Complex Schema Match Discovery and Validation through Collaboration

Khalid Saleem and Zohra Bellahsene

LIRMM - UMR 5506 CNRS University Montpellier 2,
161 Rue Ada, F-34392 Montpellier
{saleem,bella}@lirmm.fr

Abstract. In this paper, we demonstrate an approach for the discovery and validation of n:m schema match in the hierarchical structures like the XML schemata. Basic idea is to propose an n:m node match between children (leaf nodes) of two matching non-leaf nodes of the two schemata. The similarity computation of the two non-leaf nodes is based upon the syntactic and linguistic similarity of the node labels supported by the similarity among the ancestral paths from nodes to the root. The n:m matching proposition is then validated with the help of the mini-taxonomies: hierarchical structures extracted from a large set of schema trees belonging to the same domain. The technique intuitively supports the collective intelligence of the domain users, indirectly collaborating for the validation of the complex match propositions.

Keywords: Complex Schema Matching, Mini-taxonomies, Collaboration, Tree Mining, Large scale.

1 Introduction

Schema matching relies on discovering correspondences between similar elements of two schemata. Several types of schema matching techniques [8] have been studied, demonstrating their benefit in different scenarios. Schema matching is categorized as *simple element level matching* and *complex structural level matching* [8]. Simple matching comprises of 1:1, 1:n and n:1 match cardinality, whereas n:m match cardinality is considered to be complex. Figure 1 demonstrates the 1:1 (author : writer) and n:m ({FName,LName} : {FirstName,MI,LastName}) match scenarios. Most of the existing approaches and tools give good 1:1 local and global match cardinality but lack the capabilities for discovering n:m matches among the schemata. In this paper, we present an idea for complex match proposition and its validation through an indirect user collaboration technique. We consider the schemata to be rooted, labeled trees, with the tree nodes representing the schema elements. This supports the computation of contextual semantics of the elements in the tree hierarchy. Research literature supports the assumption that there exists a hierarchy among the schema elements of most of the data models. For example, He et al. [5] extracted the hierarchical structures out of the web interface forms and Lee et al. [6] converted the relational database into XML structures, which are inherently trees.

R. Meersman, T. Dillon, P. Herrero (Eds.): OTM 2009, Part I, LNCS 5870, pp. 406–413, 2009.

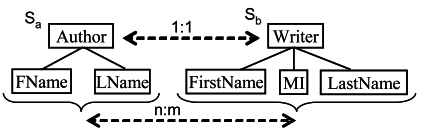

Fig. 1. 1:1 and n:m mappings between two schemata

Our approach is XML/ web interface forms schemata centric, where data lies at the leaf nodes. If a 1:1 correspondence is found between two non-leaf schema elements, we make the proposition that a complex match may exist between the descendant leaf nodes of the respective non-leaf nodes. Next, we utilize an already available repository (automatically generated) of mini-taxonomies [9] to validate the complex match proposition.

Mini-taxonomy is basically a small hierarchical structure representing a concept, used in the input schemata. For example the two element hierarchies given in Figure 1 are mini-taxonomies representing the same concept *Author Name*. Mini-taxonomies repository is built with the help of a tree mining technique [9]. Tree mining extracts similar and frequent sub-trees from a large set of schema trees. Therefore the mini-taxonomies repository is in fact the collection of representations of the domain concepts, most frequently utilized in the respective domain. The validation process thus presents a collaborative support of the domain users, in the matching process.

The approach provides an automated technique with an approximate complex schema match quality [2], supporting a large scale scenario (schemata have large number of elements). The individual semantics of the node labels within the schema trees, have their own importance. We utilize linguistic matching algorithms, based on tokenisation, and synonym and abbreviation tables, to extract the concepts hidden within them.

Our Contributions

In this paper, we focus particularly, though not exclusively, on n:m complex mappings. We present a methodology for matching two schemata, covering the element level and structural level matchings discovery. The main features of our approach are as follows:

1. The approach is based on a tree mining technique. It utilizes the ancestor nodes details of the schema nodes, within the tree mining framework, to enable fast calculation of the contextual (hierarchical) similarity between the schemata.
2. It suggests the possible n:m matches between the two schemata and then validates these propositions using already available mini-taxonomies [9], representing the domain concepts.
3. The mini-taxonomies are automatically extracted from a large set of domain specific schemata using the tree mining technique. Mini-taxonomies represent the domain users perspective about the concepts' structural representation within the schemata. Therefore the technique demonstrates the use of collective intelligence in schema matching, in an indirect collaborative manner.
4. Overall, the approach is automatic and hybrid in nature to support the large scale scenarios.

