CONCEPTUAL FRAMEWORK FOR XML SCHEMA MAPPING

Myriam Lamolle, Amar Zerdazi
LINC, IUT of Montreuil-University of Paris 8, 140 rue de la Nouvelle France, 93100 Montreuil, France

Ludovic Menet
Orchestra Networks, 75 boulevard Haussman, 75008 Paris, France

Keywords: Mapping, node similarity, schema integration, XML Schema.

Abstract: Today's web-based applications and web services publish their data using XML, as this helps interoperability with other applications and services. The heterogeneity of XML data has led to recent research in schema matching, schema transformation, and schema integration for XML. In this paper, we propose an approach for mapping integration for XML schema. The basic idea is to drive direct as well as complex matches with their associated transformation operations from the computed element similarities. The representation of a mapping element in a source-to-target mapping clearly declares both the semantics correspondences as well as the access paths to access and load data from source into a target schema. We detail our mapping generation process and proceed in four steps to specify a formal representation of a source-to-target mapping in order to discover structural mappings between XML schemas.

1 INTRODUCTION

Today's web-based applications and web services publish their data using XML, as this helps interoperability with other applications and services. The heterogeneity of XML data has led to recent research in schema matching, schema transformation, and schema integration for XML.

In this paper, we propose an approach for mapping integration for XML schema. The basic idea is to drive direct as well as complex matches with their associated transformation operations from the computed element similarities. The representation of a mapping element in a source-to-target mapping clearly declares both the semantic correspondences as well as the access paths to access and load data from source into a target schema.

The paper is organized as follows. Some related works are presented in section 2. In sections 3 and 4, we respectively give a brief overview of our formal model for XML schema (XML Schema graph) and highlight a set of structure transformation operations. Section 5 presents the core of this paper. We detail our mapping generation process and proceed in four steps to specify a formal representation of a source-to-target mapping in order to discover structural mappings between XML schemas. Section 6 concludes this paper.

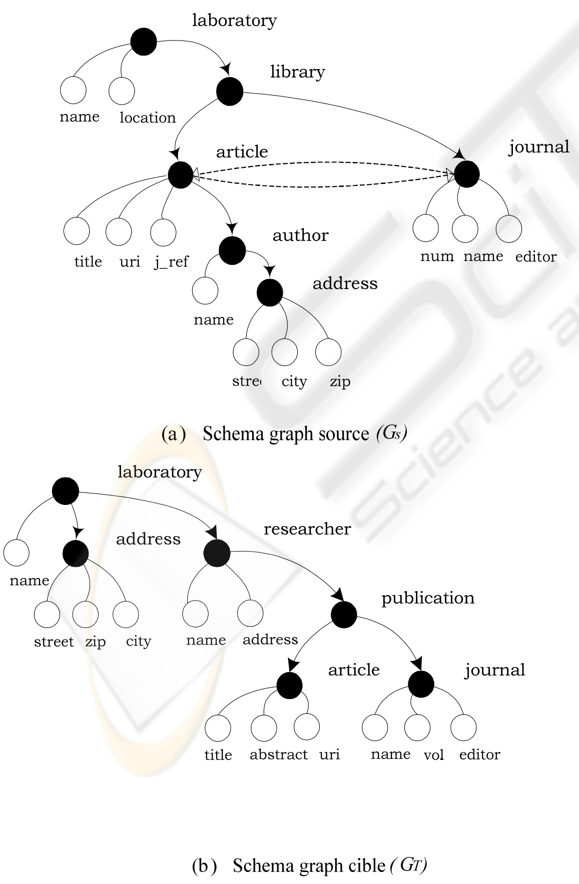
2 RELATED WORKS

Several approaches (Bouzeghoub et al., 2003), (Miller et al., 2000), (Li et al., 2000) have been proposed to generate mappings when the target and the source schemas are expressed using the relational model. The approach presented in (Kedad and Xue, 2005) generates a set of mappings from a set of source schemas using linguistic correspondences between target attributes and source attributes expressing the idea that these elements represent the same concept.

