ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 9, September 2013

A Novel Approach on Web Page Modification Detection System at multiple nodes.

Neha Batra[1], Chandna Jain[2]

M-TECH Student, Computer Science & Engineering, JCDM College of Engineering, SIRSA, INDIA[1]

Assistant Professor, Computer Science & Engineering, JCDM College of Engineering, SIRSA, INDIA[2]

Abstract: In this paper, we describe the technique to detect the multiple change in the web document in the form of addition, deletion of the text and content change. We know that World Wide Web today is growing at phenomenal rate. People are using internet for exchange of the information. The information on the web changes continuously and rapidly. So it is very difficult for us to observe every change in the web document and to retrieve the latest information. In this paper, the web page modification detection system at multiple nodes based on the signature of nodes corresponds to HTML web pages. So that user can retrieve the latest information easily and in the short time. In this first input web page then build the tree from HTML page i.e. DOM (data object modelling).The node signature algorithm is developed to compare the trees of old web page and modified web page to find changes. In this way, system detect the changes i.e. addition and deletion of the nodes, attributes change, structure changes etc and helpful for keeping with up to date information.

Keywords: Web page modification detection, Tag Value of leaf nodes, Signature of nodes i.e. non leaf nodes, Find Changes.

I. INTRODUCTION

Due to increasing the usage of the internet by the people for the information user are only interested to access those pages where new information has been updated like online shopping, online trading. For e.g. the stock traders will be interested to know how the information is changing continuously. Stock price can be changed. So this system is helps to reduce the browsing time of the user and allow the user to find the item on web which changed frequently. In June 2009, a new algorithm is designed by S.Goel and R.R.Aggarwal to detect the structural as well as content based changes in web pages [1].In 2008, algorithm is developed by H. Artail, K. Fawaz, detection in web page is based on restricting the similarity computations to sub tree nodes having same HTML tag and showed the speed improvement [2].in June 2010, node signature algorithm is developed by H. P. Khandagale and P. P. Halkarnikar, system traverse the nodes and detect the changes in between the node signature. The result of this system shows that the node signature is faster than generalize algorithm [3].In [4, 5, 6] various algorithm and tools are used to detect the changes in web documents. In paper [7] technique is implemented in CMW system providing a change monitoring service on web. In [8] change id detected on the contents of the document and visualized by WebVigil. [10] Change detection based on the three optimizations that are implemented on top of Hungarian algorithm.

II. MODIFICATION DETECTION

The web page modification detection system at multiple nodes helps the user to detect changes efficiently and in minimum browsing time. The changes are like structure changes, presentation changes, content changes and the behavioural changes. We classify the changes web pages in to different types.

Changes are classified as:-

i. Structural changes
ii. Content changes
iii. Behavioural changes
iv. Presentation changes

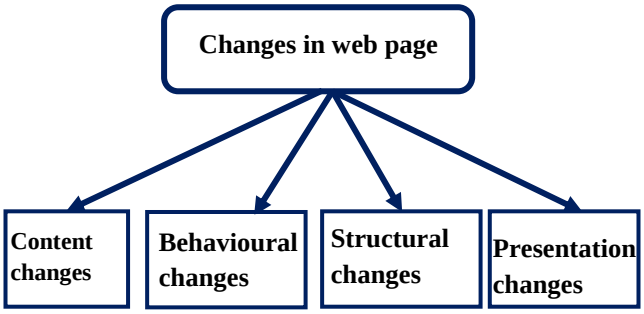

Fig1. Classification of changes in web page

Structural changes: - In the structure changes the structure of the web document is changed means new items are added or deleted from the web page. In this structure change contents of the web page remains same but new tags are added or deleted which causes a structure change in web page.

Content changes: - In the content changes the content of the web pages are changed like change in the status of the market and the current price of the share.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 9, September 2013

Behavioural changes: - In the behavioural changes, changes are in the active component which are present in web pages. For e.g. in web pages scripts and applets are as active components.

Presentation changes:-In presentation changes, contents of the web page remains same but way of presentation of contents of web pages changes.

III OBJECTIVES OF THE RESEARCH

The main objective of the web page modification detection system is to detect the structural as well as content changes at multiple nodes at one time so as to reduce the user time and allow the user to find the item in the web pages which change frequently.

- Addition of new nodes to the tree
- Deletion of nodes from the tree.
- Detecting the change in the content of multiple nodes.

IV. STEPS FOR WEB PAGE MODIFICATION DETECTION

I. Input web page
II. Parse the web page and filter out tag structure
III. Arrange tags in order of hierarchal relation
IV. Tree is generated from tree generated module

A. Algorithm used to detect changes.

class Node

```
{
   string nodename
   int nodeno
   int tag
   string childs
   string content
}
```

Where nodename contains name of node, nodeno means node number which is assisgned according to bfs,tag contains the hash value , childs contains node numbers of chidren and content contains content of leaf node.

first of all we have to extract tags from the input file and then build tree from it. To assign node numbers we use breadth first search technique. the algorithm to assign node number is given below.

