Difference computation using change identification techniques for structured web documents

J Arora and K R Ramkumar
Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India,
jatin.arora@chitkara.edu.in

Abstract. In this era of the competitive world, one needs to stay updated with all the information that is required for their professional and personal growth. But due to vast information, it is difficult to cope up with the ever-changing data. WWW is the main source of data. It consists of thousands and millions of web pages, out of which some are static pages while the other is dynamic pages. The contents of the dynamic web pages such as news websites, stock prices, weather broadcast change frequently. The changes include the insertion of some new data, deletion of data, or modification of existing data. To detect the changes that are variedly occurring in these web pages, various change detection tools and algorithms have been developed. This paper will discuss the categories of changes occurring on the structured document and their corresponding algorithms and tools which can be applied to retrieve the relevant changes.

1. Introduction
The documents on the web are increasing at a rapid rate and with the same rate changes which are occurring in these documents are also increasing. Information is exchanged on the web with the help of XML documents as XML is used as a standard format for passing documents over the web [1]. In industries and academic institutions, XML documents are used to store the data, and then these documents are transferred for use. The main concern here is that the documents should reach the intended receiver intact. That is there should not be any changes that are being done in these documents either intentionally or unintentionally and hence change detection in XML documents is required.

Certain websites change frequently their data [2], to detect and notify the respective changes it is necessary to examine the changes. So identifying the changes and computing the difference in web documents is a research area that has a huge potential. The changes in these web pages can be measured based on various parameters such as a change in content, change in layout, presentational changes, behavioral changes, and changes that occur due to type modification.

Changes in content here refer to any change that is being done in the text such as a change in the value of gdppc data, country name, year, etc. Changes in layout occur due to the repositioning of tags within the document e.g. the position of year and rank tag is interchanged. Presentation changes occur when the format of data representation on the web is changed such as a change in font, color, size. If the location of a country is responding unpredictably then behavioral change can be said to occur. Type changes happen if certain tags are replaced by some other tag to make the original tag
inaccessible or behave abnormally. Consider an example of the country database in XML format as shown in Figure 1.

```xml
<?xml version="1.0"?>
<data>
  <country name="Liechtenstein">
    <rank updated="yes">1</rank>
    <year>2008</year>
    <gdppc>141100</gdppc>
    <neighbor name="Austria" direction="E"/>
    <neighbor name="Switzerland" direction="W"/>
    <location>url to google maps</location>
  </country>
  <country name="Singapore">
    <rank updated="yes">2</rank>
    <year>2011</year>
    <gdppc>59900</gdppc>
    <neighbor name="Malaysia" direction="N"/>
    <location>url to google maps</location>
  </country>
</data>
```

Figure 1. Country XML Document.

To identify and detect the above changes as well as to find the exact duplicates of web pages or structured documents, different change detection algorithms and tools have been developed [3]. This paper will give a brief description of the various change detection algorithms that have been used to identify the operations such as insertion, deletion, update, and moving of structured documents. These operations are responsible for the occurrence of changes in web pages. Most of these documents are first converted into a DOM structure and then depending upon the application, the change detection algorithm is applied to it. The algorithm follows matching the two versions of the documents and comparing for any changes by computing the difference between the nodes of the two trees.

2. Change Detection Algorithms

A major concern of change detection in the structured document is the ability to identify changes occurring over time. Multiple change detection algorithms were developed to identify the differences between various documents. Table 1 illustrates the popular change detection techniques available for detecting changes in hierarchically structured documents. One of the well-known techniques used for change detection is diff utility. By following the method used in diff utility, various algorithms have been developed. The basic idea of change detection technique is to represent the structured document in the tree form and then perform tree to tree comparison [4].

2.1 Shingling algorithm

This algorithm deals with the problem of finding near-duplicates web documents. To identify such documents, a general approach is used which generates the similarity index of ngram vectors from these documents which is then matched based on similarity metric and if the value of this metric reaches a certain threshold then these documents are considered to be near-duplicates of each other. But in the Shingling algorithm, a shingle vector is made with the help of ngram vectors encoded in 64 bit Rabin fingerprint format [5]. Then these shingle vectors are matched using the Jaccard coefficient and similarity between these documents is verified.
2.2 Johnson Algorithm
In the domain of ever-changing web documents, the Johnson algorithm was designed to acknowledge the changes which are made to these documents. This algorithm computes the difference between two web pages by measuring the difference between the computed values of the weights of addition or deletion of keywords, contents, or paragraphs. If the computed distance is greater than the target value then that document can be discarded also, otherwise if it is within the specified range then the users are being made aware of the changes that happened [6].

