Semantic and Structure Based XML Similarity: The XS3 Prototype.

Conference Paper · January 2006
Source: DBLP

CITATIONS
0

READS
51

3 authors:

Joe Tekli
Lebanese American University
38 PUBLICATIONS 285 CITATIONS

Richard Chbeir
Université de Pau et des Pays de l'Adour
187 PUBLICATIONS 698 CITATIONS

Kokou Yetongnon
University of Burgundy
126 PUBLICATIONS 613 CITATIONS

Some of the authors of this publication are also working on these related projects:

Project
Modélisation et traitement de requêtes images complexes View project

All content following this page was uploaded by Richard Chbeir on 05 July 2017.

The user has requested enhancement of the downloaded file. All in-text references underlined in blue are added to the original document and are linked to publications on ResearchGate, letting you access and read them immediately.
Semantic and Structure Based XML Similarity:  
The $XS^3$ Prototype

Joe Tekli  
LE2I Laboratory CNRS  
University of Bourgogne  
21078 Dijon Cedex  
France  
joe.tekli@khali.u-bourgogne.fr

Richard Chbeir  
LE2I Laboratory CNRS  
University of Bourgogne  
21078 Dijon Cedex  
France  
richard.chbeir@u-bourgogne.fr

Kokou Yetongnon  
LE2I Laboratory CNRS  
University of Bourgogne  
21078 Dijon Cedex  
France  
kokou.yetongnon@u-bourgogne.fr

1. Introduction
Due to the ever-increasing web availability of XML-based data, an efficient approach to compare XML documents becomes crucial in information retrieval. Such comparison of XML documents has applications in version control (finding, scoring and browsing changes between different versions of a document), change management and data warehousing (support of temporal queries and index maintenance) [3, 4, 5]. XML query systems (finding and ranking results according to their similarity in order to retrieve the best results possible) [10, 12] as well as in the classification/clustering of XML documents gathered from the web against a set of DTDs declared in an XML database (just as schemas are necessary in traditional DBMS for the provision of efficient storage, retrieval, protection and indexing facilities, the same is true for DTDs and XML repositories) [1, 2, 8].

Here, we present our XML comparison prototype $XS^3$ (XML Structural and Semantic Similarity) able to integrate IR semantic similarity assessment in an edit distance structural similarity algorithm, seeking to amend similarity judgments when comparing heterogeneous XML-based documents.

2. System architecture
$XS^3$ prototype (implemented using C#) is made up of four components: a validation component, an edit distance component, a synthetic XML data generator, and a taxonomic analyzer. They are respectively presented in the following subsections.

Figure 1 depicts the overall architecture of our system. XML documents (synthesized via our XML generator or attained from some external source) are tested using our validation component before being passed to the edit distance component for similarity computations. Semantic similarity values between pairs of words/expressions would have to be computed by the taxonomic analyzer prior to the comparison phase, so as to be used (via our SCM) in identifying the semantic relatedness between element/attribute node labels.

![Fig. 1. Overall system architecture](image-url)

2.1 Validation component
The validation component verifies the integrity of XML documents, testing if the documents are well formed with respect to the basic XML syntactic rules [9]. Note that we have considered the following simplifications in order to gain in computation and time complexity:

- XML documents to be treated by the validation component should have attributes appearing as children of their containing elements.
- Empty elements should be handled as PCDATA elements with the constraint of having null contents.
In addition, the validation component transforms documents to their Ld-pair representations [11] in order to be treated by the edit distance component.

2.2 Edit distance component
The edit distance component in XS³ undertakes XML similarity computations following the process developed in our study [11]. It basically comprises of an original operation cost model (SCM), utilized with Chawathe’s classical edit distance algorithm [5], in order to take into account the semantic relatedness of XML element/attribute labels in the comparison process.

2.3 Synthetic XML documents generator
A synthetic XML data generator was also implemented in order to produce sets of XML documents based on given DTDs. The synthetic XML generator accepts as input: a DTD document, a MaxRepeats¹ value designating the maximum number of times a node will appear as child of its parent (when * or + options are encountered in the DTD), as well as an NbDocs value underscoring the number of synthetic XML documents to be produced.