B. Assign node no to nodes of tree.

Assign node no	
Input: Empty tree which just contains node names Output: Tree which contains nodenumber according to bfs	
1.	$Let\ countopen \leftarrow 0, countclose \leftarrow 0, closetag \leftarrow "", nexttag \leftarrow ""$
2.	$Let\ tree[0].nodeno \leftarrow 1$
3.	$Repeat\ for\ i\ from\ 0\ to\ tree.count$
4.	$Let\ countclose \leftarrow 0, countopen \leftarrow 0$
5.	$Let\ closetag \leftarrow$ "/" + tree [i].nodename
6.	$Let\ nexttag \leftarrow tree[i+1].nodename$
7.	$Let\ j \leftarrow i+1$
8.	$While\ (nexttag! = closetag)$
9.	$Let\ subs \leftarrow tree[j].nodename.substring(0,1);$
10.	$If\ (!subs.equals("/"))\ then$
11.	$countopen \leftarrow countopen + 1$
12.	$Otherwise, countclose \leftarrow countclose + 1$
13.	$end\ if\ else$
14.	$if(countopen - countcloe = = 1\&\&!subs.equals("/")\&\&$
15.	$tree[j].nodeno == 0)$
16.	$tree[j].nodeno = k$
17.	$k \leftarrow k+1$
18.	$End\ if$
19.	$j \leftarrow j+1.nexttag \leftarrow tree[j].nodename$
20.	$End\ loop$
21.	$End\ loop$

In above algorithm we are starting from root node and moving towards next node until we find close tag of root node and so on . Thus allocating the nodenumber to nodes in breadth first search.

C. Assign Tag value to leaf nodes

Algorithm: - Assign hash value to leaf nodes
Input: - Root of the tree T1 and T2.
Output: - Hash value of the leaf (child) nodes.
step 1 Input root node of tree **Step 2** Repeat foreach node n of the tree if(n.children == null) then **Step 3** Set n.tag = assign tag value // assign tag value using GetHash () **Otherwise** **Step 4** goto step 2

In GetHash() function we are decide hash value for each node which depends on level of tree at which child is presently and node name

D. Calculate signature value of non leaf nodes.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 9, September 2013

Algorithm: - Calculate signature value of all the non leaf nodes
Input: All nodes of the tree T1 and T2
Output: - Signature value of all non leaf nodes including root node of the tree.
Step 1 Repeat for currentnode from n to 1 // where n is highest node number **Step 2** If currentnode.tag !=null goto step 1 **Otherwise** **Step 3** $\boldsymbol{currentnode.tag = \sum_0^m children.tag}$ // calculate signature value of the nodes where m is total number of children of currentnode and children represents the children of currentnode. **Step 4** Ends

In the signature calculation, signature value of the non leaf nodes is calculated by summation of hash value of leaf nodes.

E. To find content changes.

Algorithm : Content change()	
Input:-old tree T1 and New tree T2 to find changes. Output: Find content changes	
1	$Let\ changecontent \leftarrow$ " "
2	$Repeat\ for\ i\ from\ 0\ to\ tree.count$
3	$If(!\,t1[i].childs.equals(Null))$
	$Compare\ content\ of\ t1[i]\ and\ t2[i]$
	$If\ content\ of\ t1[i]\ and\ t2[i] is\ not\ equal\ then\ note\ the\ change$
	End

F. Find changes in trees T1 and T2

Algorithm: - Find changes in tree T1 and T2	
Input:-old tree T1 and New tree T2 to find changes. **Output: - node added in the new tree or deleted in old tree.**	
1	$Compare\ root\ nodes\ of\ tree\ T1\ and\ T2$
	$if(signature\ of\ root\ node\ of\ T1 = = signature\ of\ root\ node\ of\ T2)$
	$then\ no\ change\ found,$ goto contentchange() // compare the Signature Value
	Otherwise
2	$Compare\ number\ of\ children\ at\ each\ level\ of\ tree$
	$If\ number\ of\ children\ differ\ at\ any\ level\ compare\ signature\ of\ parents\ of\ nodes\ at\ that\ level$
	$if\ (signature\ of\ t1_i\ ! = signature\ of\ t2ithen\ changes\ found.//$ compare signature value of child node {
	$if\ signature\ value\ of\ node\ t1_i > signature\ value\ of\ node\ t2_i\ then\ node\ added = T2.signature - T1.signature$ ***oterwise*** $node\ deleted = T1.signature - T2.signature.$ // changes found in text node added or deleted in tree.
	$If\ the\ child\ whose\ tag\ value\ is\ equal\ to\ difference\ calculated\ above$ }
3	End

In this way, multiple changes are found in the web page by comparing the nodes of the trees T1 and T2.

