XML Schema Matching – Using Structural Information

A.Rajesh
Research Scholar
Dr.MGR University, Maduravoyil, Chennai

S.K.Srivatsa
Sr.Professor
St.Joseph's Engineering College, Chennai

ABSTRACT
Schema matching is the task of finding semantic correspondences between elements of two schemas. It takes two schemas as input and returns a mapping that identifies corresponding elements in the two schemas. Schema matching is an important and vital step in many schema and data translation and integration applications, such as integration of web data sources, data warehousing, XML message mapping etc. In this paper, we describe different characteristics exhibited by matching element pairs at various levels in the XML schema and propose a system which uses these characteristics to perform the matching process. The novelty in the system is the architecture of the system which comprises of a linguistic matcher in combination with a set of filters operating based on the level of the element pairs to be matched in the source and target XML schemas.

General Terms
Datamining, Datawarehousing, Information Retrieval, XML Message Mapping

Keywords
Schema Matching, Hybrid Schema Matching, Schema Integration, XML Schema Matching, XML Schema Integration, Semantic Matching.

1. INTRODUCTION
Schema matching is the task of finding semantic correspondences between elements of two schemas [4,5,6,8]. It takes two schemas as input and returns a mapping that identifies corresponding elements in the two schemas [6]. Schema matching is an important and vital step in many schema and data translation and integration applications, such as integration of web data sources, data warehousing, XML message mapping etc. Schema matching is a challenging task due to structural and naming conflicts present even in identical schemas and different data models used to build the schemas.

In this paper, we propose a schema matching system for XML schemas that operate based on the characteristics exhibited by matching element pairs at various level in the XML schema. We have concentrated on matching XML schemas due to the increased popularity of XML model and the huge proliferation of XML documents online. The system operates on schema trees. That is the XML document structure is parsed to form a schema tree as shown in Figure 1 which is given as input to the system. The system comprises of a basic linguistic matcher followed by a set of filters which extract the matching element pairs from the candidate match pairs generated by the linguistic matcher.

The rest of the paper is organized as follows: section 2 gives an overview of schema matching, section 3 gives an overview of our proposed system, section 4 gives in detail the composition of the system, section 5 gives the experimental evaluation of the system and section 6 gives the conclusion.

2. SCHEMA MATCHING OVERVIEW
A recent survey of the schema matching algorithms in the literature indicate that these techniques in general rely on the following factors: label similarity, structural similarity, constraint similarity, and in addition may also use auxiliary information such as dictionaries, thesauri and domain specific ontology [1,2,3,4,5,6,7,8,9] for the purpose of matching. Some of these techniques use only Schema information – schema based [1,2,3,5,6,7,9] – for identifying matches and some use instance level – instance based [5,8] – for identifying matches. A still broader taxonomy of the schema matching approaches is presented in [10].

We just discuss a few of the prominent approaches for schema matching problem to give an idea regarding the progress of the research in this area and also to highlight how our proposed system differs from these existing approaches.

LSD[10] (Learning Source Descriptions) uses a machine learning approach for schema matching problem. It uses a set of base learners to learn linguistic, structural, data and domain specific information. The results from the base learners are given as input to a meta learner which determines the match between the schema elements.

Cupid[6] is a hybrid matching system that uses a combination of linguistic and structural matching methods to perform the matching process. The linguistic method proposed in the system tokenizes the label of the given elements and determines the similarity between the elements based on the number similar tokens between the labels of the elements. The structural approach determines the similarity between the elements based on the number of similar leaf elements in the subtree originating from these elements. The final similarity between elements is computed by adding weighted linguistic and structural similarity measure.

COMA[4] is a composite schema matching system which proposes an architecture to include multiple matching algorithms to perform the schema matching process. It also proposes methods to combine the results of the various matching algorithms in the system and determine the similarity between the element pairs.

QMatch[9] is a hybrid matching system which uses a combination of linguistic, property, and structural matching approaches to perform the matching process. The structural matching process uses a path based approach dominated by the number of similar child elements to determine the similarity of between the elements in the schema.

