Estimating Similarity of XML Schemas using Path Similarity Measure

Aruna Tiwari
Assistant Professor
School of Computer Engineering,
IIT, Indore, India

Veena Trivedi
Dept. of Computer Engg.
Shri. GS Institute of Tech. & Sci., Indore,
India

ABSTRACT

In this paper, an attempt has been made to develop an algorithm which estimates the similarity for XML Schemas using multiple similarity measures. For performing the task, the XML Schema element information has been represented in the form of string and four different similarity measure approaches have been employed. To further improve the similarity measure, an overall similarity measure has also been calculated. The approach used in this paper is a distinguished one, as it calculates the similarity between two XML schemas using four approaches and gives an integrated values for the similarity measure.

Keywords

component; XML Schema, XML Schema Resemblance, XML Schema Matching, Similarity Measure.

1. INTRODUCTION

Nowadays, XML has became popular standard for effectively and appropriately data presentation and interchange through Web. Together with the increasing demand , a huge amount of XML schemas are created which have been employed limitless to express and exchange information among many enterprise applications over the world. This use of XML technology in the data presentation has gradually given rise to the problem of XML Schema matching. Different organizations overall need to exchange information which is mainly all in different XML Schemas and these schemas need to be transformed and evaluated on some similarity parameters for performing various tasks like data integration , clustering, matching etc. As schemas have been used very extensively, requirement arises for developing schema mapping techniques for obtaining structural similarity which can be useful for classification or clustering purpose.

Measuring similarity or distance between two entities is a key step for many knowledge discovery tasks. These have brought challenges for the eff ective and efficient organization of information. One of the techniques useful for organizing the contents is to classifiy them on similarity measures A variety of similarity or distance measures have been proposed and widely applied, such as cosine similarity and the Jaccard correlation coefficient[4][2].

Similarity plays a crucial role in many research fields. Similarity serves as an organization principle by which individuals classify objects, form concepts etc[4][6]. Similarity can be computed at different layers of abstraction: at data layer, at type layer or between the two layers. Measuring similarity or distance between two data points is a core requirement for several data mining and knowledge discovery tasks that involve distance computation[11].

Similarity measure has been a key concern here in context to XML schema matching. XML Schemas are the structural representation of the XML Documents. Schema matching is a schema manipulation process that takes as input two heterogeneous schemas and possibly some auxiliary information, and returns a set of similarities identifying semantically related schema elements.

In practice, schema matching is done manually by domain experts and it is time consuming and error prone. As a result, much effort has been done toward automating schema matching process. This is challenging for many fundamental reasons. According to [11], schema elements are matched based on their semantics. Semantics can be embodied within few information sources including designers, schemas, and data instances. Hence schema matching process typically relies on purely structure in schema and data instances[5]. Schemas developed for different applications are heterogeneous in terms of structure and syntax[1]. To resolve schematic and semantic conflicts, schema matching often relies on element names, element data types, structure definitions, integrity constraints, and data values. However, such clues are often unreliable and incomplete. Schema matching cannot be fully automated and thus requires user intervention, it is important that the matching process not only do as much as possible automatically but also identify when user input is necessary.

Computing Schema Matching may be termed as one of the crucial process in schema manipulation as it can help out to distinguish a range of related schemas in their respective domain[5]. Most of the researchers have talked about the schema matching process based on the semantic approach and structural similarity[1].

Various systems and approaches have recently been developed to determine schema matches, e.g., Autoplex [12], Automatch [5], Clio [22, 16], COMA [7], Cupid[6],Similarity Flooding (SF) [13], and TranScm [14]. While most of them have emerged from the context of a specific application, a few approaches (Clio, COMA, Cupid, and SF), try to address the schema matching problem in a generic way that is suitable for different applications and schema languages. Some similarity schemes have been discussed in [6].

In this paper, an algorithm has been developed which calculates the similarity measures for two XML schema strings which can be used for matching of diverse data structures. The main contributions from the paper are as follows-

The hierarchical structure of the XML Schema is stored in the relational database and has been considered in the form of string from the database. This facilitates the task of information presentation and the computation.

The longest common subsequence (LCS) algorithm has been employed for measuring similitude between strings which takes into account the maximum similar characters encountered in the given strings and returns the matched substring.The proposed algorithm also takes into account the place of the token in a string for calculation of similarity measure The multiple measures used in a proper proportion also generate the overall similarity between two XML Schemas. An impact of the ratio of the similitude measures have been also considered in the analysis and presented in the form of the graphs by using different strings for comparison and also varying the similarity measure values.The organization of this paper is as follows- section II describes the preliminaries of the XML Schemas useful for the development of the algorithm. Section III of the paper discusses the different similarity measures. Section IV discusses the proposed algorithm. Section V is presented with the experimentation carried out and the results generated In section VI, concluding remarks are given.

