Comparative Study of Different Tools and Algorithms for Web Page Change Detection

Parul[1] Dr Pardeep Kumar[2]
[1]Research Scholar [2]Assistant Professor
[1,2]Department of Computer Science and Applications
[1,2]Kurukshetra University, Kurukshetra

Abstract— Now a days people are surfing internet actively for exchange of information across the world resulting in uploading and downloading of information and updating of new web pages very frequently. The contents of web page changes continuously & rapidly. Hence it becomes very difficult to observe the changes made in web pages and retrieve the original web pages. For efficient retrieval and monitoring the changes made in two versions of a webpage various tools or algorithms are there. In this paper, we will explain some of tools and algorithms to detect the changes in web pages.

Key words: HTML, Web Page Change Detection, tree similarity, node comparison, types of changes

I. INTRODUCTION

Tremendous amount of data sources are available online. Web surfers are flooded with huge collection of web pages managed by various sources. About 60% of the contents on the Web is dynamic and approximately 14% of the links in search engines is broken. It is quite possible that after downloading web pages, their local copies residing in the repository/database of a search engine become obsolete compared to the copy on the web server end. Therefore, need arises to refresh the web pages of database at regular interval. Once it is decided to update the document, it should be ensured that minimal resources are used in the updating process. Updating only those documents to the database, which have undergone actual changes. Which require a small amount of data processing as compared to the complete web sites/pages.

Secondly, the user interest has changed from just seeking information to monitoring the changes. The users are interested to seek changes in particular section or portion and not on entire web page. Therefore there is a need to design a system helpful to the user to detect changes in entire page or particular section of web page. For example in a cricket match the statistics changes with every ball, in stock market the rates of shares changes frequently, news are updated rapidly. To understand these changes and to compare the changes with respect to specified time zone the web detection system is very important. This system helps the user to understand the changes effectively and efficiently in minimum browsing time.

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. Most change detection approaches are computationally complex and require non-polynomial running time [2, 3]. Some of the well-known systems that fit the above characteristics are HTMLDiff [4], NetMind [5], WebCQ [6], WebVigiL [7], and CMW [1]. These basically work by estimating the rate of change that occurred between the reference web page and its updated version, and eventually locating the differences between them.

II. LITERATURE REVIEW

In 2003 [8] the change-detection tool was created to determine whether the two versions are identical or not.

In 2003[1] CMW system was implemented which provided change monitoring service on web. In 2008,[11] an algorithm was developed by H. Artail, K. Fawaz, for detection of changes in web page which was based on restricting the similarity computations to sub tree nodes having same HTML tag. It was shown that the speed was improved using this algorithm. In June 2010, [12] node signature algorithm was developed by H. P. Khandagale and P. P. Halkarnikar, system traverse the nodes and detect the changes in between the node signatures. The result of this system shown that the node signature is faster than generalize algorithm. In June 2009, [13] another algorithm was designed by S.Goel and R.R.Aggarwal to detect the structural as well as content based changes in web pages.

In [11], [15] and,[9] various different algorithms were described for detecting changes in XML documents. The algorithm in [17] was based on finding and then extracting the matching nodes from the two trees that were being compared. From the non matching nodes, the change operations were next detected. Matching of nodes was based on comparing signatures (functions of node content and children) and order of occurrence in common order subsequences of nodes. The works in [15] and [9] transformed the pages to trees according to the XML structure and use edit scripting to compare them. The strength of these algorithms lies in their low time-complexity, which is in the order of O (nlogn). Recent work in change detection has focused on computing differences between flat files. The GNU diff utility and AT &Ts HtmlDiff [16,17, and 18] are examples of this category. These examples use the LCS (Longest Common Subsequence) algorithm [17] to compare two plain text files or two HTML pages. This LCS works well for flat files [2].

A number of change detection products are available now-a-days. Some of which are described below:

One of that products is AIDE [20] 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 page are not supported.

ChangeDetect[21] 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.

Copernic Tracker [22] 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.

NetMind [23] 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.

WebCQ [24][6], 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.

Website Watcher [25] 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.

WYSIGOT [26] 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.

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

XML TREE DIFF [27] algorithm presents support for change control in the context of the Xyleme project that was investigated dynamic warehouses capable of storing massive volume of XML data. This algorithm was efficient in speed and memory space. It uses operations such as change node, delete node and insert node. Delta was 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 was that there was some loss of quality and need of gathering more statistics about the size of deltas and in particular for real web data.

BIODIFF [28] algorithm covers some limitation of X-Diff algorithm. Unlike X-DIFF algorithm, BIODIFF designed for genomic and proteomic data. It outperforms 1.5 – 6 times faster than X-DIFF for different datasets. But one limitation of this algorithm was that, if a database has more nodes that require min-cost max-flow matching, the improvement of Bio Diff was less as compared to XDIFF. Another limitation was that it take more time to assign different matching types to the nodes in XML tree.