Based on the analysis of the various approaches used for schema matching process in the literature, we have identified certain characteristics exhibited by similar elements. These characteristics vary depending on the level of occurrence of the elements within the XML schema – leaf, interior nodes, and root. In the following sections we would just see the similarity characteristics exhibited by the elements at the various levels in the schema tree.

3. SIMILARITY CHARCTERISTICS

The similarity characteristics exhibited by the elements at different levels in the XML schema are described in the following subsections.

Leaf – Leaf similarity

When two leaf elements are found to be similar, the respective paths to these elements from the root are also found to be similar. Here by similarity of path we mean that the paths share the highest number of similar elements between them. For example, the leaf elements line and lineNumber in Figure 3 are similar. The paths to these elements po-poLines-item-line and purchaseOrder-items-item-lineNumber are also similar as they share the maximum number of similar elements i.e., po:purchaseOrder, poLines:items, item:item, line:lineNumber.

Root – Root similarity

When two root elements are found to be similar, the trees originating from these roots share the highest number of similar leaf elements between them. This doesn't mean that the elements other than the leaf are dissimilar, they also can be similar. But we give emphasis to leaf similarity as the maximum information is conveyed only by the leaf elements. For example, the root elements po and purchaseOrder in Figure 2 are similar. They also share the highest number of similar leaf elements in the trees originating from them i.e., street:street, city:city, line:lineNumber, qty:quantity, uom:unitOfMeasure.

Interior – Interior similarity

When two interior elements are found to be similar, they share a large number of similar direct descendants and / or similar siblings between them. Since descendants and siblings provide a context to the elements to be matched, similar elements reside in similar contexts.For example, the interior elements poLines and items in Figure 2 are similar. They also share the highest number of descendants – item:item,count:itemCount and highest number of siblings – poShipTo:deliverTo, poBillTo:invoiceTo.

Interior – Root similarity and vice – versa

When an interior element is found to match a root element or vice – versa, they share a large number of similar descendants between them. In a hierarchical structure, the descendants usually are specific descriptions of their parent elements. Hence similarities of the children indicate similarity of their parents. For example, the elements purchaseInfo and purchaseOrder in Figure 1 are similar. They also share the highest number of descendants – shippingAddr:shipTo, billingAddr:billTo, lines:items.

Interior – Leaf Similarity and vice – versa

When an interior element is found to match a leaf element or vice – versa, they share a large number of similar siblings between them. Again we can say that since siblings of an element give the context in which the element is present, elements in similar context are more likely to be similar. For example, the elements day and day in Figure 3 are similar. They also share the highest number of similar siblings – section:section, title:title, instructor:instructor, credits:units, place:place.

In the proposed matching approach we have utilized these characteristics to match the elements between the two schemas. The novelty in this approach is that we apply different matching strategies depending on the characteristics exhibited by similar elements at leaf, interior, and root levels in the XML schema. The matching strategies are implemented as a set of filters which extract the matching pairs from the source and target XML schemas.

4. MATCHING PROCESS

We first measure the linguistic similarity between the elements of the two schemas. Any linguistic matching approach can be used in the base matcher. The element pairs whose linguistic similarity measures exceed a predefined threshold (lt) are taken as the candidate matching pairs. These candidate matching pairs are filtered using a set of filters to obtain the final set of matching pairs. These filters operate based on the characteristics exhibited by the matching element pairs at various levels in the schema tree. The various filters used in the proposed system are described in the following subsections

4.1 Leaf – Leaf match filter

This filter extracts from the candidate matching pairs those pairs which satisfy the following conditions,

- Both elements that form the pair are leaf elements
- The paths to these elements also match

Here matching path means that the paths to these leaf elements must share the highest number of similar elements between them. The path match between any two paths is computed as (Equation 1),

4.2 Root – Root match filter

This filter extracts from the candidate matching pairs those pairs which satisfy the following conditions,

- Both elements that form the pair are root elements
- The trees originating from these root elements share the highest number of similar leaf elements

The similarity between the root elements is computed as (Equation 2),

4.3 Interior – Interior match filter

This filter extracts from the candidate matching pairs those pairs which satisfy the following conditions,

- Both elements that form the pair are interior elements
- Both share the highest number of similar descendants and siblings