In (Popa et al., 2002), an approach is proposed for generating mappings from one source schema to a target schema when these schemas are represented by XML Schema. In (Yu and Popa, 2004), a query rewriting algorithm which uses these mappings is proposed for integrating data sources. Other approaches have been proposed (Claypool et al., 2003), (Yang et al., 2003), and (Zamboulis and Poulovassilis, 2004) to generate mappings from several source schemas. These approaches comprise two steps: (i) the definition of rules to restructure each source schema according to the structure of the target schema, and (ii) the generation of mappings from these restructured schemas. In these approaches, source schemas must be restructurable with respect to the target schema in order to use them for mapping definition.

3 THE DATA MODEL

As already mentioned in section 2, up to now few existent XML schema matching algorithms have focused on structural matching exploiting all W3C XML schemas (W3C, 2001) features. In this section, we propose an abstract model that serves as a foundation to conceptually represent W3C XML schemas and potentially other schema languages. We model XML schemas as a directed labeled graph with constraint sets; a so-called schema graph. This graph is used in the matching process for the measure of node context similarity.



(a) Schema graph source (G_S)

(b) Schema graph cible (G_T)

Figure 1: A schema graph example.

A schema graph consists of series of nodes that are connected to each other through directed labeled links. In addition, constraints can be defined over nodes and links. We proposed model for XML schemas in order to define a formal framework for solving matching problems.

4 OPERATIONS ON MAPPING

A mapping element relates, in its simplest form, a construction (which refers to nodes and edges in the schema graph) from S to a semantically similar construction from T (respectively source and target schema graphs). Although, a source may not have a construction that directly corresponds to the target one, nevertheless target constructions may be derived from source constructions by applying a set of predefined operations. This mechanism is somehow similar to *virtual views creation* in data integration, where a virtual view over the source has to match a mediated schema (e.g., the concatenation of source elements *firstname* and *lastname* gives rise to a virtual element *name* that matches the target element *name*). Virtual views are also applicable to edges (e.g., the two edges *author→firstname*, and *author→lastname* are merged together into a virtual edge *author→name* that matches the edge *author→name* in the target schema). Based on this observation we borrow the notion of virtual view to formally define mappings between source and target schemas. We have been inspired essentially by research in the field of generating virtual views in data integration systems (Biskup and Embley, 2003), (Dobre et al., 2003), (Xu and Embley, 2003).

Definition 1 *(Schema alphabet):* Given a schema S (in keeping with the schema graph formalism), we define the alphabet of S, Σ_S as the union of nodes and edges in the S schema graph. $\Sigma_S=N_S\cup E_S$.

Definition 2 *(Virtual view):* Given a source schema S and its alphabet Σ_S, a virtual view over S, v_S is defined as a derivation from the alphabet Σ_S. Applying a set of predefined operations $O=\{o_1,\ldots o_n\}$.

Definition 3 *(Mapping element, mapping direct, mapping complex):* Let V_S denotes the set of possible virtual views constructed over Σ_S. A *mapping element* is a function μ: $\Sigma_S\cup\Sigma_T$, that associates an element s in $\Sigma_S\cup V_S$ to an element t in Σ_T. A mapping element is *direct mapping* if it binds an element in $\Sigma_S(\subseteq V_S)$ to an element in Σ_T and a

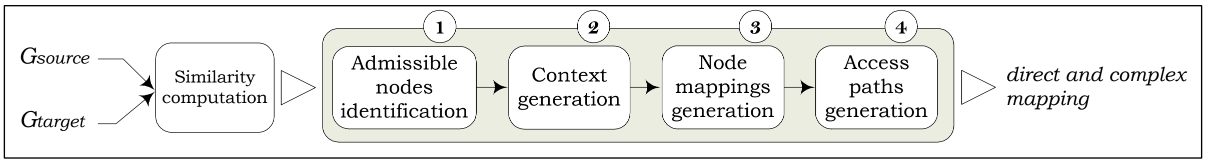


Figure 2: Mapping generation process.

complex mapping if it binds a virtual element $v_S \in V_S$ to a target element in Σ_T through a mapping expression defined over *O*.

Mappings can be combined by means of some operators giving a result that in turn is a mapping. In (Zerdazi and Lamolle, 2005), we extended the relational algebra so that we were able to formally define a set of operators that are defined over source and target schema nodes and that can be used to combine nodes to form complex mapping expressions.