Note that for the sake of computational simplicity, the DTDs to be treated by our generator should respect certain criterion in order to yield well formed documents:

- Attributes must be introduced as children of their containing elements:
  - `<!ELEMENT Employee (#PCDATA)>`  
  - `<!ATTLIST Employee Name CDATA>`  
  The preceding should be replaced by the following:
  - `<!ELEMENT Employee (Name)>`  
  - `<ELEMENT Name (#PCDATA)>`.

- Since attributes are treated as sub-elements, we disregard empty elements:
  - Declarations such as `<!ELEMENT Employee (EMPTY)>` should not appear in the DTDs presented to our generator.

- We also disregard elements without restrictions:
  - Declarations such as `<!ELEMENT Employee (ANY)>` should not appear in the DTDs presented to our generator.

- The “OR” operator (“|”), representing an alternative of elements, is not considered in our generator. We only consider the “AND” operator, representing *sequences*:
  - Declarations such as `<!ELEMENT University (Faculty | Department)>` are not analysed by our generator. Consequently, declarations containing overlapping brackets should not appear in our input DTDs:
  - `<!ELEMENT University (Faculty, (A. Section) | Laboratory)>`.

- Input DTDs should not contain entity declarations:
  - `<ENTITY … >`.

Fig. 2. Sample DTD

Semantic similarity computational details are given in the main paper [11].

Note that we utilized an indexed Oracle 9i DB² table to store, and subsequently access similarity values. In fact, we had considered computing semantic similarity each time it was needed (practically with each edit distance operation execution). However, pre-computing similarity values for each pair of nodes in the taxonomy at hand and, subsequently, managing them via an indexed table proved to be less time consuming. For example, an average of 0.25 seconds per pair-wise semantic similarity assessment was saved, when exploiting a 677 words WordNet-based taxonomy, owing to that procedure.

3. XS³ comparison modules
XS³ enables four comparison functionalities, made available via dedicated comparison modules.

3.1 One to One (1/1) comparison module
It is the basic module of the system upon which are built all other three modules. Its function comes down to comparing one XML document X₁ to another document X₂, the user being able to choose between the intuitive cost model ICM (assigning unit operation costs) mostly used in the literature and our semantic cost model SCM [11] (cf. Figure 3).

[^1]: A greater MaxRepeats value increases the probability of attaining greater size and variability when generating synthetic XML documents

[^2]: Oracle uses the B-Tree indexing technique
3.2 One to Many (1/∞) comparison module

It utilizes the one to one module in order to compare one XML document $X_i$ to the set of XML documents contained in the same folder. This functionality allows the ranking of documents according to their similarity to $X_i$ (cf. Figure 4).

3.3 Many to Many (∞/∞) comparison module

It uses the one to many comparison module in order to compare XML documents contained in the same folder, one by one. This functionality would allow the clustering of similar documents, the clustering phase not being implemented in our prototype (cf. Figure 5).

3.4 Set comparison module

It compares sets of XML documents by computing corresponding average similarity scores (cf. Figure 6). This functionality was proven useful while undertaking our experiments in order to validate our approach [11].
4. XS³ Interfaces

XS³ provides a set of visual interfaces easy to be used and managed. *Figure 7* is a screen shot of its 1/1 comparison interface. The user starts by identifying the cost model to be employed and the XML documents to be compared. Detailed edit distance computations can be depicted if the user so wishes. In addition to the edit distance value and the corresponding similarity value, the time span needed to perform the comparison process is also reported on screen.

![Fig. 7. The One to One comparison interface.](image)

*Figure 8* shows a screen shot of the taxonomic analyzer interface while computing similarity values between words/expressions. After storing a given taxonomy, along with corresponding word occurrences, in dedicated Oracle 9i DB tables, the user utilizes the taxonomic analyzer interface to compute word frequencies as well as corresponding word similarity values. In addition, the user can test the speed of the system, comparing the time to compute the similarity value between a pair of words, with the time to access it in the indexed Oracle 9i DB table dedicated to storing similarity values (if the table is already populated).

XS³ is available for research purposes and can be obtained from the authors.

References