Outline

The remainder of the paper is organized as follows. Section 2 gives the related work in schema matching, specifically the complex schema matching. In Section 3 we describe the overall architecture of our approach. Section 4 discusses the core of the paper, complex match proposition validation appproach with the help of an example. Section 5 presents the lessons learned from the evaluation of the approach. Section 6 outlines the future perspective and concludes.

2 Related Work

Simple matching with acceptable quality has been successfully demonstrated in [1,4,7] by utilizing the element level and structural level schemata knowledge. There has been very limited work on the complex schema matching.

Formally speaking the complex schema matching is n:m match cardinality problem [8]. But most of the work done in the complex schema matching research domain, revolves around the 1:n match discovery. In n:m match discovery, SCIA [11] is one such work but it is dependent on the manual input during the matching process. Another research, by Embley et al. [3] uses manually created mini-ontologies for the domain specific concepts, to generate n:m correspondences between the two schemata. Work in our paper is similar to [3] but the ontological structures, i.e., mini-taxonomies, are generated automatically [9] and we follow the n:m notion of the complex matching.

3 Complex Match Discovery - Our Approach

The architecture of the approach for complex match discovery is shown in Figure 2. It is composed of four modules: (i) *Pre-Phase*, (ii) *Mini-Taxonomies Generation*, (iii) *Simple Schema Matching*, and (iv) *Complex Match Proposition Validation*, supported by a repository which houses the synonym and abbreviation lists, mini-taxonomies, schemata and match results for future reuse.

The *Pre-Phase* module processes the input schemata as trees, calculating the depth-first node number and the scope (number of nodes in the schema tree rooted at that node), for each of the nodes in the input schema trees [10]. At the same time, for each schema tree a listing of the nodes is constructed, sorted in depth-first traversal order. As the trees are being processed, a sorted global list of labels over the whole set of schemata is created by the *Terms Extraction* sub-module.

In the *Similar Terms Computation and Clustering* sub-module, label concepts are computed using the linguistic techniques. The labels are tokenized and the abbreviated tokens are expanded using an abbreviation oracle. Currently, a domain specific user defined abbreviation table is being utilized. Label comparison is based on the similar synonym token sets, supported by a manually defined domain specific synonym table. The architecture is flexible enough to employ additional abbreviation, synonym oracles or arbitrary string matching algorithms. Similar labels are clustered together, intuitively clustering nodes with similar labels [10].

First task performed by the *Mini-Taxonomies Generation* module is to compute the frequency of each term in the forest of trees. Next, within each labels cluster, the term with the highest frequency in the forest of schema trees is taken as the symbol representing the cluster. The frequency of the cluster symbol is computed by adding the frequencies of all the terms in the cluster. From here on the algorithm executes similar to the frequent sub-tree mining algorithm given in [12]. The cluster representative symbols act as the starting labels for the data structure storing frequent sub-tree patterns. The output of the process is a list of the sets of mini-taxonomies. Each list representing a set of mini-taxonomies of same size. Next, these mini-taxonomies lists are replicated by replacing the cluster level similar labels in the list, thus producing all the possible mini-taxonomies which can be considered as the concept representation, frequently utilized by the domain users.

