A Tool for Mapping Discovery over Revealing Schemas

Verena Kantere
Ecole Polytechnique Fédérale de Lausanne
1015 Lausanne, VD, Switzerland
verena.kantere@epfl.ch

Dimos Bousounis
National Technical University of Athens
Heroon Polytechniou 9, 15780 Zografou, Greece
dbous@dblab.ece.ntua.gr

Timos Sellis
Institute for the Management of Information Systems
(R.C. 'Athena'), Athens, Greece
timos@imis.athena-innovation.gr

ABSTRACT

In a world of wide-scale information sharing, the decentralized coordination has to consolidate a variety of heterogeneity. Shared data are described in different formats, i.e. data structures, values and schemas. Querying manifold such sources entails techniques that can bridge the data formats. Some of these techniques deal with producing mappings for the schemas of data. The existing techniques view complementary aspects of the schema mapping problem. Important ones, consider producing all the possible mappings for a pair of schemas, insinuating any accompanying semantics in the mappings and adapting correct mappings as schemas evolve. Towards this end we have developed a solution that is fine-tuned for the discovery of mappings as schemas of autonomous sources are gradually revealed. In this demonstration we exhibit a new prototype tool that implements this solution. The tool provides a mechanism that realizes discovery of correct mappings as schemas are revealed. Mapping discovery is schema-centric and incorporates new semantics as they are unveiled. Mapping experience is reused and possible mappings are ranked so that the best choice is presented. The core mechanism collaborates with an automatic schema matching tool and the user that lightly guides the mapping process. The demonstration presents two application scenarios that prove the suitability of this prototype tool and the effectiveness of the implemented mapping solution in realistic situations of of data integration and exchange between heterogeneous autonomous sources.

1. INTRODUCTION

Autonomous computing is the new trend in research since the need for the information sharing increases. In the networked world shared data are described in different formats, i.e. data structures, values and schemas. Thus, the decentralized coordination of autonomous sources has to consolidate a variety of heterogeneity. Some of the techniques that deal with this problem focus on the necessity to specify the relationship of their schemas and data, i.e. the construction of schema and data mappings. Automating the discovery of schema (and data) mappings is one of the fundamental unsolved challenges of data interoperability.

The problem of schema mapping has been approached from several complementary aspects [7, 11, 3, 4] etc. Clio [7] is a tool that provides mappings between pairs of relational or XML schemas based on data constraints in order to preserve data associations. The emphasis is on finding all or a set of possible mappings, based on user-defined data value correspondences. ToMAS [11] is a tool concerned with the influence of schema evolution on existing mappings. ToMAS takes a mapping-centric approach and considers that initial mappings have correct semantics that should be preserved. Therefore, user choices w.r.t. initial mappings are maintained as far as possible while the schemas change. Muse is a mapping design wizard; it draws examples of data in order to solve ambiguities in schema mappings, concerning the grouping semantics of sets of data. Towards the same end, the technique in [4] explores the benefits of possibly available additional semantic information.

A complementary aspect on the schema mapping problem is how to maintain mappings in a dynamic autonomous environment. Our interest is on autonomous sources that desire a priori discretion of their schemas and their implied semantics, and are willing to gradually reveal them, as their communication needs increase. In such a setting it is required that the mapping process can take place in lack of additional semantics, but, also, in lack or shortage of data.

Motivating Applications. Our focus is on dynamic settings of autonomous heterogeneous sources that share data. We consider applications that motivate the necessity for a specialized mapping technique. Such are applications of federated databases and Peer Data Management Systems (hereafter PDMSs). Figure 1 shows two relational revealing schemas from autonomous sources in the education domain.

Federated databases are apt to retaining their autonomy, and, thus, their heterogeneity, as long as their provisional requirements for data integration can be fulfilled. In such a setting the sources can benefit from a dynamic solution that allows incremental schema mapping. Sources can map their schemas in a piecemeal fashion that gives the opportunity for temporal partial integration, as well as monitor the gradual mapping improvement and estimate how much of their schema/data autonomy they are willing to surrender.

PDMSs consist of peers that share structured data [5] by expressing queries on their local schema. These queries are answered by remote peers after they have been rewritten through chains of schema mappings [6]. Pairs of remote peers may exchange queries and data, which gradually reveal local schema and data semantics to each other. Such peers may want to construct mappings between their schemas, in order to become directly acquainted in the overlay. GrouPeer [9] is a framework that enables such familiarization of remote peer databases.