2.3 Proportional Algorithm
Proportional algorithm is an advancement of the Johnson algorithm which provides the distance value of the two documents by first normalizing the weighted parameters by calculating the signature of the relevant parameter of the document individually and then measuring it proportionally for all the changes [6]. It is a Signature-based approach and is highly used for all kinds of web pages. This systematic approach leads to provide a better result for change detection and notification.

2.4 Cosine Algorithm
This approach is used to calculate the context-based changes. It starts with finding the weighted value of the term vectors for each web page using any heuristic and then saving these term and weighted vector for all pages into a file. Then it takes the composition of vectors and computes the context term and weighted vector of the unchanged web pages followed by the calculation of cosine similarity angle. This cosine angle is compared with the previously computed angle of the web pages and hence determines the degree of similarity of these web pages [7].

2.5 Fuzzy Tree Algorithm
As the name signifies this algorithm detects the changes made to web documents by converting the document into a tree structure and calculate the difference in two trees by applying a fuzzy hash function to the nodes of a tree. This algorithm removes all the irrelevant changes that have been made. To do this it first defines what will be considered as relevant changes and what will not. Based on this parameter it compares the threshold value of the computed DOM tree and specifies what changes have been made to the document [8].

2.6 BULD Algorithm
Cobena et al. proposed a BULD algorithm which stands for Bottom-Up, Lazy-Down propagation [1][9]. In this algorithm Bottom-Up, tree parsing is done, as first, it converts structured web documents into a tree and then computes the similarities between the common large subtrees. Then in a lazy down manner, it propagates the similarities in a bottom-up manner. It matches nodes of the tree and constructs a delta corresponding to it. Delta is the difference between two trees which is then compared to check the similarity of documents.

2.7 CX-Diff
CX-Diff algorithm also works on the principle of constructing a DOM tree from an XML document in the distributed environment. This algorithm is designed to detect the changes which the user request and then notify them for the same. Here it is considered that nodes of the trees are arranged in an orderly manner but the attributes of nodes can be present in an unordered way. So, depending upon the changes it uses the concept of Event-Condition-Action rules and extracts the changes using the push/pull paradigm [10].

2.8 X-Diff
Wang et al. studied the X-Diff algorithm which focuses on an unordered model of a tree while most other algorithms assume structured documents that can be represented in an ordered manner [11]. This kind of model is helpful in database applications. It works on matching the node signature and new
matching between the two versions of the document which resulted in finding the value of minimum-cost matching and thus generating minimum cost edit script.

2.9 Vi-DIFF
As the name suggests, Vi-DIFF stands for detecting visual changes that occur on the web pages. This algorithm can identify changes that are done either on text, images, links, or visual appearance of it. It compares the web pages by first creating the visual semantic block based on the segmentation of web pages followed by comparing the block nodes of two documents. It then applies a change detection technique that finds delta between the two semantic versions of the web pages and thus calculates the changes which have occurred [12].

2.10 Level order Traversal
Level order traversal is based on a breadth-first strategy. It detects the changes in web documents by constructing the tree, traversing the nodes of a tree, and then calculating the R.M.S value of the node tag and content such as paragraph, heading, etc. It then matches the number of nodes of both documents which result in determining the adding, deleting of new nodes, or modification of the existing tag [13].

2.11 LA-Diff
LaDiff change detection algorithm is used for detecting changes in LaTeX documents. The algorithm uses FastMatch which cannot be used for detecting changes in an XML document. It forms a basis for developing other change detection algorithms. LaDiff handles updates, inserts, deletes, and even moves. But LaDiff only supports certain LaTeX elements, therefore its FastMatch cannot be used as an algorithm for change detection in XML documents, but its underlying algorithms have inspired others in creating their algorithms. FastMatch uses the longest common subsequence to match the nodes in a sequence starting from leaf nodes [14].