The similarity between the two interior nodes is computed as (Equation 5),

4.4 Interior – Root match filter

This filter extracts from the candidate matching pairs those pairs which satisfy the following conditions,

- One of the elements in the pair is an interior element and the other a root element or vice – versa
- Both share the highest number of similar descendants (Equation 3)

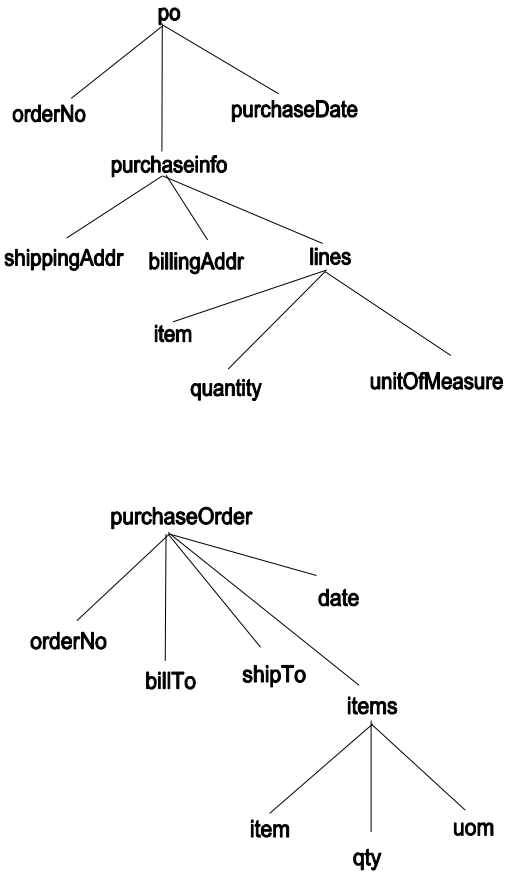

Figure 1 An example schema tree of purchase order schema

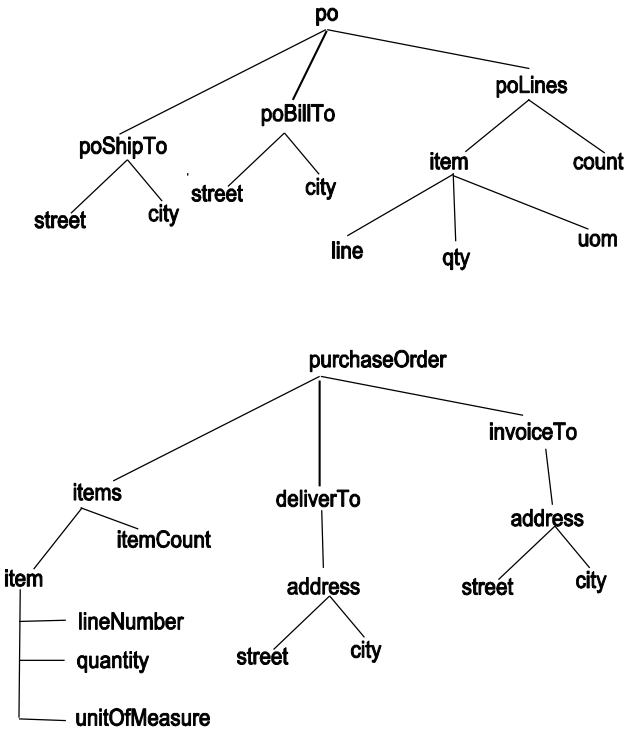

Figure 2 po, purchaseOrder schema variant 2

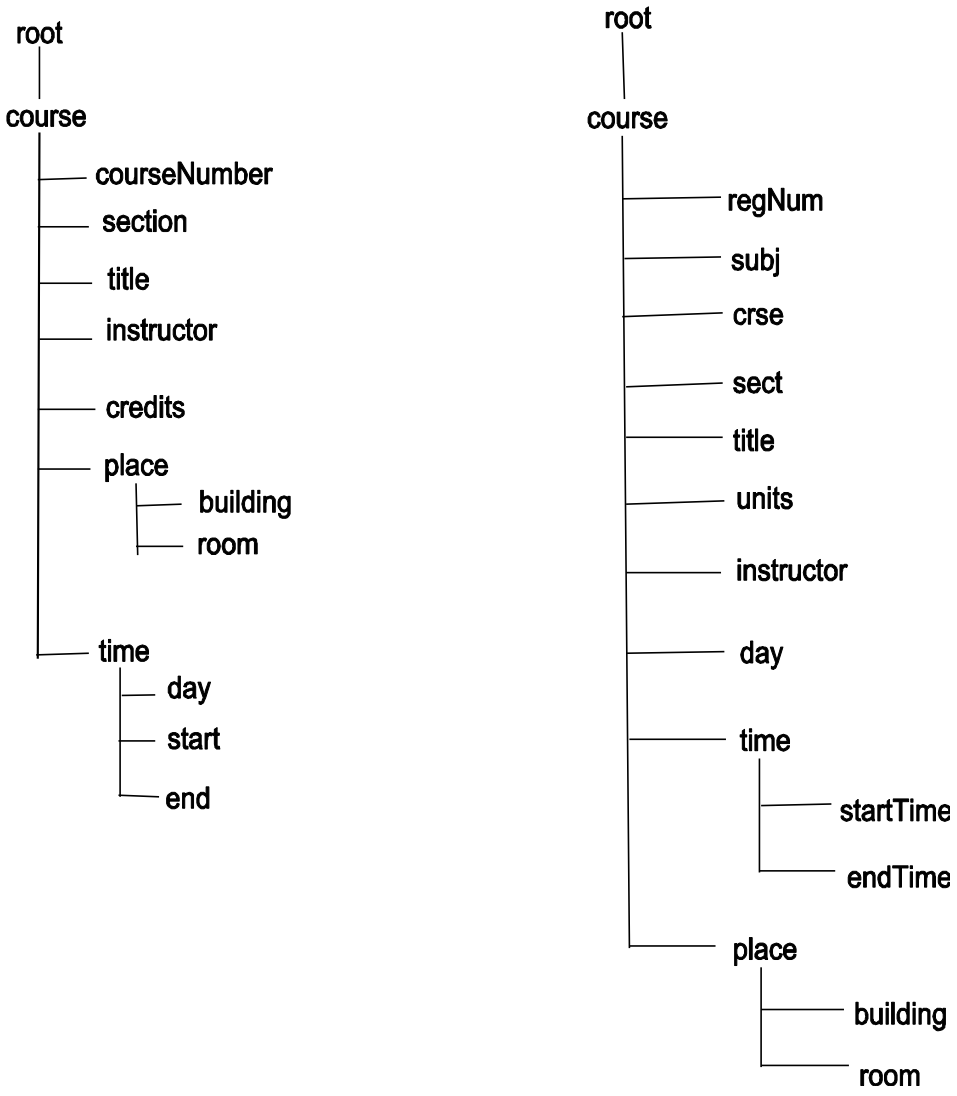

Figure 3 course domain schema variant 1

$$\text{Path similarity} = \frac{\text{Number of linguistically similar elements between the source and target paths}}{\text{Total number of elements in the source and target paths}} \quad (1)$$

4.5 Interior – Leaf match filter

This filter extracts from the candidate matching pairs those pairs which satisfy the following conditions,

- One of the elements in the pair is an interior element and the other a leaf element or vice – versa
- Both share the highest number of similar siblings (Equation 4)

5. EXPERIMENTAL EVALUATION

We use the architecture shown in Figure 4 to test our approach. The aim our experimentation is to evaluate the performance of our system and compare its performance with that of the other matching systems performance. By this we try to emphasize the performance gain achieved using the proposed system.

5.1 Methodology

We follow the following methodology to evaluate the proposed system.

- Identify data sources to test the system
- Evaluate the system using the test data sources
- Compare the performance with other matching systems

5.2 Data Sources

The test data for the system are schemas with varying characteristics used extensively to test similar systems in the literature. The characteristics of the test data are shown in Table I.