2. PRELIMINARIES

XML is a markup language for documents containing structured information. XML specifies neither semantics nor a tag set. XML is based on two simple ideas: represent documents and data as trees, and represent the types of documents and data using tree grammars. Tree grammars are represented using DTDs [3] or XML Schema. It is a meta-language for describing markup languages[9][10]. XML provides a facility to define tags and the structural relationships between them. All of the semantics of an XML document will either be defined by the applications that process them or by style sheets. XML also defines the document types called DTDs(Document Type Definition) and XML Schemas that describes the structure of documents. XML Schemas can be considered more powerful than DTDs. Among other things, it uses a uniform XML syntax, supports derivation of document types (similar to sub classing in object-oriented languages),[3] XML Schema is also more complex than DTDs, requiring a couple of hundred pages to describe, as opposed to the thirty or so in the original specification of XML 1.0 (which included DTDs). Schema[7][8].

The use of XML ranges over information formatting and storage, database information interchange, data filtering, as well as web services interaction. Due to the ever-increasing web exploitation of XML, an efficient approach to compare and classify XML-based documents becomes crucial in information retrieval. It has a nested structure and is self describing making use of its own user define tags for a given application[10][9].

3. SIMILARITIES MEASURES

In this paper, the focus is on formalising the problem of structural XML Schema matching. The algorithm developed gives the results based on four ways of the structural matching. The similarity parameters include finding the values using LCS algorithm, considering the position of a element in the tree, altering the position of the string, considering the braces between the elements.

The concept can be explained with an example

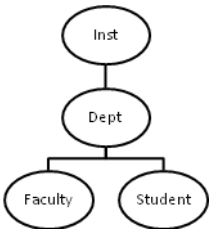

Fig 1.a

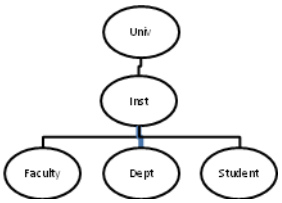

Fig 1.b

The structural information represented by the Figure 1.a and 1.b can be depicted as inst/dept/Faculty/Student and Univ/inst/Faculty/Dept/Student.

The root node is not matching , but still some other node values are in match and thus an overall similarity measure can be generated in the range 0..1

The similarity measure evaluated is on string data and hence the first requirement is the representation of the XML schema information in the form of string which is performed with DFS based search technique.The four similarity measures which are utilized for calculation purpose can be explained below. After calculation of the values, they help for overall calculation of the similarity measure where some fraction of the values are used.

Similarity Measure using LCS

Two strings can be considered representing the structural information of XML Schemas. Let S1 be the string comprising of tokens T11,T12,T13...T1N and S2 be the string comprising of tokens T21,T22,T23...T2M where N and M represent the tokens in the two strings.The Longest Common Subsequence(LCS)algorithm works on the string comparison in an ascending fashion matches character by character. The similarity measure is given by

$$Sim1(S_1, S_2) = 2 * depth(LCS)/[depth(S_1) + depth(S_2)] \text{ ---(1)}$$

Similarity Measure with String Position

The starting position i.e. root element of the string can be also varying and thus the position can also be a considerable factor and it can be calculated based on the current and the optimal position

$$Sim2(S1,S2)=1+(\log10(calculated+optimal)/(calculated+optimal)\ (2*optimal+1)) \text{ ----------(2)}$$

This similarity measure calculates the similarity Values as 1 in case of exact match, and >1 for similar matches

Similarity Measure with breaches in the string token values

Another aspect to be considered in the alteration in context to the variation in the string values. As shown in the Fig 1.a and Fig 1.b–, the position of the token department is at different levels, where in one level it is treated as a parent node–, whereas it is treated as a peer token to others. This aspect of similarity measure is for considering the breaches between the tokens in ascending order as given by the LCS algorithm. In the LCS algorithm , all the tokens with the complete token match are considered.

Sim3 = | (1.0/(1.0-gap)) |----------(3)

In case of all matches a value of zero is generated depicting that all the tokens in the two strings are identical

Similarity Measure with alteration in the element position

Another aspect which is often left unattended is the hierarchy order of the token in the string. For ex.

/Univ/inst/faculty and /Univ/faculty/inst.