2.12 MH-Diff
MH-DIFF stands for Meaningful change difference in hierarchical structure data for an unordered tree. This algorithm implements subtree copy and subtree move operation also, besides, to insert, update and delete operations. It transforms the task into an edge cover problem and thus compares two trees and forms a minimum-cost edit script, to transform the original tree into a modified tree. It is an efficient algorithm that extracts semantic meaningful changes only [15].

2.13 XyDiff
The XyDiff change detection is applied to XML documents. It takes two XML documents and calculates the value of hash and the corresponding weight of each node in the tree in a bottom-up fashion. It assigns a unique identifier to all nodes of the tree and forms a document in prefix order of nodes with the help of XidMap. It tries to find the exact match between the subtrees and if a match is found it uses heuristics to pick the nodes and propagates its updated weights higher in the tree [16].

2.14 DiffX
This algorithm is used to detect changes that occur between two versions of the XML document [17]. It starts by converting the XML document into tree structure ensuring the nodes of the tree are well labeled and typed. It then identifies the largest isolated tree fragment based on matching the largest tree fragment nodes of the two versions of the documents. After identifying the largest tree fragment, it iteratively matches the entire tree to identify the changed nodes concerning both content and structure of the nodes and record the matchings in edit script to be applied to the original version of the document.

2.15 MMDiff/XMDiff
According to Sudarshan et al., the MMDiff algorithm compares the structured documents in which data being differentiated fits within the main memory and does not require any additional memory space [18]. The MMDiff creates a matrix based on the string edit problem to produce a difference script. XMDiff algorithm is closely related to the MMDiff algorithm that allows user to compare structured data which cannot completely fit within the main memory and require external memory for storing the entire data. It helps compare the structured document of a very large size. It produces minimum cost edit script

2.16 KF-Diff+
According to Xu et al., KF-Diff+ is an efficient algorithm for fast calculation of change detection for both ordered and unordered models of structured trees [19]. It breaks the traditional tree to tree correction technique which is used in the above algorithm. It works on the idea of comparing key trees which are labeled trees that consist of a unique path. It devises a mechanism to eliminate duplicate paths by identifying MI i.e. multi-instance nodes which consist of nodes that share the same parent node and label name and then computes the key path. It then matches the two trees based on the key path computed and finds the minimum cost matching for change detection.

3. Applications of Change Detection System
In this section, we have mentioned some of the primary application areas of change detection algorithms.

3.1 XML Document Version management
A version management system keeps information of all the data and files which are revised and being sent out to the user. E.g. a news website sends updates and notifications to its users. They have to keep track of all the versions they send out related to updated information. It archives the versions of data that are communicated and keeps track of all the previous versions also. So in case of any updates or changes of web pages, version management is responsible for storing and managing all edited documents [20].

3.2 Website content change tracking
For providing remote access to documents and authoring changes in webpages WebDAV (Web-based Distributed Authoring and Versioning) is used. It provides a mechanism that helps service providers to create, update, and move documents on the server. It also maintains the log of document properties such as modification in a namespace, author detail, collection, and so on. It is an extension to the famous HTTP protocol which is used by web browsers or servers to interact with each other. It also behaves like a file server and allows for file lock and sharing features. It keeps tracks of various versions and controls them.

3.3 Synchronize data in a dynamic environment
To synchronize with the changes that are occurring in documents or web pages, the git-merge tool can be used. It merges the old and new document by combining the common changes and to control the conflicts if any due to merging kdiff3 can be used. This will create a new merged document and will remove all duplicate entries of the past. Likewise for merging documents 2, merge for 3 document can also be used [21]. A 2-way merge file system only keeps tracks of changes that occur on website or documents and after identifying such changes it reintegrates them into one merged file. Whereas in 3 ways merging the files need to be kept, one is the original file and the other two are modified versions of the original file. Delta XML Tool is capable of performing 2 way and 3 way merging of documents [22].
Table 1. Change detection Algorithm and its key features