EDBT 2009, March 24–26, 2009, Saint Petersburg, Russia.

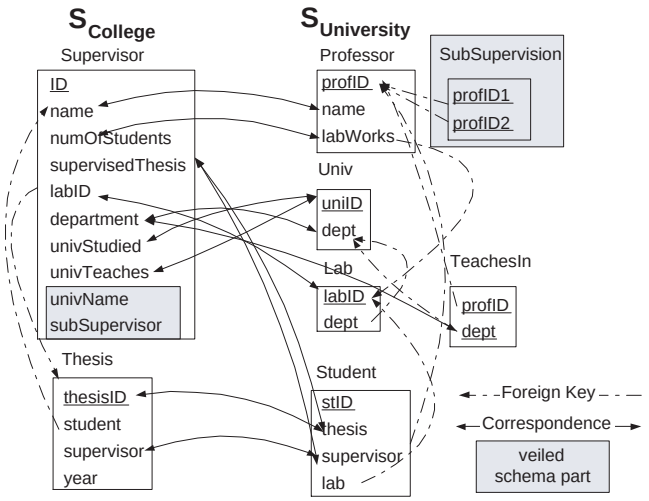

Figure 1: A pair of revealing schemas.

In both settings the task of schema mapping is vital and should be performed in a dynamic manner. Initial mappings on partial schemas should be progressively refined as schemas are revealed, and adhere to the new overall knowledge of semantics. It is important that possible mappings are prioritized, so that users are presented with correct mappings instead of trying to navigate among numerous ambiguously different mapping versions. Furthermore, mapping experience should be appropriately memorized and reused in each new phase of schema reveal.

The Mapping Mechanism. The work in [8] describes a schema mapping solution that is fine-tuned for autonomous heterogeneous sources that share data. As schemas are gradually revealed the mechanism discovers mappings in a semi-automatic manner. At each disclosure of schema parts the mappings change so that they reflect the incrementally unveiled schema semantics. The mechanism searches efficiently the mapping space for possible mappings, so that the more accurate mappings are discovered first. Possible mappings are ranked so that the user is presented with the best choice. An intuitive and simple interface enables the user to lightly guide the schema mapping through coarsely expressing her opinion on the mapping structure. Accumulated experience during the mapping process is valuable and exploited by the mechanism in future mapping phases. Appropriate memorization of this experience facilitates mapping adaptation to new schema semantics, such that the same mistakes are avoided but, also, search in the mapping space becomes more efficient. The mechanism can take advantage of value constraints on the schema matching, so that it produces value-conditional mappings. Depending on the application setting, value constraints are input either by query traffic between the mapped sources, or by respective data samples.

Demonstration Proposal. This demonstration presents a new prototype tool that implements the discussed mapping solution. Two application scenarios are displayed, that prove the suitability of the tool for settings of heterogeneous autonomous environments that integrate or exchange data. Specifically, examples from a federated database system that performs dynamic data integration and a peer data management system [6] that performs data exchange show multifarious situations of gradual schema reveal that require adaptive schema-centric mapping discovery, provided by the prototype mapping tool.

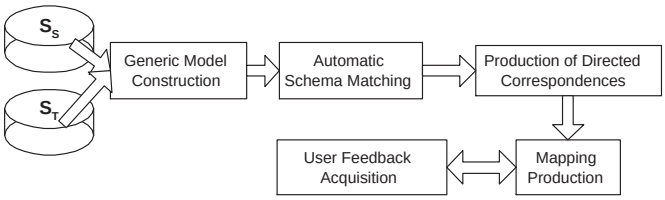

Figure 2: The workflow of the mapping procedure

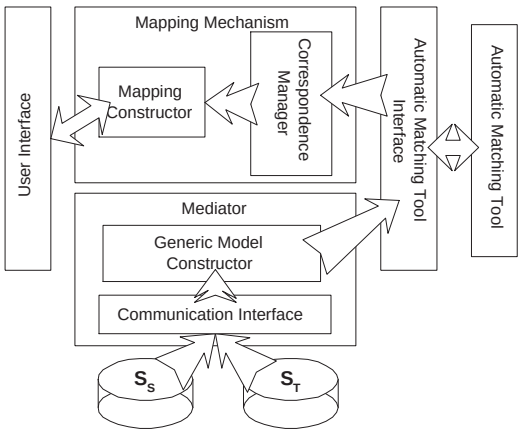

Figure 3: Architecture of the mapping tool.

