Data & Knowledge Engineering 72 (2012) 257–284

Contents lists available at SciVerse ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

# A change detection system for unordered XML data using a relational model

Sathya Sundaram, Sanjay K. Madria *

*Department of Computer Science, Missouri University of Science and Technology, Rolla, MO, USA*

ARTICLE INFO

*Article history:*
Received 19 November 2010
Received in revised form 5 November 2011
Accepted 7 November 2011
Available online 15 November 2011

*Keywords:*
XML
Change detection
Edit distance
Tree comparison
SQL

ABSTRACT

The dramatic increase in the evolution of XML data available on the Internet requires a change detection system to keep track of important changes occurring during their life time. In this paper, we introduce a novel approach of detecting changes between two versions of unordered XML data stored in a traditional relational database using approaches like XRel. Most of the existing work in the area of XML change detection is mainly focused on detecting changes between two versions of XML data by constructing their Document Object Model (DOM) trees and then comparing these two tree structures based on Longest Common Sequence (LCS) using minimum edit distances. The basic tree comparison approach is not efficient in handling large XML files due to the fact that (1) an equivalent XML DOM tree will be twice as large as the original document and (2) the entire trees of both versions have to be memory resident during the comparison process. These two issues are constrained by the available main memory. In addition, existing approaches fail to detect changes among versions of XML data stored in relational databases as reverse mapping is not loss-less. We propose an efficient algorithm (XRel_Change_SQL) for detecting unordered changes between two XML data files stored in XRel as the underlying relational data model, using Structured Query Language (SQL). We compare the efficiency and quality of our change detection algorithm with existing XML change detection tools like X-Diff, DeltaXML and XANDY. We provide an experimental evaluation of the results obtained from the benchmark datasets as well as some synthetic datasets to show that our approach is highly scalable, and results in a much better efficiency and delta quality than the aforementioned approaches and tools.



## 1. Introduction

XML has a semi-structured format for representing data and documents. In many web applications, users are not only interested in current versions of XML data, but also in the past and future updates. The deltas (changes) among versions can be used in a wide variety of applications such as in market data analysis, versioning systems, and content management systems. For example, in a content management system, to construct a complex view from the versions of data from different sources, a change detection tool is a necessity.

There are two types of XML documents; data-centric and document-centric. Data-centric is structured data wrapped inside XML and has a pre-defined record oriented structure with a one-to-one relationship with the data it represents. Most data-centric XML is generated from structured sources like RDBMSs or legacy applications capable of producing structured data. Document-centric XML is semi-structured text with either no XML Schema (no pre-defined element) or have a frequently changing schema. We are mainly focused on data-centric XML where the schema is assumed to be fairly structured and does not change often but the data does. Further, we believe that most data-centric documents are unordered where only ancestor relationships are considered significant (left-right ordering among siblings are considered insignificant) since they are used in database

* Corresponding author.
*E-mail address:* madrias@mst.edu (S.K. Madria).

