level Order Traversal [13] is another form of the breadth first traversal. It includes document tree construction, document tree encoding and tree matching (based upon the concept of R.M.S. value of the content), for the detection of structural changes and content changes. Parameters used in this algorithm are node id, child node, parent node, level, tag name, content value or RMS value (sum of multiply the position of the character with its ASCII value). It has linear time complexity because it traverses only the changed portion of the tree rather than the whole tree and hence saves the time. It also extracts effectively the changed content from different versions of a web page. It is simple, less cost, and understandable and can reduce the network traffic by using HTTP meta data. It can successfully retrieve the summary but not the complete content of the newly created page which is insufficient information for the user.

Optimized Hungarian algorithm [14] introduced three running time optimizations that control the operations of the Hungarian by considering time and accuracy analysis. This algorithm focuses on finding the most similar subtree, finding out of order tags or unclosed tags, edit scripting to find minimum edge weight monitoring for bipartite graph. Three measures for detecting changes are also considered which are *intersect*(percentage of similar words) ,*typedist* (position of elements),*attdist*(relative weight of similar attributes). This algorithm also defines that performance is inversely proportional to the depth of tree. Limitation of this algorithm is that running time may be large. Another is that user selected zone may not cover the MSST.

Hashing based [15] saves computation time by limiting the similarity computations between two versions of a web page to nodes having the same HTML tag type, and by hashing the web page in order to provide direct access to node information. To speed up the process of web change detection system a hashing based technique is used for direct lookup of subtree node information during comparisons, and eliminated irrelevant node comparisons by limiting them to nodes of the same type. This algorithm is also applied to RSS(really simple syndication) feeds change detection for delivering regularly changing web content , such as news. Original and enhanced approaches are introduced to improve the comparisons. This algorithm suffers from one limitation that inability to detect changes when root tag is changed. Multithreading is used for improving the performance. This algorithm can be further implemented on dynamic web pages and can also be defined in XML file as future work.

Node Signature Comparison [16] detects the changes in attribute, text. This follow a top down approach it start with the root node by comparing the signature values of each node to its corresponding node in the two trees. To reduce the number of comparison we use node signature by assigning the hash value to all child nodes in the trees the child nodes are basically the text nodes. Signature is mainly the function of the hash value calculated from the contents of the node. Signature of child node is basically the summation of its entire child node signature except the leaf node. The system is very useful for saving browsing time. Limitation of this algorithm is that accuracy and running time is not discussed.

Tree traversing [17] is developed for detecting the structural as well as content change. This algorithm is divided into two parts tree development and change detection. The proposed algorithm used bottom up approach for assigning hash value to each leaf node and tag value to the non-leaf nodes. This algorithm include extracting the tags from html code of the page, assigning node number to each node, finding multiple childs of node, calculating the hash value and tag value by assigning hash function, comparing tags. This algorithm is based on depth first search. This algorithm is simple to understand and it saves the browsing time. But the drawback of this algorithm is that performance is not defined when depth or levels of the tree is increased.

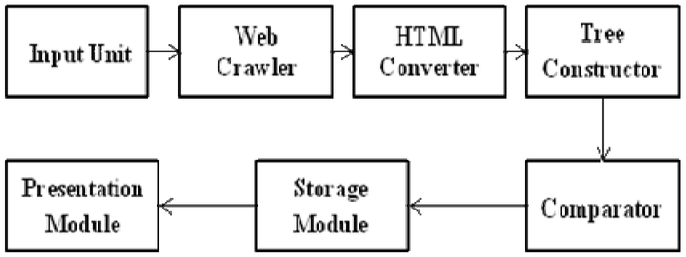
Document Tree based Approach [19] detects the structural and content changes. This Tree based approach is good for comparing the nodes of both the tree as old web page tree and modified web page tree. It gives the relevancy to the web pages and notifies the user about detecting the changes. For detecting structural change document tree is constructed and then signature values assigned to the root nodes and child nodes of the old and new web pages are compared. And for detecting content change first calculates the ASCII value of each character and then it is divided by those particular characters which are occurred in web page only once. Then it determines the text code for the different versions of the page and compares them.

Text code = Summation of ASCII characters/distinct character count

This algorithm defines the good comparison study for the different algorithms and provides simple method for detecting changes. Limitation of this paper is that comparison becomes longer if numbers of nodes are increased. So it's difficult to compare signature for each and every child node.

III. GENERAL DESIGN OF A WEB PAGE CHANGE DETECTION SYSTEM

The design of the Detection system includes Input unit, web crawler, html converter, tree constructer, comparator, storage module, Presentation module. Sometimes design depends upon the techniques used for detection. Here input unit Stores the user input i.e desired web page which is chosen by the user for detection. Web crawler fetches the Web page from the input unit or can download from the internet. Web Crawler is also called Web spiders, web bots, web robots, walkers and wanderers [18]. Html converter converts the web page into html form for extracting the html tages used in the user page and assigns the values. Tree constructer builds a tree from the html tags by defining root node and child nodes. Comparator makes the comparison between the nodes of old and new web page by using some algorithm/technique. Algorithms may be used basis on hash function, signature values, RMS values etc. Storage module stores the comparison result introduced by comparator. Finally the desired web page with the changes is presented to the user for navigation. Hence this design of web page change detection system used in various algorithms for efficient change detection and helps the user for reducing the browsing time of the user.