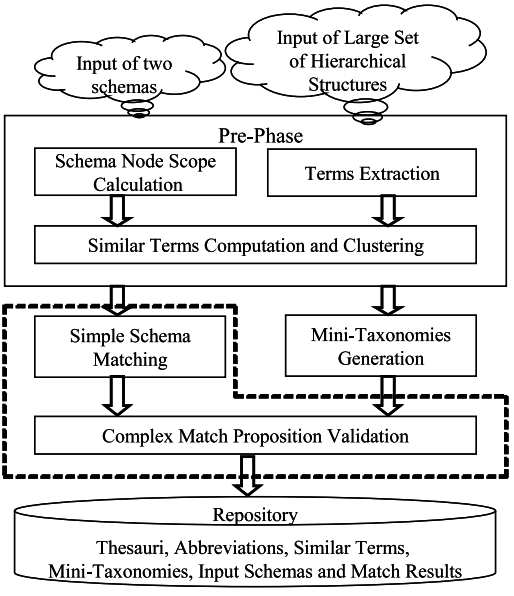

Fig. 2. Architecture for complex matching discovery and validation using automatically generated mini-taxonomies

The *Simple Schema Matching* module tries to compute a correspondence for every node from source schema tree to target schema tree. For each input node a set of possible matching nodes in the target schema is created, producing the target search space, based on node label similarity. For contextual similarity, *ancestor node match* is checked for each possible target matching node, to confirm that there exists a match between an ancestor node of the current source node and some ancestor node of the target node, except for the root node. This contextual proximity is calculated as α:

$$\alpha = 1/(ddif_s + ddif_t)$$

where $ddif_s$ is the depth difference between the current source node and the ancestor node for which a match exists and $ddif_t$ is the depth difference between the candidate target node and the ancestor node in target schema to which the source ancestor node is matched. The candidate target node with the highest α value is selected as the mappable target node.

Next, the method proposes the type of mapping (β). The initial matches are marked as 1:1 and extended to 1:n, n:1 or n:m categories, depending upon the leaf or non-leaf status of the the matched nodes. The complex match propositions

discovered between the leaf nodes of the similar non-leaf nodes are validated in the complex match proposition validation part.

The *Complex Match Proposition Validation* (CMPV) module forms the main core of this research work. We use a novel method to validate the proposed complex matches with the help of automatically generated domain specific mini-taxonomies.

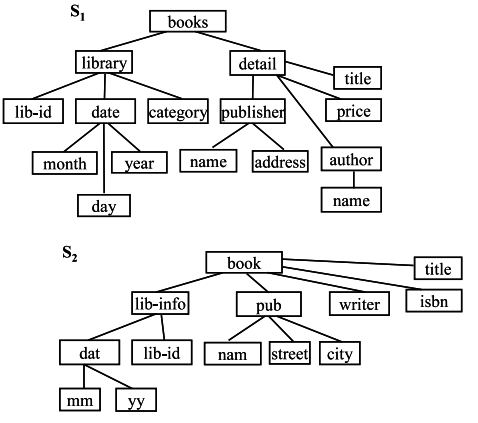

Fig. 3. Two schema trees S_1 and S_2 for complex match discovery

4 Complex Match Proposition Validation Algorithm

The Complex Match Proposition Validation (CMPV) module validates the n:m match propositions discovered in the simple schema matching module. The technique utilizes small conceptual taxonomies already generated from a large number of schemata within the specific domain.

4.1 Complex Match Validation Example

Figure 3 shows two schema trees from the books domain. A list of correspondences is shown in Figure 4, after the execution of simple matching module. The result is discovery of one to one matches along with complex match propositions (in brackets). The scenario presents one n:1 and four n:m complex match situations. The size of the sub-tree rooted at the non-leaf nodes is verified by scanning the scope of the nodes[1]. Large size sub-trees tend to be collection of concepts rather than a single concept.

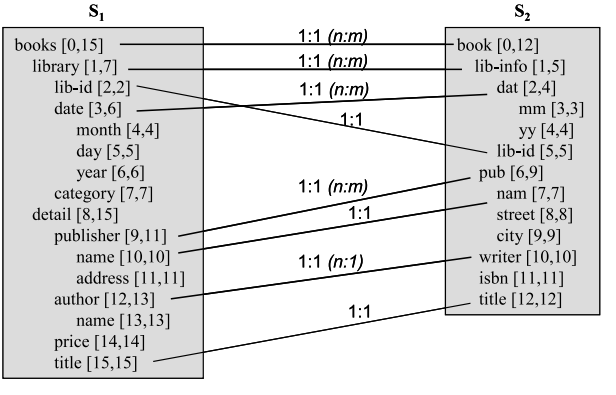

Fig. 4. Element level matches between schemata S_1 and S_2 after execution of simple schema matching

[1] For any node : label[X,Y]; X is depth-first order number and scope is given as (Y-X) i.e, the number of nodes in the sub-tree rooted at that node.