CH-DIFF and CX-DIFF [29] were developed by the Webvigil [29], a system that automated the change detection and send timely notification of HTML/XML pages based on user specified changes of interest. CH-DIFF was detected changes to various components such as links, images, keywords, phrases and any change using Longest Common Subsequence (LCS). But using LCS was computationally expensive. It was improved by introducing the concept of window-based change detection. CX-DIFF algorithm was consist 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 was also introduced in the paper. Immediate, Best-effort, Interval based, Interactive notifications were provided to the user. Drawback of this paper was that expensive computation and sentinels or user requests can be overloaded on the single server.

Level Order Traversal [30] was another form of the breadth first traversal. It included 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 was 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 was insufficient information for the user.

Optimized Hungarian algorithm [12] introduced three running time optimizations that control the operations of the Hungarian by considering time and accuracy analysis. This algorithm focused on finding the most similar sub-tree, finding out of order tags or unclosed tags, edit scripting to find minimum edge weight monitoring for bipartite graph. Three measures for detecting changes were also considered which were intersect (percentage of similar words) ,typedist (position of elements),attdist(relative weight of similar attributes). This algorithm also defined that performance was inversely proportional to the depth of tree. Limitation of this algorithm was that running time may be large and user selected zone may not cover the MSST.

Hashing based [13] saved 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 was 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 was also applied to RSS(really simple syndication) feeds change detection for delivered the regularly changed web contents , such as news. Original and enhanced approaches were introduced to improve the comparisons. This algorithm suffered from one limitation that inability to detect changes when root tag is changed. Multithreading was 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 [14] detected the changes in attribute, text. It followed top down approach. It started 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, It use node signature by assigning the hash value to all child nodes in the trees the child nodes were basically the text nodes. Signature was mainly the function of the hash value calculated from the contents of the node. Signature of child node was basically the summation of its entire child node signature except the leaf node. The system was very useful for saving browsing time. Limitation of this algorithm was that accuracy and running time was not discussed.

Tree traversing [15] was developed for detecting the structural as well as content change. It was divided into two parts 1.) tree development and 2.) 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. It included extracting the tags from html code of the page, assigning node number to each node, finding multiple child of node, calculating the hash value and tag value by assigning hash function, comparing tags. It was based on depth first search. It was simple to understand and it saved the browsing time. But the drawback of this algorithm was that performance was not defined when depth or levels of the tree was increased.

Document Tree based Approach [31] detected the structural and content changes. This Tree based approach was 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 was constructed and then signature values assigned to the root nodes and child nodes of the old and new web pages were compared. And for detecting content change first calculates the ASCII value of each character and then it was divided by those particular characters which were 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 defined the good comparison study for the different algorithms and provided the simple method for detecting changes. Limitation of this paper was that comparison became longer if numbers of nodes were increased. So it was difficult to compare signature for each and every child node.

III. CONCLUSION

From the above comparison of different algorithm [Table 3.1] or tools [Table 3.2] of web Page Change Detection, we can easily obtain the changes between two versions of a web Page. We can detect any type of change in the web page in form of insertion, updation or deletion in text and attributes. 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 tree and 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.

Algorithm	Speed	Time Complexity	Ordered/ Unordered	Space	Use	Issue	References
XML Tree diff	Faster	Quadratic O(n^2)	Both	Small	Good for massive volume of data, support delete, Update, insert/ move operations	More Time assignment to matching types	27
BIO DIFF	Faster	Quadratic O(n^2)	unordered	Large	For genomic and proteomic data	Loss of quality, Require large statistic information for detection	28
CH DIFF and CX DIFF	Faster			small	Timely Notification, Better Space Utilization, Efficient type monitoring	Expensive computation Overloading on single server	29
Level Order Traversal	Faster	Linear	ordered	medium	Reduce Network traffic, less cost, simple	Create insufficient information for the user	30
Optimized Hungarian Algorithm	Slow	Linear O(n^2)	ordered	small	Good details of time and accuracy analysis. Good performance results	More Running time. Inefficient for user selected zone	12,10

Hashing Based Algorithm	Faster			small	Less computation time, Used for RSS feeds change detection	Sometimes not support change delete operations, not good when root node is changed	13
Node signature	Faster	Linear	ordered	Depends upon number of nodes	Suitable for attribute and text changes	No discussion about accuracy and running time	14
Tree Traversing	Faster			Depends upon number of nodes	Simple to understand, less browsing time	Performance not defined when depth of tree is more	15
Document Tree Based Approach	Faster			Depends upon number of nodes	Good comparison study of different algorithms. Simple to understand	Comparison is difficult if more number of nodes	31

Table 3.1: Showing Comparison Study of Various Algorithms

Tool	Client/Server	Recursion	Notification	Show Difference	Paid/Free
AIDE	Server	yes	Yes (with Email)	yes	
ChangeDetect	Server	yes	Yes (with Email, ICQ, text message)	Yes (with color coded highlights changes)	Free service
Copernic Tracker	Server	yes	Yes (with Email, including a copy of the web page with the changes highlighted, or by desktop alert.)	Yes (with highlights changes)	Paid service
NetMind	Server	yes	Yes (with Email or call)	Yes (indicating degree and location of change)	paid
WebCQ	server	yes	Yes(with Email)	Yes (analyzes page structure)	free
Website Watcher	both	yes	Yes		paid
WYSIGOT	both	yes		Highlight anything new	Both versions availble