General design of Web Page change Detection System

IV. COMPARISON STUDY OF VARIOUS ALGORITHM

Algorithm	Speed	Space	Use	Issues	References
BIODIFF [2004]	Faster	Large	For genomic and proteomic data	More Time assignment to matching types	10
XML Tree Diff [2002]	Faster	Small	Good for massive volume of data, support delete, Update,insert/move operations	Loss of quality, Require large statistic information for detection	11
CH Diff and CX diff [2006]	Faster	Small	Timely Notification, Better Space utilization, Efficient type monitoring	Expensive computation, Overloading on single server	12
Level order Traversal [2007]	Faster	Medium	Reduce Network traffic, less cost, Simple	Create Insufficient Information for the user	13
Optimized Hungarian algorithm [2007]	Slow	Small	Good details of time and accuracy analysis, Good performance results	More Running time, Inefficient for user selected zone	14
Hashing Based Algorithm [2008]	Faster	Small	Less computation time, Used for RSS feeds change detection	Sometimes not support change delete operations, not good when root node is changed	15

Node signature comparison [2010]	Faster	Depends upon number of nodes	Suitable for attribute and text changes	No discussion about accuracy and running time	16
Tree Traversing [2012]	Faster	Depends upon number of nodes	Simple to understand, less browsing time	Performance not defined when depth of tree is more	17
Document Tree based Approach [2013]	Faster	Depends upon number of nodes	Good comparison study of different algorithms, simple to understand	Comparison is difficult if more number of nodes	19

V. Conclusion and Future Work

From the analysis of various research papers it is concluded that using a web page change detection system saves the user browsing time. Every algorithm plays some role in improving the performance of previous algorithm. Detection of changes in structure, content, presentation, behavior of the web page helps the user to know about the updated information.

As a future work various performance parameters like running time, accuracy, complexity of computation, number of nodes, depth of tree, load balancing of user requests for notification in various algorithms can be measured and improved. Algorithms can also be implemented in dynamic web pages or in password protected web sites.

References

[1] S. Flesca, E. Masciari, Efficient and effective web change detection, Data and Knowledge Engineering 46 (2) (2003) 203–224.

[2] F. Douglis, et al., “The AT&T Internet Difference Engine: Tracking and Viewing Changes on the Web,” WWW, vol. 1, no. 1, pp. 27–44, Jan. 1998.

[3] D. Hirschberg, “Algorithms for the longest common subsequence problem,” Journal of the ACM, vol. 24, no. 4, pp. 664–675, 1997.

[4] WebCQProduct,http://www.cc.gatech.edu/projects/disl/ WebCQ, 2006.

[5] L. Liu, C. Pu, and W. Tang, “WebCQ—Detecting and Delivering Information Changes on the Web,” Proc. Ninth Int’l Conf Information and Knowledge Management, pp. 512-519, 2000.

[6] Copernic Technologies, Copernic Tracker Product, 2006, http://www.copernic.com/en/products/tracker/tracker-features.html.

[7] ChangeDetect, http://changedetect.com/.

[8] M. Aignesberger WebSite-Watcher Product, http://www.aignes.com, 2006.

[9] Wysigot, http://www.wysigot.com.

[10] Song, Bhowmick. “BioDIFF An Effective Fast Change Detection for Genomic and Proteomic Data” in a Proceedings of the Thirteenth ACM conference on Information and knowledge management., Pp146-147, Nov. 2004.

[11] G. Cobena, S. Abiteboul, and A. Marian, “Detecting Changes in XML Documents,” Proc. 18th Int’l Conf. Data Eng., pp. 41-52, 2002.

[12] S. Chakravarthy, Subramanian, “Automating change detection and notification of web pages” , Proc 17th Int’l Conf. on DEXA, IEEE , 2006.

[13] D.Yadav,A.K. Sharma, J.P. Gupta”Change Detection In Web page” in a proceeding of 10th international conference on information technology,pp 265-270,2007.

[14] I. Khoury, Student Member, IEEE, R. M. El-Mawas, Student Member, IEEE,O. El-Rawas, Elias F. Mounayar, and H. Artail, Member, IEEE ,An Efficient Web Page Change Detection System Based on an Optimized Hungarian , in IEEE Transactions on knowledge and data engineering, VOL. 19, NO. 5,pp 599-612, MAY 2007.

[15] H. Artail *, K. Fawaz, A fast HTML web page change detection approach based on hashing and reducing the number of similarity computations, journal homepage: www.elsevier.com/ locate/datak, Data & Knowledge Engineering 66 (2008),pp 326–337

[16] H. P. Khandagale and P. P. Halkarnikar A Novel Approach for Web Page Change Detection System in a International Journal of Computer Theory and Engineering, Vol. 2, No. 3, pp 364-367,June, 2010.

[17] G. Srishti, Rinkle R. A. “An efficient fr web page change detection” , IJCA, VOL. 48, NO.10, June

2012

[18] Heydon and .Najork.Mercator: “A scalable, extensible Web crawler”. World wide web2 (4):219-229,1999.

[19] Varshney Naveen Kumar and Dilip Kumar Sharma “A Novel Architecture and Algorithm for Web Page Change Detection” IEEE IACC, 782-787, 2013.

[20] Mind-it,http://www.netmind.com/.