2. OVERVIEW OF THE MAPPING PROCESS

The mapping process formulates the mappings for a pair of revealing schemas. We outline the workflow of the mapping process and we summarize the core mapping algorithm.

2.1 The mapping functionality

The mapping process, (depicted in Figure 2), is deployed in iterative phases that construct the mappings between two schemas S_S and S_T, (as in Figure 1). Each phase takes as input the currently revealed schema parts as well as the user feedback. The initial phase initiates all the associative structures used by the mapping mechanism. The following phases incrementally change these structures in order to reflect the current mapping experience. In each phase there is new user feedback and/or newly revealed schema information.

The schema information is processed and formatted in a generic schema model that encapsulates the schema elements, foreign key and, possibly, data-value constraints, as well as deduced constraint information. This information is directed to the automatic schema matching tool, which produces $1-1$ concept correspondences paired with some confidence value, that show the certainty of the matcher about this result. This set of undirected correspondences is processed by the mechanism, that produces two sets of directed correspondences, $\mathcal{C}_{\mathcal{D}_S}$ and $\mathcal{C}_{\mathcal{D}_T}$ that are used for the construction of mappings for the two schemas. The mappings are initialized in the initial phase or improved in the following ones by the mapping algorithm summarized in Section 2.2. These mappings are presented to the user, who can give feedback w.r.t. to their correctness. The feedback is coarse intuitive estimations that can be made by an non-advanced user. This feedback enables the mechanism to learn about mistakes on the semantics of the current mappings, which are memorized and avoided in future mapping phases.

2.2 The mapping algorithm

The mapping process is realized by an iterative algorithm that searches and presents the most likely correct mapping in each phase of schema reveal and/or user feedback. The algorithm constructs schema mappings of the forms GAV (Global-As-View) or LAV (Local-As-View) [10] between schemas S_S and S_T. Both schemas are gradually revealed and matched by the automatic matching tool, which produces a set of undirected correspondences of schema concepts, $\mathcal{C}_{\mathcal{U}}$.

Managing Correspondences The undirected correspondences between the source and target schema, $\mathcal{C}_{\mathcal{U}}(\mathcal{S}_{\mathcal{S}}, \mathcal{S}_{\mathcal{T}})$ are broken into directed ones, that show subsumption of the corresponded concepts. These are stored in the sets $\mathcal{C}_{\mathcal{D}_{\mathcal{S}}}$ $\mathcal{C}_{\mathcal{D}_{\mathcal{T}}}$ that are intended to be employed in mappings for the source and the target schema, respectively. The two sets are augmented with inferred directed correspondences. Roughly, for a directed correspondence $C_D(E_S, E_T^f)$, the inferred correspondence $C_D(E_S, E_T^p)$ is inferred, iff E_T^f is a foreign key to E_T^p, in schema S_T; this is inserted in $\mathcal{C}_{\mathcal{D}_{\mathcal{T}}}$. Analogous inferences are inserted in $\mathcal{C}_{\mathcal{D}_{\mathcal{S}}}$.

Discovering Mappings. The mechanism discovers a GAV and a LAV mapping for a relation R_S, from S_S to the target schema S_T, denoted as $M_G(R_S, S_T)$ and $M_L(R_S, S_T)$. A special structure defined as $M_{atrix}(R_S, S_T)$ is associated with each of these two mappings, which keeps track of changes, along the iteration of the mapping process. Briefly, for each attribute of R_S, $M_{atrix}(R_S, S_T)$ logs the currently chosen correspondence, as well as the rest possible correspondences; moreover, it keeps track of the correspondences that were selected in previous mapping iterations and were proven to be wrong. For the chosen correspondence of each attribute, M_{atrix} also indicates the respective join path in S_T that should be incorporated in the mapping. Also, it keeps track of join paths that were proven to be inappropriate. Intuitively, a join path is the combination of joined relation attributes of the target schema that is used so that the correspondence can be included in the specific mapping.

The algorithm finds the most *appropriate*, i.e. short, join paths for each one of the selected correspondences or the attributes of R_S in an efficient manner. Visualizing the relations of S_T to be involved in the mapping as nodes of a graph, the algorithm aims to inter-connect the nodes such that the paths between all pairs of nodes are short. Such a graph connection is translated as a tight correlation of the concepts involved in the respective mapping.