In the above two strings , the LCS algorithm shows a match of only two tokens , due to the alteration of the hierarchical order. This aspect can be managed with the similarity measure which also considers with the match at different levels.

No_match = Total_no_of_token match_between_String S1 and S2

Distance_similarity = Sum_of_all Matches based on token(1.0 – 0.2*|Token1i –Token1j |))

Sim4 = Distance_similarity/no_match ----------(4)

Thus the overall similarity can be generated taking into account all the four similarity measures

Overall similarity = α*sim1+ β*sim2+ γ*sim3+ δ*sim4)-(5)

Where α, ß, λ and δ are positive parameters ranging between 0 and 1 that represent the impact of each approach. Here the values associated with α, β, γ, δ can be altered depending upon the application requirement.

4. ALGORITHM

For finding similarity between the two strings, the equations generated in the previous section have been employed to develop an algorithm for calculating the similarity measure. The following algorithm summarizes the computation of similarity measure using the above formulas

```
Input: String S1 , S2
Output: Overall Similarity(S1 , S2 )
Begin
//Calculate Similarity 1 using LCS
Sim1 =2.0*|LCS(S1 , S2 )|/ |S1|+|S2|
//Calculate Similarity 2
Sim2 = 1.0-√ (|log10(calculated+optimal)/2*optimal)|
//Calculate Similarity 3
Sim3 = |1.0/(1.0-gap) |
//Calculate Similarity 4
No_match = Total_no_of_token match_between_String S1 and S2
Distance_similarity = Sum_of_all Matches based on Tokens( 1.0 – 0.2*|Token1i –Token1j |))
Sim4 = Distance_similarity/no_match
overall_similarity = α*sim1+ β*sim2+ γ*sim3+ δ*sim4);
print overall_similairty
End
```

5. EXPERIMENTATION AND RESULT

Experiments are carried out on a Intel Processor with 3.00GHz and 4GB RAM memory. The algorithm has been implemented in Java. The example XML Schemas shown in the Figure 1.(a) and (b) have been used for checking the performance of proposed algorithms.

In the match process, following observation can be made for the two strings under study. The String S2 is taken in a fashion such as to have two/three/../n token similarity and the overall Similarity is noted which is presented in the following table for few cases

S1 = univ/inst/faculty/dept/stud

I. String S2	II. Overall Similarity
III. Univ/inst/faculty/dept/stud	IV. 1.0
V. Inst/univ/ faculty/dept/stud	VI. 0.87
VII. Inst/faculty/dept/stud	VIII. 0.78
IX. Inst/dept/faculty/stud	X. 0.69
XI. Univ/dept/faculty/stud	XII. 0.69
XIII. Univ/faculty/dept/stud	XIV. 0.82
XV. College/faculty/dept/stud	XVI. 0.7
XVII. Inst/dept/stud	XVIII. 0.66
XIX. Univ/faculty/dept	XX. 0.78

Table 5.1

The tabular comparison can be presented in a form of graph as shown in the Figure 5.1

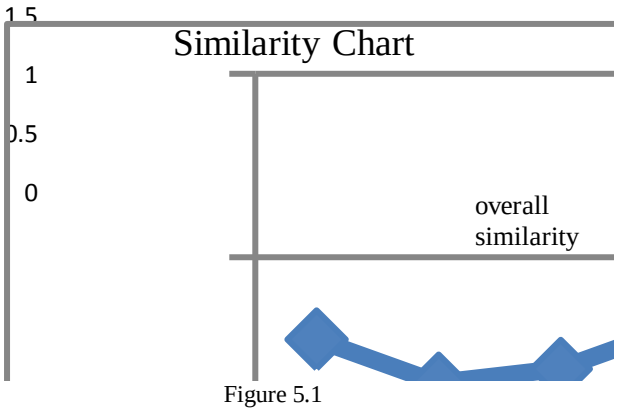

Figure 5.1

Even in the matching process, an optimal values of α, β, γ, δ is to be selected with good judgement which also gives variation in the results.