| S No | Name of Algorithm      | Content Change | Structural Change | Presentation Change | Behavioural Type Changes | Context Change | Key Feature                                                                 |
|------|------------------------|----------------|-------------------|---------------------|--------------------------|----------------|-----------------------------------------------------------------------------|
| 1    | Shingling Algorithm    | Y              | N                 | N                   | N                        | N              | Website/Document content duplication detection.                             |
| 2    | Johnson's Algorithm    | Y              | N                 | Y                   | N                        | N              | Computes the distance between two documents.                                |
| 3    | Proportional Algorithm | Y              | Y                 | Y                   | N                        | N              | Computes the distance between two documents including structure change.     |
| 4    | Cosine Algorithm       | N              | N                 | N                   | N                        | N              | Computes the degree of change based on the Cosine angle between context vector and change term vector. |
| 5    | Fuzzy Tree Algorithm   | Y              | N                 | N                   | N                        | N              | Extracting modifications between two websites by using fuzzy tree algorithm. |
| 6    | BULD Diff              | Y              | Y                 | N                   | N                        | N              | Build algorithm computes difference between two XML documents in linear time. |
| 7    | CX-Diff                | Y              | Y                 | N                   | N                        | N              | CX-Diff identifies customized changes on XML Documents.                     |
| 8    | X-Diff                 | Y              | Y                 | N                   | N                        | N              | X-diff determines the difference between two tree representation of documents based on unordered model. |
| 9    | Vi-Diff                | Y              | Y                 | N                   | N                        | N              | Detect visual representation of web pages based upon semantic differences between two versions of HTML pages. |
| 10   | Level order Traversal | Y              | Y                 | N                   | N                        | N              | Compare DOM tree representation based on the RMS value of the content.      |
| 11   | Tai                    | Y              | Y                 | N                   | N                        | N              | Perform tree to tree comparison based on longest common                      |
| 12 | LA-Diff | N | Y | N | N | N | N |
| 13 | MH-Diff | Y | Y | N | N | N | N |
| 14 | XyDiff | Y | Y | N | N | N | N |
| 15 | DiffX | Y | Y | N | N | N | N |
| 16 | MM-Diff | Y | Y | N | N | N | N |
| 17 | XM-Diff | Y | Y | N | N | N | N |
| 18 | KF Diff+ | Y | Y | N | N | N | N |

| 19 | Change Detection by Level | Y | Y | N | N | N | N |

subsequence.

Comparates two LaTeX documents using fast match algorithm.

Comparates two hierarchically structured documents in meaningful manner.

XyDiff uses simple heuristics to select the node and followed by computing and comparing the signature of two nodes.

Performs an isolated tree fragment mapping to identify the largest matching fragments between the two trees.

Find differences between two ordered trees based upon shortest path and fit in main memory.

Find differences between two ordered trees based upon shortest path and fit in external memory.

KF-Diff+ transforms the traditional tree to tree correction into the comparing of the key trees.

Compare preprocessed trees by matching longest path subtree.

3.4 Improved Query system

If the user query about present data or previous data, then it should be delivered exactly and a record of all changed values and unchanged values must be kept for dealing with any query. The delta in change detection algorithms keep track of all the changes and modifications done to the document but the document should also keep an identification of all the past values. Versioning of various documents plays a key role here. Also, the user may ask a query to notify him in case of any specific changes so a trigger should be maintained and fired automatically when delta changes to those specific changes. Proper surveillance and monitoring of documents must be done to improve the query system [23].
3.5 Visualize changes
Displaying and viewing changes is an important component of any change detection system. Visualizing changes allows an individual to easily check the differences that occurred on the web page. Most of the visualization tools display the changes in the same interface as provided by the CDN system [24].

3.6 Patching changes
The change detection algorithms find out the changes between two documents and generate an edit script consisting of mainly delete, insert, and move operations. By applying or patching these operations to one document, it will produce another version of the document.

4. Recent CDN system
Change Detection and Notification (CDN) tools are useful for providing the updated information related to web pages and web documents to the user. Whenever any changes occur in websites these tools play a key role in notifying the subscribed user of it. Some considerations are to be made before notifying users such as when an intended user must be notified, the medium of notification, and what exactly to notify. Nowadays, the CDN system has many more parameters to consider in addition to the above parameters. It includes monitoring an entire web page, part of a web page, or an entire website [25]. It also focuses on which documents to notify such as pdf file, image file, text file, etc. and how frequently to notify. According to these considerations various CDN systems are available and currently used which are as follows:

4.1 WebVigiL (2006)
It is a very popular tool for identifying the changes made in an HTML document or XML document. It retrieves the essential changes and notifies the user on time. It uses the ECA rules and push/pull technology for monitoring changes in the documents over the web [26].