Incorporating user feedback The initialization of $M_{atrix}(R_S, S_T)$ is improved iteratively as S_S is revealed[1] and as the user gives feedback on the quality of the presented mapping. Table 1 shows an instance of M_{atrix} ($Supervisor$, $S_{University}$). The user annotates each row of the M_{atrix} with pre-defined characterizations; these provoke predefined actions that aim to memorize the user feedback and employ it for mapping refinement. These characterizations are intuitive, such that the user has no difficulty in assigning them.

3. THE MAPPING TOOL

The mapping tool, depicted in Figure 3 is composed of the core module that implements the mapping mechanism, as well as some peripheral modules that perform the communication with the external environment, i.e. the user, the relational sources, and the automatic matching tool.

Schema/Data Mediator. A mediator that communicates with the external relational sources imports information about the pair of revealing schemas. The mediator comprises a communication interface, as well as a module that preprocesses the incoming information and formulates it appropriately, so that it can be input in the mapping mechanism. The interface imports elements of the revealing schemas, i.e. relations, attributes, constraints, and, occasionally data values. This information is processed in the generic model constructor, which formulates it in a structure that is: (i) incremental, so that new elements can be added when they are revealed, and (ii) it extracts implicit information, such as inheritance of schema constraints.

Automatic Matching Tool Interface. The mapping tool leverages the task of primitive conceptual matchings for the pair of incoming schemas to an automatic matching tool . In this prototype we have employed the well-known free matcher COMA++ [1]. The interaction of the mapping mechanism and the matcher is performed by the automatic matching tool interface. Through this interface the matcher receives the schema (and data) information and as well as a predefined set of good element correspondences. Based on this input, the matcher captures $1-1$ correspondences between schema elements, input to the mechanism.

User interface. The mapping mechanism interacts with the user through the user interface. The latter enables the user to visualize the current mapping phase and communicate the human rationale to the mapping mechanism. The user is presented with the current mapping and is requested to make coarse intuitive estimations that can lightly guide the next phases of the mapping process.

Mapping Mechanism. The core module of the mapping tool is the mapping mechanism. The latter performs central interaction with all three communication ends and implements the mapping algorithm. Concerning interaction, the mapping mechanism receives, processes and redistributes information from and to the external modules. The generic schema models received from the mediator are processed so that primitive a priori concept correspondences are defined. Together with the generic models, these correspondences are input to the automatic matching tool interface. The output of the latter is processed so that sets of directed correspondences are formed. Furthermore, the mapping mechanism outputs the current mapping to the user interfaces and receives from the latter the user feedback, to be taken into account throughout the next mapping phases.

Beyond interaction, the mechanism realizes the algorithm that produces initial sets of GAV/LAV mappings, and iteratively improves these mappings along mapping phases with new user feedback and newly revealed schema elements.

4. DEMONSTRATION

In this demonstration the visitors are presented with two application scenarios of the prototype mapping tool which prove the suitability of the latter in a variety of autonomous environments. The first scenario is related to the data exchange field and presents the utility of the mapping tool in a PDMS, and the second scenario is related to the data integration field and presents the utility of the mapping tool in a system of federated databases. The examples demonstrate situations of gradual schema reveal that require adaptive schema-centric mapping discovery, provided by the tool.

For both scenarios the demonstration shows running examples with source schemas from the domain of education and medicine [2]. The schemas are presented in variations in order to illustrate the mapping process for pairs of schemas of different similarity, i.e. of different degrees of schema matching. The variations comprise schemas with different structure, which exemplify different

[1]Note that S_T can be gradually revealed, too.

Attribute	Correspondence	Alias	Possible Corr.	Bad Corr.	Join Path	Bad Join Paths
ID	Professor.ID	-	Student.supervisor	-	-	-
name	Professor.name	-	-	-	-	-
numofstudents	-	-	-	-	-	-
supervisedThesis	Student.thesis	-	-	Student.supervisor	Professor,Student	-
labID	Lab.ID	-	Student.lab	-	Professor, Lab	-
department	Lab.department	-	TeachesIn.dept, Univ.dept,	-	TeachesIn, Professor	-
univStudied	Univ.unID	-	-	-	Professor, Univ	-
univTeaches	Univ.unID	Univ$1	-	-	Professor, TeachesIn, Univ$1	-
univName	Univ.name	Univ$1	-	-	Professor, TeachesIn, Univ$1	Professor, Univ

Table 1: $M_{atrix}(Supervisor, S_{University})$

degrees of difficulty in search for the correct mapping semantics. For example, the number of schema constraints, as well as their distribution in the schema relations, and the size of the latter, play an important role in the mapping discovery progress. During the scenarios the visitors can interact with the tool and view the effect of their feedback on the mapping discovery process.