Our mapping algebra concerns a set of transformation operations, as listed below:

- Rename: *t=rename⟨s⟩*, generates a construction that is the same as a construction *s*, but with a different name *t*. For example, *editor=rename⟨publisher⟩*;
- Merge: *t=merge⟨*s_1*,..*s_i*⟩*, generates a construction *t* whose value is obtained by concatenating s_1,..s_i in the case of strings. For example, *address=merge⟨street, city, zip⟩*;
- Split: *(*t_1*,..*t_i*)=split⟨s⟩*, where t_1,..t_i are obtained by splitting a construction *s* with respect to a separation criterion. For example, *(first-name, last-name)=split⟨name⟩*;
- Parity: *t=parity⟨s⟩*, generates a construction *t* which has the same content and label as *s*. for example, *author=parity⟨author⟩.*

5 DISCOVERY OF NODES AND EDGES MATCHES

Most schema matching algorithms (Li et al., 2000), (Doan et al., 2003), (Mitra et al., 2000), (Milo and Zohar, 1998), (Palopoli et al., 2000), (Castano et al., 2004), (Do and Rahm, 2001), (Melnik et al., 2002), (Noy and Musen, 2001), (Giunchiglia et al., 2004) produce similarity scores between source and target schemas nodes such as the ones we produce in section 5.3; however, such a mapping result partially solves the problem. First, produced similarities between individual nodes are not enough to produce *access paths* for retrieving data from the available sources. For example, based on previous matching techniques, we obtain a set of node matches such as the match between *laboratory* and *laboratory*, or between *researcher* and *author* or between *book* and *book*. Without matches between edges, however, it may be impossible to distinguish authors that wrote books, from authors that wrote articles. Intuitively, a source-to-target mapping should describe all the produced mappings such as one-to-one mappings, complex mappings identified using type hierarchy have to be incorporated in the matching result and further complex mappings have to be discovered. To do this, we proceed in four steps (figure 2).

5.1 Admissible Node Identification

While generating mapping elements, we apply a *top-down strategy* (We use the same top-down strategy as in (Xu, 2003b). However the difference is that this technique is used to discover structural similarity. In our approach, it is just for mapping generation; the structural matching has been already performed). At the top level, we establish correspondences between complex nodes of the target and source schemas. Matched complex nodes are called compatible nodes. In figure 1(a), the complex node *laboratory* is considered to be a compatible node since it is matched to node *laboratory* in the schema graph of figure 1(b), while node *library* is not a compatible node. Then, at the bottom level, with the guide of compatible nodes between the target and source schemas, we establish the finer-level correspondences between nodes and edge sets. Figure 3 illustrates compatible nodes. Visually compatible nodes are depicted as colored circles and dashed lines.

5.2 Context Generation for Admissible Node

After identifying admissible nodes, we proceed to construct a context for each admissible node by taking edges around a complex node *n* into account. We cluster a set of nodes and edges with a complex node as a conceptual component in the schema graph. We call this the context of *n*. The context for an admissible node *n* consists of a set of nodes and a set of edges among those nodes. For a given

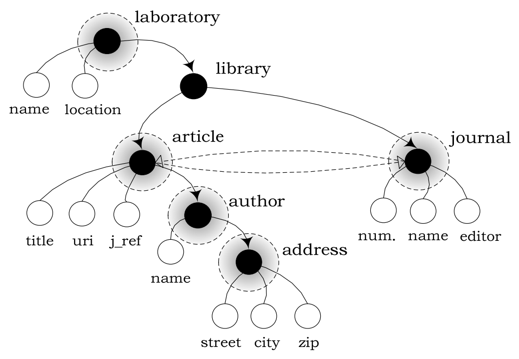


Figure 3: Example of a schema graph after admissible node identification.

admissible node *n*, we construct such a context as follows:

1. Including all atomic directly node related to the admissible node n.
2. Including all non-admissible nodes directly connected to *n* with their connected atomic nodes and connected non-admissible nodes. We continue this procedure until we find an admissible node.
3. If a directly connected admissible node is also similar to an atomic node, it is also included in the context of *n*.
4. Including all nodes having an association relationship with *n* and their respective context.
5. Including all containment relationships between nodes in the context of *n*.

