Queries, the Missing Link in Automatic Data Integration

Aibo Tian, Juan F. Sequeda, and Daniel P. Miranker

Department of Computer Science, The University of Texas at Austin
Austin, Texas, USA
atian@utexas.edu, {jsequeda, miranker}@cs.utexas.edu

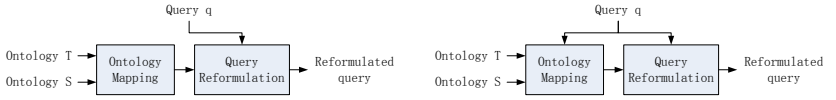
Abstract. This paper introduces the ontology mapping approach of a system that automatically integrates data sources into an ontology-based data integration system (OBDI). In addition to the target and source ontologies, the mapping algorithm requires a SPARQL query to determine the ontology mapping. Further, the mapping algorithm is dynamic: running each time a query is processed and producing only a partial mapping sufficient to reformulate the query.
This approach enables the mapping algorithm to exploit query semantics to correctly choose among ontology mappings that are indistinguishable when only the ontologies are considered. Also, the mapping associates paths with paths, instead of entities with entities. This approach simplifies query reformulation. The system achieves favorable results when compared to the algorithms developed for Clio, the best automated relational data integration system.

We have developed an Ontology-based Data Integration (OBDI) system that departs from the conventional OBDI organization. The goal is to include automatic integration of new data sources, provided those data sources publish a self-describing ontology. A consequence of that goal is there is no longer the opportunity for an engineer to review and correct an ontology matching prior to its use by the query reformulation system. As ontology matching is understood to be an uncertain process, some other method of mapping refinement is needed. Our system uses queries for this purpose.

Ontology mapping in conventional OBDI systems is determined prior to, and without knowledge of the queries to be executed [3]. A static representation of a mapping between target and source ontologies serves as input to a query reformulation module (Fig. 1(a)). In the system described here, ontology mapping is a dynamically computed component whose result depends on the query that is being processed (Fig. 1(b)). In effect, the query becomes a third argument to the ontology mapping algorithm. The query provides context for selecting among competing mappings. Since a mapping is specific to a query, the results may be limited to the partial mapping required by the query reformulation system.

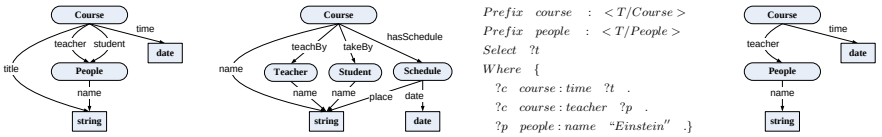
The organization was motivated by the following observations. A mapping method may determine that an entity in one ontology maps with equal likelihood to two or more entities in the other ontology. The mapping and reformulation of certain queries is correct only if one pairing is chosen. The correct choice may be different for different queries. The query itself may lend additional semantics that correctly resolve the ambiguity.

These observations are supported by the example in Fig. 2. Looking at the ontologies alone, there is insufficient information to determine if the class $T{:}People$ should



(a) Traditional (b) The proposed

Fig. 1. OBDI systems with the traditional and the proposed ontology mapping component.



(a) Ontology T (b) Ontology S (c) SPARQL query (d) Query graph

Fig. 2. Example ontologies and SPARQL query.

be mapped to S:$Teacher$ or to S:$Student$. A third possibility is a one-to-many mapping entailing both. However, given the SPARQL query in Fig. 2(c), which asks for the time of the course that is offered by "Einstein", it is clear that T:$People$ should only be mapped to S:$teacher$. A complementary query addressing student enrollment requires T:$People$ to be mapped to S:$student$. Correct answers from a reformulated query can be achieved only through mutually exclusive query dependent mappings.

An overview of the matching algorithm is as follows. To exploit the context implicit in a query, the algorithm maps paths in the target ontology to paths in a source ontology. A path contains a datatype and multiple classes connected by properties. Each element can be considered as the context of other elements in the path. For example, if a path contains a class $People$, and a property $teaches$, then we can infer that $People$ is a $Teacher$ instead of a $Student$. It follows that not all entities in one path have a corresponding entity in the corresponding path. Path-based mapping must necessarily accommodate this. Since the number of unconstrained paths in a graph is much larger than the number of vertices, the properties suggest that path mapping is a combinatorially much harder problem. However, as the organization stipulates that ontology mapping is dynamic and specific to a query. Thus, the mapping may be limited to only those mappings required to reformulate the query. Relative to ontologies, queries are very small, and only the paths in the target ontology corresponding to the query need to be mapped. These constraints limit the search problem to a manageable size.

Consider the specific SPARQL query in Fig. 2(c). Fig. 2(d) illustrates the part of ontology that corresponds to q, which is called a query graph. The task is to generate mappings that can be used to reformulate q in terms of ontology S. The algorithm must address the following challenges. The class $People$ in T can be mapped to $Teacher$ or $Student$ in S, and some entities, such as class $Schedule$ in S, do not have any mapped entity in T. However, the mapping for $Schedule$ is necessary to reformulate the query.

All of these challenges are met by mapping paths, represented as sequences of labels, where the sequence comprises alternating vertex and edge labels. In the example, query q has two paths in its query graph:

$\{T{:}Course, T{:}teacher, T{:}People, T{:}name, string\}$ and $\{T{:}Course, T{:}time, date\}$

We search for a subgraph with two paths in ontology S, which have the highest probability such that each of the paths in S are mapped to paths that correspond to the query graph of q. The probability of each path mapping is determined by scoring the similarity of all labels in the paths.