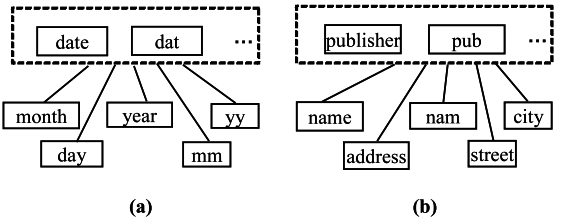

Fig. 5. Mini-taxonomies extracted from large input of books domain schemata

Next, the validation of the proposed complex matchings is done by our Complex Match Validation algorithm with the help of a set of already acquired mini-taxonomies.There are two mini-taxonomies, shown in Figure 5, representing (a) the *date* and (b) the *publisher* concepts respectively. The *date* concept can be represented by *date*, *dat* or some other similar string, as the root node for the concept. The leaf nodes collection of *month*, *day* , *year*, *mm*, *yy* represent attributes describing the concept. Within one instance of the mini-taxonomy, the presence of synonymous leaf nodes is not possible e.g. if *month* and *mm* are synonymous then the two nodes will not exist together in a mini-taxonomy with root node *date* or *dat* [5]. Similarly, Figure 5b presents the mini-taxonomy instances of the concept *publisher address*.

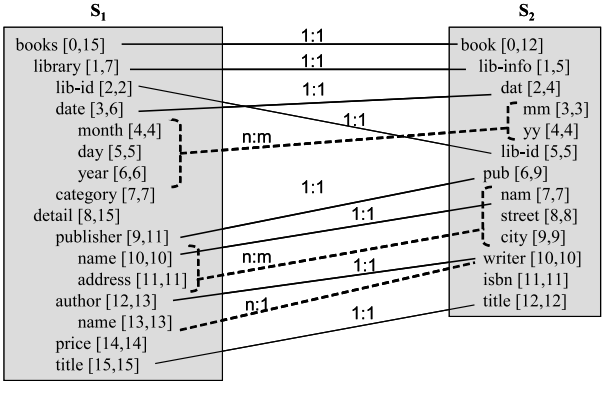

Fig. 6. Mappings between schemata S_1 and S_2 after execution of complex match validation algorithm

The execution of the algorithm occurs for each of the four n:m propositions. In case of S_1. books[0,15] $\leftrightarrow$ S_2.book [0,12] and S_1.library [1,7] $\leftrightarrow$ S_2.lib-info[1,5], no mini-taxonomy is found with root element similar to *books* or *book* and *library* or *lib-info*. As a result, the two propositions are discarded and only the 1:1 simple match is considered. In the other two cases the algorithm finds mini- taxonomies in the form of date-month/day/year, dat-mm/yy, publisher-name/address and pub-nam/street/city[2]. The algorithm substantially authenticates the propositions and creates two more complex matches as S_1.(month[4,4], day[5,5], year[6,6])$\leftrightarrow S_2$.(mm[3,3],yy[4,4]) and S_1.(name[10,10], address[11,11])$\leftrightarrow S_2$.(nam[7,7], street[8,8],city[9,9]). The final correspondences are shown in Figure 6.

5 Lessons Learned

Several data sets[3] were selected as the input for the experiments. Characteristics of these sets of schemata are given in Table 1.

[2] - and / delimiters denote downward and upward traversal, respectively.

[3] http://metaquerier.cs.uiuc.edu/repository

Table 1. Characteristics of domain schema trees used in the CMPV experiments

Domain	BOOKS1	BOOK SEARCH	JOBS	AUTO	AIR TRAVEL	REAL ESTATE	COURSES
Type	Synthetic	Real	Real	Real	Real	Real	Real
Number of schemata	176	19	20	14	20	14	42
Average nodes per schema	8	6	5	5	14	9	8
Largest schema size	14	12	8	10	21	20	17
Smallest schema size	5	3	4	3	6	4	2
Schema Tree Depth	3	2	2	2	2	2	4
n:m match propositions	2	2	1	1	3	2	2