Example: Figure 4 illustrates a schema graph of Figure 3 after context construction. The context of the admissible node *laboratory* includes atomic nodes *name* and *location* and non-admissible node *Library*. The context of admissible node *Article* includes referential node *Journal* and its context. The context of node *laboratory* includes the admissible node *address*, since *address* is similar to a leaf node (*location*) belonging to the context of a matched node *laboratory*.

5.3 Node Mappings Generation

At this stage, we have completed with the top-level comparison between source and target schema graphs. We are now ready to detect node and edges matches at the bottom level. For each matching pair *(n_S , n_T)* which represented two admissible nodes in source and target schema graphs, we make use of the node similarity score generated in previous work

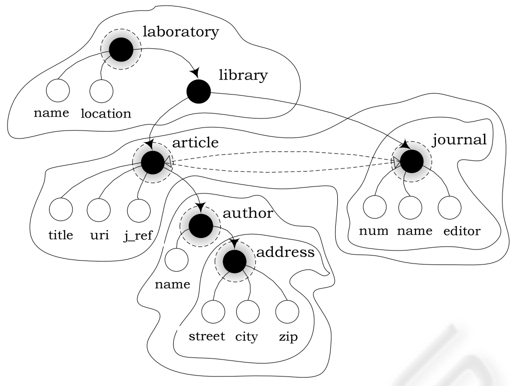


Figure 4: Example of a schema graph after context generation.

(Zerdazi and Lamolle, 2007) to settle node matches. The following gives examples on how we proceed.

Example 1: Let the schema in Figure 5(a) be the source schema and the schema in Figure 5(b) be the target schema. Consider the two admissible nodes, source node *laboratory* (*laboratory$_S$*) and the target node *laboratory* (*laboratory$_T$*), we first settle node-set matches between both source and target contexts that hold with the highest node similarity score. As an example, we settle the match pair *($name_S$, $name_T$)* using a *parity* operator.

Example 2: The target node *address$_T$* is both similar to the source nodes *laboratory/location$_S$* and *author/address$_S$* with approximatly the same scoring. This is due in the case of *laboratory/location* to the fact that ancestor context similarity is high and in the case of *author/address* to the fact that leaf context similarity is high (Zerdazi et al., 2006). Since target node *laboratory/address* and source node *author/address* belong to non-admissible complex nodes while target element *laboratory/address* and source element *laboratory/location* belong to two admissible contexts, a match is then derived between source node *laboratory/location* and target node *laboratory/address*.

Moreover, since we have decided to map a non-leaf node with a leaf node, a complex mapping with *split* operation can be deduced.

5.4 Access Path Generation

With the available correspondences between nodes in source and target schemas, we further discover matches between edges. As in (Xu, 2003), the recognition of edges matches starts by locating an