Mapping results should be:

$$\{T{:}Course, T{:}teacher, T{:}People, T{:}name, string\} = \{S{:}Course, S{:}teachBy, S{:}Teacher, S{:}name, string\}$$

$$\{T{:}Course, T{:}time, date\} = \{S{:}Course, S{:}hasSchedule, S{:}Schedule, S{:}date, date\}$$

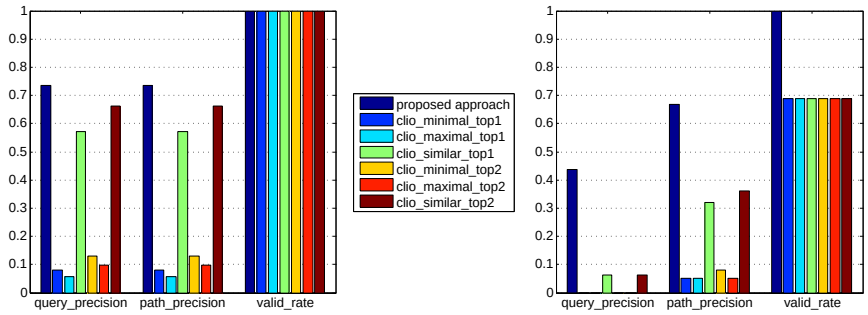
Note that this mapping is specific to the query. If another query asks for the time of the course that is taken by "Einstein", the mapped path should contain $S{:}Student$ instead of $S{:}Teacher$. Thus, defining similarity to a sequence of labels identified by the query introduces context. Limiting the problem to the paths in the query graph not only limits the size of the computation, but also removes any consideration of potentially conflicting interpretation. Given that sequence matching is an endemic problem in genomic data processing, there are many avenues open to exploration.

Experimental Setup: The evaluation comprises real world ontologies from bibliography domain. DBLP ontology is generated by direct mapping the relational schema of the DBLP metadata database using Ultrawrap [2], and the UMBC ontology is from the OAEI benchmark track[1]. These ontologies can be found on our website[2]. We manually generated groundtruth mappings between paths. Subsequently, a computer program systematically generates two kinds of SPARQL queries for each ontology. (1) A *PathOnly* query has query graph consisting of only one path in the groundtruth mappings. (2) A *ClassAll* query has query graph consisting of the set of all paths that share a same source in the groundtruth mappings. For each query, the generated path mappings are evaluated by comparing to the groundtruth mappings. The mapping results are evaluated by three metrics. $valid\_rate$ measures whether the generated mappings contain all paths in a query, regardless of correctness. $path\_precision$ measures the correctness of individual path mapping. $query\_precision$ measures whether all path mappings are correct for a query.

Baseline: Clio is a semi-automatic relational schema mapping system, however, the resulting algorithms are applicable to ontologies [1]. We implemented multiple configurations of Clio as baselines. All baselines first generate mappings between datatype properties by picking the ones with highest similarities. Given a query, the baselines find the mapping candidates that contain all the mapped datatype properties. If there exists more than one candidates, Clio asks a user to make the decision, which is not allowed in our automatic setting. We implement three baselines to approximate this process: *clio-minimal*, *clio-maximal*, and *clio-similar*, which chooses the mapping candidate with minimal summation of path lengths, maximal summation of path lengths, and

[1] http://oaei.ontologymatching.org

[2] http://www.cs.utexas.edu/~atian/page/dataset.html



(a) Bibliography, *PathOnly* (b) legend (c) Bibliography, *ClassAll*

Fig. 3. The evaluation results for bibliography data sets.

the highest similarity between the sources of the paths. We also enhance the baselines, by keeping 2 mappings instead of 1 for each datatype property. We generate mapping for each of them, and consider the mapping as correct if any of them is correct.

Results: Overall, our approach dominates over all baselines as shown in Fig. 3. For *PathOnly* queries, $query\_precision$ and $path\_precision$ are the same, and all approaches have 100% $valid\_rate$, because each query only consists of one path in the query graph. In terms of $query\_precision$ and $path\_precision$, our approach is 0.15 higher than the best top1 baselines. The three top2 baselines are improved with respect to the top1 approaches. However, even though the top2 approaches choose the correct mapping from large number of candidate mappings, it would still not yield a mapping with higher precision than our approach. clio_similar has better performance comparing to clio_minimal and clio_maximal. This indicates that the similarity between sources is important to the path mapping. For *ClassAll* queries, none of the baselines are competitive with respect to our approach. This is because the baselines determine the datatype property and source mappings first, and only use query to find valid path mappings. In some cases, the valid mappings are not existed. For our approach, the path mappings are jointly determined by both entity mappings and the query, so we have higher chances to find valid correct mappings.

References

1. R. Fagin, L. Haas, M. Hernández, R. Miller, L. Popa, and Y. Velegrakis. Clio: Schema mapping creation and data exchange. *Conceptual Modeling: Foundations and Applications*, pages 198–236, 2009.
2. J. F. Sequeda and D. P. Miranker. Ultrawrap: Sparql execution on relational data. Technical Report TR-12-10, University of Texas at Austin, Department of Computer Sciences, 2012.
3. H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-based integration of information-a survey of existing approaches. In *IJCAI-01 workshop: ontologies and information sharing*.