0169-023X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2011.11.003

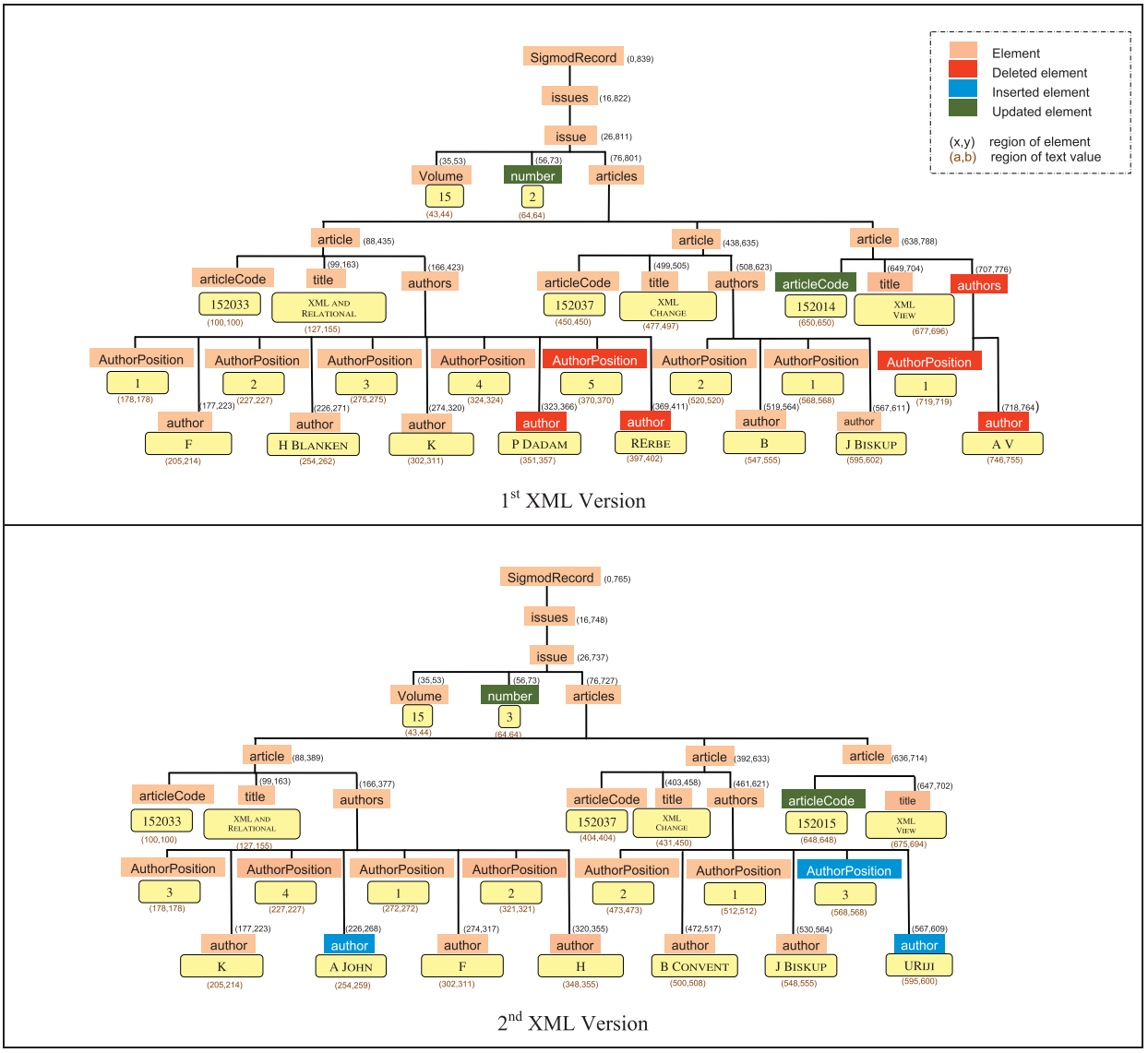


**Fig. 1.** DOM trees of the versions in Table 3.

applications and therefore, we consider changes in data-centric XML in our work. Note that our approach may be suitable for document-centric XML change detection, where 'round-tripping'[1] is not present. For storing data-centric XML documents, relational databases have many advantages due to their proven scalability and speed up. In our approach, we have modified XRel [31] for storing XML documents which uses a fixed relational schema to help in the change detection process.

The change detection algorithms (GNU Diff, HTML Diff) and [7] based on Longest Common Subsequence (LCS) [17] are not suitable for XML documents since they are hierarchical and semi-structured in nature. Much of the other previous work on change detection is not scalable because it works by comparing two XML DOM tree versions; unfortunately tree comparison problem is NP-Hard [20]. DiffXML [9], XyDiff [5] and DocTreeDiff [23] are algorithms for detecting changes among ordered XML documents, whereas X-Diff [28], CDL [24] and DeltaXML [11,12,18] handle the more difficult problem of detecting changes between unordered XML documents.

In order to use existing tree-based comparison methods to detect changes among unordered XML data stored in a relational model, data need to be converted back to their XML representation from their current formats. Then the two versions of XML documents in question are parsed and represented as DOM trees which have to be memory resident during the comparison process. This is clearly not a scalable approach and in addition it has two significant shortcomings. First, any reverse mapping of shredded

[1] 'round-tripping' is a term used when we are unable to generate the exact XML document from a relational database.