5.3 System Evaluation

The parameters used to fine tune the system are,

- Linguistic similarity threshold (lt)
- Root similarity threshold (rt)
- Descendant similarity threshold (dt)
- Sibling similarity threshold (st)

Only the similarity measures which exceed these predefined thresholds are considered for evaluation. On testing with various values for these parameters we found the system gave optimum performance for the following values:

Lt=0.4, rt=0.5, dt=0.4, st=0.4

The quality of the results is determined by the following factors:

- Precision – The number of real matches identified from among the candidate matches returned by the system. This gives an estimate of the reliability of the match predictions.
- Recall – The number of real matches identified from among the number of real matches identified manually. This specifies the share of real matches discovered by the system.

The results of our experimentation are summarized in the Figure 5. The comparison of system performance with other similar systems is shown in Table II

$$\text{Root Similarity} = \frac{\text{Number of linguistically similar leaf elements in the trees originating from the source and target root elements}}{\text{Total number of leaf elements in the trees originating from the source and target root elements}} \quad (2)$$

$$\text{Descendant Similarity} = \frac{\text{Number of linguistically similar children between the source and target elements}}{\text{Total number of children of the source and target elements}} \quad (3)$$

$$\text{Sibling Similarity} = \frac{\text{Number of linguistically similar siblings between the source and target elements}}{\text{Total number of siblings of the source and target elements}} \quad (4)$$

$$\text{Interior Similarity} = \frac{\text{Descendant Similarity} + \text{Sibling Similarity}}{2} \quad (5)$$

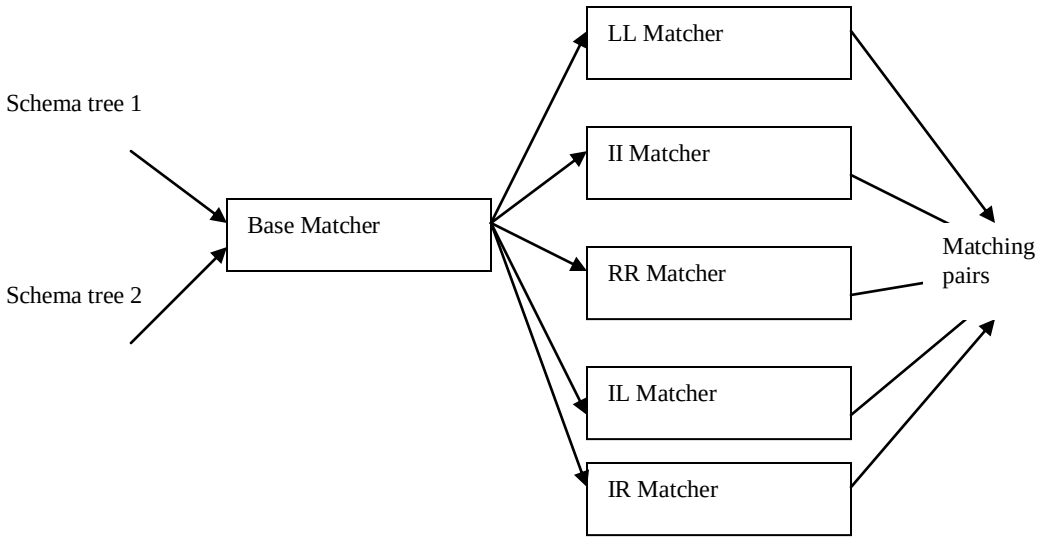

Figure 4: System architecture

Table I: Characteristics of the data sources used in the experimentation

Xml Schemas	No. of elements	Max. depth	No. of leaves
po,purchase Order variant 1	10 x 9	4 x 3	7 x 7
po,purchase Order variant 2	13 x 15	4 x 4	8 x 8
po,purchase Order variant 3	40 x 43	4 x 4	33 x 33
course schemas variant 1	14 x 16	4 x 4	10 x 12
course schemas variant 2	14 x 20	4 x 5	10 x 15
course schemas variant 3	14 x 20	4 x 4	10 x 16
Supplier schemas	17 x 43	5 x 2	10 x 34