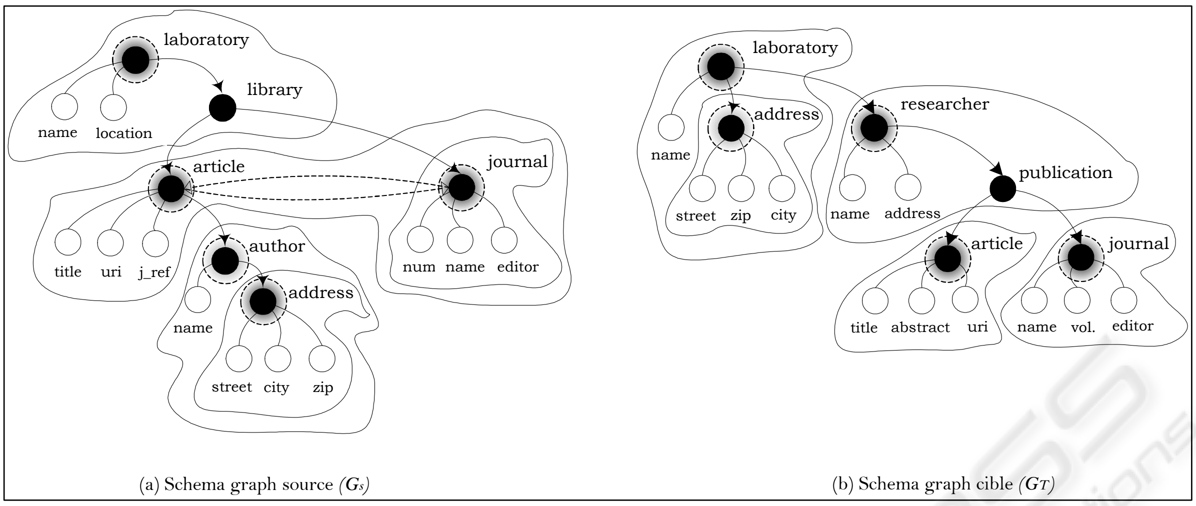


Figure 5: Source and target schema graphs after context generation.

edge set e_T in G_T. Then, based on nodes n_T connected by e_T, we can locate a set of nodes that correspond to n_T in G_S, from which we either locate or derive an edge set e_S that corresponds to e_T. We essentially focus on the discovery of *access paths* in order to retrieve source data when performing transformations. For each target element *t*, we first define the access path indicating where matched source elements are localized, then the discovered transformation operation and finally the conditions under which the mapping element holds true.

It is widely accepted that the matching process cannot be fully automated. Thus is necessary to incorporate user feedback on the matching task. The following stage will be validating mapping result and constraint filtering.

6 CONCLUSION

In this paper, we have presented a framework for discovering structural mappings between XML schema. This approach produces a set of mappings for a target schema considering a set of source schemas and a set of correspondences; each target mapping has different semantics.

Since the result of our approach is a set of structural mappings, one interesting perspective is to take advantage of these multiple semantics; the system should be able to select the mapping that best fits the needs of a specific user, using preferences or quality criteria. To achieve this goal, the users are invited to validate the final mapping result with having the possibility to add, delete or update mapping elements. The accuracy of a matching system considerably reduces post-matching user efforts. Another perspective of our work is the maintenance of the mappings. In dynamic environments such as the Web, data sources may change not only their data but also their schemas and their semantics. Such changes must be reflected in the mappings. Mappings left inconsistent by a schema change have to be detected and updated. It is important therefore to develop techniques for automatically adapting mappings as schemas evolve. Authors in (Velegrakis et al., 2003) proposed a mapping adaptation technique, but which essentially deals with relational schema changes).

REFERENCES

Biskup, J., and Embly, D.W., 2003. Extracting information from heterogeneous information sources using ontologically specified target views. *Information Systems, 28(3)*, 169-212.

Bouzeghoub, M., Farias Lóscio, B., Kedad, Z., Salgado, A.-C., 2003. Managing the evolution of mappings. Proc. of the 11th. In *International Conference on Cooperative Information Systems (CoopIS'03)*, Catania, Italy, 22-37.

Castano, S., Ferrara, A., Montanelli, S., and Racca, G., 2004. Semantic information interoperability in open networked systems. In *Proceedings of the International Conference on Semantics of a Networked World (ICSNW), in cooperation with ACM SIGMOD*, 215–230.

Claypool, K. T., and Rundensteiner, E. A., 2003. Gangam: A Transformation Modeling Framework. In *Proceeding of Eighth International Conference on Database Systems for Advanced Applications (DASFAA'03)*, Kyoto, Japan, 47-54.

Do, H.H., and Rahm, E., 2001. COMA - a system for flexible combination of schema matching approaches. In *Proceedings of the Very Large Data Bases Conference (VLDB)*, 610-621.

Dobre, A., Hakimpour, F., Dittrich, K.R., 2003. Operators and Classification for Data Mapping in Semantic Integration. In *Proceedings of the 22nd International Conference on Conceptual Modelling (ER2003), Springer Verlag*, (Chicago, Illinois, 534-547.

Doan, A., Madhavan, J., Domingos, P., Halevy, A., 2003. Learning to map ontologies on the semantic web. In *Proceedings of the International World Wide Web Conference (WWW)*, 662–673.

Giunchiglia, F., Shvaiko, P., Yatskevich, M., 2004. S-Match: an algorithm and an implementation of semantic matching. In *Proceedings of the European Semantic Web Symposium (ESWS)*, 61-75.