V. RESULT

A. Deletion of nodes from the tree

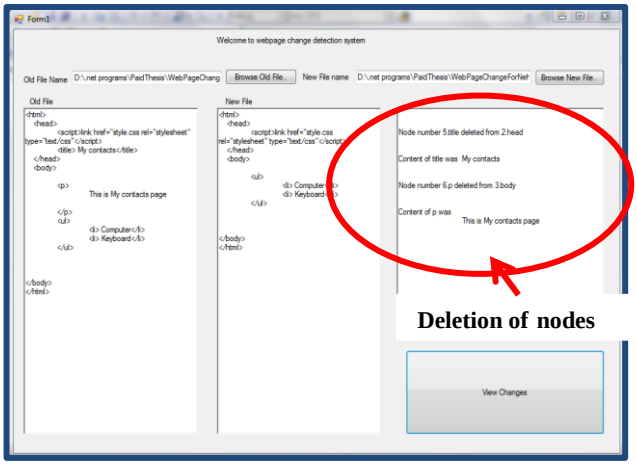

Fig2. Deletion of nodes in web page

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 9, September 2013

B. Addition of nodes to the tree

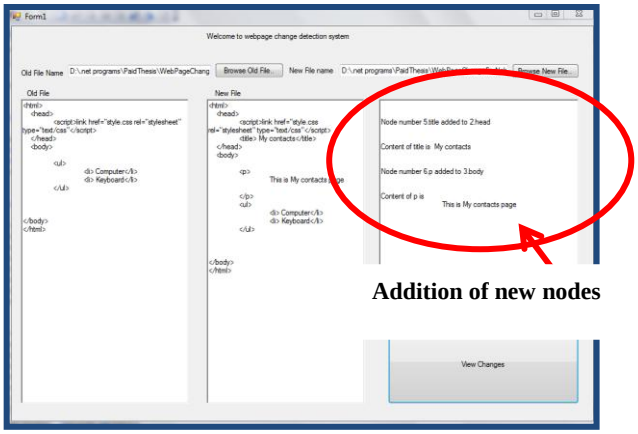

Fig3.Addition of nodes in web page

C. Content Changes in the two trees

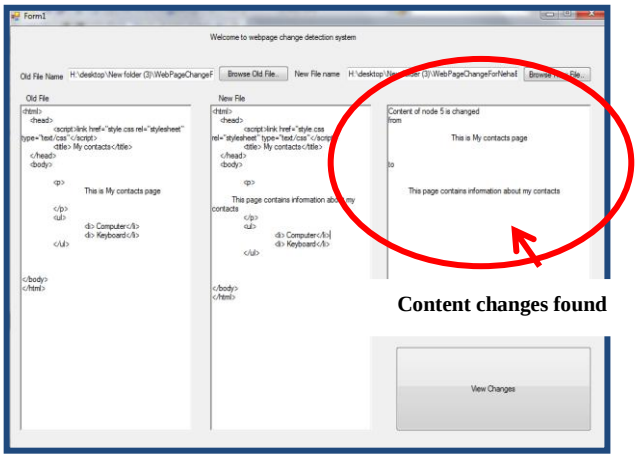

Fig4. Content changes in web page

V. CONCLUSION

In this, system detect the changes at text node i.e. the new text nodes are added or deleted in the tree and reduce the user time. To find the changes, we traverse each node in the tree by breadth first search traversing technique. This system is useful for saving the user time and gives the user time to time update about the changes occurs in web pages at multiple nodes.

ACKNOWLEGDEMENT

The authors of this paper would like to express their heartfelt gratitude to the Management, Principal of their respective institutions and the faculty members of their respective departments who have contributed kind support to publish this paper. In addition, authors also wish to express their great thanks to their family members who have rendered kind support to bring this research paper for the publication. The authors also express their sincere thanks to anonymous reviewer of IJARCCE for reviewing this work and providing necessary comments.

REFERENCES

[1] S.Goel, R.R.Aggarwal, "An Efficient Algorithm for Web Page Change Detection" *international journal of computer application*, vol48, no10, pp 29-33, June 2012.

[2] 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, pp 326–337,2008.

[3] 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.

[4] 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.

[5] 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.

[6] E. Berk, "HtmlDiff: A Differencing Tool for HTML Documents", *Student Project*, Princeton University, http://www.htmldiff.com.

[7] S. Flesca, E. Masciari "Efficient and effective Web Page Detection", Data & Knowledge Engineering, vol. 46, issue 2, pp. 203–224, 2003.

[8] 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.

[9] 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.

[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.