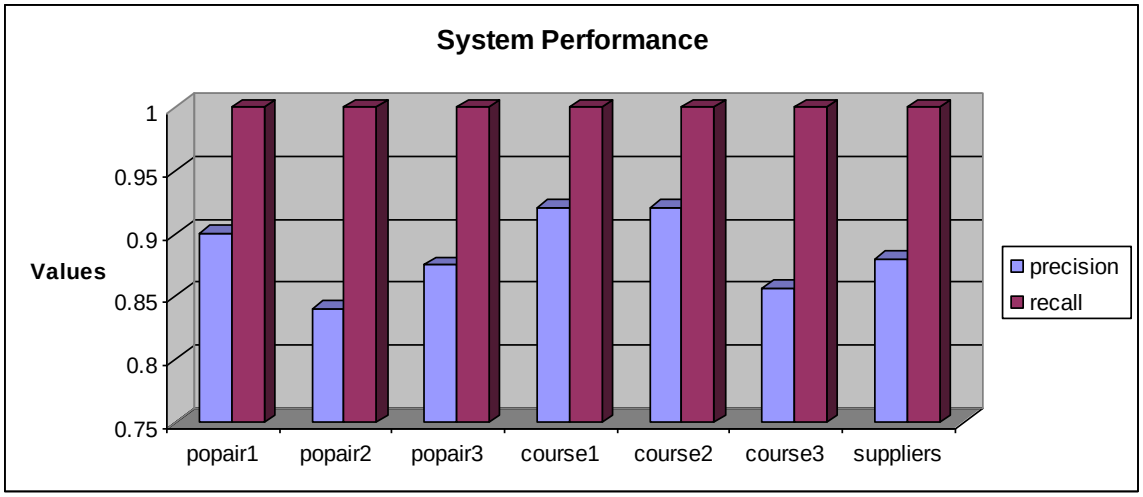

Figure 5 The Precision and Recall values for various schemas

Table II: Comparison of HMAT with other systems in the literature

Xml Schemas	Cupid	Similarity Flooding	COMA	QMatch	Proposed approach
Average Precision	0.66	0.6	0.87	0.83	0.88
Average Recall	1.0	1.0	0.85	1.0	1.0

6. CONCLUSION

Our approach demonstrates a simple and novel method for determining one to one matches between schema elements. This can be extended to determine one to many and many to many matches between schema elements. We have shown the characteristics exhibited by similar elements at various levels in the schema tree and how to use them to aid us in our matching process. This information can be used to design matchers that could operate efficiently at various levels and also utilize this information as features to matchers that could learn to match using these features. We could also try to incorporate user feed back to further fine tune the system performance.

7. REFERENCES

[1] Bergamaschi, S., Castano, S., Vincini, M., Beneventano, D., "Semantic Integration of Heterogeneous Information Sources," Data & Knowledge Engineering, vol. 36, no. 3, pp. 215–249, 2001.

[2] Bright, M.W., Hurson, A.R., and Pakzad, S. H., "Automated Resolution of Semantic Heterogeneity in Multidatabases," in the proceddings of TODS, vol.19, no. 2, pp. 212–253, 1994.

[3] Berlin, J., and Motro, A., "AutoPlex: Automated Discovery of Content for Virtual Databases," in the proceedings of CoopIS, pp.108–122, 2001.

[4] Hong Hai Do and Rahm, E., "COMA - A System for Flexible Combination of Schema Matching Approaches," in the proceedings of Int. Conference on Very Large Data Bases, 2002.

[5] Li, W., Clifton, C., "SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks," Data & Knowledge Engineering, vol. 33, no. 1, pp. 49-84, 2000.

[6] Madhavan, J., Bernstein, P., and Rahm, E., "Generic Schema Matching with Cupid," in the proceedings of Int. Conference on Very Large Data Bases, pp. 49–58, 2001.

[7] Melnik, S., Garcia-Molina, H., Rahm, E., "Similarity Flooding: A Versatile Graph Matching Algorithm," in the proceedings of ICDE, 2002.

[8] Miller, R.J., et a, "The Clio Project: Managing Heterogeneity," SIGMOD Record vol. 30, no. 1, pp. 78-83, 2001.

[9] Naiyana Tansalarak, Kajal T. Claypool, "Qmatch – Using paths to match XML Schemas," Data & Knowledge Engineering, vol. 60, pp. 260 – 282, 2007.

[10] Rahm, E., Bernstein, P.A., "A Survey of Approaches to Automatic Schema Matching," VLDB Journal, vol. 10, no. 4, 2001.