Kedad, Z., and Xue, X., 2005. Mapping Generation for XML Data Sources: a General Framework. In *Proceeding of the Int. Workshop on Challenges in Web Information Retrieval and Integration (WIRI'05), in conjunction with the 21st Int. Conf. on Data Engineering (ICDE'05)*, Tokyo, Japan.

Lamolle, M., and Zerdazi, A., 2005. Intégration de Bases de données hétérogènes par une modélisation conceptuelle XML, In *Conférence sur l'Optimisation et les Systèmes d'Information (COSI'05)*, 216-227.

Li, W., and Clifton, C., 2000. SEMINT: A tool for identifying attribute correspondence in heterogeneous databases using neural networks. *Data and Knowledge Engineering,* 33, 49-84.

Melnik, S., Garcia-Molina, H., Rahm, E., 2002. Similarity flooding: A versatile graph matching algorithm. In *Proceedings of the International Conference on Data Engineering (ICDE)*, 117-128.

Miller, R.J., Haas, L. M., Hernández, M. A., 2000. Schema Mapping as Query Discovery. In *Proceeding Of the 26th International Conference on Very Large Data Bases (VLDB'00),* Cairo, Egypt, 77-88.

Milo, T., and Zohar, S., 1998. Using schema matching to simplify heterogeneous data translation. In *Proceeding of 24th International Conference On Very Large Data Bases*, 122-133.

Mitra, P., Noy, N., Jaiswal, A., 2004. OMEN: A probabilistic ontology mapping tool. In *Proceedings of the Meaning Coordination and Negotiation workshop at the International Semantic Web Conference (ISWC)*.

Noy N., and Musen, M., 2001. Anchor-PROMPT: using non-local context for semantic matching. In *Proceedings of the workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI)*, 63-70.

Popa, L., Velegrakis, Y., Miller, R.J., Hernandez, M.A., Fagin R., 2002. Translating web data. In *Proceeding of the 28th International Conference on Very Large Data Bases (VLDB'02)*, Hong Kong, China, 598-609.

Palopoli, L., Terracina, G., Ursino., D., 2000. The system DIKE: Towards the semi-automatic synthesis of cooperative information systems and data warehouses. In *ADBIS-DASFAA, Matfyzpress*, 108-117.

Velegrakis, Y., Miller, R.J., Popa, L., Mylopoulos, J., 2003., ToMAS: Mapping Adaptation under Evolving Schemas. In Second *Hellenic Symposium for Data* Management *(HDMS).*

W3C. Fallside, D.C. (editor)., 2001. XML Schema Part 0: Primer, W3C Recommendation 2 May 2001. *Available at http://www.w3.org/TR/2001/REC-xmlschema-0-20010502*.

Xu, L., and Embly, D.W., 2003. Discovering Direct and Indirect Matches for Schema Elements. In *Proceedings of the 8th International Conference on Database Systems for Advanced Applications (DAFSAA)*.

Xu, L., 2003. *Source Discovery and Schema Mapping for Data Integration*. PhD thesis, Brigham Youg University.

Yang, X., Lee, M. L., Ling, T. W., 2003. Resolving structural conflicts in the integration of XML schemas: a semantic approach. In *Proceeding of 22nd International Conference on Conceptual Modeling (ER'03)*, Chicago, 520-533.

Yu, C., and Popa, L., 2004. Constraint-based XML query rewriting for data integration. In *Proceeding of International Conference ACM SIGMOD (SIGMOD'04)*, Paris, France, 371-382.

Zamboulis, L., and Poulovassilis, A., 2004. XML data integration by Graph Restructuring. In *Proceeding of the 21st Annual British National Conference on Databases (BNCOD21)*, Edinburgh, 57-71.

Zerdazi, A., and Lamolle, M., 2006. Intégration de sources hétérogènes par matching semi-automatique de schémas XML étendus. In *25ème Congrès Informatique des Organisations et systèmes d'Information et de décision (INFORSID),* Tunisia, 991-1006.

Zerdazi, A., and Lamolle, M., 2007. Matching of Enhanced XML Schema with a measure of structural similarity. In *Proceedings of the Third International Conference on Web Information Systems and Technologies (WEBIST),* Barcelona, Spain, 128-133.