Data exchange scenario This scenario displays pairs of peer databases that are supposed to be remote in the overlay but share similar semantics, reflected in their peer schemas. The peers receive and answer queries on behalf of the other party, which reveal the schemas to each other. [2]. Details on the PDMS setting can be found in [9]. Note that the queries are rewritten on the local schema [9, 6]. The mapping tool is employed by both peer communication ends in order to discover mappings between the revealing schemas that embrace their semantics accurately.

The scenario includes examples on a range of peer situations that exhibit the diversity scale of the mapping discovery problem in such a setting. The visitors can watch the mapping discovery from both ends of the peer communication, i.e. from the point of view of each of the two remote peers. Specifically, the local peer schema is fully known a priori, whereas the remote one is gradually revealed through the incoming queries. The demonstration shows the progressive composition of the remote schema on each end, as well as the mappings that are discovered. It is interesting to see that the the progress of the mapping discovery may differ on the one and the other end. This is due to: (i) different pace of schema reveal because of diverse incoming queries from the other party, but also, (ii) the different schema structures.

Furthermore, the running examples illustrate the discovery of conditional mappings, i.e. mappings comprise data value constraints. These constraints originate only in the incoming queries from the remote peer, since the automatic matching tool does not provide concept correspondences that hold under value conditions. The interest focuses on how the query constraints are deduced and introduced in new concept correspondences that are merged with the correspondences produced by the automatic matcher.

Data integration scenario This scenario displays two autonomous databases that belong to a federated system. The databases desire a contingent integration of their schemas and data, so that they can selectively retain or yield their autonomy. The demonstrated mapping tool takes as input this pair of revealing schemas and discovers mappings.

The scenario includes examples that exhibit the variety of circumstances under which mapping discovery takes place, w.r.t. how big and what schema parts are revealed. Schema elements are unveiled on demand in order to watch different courses of progress in mapping discovery. The visitors can discern the feasibility of data integration based on the revealed schema parts and the discovered mappings. The examples include the construction of conditional mappings based on value constraints that the user feeds in the mapping mechanism. These are deduced from data samples.

[2]For details on the PDMS setting the reader is referred to [9]

5. REFERENCES

[1] http://dbs.uni-leipzig.de/Research/coma.html.

[2] http://www.dbnet.ece.ntua.gr/˜vkante/mapping/schemas.

[3] Bogdan Alexe, Laura Chiticariu, Renée J. Miller, Daniel Pepper, and Wang Chiew Tan. Muse: a system for understanding and designing mappings. In *SIGMOD Conference*, pages 1281–1284, 2008.

[4] Yuan An, Alexander Borgida, Renée J. Miller, and John Mylopoulos. A semantic approach to discovering schema mapping expressions. In *ICDE*, pages 206–215, 2007.

[5] Marcelo Arenas, Vasiliki Kantere, Anastasios Kementsietsidis, Iluju Kiringa, Renée J. Miller, and John Mylopoulos. The hyperion project: from data integration to data coordination. *SIGMOD Record*, 32(3):53–58, 2003.

[6] A. Halevy, Z. Ives, J. Madhavan, P. Mork, D. Suciu, and I. Tatarinov. The Piazza Peer Data Management System. In *IEEE TKDE*, 2003.

[7] Mauricio A. Hernández, Renée J. Miller, and Laura M. Haas. Clio: A semi-automatic tool for schema mapping. In *SIGMOD Conference*, page 607, 2001.

[8] V. Kantere, D. Bousounis, and T. Sellis. Mapping Discovery over Revealing Peer Schemas. Submitted for publication. http://www.dbnet.ece.ntua.gr/˜vkante/mapping.

[9] V. Kantere, D. Tsoumakos, T. Sellis, and N. Roussopoulos. GrouPeer: Dynamic clusterinf of P2P Databases. In *Information Systems, doi:10.1016/j.is.2008.04.002*, 2008.

[10] M. Lenzerini. Data Integration: A Theoretical Perspective. In *21th ACM PODS*, 2002.

[11] Yannis Velegrakis, Renée J. Miller, Lucian Popa, and John Mylopoulos. Tomas: A system for adapting mappings while schemas evolve. In *ICDE*, page 862, 2004.