4.2 Distill (2013)
This change detection tool automatically monitors the changes in websites. Users can view the updated data any time they require. It also provides the facility of alert messages, emails, notifications, and pop-ups on the phone or any other devices by registering the user’s account. It maintains a version history of documents as well as provides data in any format user requires.

4.3 Visualping (2017)
It is just one click tool that notifies the user about the changes on the website itself. Users can select the content on the website for which to keep track of and in case of any modifications in that part, a user is sent an email alert. As the name suggests, in visual ping users can compare the data in visual mode also. Also, users can track the entire webpage, in that case when any changes occur on the webpage; it is highlighted for easy reference.

4.4 Wachete (2014)
It performs complete as well as selected content monitoring on the webpage. Change notifications are sent to the intended subscriber via email of notification on APPs. It can also monitor protect pages where the user needs to log in and provide their user credentials.

4.5 ChangeTower (2017)
It is a cloud-based CDN tool. It is automated and monitors the website. It has all the features of a modern CDN such as visual compare, code compare, text compare, snapshots, and alerts. They have a powerful monitoring tool that tracks any changes that are made to the website. The change may include keyword, image, visual, sentence, or anything which can be asked for. It uses ChangeTower’s feed for notification.
4.6 Pagescreen (2018)
It is a highly popular tool with all the functionalities captured in this single tool. It captures archives and compares the web pages. It is used for business intelligence also as change detection technology it uses is highly effective. It provides facilities for custom alerts. Users can get notification by email or through Webhooks. Webhooks and Page screen of Page screen enables the users to integrate their solutions with the current workflow.

5. Discussion
Change detection of structured documents provides the comparison of content so that user may accept or reject the document with identified changes, multiple versions of big data could be stored easily, and web site content change tracking information. Now web pages are designed and developed using more dynamic content and user-specific layouts which become challenging for change detection algorithms to identify those changes. Considering only static content for change detection would not be beneficial because static pages changes rarely and dynamic pages are getting changed very frequently and their change detection becomes necessary. With the advancements of multimedia content and other visual effects, user interface change detection also becomes a very challenging and cumbersome task. As per the survey performed, the modern change detection algorithms and earlier developed change detection algorithms are more focused on the content change rather than other change perspectives. The major application of the change detection algorithm is to provide information related to changes and sending the same information to the user beforehand. Visual change analysis, patching of changes, querying only changed information, tracking of content changes are some of the additional applications which can be benefitted by the change detection algorithms.

6. Conclusion
This paper has presented an overview of various popular change detection algorithms and their fields of application. A tabular analysis of various algorithms is performed by which it is concluded that CDN systems can identify content change and structural change on the websites or structured documents but much of the work is still required to be done for identifying presentation changes, behavioral changes, type changes, and context changes. If some algorithms or tools can be developed for detecting these changes then it would be much easier for the user to decide whether to adapt according to these changes or stay back with the previous versions. This paper also highlights the various features which are available in recent commercial tools for change detection. These tools notify the user in case of any change in the web documents. They accept user specifications and identify the changes as required by them. These tools use various change detection algorithms to automatically monitor the changes and keep the user updated with all the required information.

References

[1] Cobena G, Abiteboul S and Marian A 2002 Detecting changes in XML documents Proc. - Int. Conf. Data Eng. pp 41–52.

[2] Lee K H, Choy Y C and Cho S B 2004 An efficient algorithm to compute differences between structured documents IEEE Trans. Knowl. Data Eng. 16(8), pp 965–979.

[3] Radlinski F, Bennett P N and Yilmaz E 2011 Detecting duplicate web documents using clickthrough data Proc. 4th ACM Int. Conf. Web Search Data Mining, WSDM 2011, pp147–156.