CONCEPT	COMPLEX MAPPINGS
date/depart/return	month,day,year↔mm,yy↔month,day,time
address/location	name,address↔nam,street,city↔street1,street2,city↔ address1,address2↔AreaCode,Country↔city,state
telephone	tel_res,tel_off↔morn_tel, even_tel,night_tel↔tel_mobile,tel_fix
name	firstName,lastName↔f_name,mi,l_name
passengers	adult,child,infant↔adult,senior,child↔adult_12-65,adult_65,child_2-4, child_5-12,infant
return/depart	month,day↔month,day,year,time
schedule	days,time,room↔DayTime,room↔Times,Place↔TimeBegin,TimeEnd, Room,Building↔time,building
car model	vfrom,vto↔fyear,tyear

An analysis of these domain schemata showed that only very small sets of elements (sets of leaf nodes representing some concept), could participate in a complex mapping. The idea of using mini-taxonomies works well, if their leaf nodes can represent some real complex match with a contextual map at ancestor level. Whereas, the ancestor level mapping is highly dependent on the label matching. Working with real world schemata of the web interface forms showed that the possibility of a complex match is very limited in this domain. The reason being the depth of such schema trees is very less, due to which the ancestor level matching requirements are restricted. Secondly, there existed a very vast variation of concept name for a certain concept. For example, there were 13 different labels for passenger concept in the travel domain.

6 Conclusion and Future Work

In this paper, we have presented an approach for discovering and validating the complex matches. Our approach is based on the leaf or non-leaf status of the node, putting forward the match proposition that when a non-leaf node is matched to a non-leaf node, there is the probability of an n:m match between the leaf nodes of the two non-leaf nodes. Next, the method validates this proposition, indirectly utilizing the collective intelligence of the domain users. This is achieved by using mini-taxonomies, extracted using frequent tree mining technique, from a large number of input schemata used over the specific domain. The technique is in fact collaboration of the domain users for complex match validation.

In the future, we plan to extend the label level matching techniques, utilizing state of the art lexical matchers and linguistic dictionaries. Secondly, we intend to exploit other application domains over the semantic web with our approach of indirect collaboration of the domain users.

Acknowledgements

K.S. is funded by the Higher Education Commission of Pakistan.

References

1. Do, H.-H., Rahm, E.: Matching large schemas: Approaches and evaluation. Information Systems 32(6), 857–885 (2007)
2. Doan, A., Madhavan, J., Dhamankar, R., Domingos, P., Halevy, A.Y.: Learning to match ontologies on the Semantic Web. VLDB J. 12(4), 303–319 (2003)
3. Embley, D.W., Xu, L., Ding, Y.: Automatic Direct and Indirect Schema Mapping: Experiences and Lessons Learned. ACM SIGMOD Record 33(4), 14–19 (2004)
4. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-Match: an Algorithm and an Implementation of Semantic Matching. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 61–75. Springer, Heidelberg (2004)
5. He, B., Chang, K.C.-C., Han, J.: Discovering complex matchings across web query interfaces: a correlation mining approach. In: KDD, pp. 148–157 (2004)
6. Lee, D., Mani, M., Chiu, F., Chu, W.W.: Net Cot: Translating relational schemas to XML schemas using semantic constraints. In: CIKM (2002)
7. Melnik, S., Rahm, E., Bernstein, P.A.: RONDO: A Programming Platform for Generic Model Management. In: SIGMOD, pp. 193–204 (2003)
8. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. 10(4), 334–350 (2001)
9. Saleem, K., Bellahsene, Z.: Automatic extraction of structurally coherent mini-taxonomies. In: ER (2008)
10. Saleem, K., Bellahsene, Z., Hunt, E.: PORSCHE: Performance ORiented SCHEma mediation. Information Systems - Elsevier 33(7-8), 637–657 (2008)
11. Wang, G., Zavesov, V., Rifaieh, R., Rajasekar, A., Goguen, J., Miller, M.: Towards User Centric Schema Mapping Platform. In: VLDB Workshop Semantic Data and Semantic Integration (2007)
12. Zaki, M.J.: Efficiently Mining Frequent Embedded Unordered Trees. Fundamenta Informaticae 66(1-2), 33–52 (2005)