The String S1 is considered and a match string is considered, similarity has been considered by altering the values of parameters. The results are shown in the Table 5.2

S1 = univ/inst/faculty/dept/stud
S2 = inst/faculty/dept/stud

Table 5.2

α	β	γ	δ	Overall similarity
0.80	0.07	0.07	0.06	0.83
0.75	0.10	0.08	0.07	0.82
0.70	0.15	0.07	0.08	0.81
0.65	0.20	0.07	0.08	0.80
0.60	0.18	0.11	0.11	0.78
0.55	0.20	0.12	0.13	0.77
0.55	0.15	0.15	0.15	0.76
0.50	0.20	0.15	0.15	0.75
0.50	0.15	0.20	0.15	0.73
0.50	0.25	0.10	0.15	0.77
0.50	0.25	0.15	0.10	0.75

Table 5.2

The results can also be depicted in a graphical form as shown in Figure 5.2

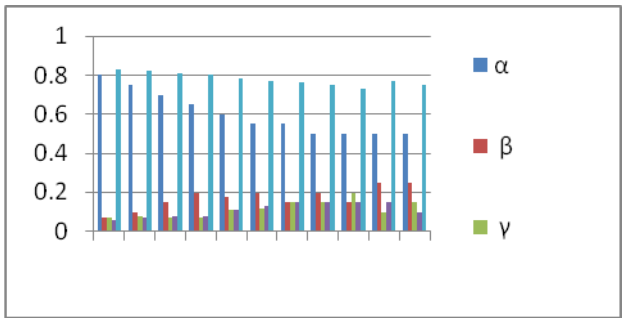

Figure 5.2

6. CONCLUSION

In this paper, an algorithm for similarity measure has been proposed and implemented using Java language. The similarity has been evaluated using four measure namely Longest Common Subsequence(LCS), the token positioning, the breaches involved and the tokens with the order altered. It has been found that the proposed algorithm for similarity matching produces better results than the results presented in [6] by Amar Zerdazi and Myriam Lamolle. In the paper presented by Amar Z.[6] results have been carried out with a fixed set of values for α, β, γ, δ parameters and variations have not been considered. It has been also observed with experimentation that the value of α, β should be preferably more than γ, δ. During experimental study it was observed that the similarity approaches associated with α, β caries more significance. On the other hand the contribution of the other two parameters i.e. γ, δ can not be neglected as it helps in the calculation when the tokens are not placed in order. Using the value obtained in the overall similarity, the proposed algorithm can be useful in clustering of the similar XML Schemas.

7. REFERENCES

[1] Algergawy, Alsayed and Nayak, Richi and Saake, Gunter (2009) XML Schema Element Similarity Measures: A Schema Matching Context. In: On the Move to Meaningful Internet Systems: OTM 2009, November 1-6, 2009, Vilamoura, Portugal, Springer Verlog publication .

[2]Anna Huang, Similarity Measures for Text Document Clustering, New Zealand Computer Science Research Student Conference ,NZCSRSC 2008 Christchurch, New Zealand, April 14–18, 2008 Proceedings

[3] Allen Brown, Matthew Fuchs, Jonathan Robie, Philip Wadler, Avaya Labs, "MSL:A model for W3C XML Schema",ACM,2005

[4] Carsten Keßler,,Similarity Measurement in Context, LNAI 4635, pp. 277–290, 2007,Springer-Verlag Berlin Heidelberg 2007

[5] Zerdazi, A., Lamolle, M.: Matching of Enhanced XML Schema with a measure of structural-context similarity. In: Proceeding of The 3rd International Conference on Web Information Systems and Technologies, WEBIST 2007 (2007)

[6]Amar Zerdazi and Myriam Lamolle, Computing Path Similarity Relevant to XML Schema Matching, LNCS 5333, pp. 66–75, 2008.© Springer-Verlag Berlin Heidelberg 2008

[7] Irena Mlynkova, Jaroslav Pokorny, From XML schema to Object Relational Database –An XML schema driven mapping algorithm,2004

[8] Ramon Lawrence, A Cost-Based Approach For Converting Relational Schemas To XML, IADC-2005 - International Advanced Database Conference (IADC-2005)

[9] WWW Consortium, Extensible Markup Language (XML) 1.0, February, 2004, hhttp://www.w3.org/TR/REC-xml/i.

[10] WWW Consortium, XML Schema Part:0 Prime, October,2004, hhttp://www.w3.org/TR/xmlschema-0/i

[11] Shyam Boriah Varun Chandola Vipin Kumar," Similarity Measures for Categorical Data: A Comparative Evaluation",SIAM International Conference on Data Mining, SDM 2008

[12] Do, H.H., E. Rahm: COMA – A System for Flexible Combination of Schema Matching Approach. VLDB 2002

[13] Melnik, S., H. Garcia-Molina, E. Rahm: Similarity Flooding: A Versatile Graph Matching Algo-rithm. ICDE 2002

[14] Milo, T., S. Zohar: Using Schema Matching to Simplify Heterogeneous Data Translation. VLDB 1998, 122–133