[4] Mallawaarachchi V, Meegahapola L, Madhushanka R, Heshan E, Meedeniya D, and Jayarathna S 2020 Change detection and notification of web pages: A survey ACM Comput. Surv. 53(1) pp 1–35.
[5] Buttler D A short survey of document structure similarity algorithms Proc. Int. Conf. Internet Comput. IC’04 1 pp 3–9.

[6] Francisco-Revilla L, Shipman F, Furuta R, Karadkar U and Arora A 2001 Managing change on the Web Proc. ACM Int. Conf. Digit. Libr. pp 67–76.

[7] Dalal Z, Suvendu D, Pratik D, Luis F, Richard F, Unmil K and Frank S 2004 Managing distributed collections: Evaluating Web page changes, movement, and replacement Proc. ACM IEEE Int. Conf. Digit. Libr. JCDL 2004 pp160–168.

[8] Borgolte K, Kruegel C and Vigna G 2014 Relevant change detection: A framework for the precise extraction of modified and novel web-based content as a filtering technique for analysis engines WWW 2014 Companion - Proc. of Int. Conf. World Wide Web 23 pp 595–598.

[9] Cobena G, Abdessalem T and Hinnach Y 2002 A comparative study for XML change detection BDA pp 1-20.

[10] Chakravarthy S and Hara S C H 2006 Automating change detection and notification of web pages Proc. - Int. Work. Database Expert Syst. Appl. DEXA pp 465–469.

[11] Wang Y, David J AND Cai J Y 2003 X-Diff: A Fast Change Detection Algorithm for XML Documents Int. Conf. Data Eng. ICDE 2003, 19 pp 519–530.

[12] Pehlivan Z, Ben-Saad M, and Gançarski S 2010 Vi-DIFF: Understanding web pages changes Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) 6261(1) pp 1–15.

[13] Yadav D, Sharma A K, and Gupta J P 2008 Change Detection in Web Pages,"International Conference on Information Technology Change 10 pp 265–270.

[14] Ronnau S, Philipp G, and Borghoff U M 2009 Efficient change control of XML documents Proc. 2009 ACM Symp. Doc. Eng. pp 3–12.

[15] Dohrn H and Riehle D 2014 Fine-grained change detection in structured text documents,” DocEng 2014 - Proc. 2014 ACM Symp. Doc. Eng pp 87–96.

[16] Peters L 2005 Change Detection in XML Trees: a Survey Third Twente Student Conf. IT, pp 1–8.

[17] Al-Ekram R, Adma A and Baysal O 2005 diffX: an algorithm to detect changes in multi-version XML documents Proceedings of the 2005 conference of the Centre for Advanced Studies on Collaborative research pp 1–11.

[18] Chawathe S S 1999 Comparing hierarchical data in External Memory Proc. of Int. Conf. Very Large Data Bases 14(17) pp 90–101.

[19] Xu H, Wu Q, Wang H, Yang G and Jia Y 2002 KF-Diff+: Highly efficient change detection algorithm for XML documents Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) 2519 pp 1273–1286.

[20] Berlik S and Fathi M 2007 Differences of structured documents - Improving their quality
2007 IEEE Int. Conf. Inf. Reuse Integration pp 486–491.

[21] Thao C and Munson E V Using versioned trees, change detection and node identity for three-way XML merging Software-Intensive Cyber-Physical System 34(1) pp. 3–16.

[22] Rubin J and Chechik M 2013 N-Way Model Merging Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering pp. 301–311.

[23] Tchendji M T, Tadonfouet L and Tchendji T T A Tree Pattern Matching Algorithm for XML Queries with Structural Preferences Journal of Computer Communication 07(1) pp 61–83.

[24] A. Oliveira, Oliveira, Tessarolli A, Gabriel G, Gleiph P, Bruno C, Fernando M, Oliveira M, Rodrigues C, Igor K, Souza M, Murta U, Braganholo L and Vanessa 2018 An efficient similarity-based approach for comparing XML documents Information System 78 pp 40–57.

[25] Oh H, Jalali L and Jain R 2015 An intelligent notification system using context from real-time personal activity monitoring International Conference on Multimedia and Expo pp 1-6

[26] Chakravarthy S, Jacob J, Pandrangi N and Sanka A 2002 WebVigil: An approach to just-in-time information propagation in large network-centric environments CEUR Workshop Proc., 702, pp 98–109.