Table 3.2: Showing Comparison Study of Various tools discuss above

REFERENCES

[1] S. Flesca, E. Masciari, Efficient and effective web change detection, Data and Knowledge Engineering 46 (2) (2003) 203–224.
[2] S. Chawathe, J. Widom, A. Rajaraman, H. Garcia-Molina, Change detection in hierarchically structured information, in: ACM SIGMOD International Conference on Management of Data, Montreal, Canada, 1996, pp. 493–504.
[3] S. Chawathe, H. Garcia-Molina, Meaningful change detection in structured data, in: ACM SIGMOD International Conference on Management of Data,vol. 26, no. 2, 1997, pp. 26–37.
[4] HTMLDiff. <http://www.componentsoftware.com/Products/HTMLDiff/index.htm>.
[5] Netmind.<http://www.changedetect.com/cd-netmind.asp>.
[6] L. Liu, C. Pu, W. Tang, WebCQ – detecting and delivering information changes on the web, in: 9th International Conference on Information and Knowledge Management, Atlanta, Georgia, 2000, pp. 512–519.
[7] S. Chakravarthy, J. Jacob, N. Pandrangi, A. Sanka, Webvigil: An approach to just-in-time information propagation in large network-centric environments, in: 2nd International Workshop on Web Dynamics, Honolulu, Hawaii, 2002.
[8] Y. Wang, D. DeWitt and J. Cai, "X-Diff: An Effective Change Detection Algorithm for XML Documents," Proc. 19th Int'l Conf. Data Eng., pp. 519-30, 2003.
[9] J. Jyoti, A. Sachde, and S. Chakravarthy, "CX-DIFF: A Change Detection Algorithm for XML Content and Change Visualization for Web Vigil," Data and Knowledge Eng., vol. 52, no. 2, pp. 209-230, 2005.
[10] 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 modification detection system at multiple nodes Based on an Optimized Hungarian Algorithm", in IEEE Transactions on knowledge and data engineering, vol. 19,no. 5, pp 599-612, MAY 2007.
[11] 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 Engineering66, pp 326–337,2008.
[12] H.P. Khandagale H.P. and Halkarnikar P.P "A Novel Approach for Web Page Detection System" International Journal of Computer Theory and Engineering, vol2, no3, pp 364-368, June 2010.
[13] S.Goel, R.R.Aggarwal "Comparative Analysis of Webpage Change Detection Algorithms" International Journal of Research in Engineering & Applied Sciences, vol2, issue 2, PP. 1382-1397, February 2012.
[14] Y.F.Chen,E.Koutsofios "The AT&T Internet Differnce Engine: Tracking and Viewing Changes on the Web" AT&T Bell Research labs,pp.11
[15] G. Cobena, S. Abiteboul, and A. Marian, "Detecting Changes in XML Documents," Proc. 18th Int'l Conf. Data Eng., pp. 41-52, 2002.
[16] F. Douglis, T. Ball, "Tracking and Viewing Changes on the Web", 1996 USENIX Annual Technical Conference, 1996.
[17] F. Douglis, T. Ball, Y.F.Chen, E.Koutsofios, "The AT&T Internet Difference Engine: Tracking and Viewing Changes on the Web", World Wide Web, 1(1):27-44, January 1998
[18] E.Berk, "HtmlDiff: A Differencing Tool for HTML Documents", Student Project, Princeton University, http://www.htmldiff.com
[19] D.McArthur, "LSR Gene Expression RFI Response", Rosetta Impharmatics, http://cgi.omg.org/cgi-bin/doc?lifesci/99-08-11,August1999.
[20] D. Hirschberg, "Algorithms for the longest common subsequence problem," Journal of the ACM, vol.24, no. 4, pp. 664–675, 1997.
[21] ChangeDetect, http://changedetect.com/.
[22] Copernic Technologies, Copernic Tracker Product, 2006, http://www.copernic.com/en/products/tracker/tracker-features.html.
[23] Mind-it,http://www.netmind.com/.
[24] webCQProduct, http://www.cc.gatech.edu/projects/disl/WebCQ, 2006.
[25] M.Aignesberger WebSite-Watcher Product, http://www.aignes.com, 2006.
[26] Wysigot, http://www.wysigot.com.
[27] G.Cobena, S.Abiteboul, and A.Marian, "Detecting Changes in XML Documents," Proc. 18th Int'l Conf. Data Eng., pp. 41-52, 2002.
[28] 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.
[29] S. Chakravarthy, Subramanian, "Automating change detection and notification of web pages", Proc 17th Int'l Conf. on DEXA, IEEE, 2006.
[30] 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.
[31] Varshney Naveen Kumar and Dilip Kumar Sharma "A Novel Architecture and Algorithm for Web Page Change Detection" IEEE IACC, 782-787, 2013.

