On Ambiguity and Query-Specific Ontology Mapping

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

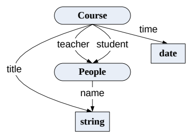
Abstract. In the course of developing an ontology-based data integration system (OBDI) that includes automatic integration of data sources, and thus, includes algorithmic ontology mapping, we have made the following observations. A mapping method may determine that an entity in one ontology maps with 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. Finally, the query itself may lend additional semantics that correctly resolve the ambiguity.

These observations suggest a targeted ontology mapping problem, *query-specific ontology mapping*. In addition to the two ontologies, a query serves as a third argument to the mapping algorithm. Further, the mapping algorithm need not produce a complete mapping, but only a partial mapping sufficient to correctly reformulate the query. We detail a number of open issues on how this problem statement might be refined, and consider features of its evaluation.

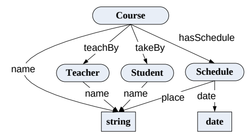
Ambiguity in Ontology Mapping: Consider the idealized representation (Fig. 1) of a critical issue in the automatic integration of new data sources in an OBDI system. T and S respectively represent target and data source ontologies. Looking at the ontologies alone, there is insufficient information to determine if the class T:People should be mapped to S:Teacher or to S:Student. A third possibility is a one-to-many mapping entailing both. Given the SPARQL query (Fig. 1c), it becomes clear that the query should be reformulated using *only* the mapping {T:People = S:Teacher}. A complementary query about students should be reformulated using *only* the complementary mapping. Thus, any static chose of one mapping will yield reformulated queries that return incorrect results.

Formulations of Query-Specific Ontology Mapping: In our system we compute a similarity matrix between all entities in the two ontologies [3]. The details may be borrowed from any ontology mapping algorithm that includes this step [2]. Given a query on the target ontology, our system uses a joint probability model to identify a maximal scoring, partial mapping that covers the target ontology entities mentioned in the query or that are needed to reformulate the query. Thus, our solution can be characterized as one that takes three arguments, and produces a partial mapping specific to the query.

There are at least two other approaches that may be considered and that produce a complete mapping and thus retain more of the standard definition of ontology matching. First is to consider complex mappings. For example, instead of choosing {T:People =



(a) Ontology T



(b) Ontology S

```
Prefix  course  :  <T/Course>
Prefix  people  :  <T/People>
Select  ?t
Where  {
  ?c  course:time  ?t  .
  ?c  course:teacher  ?p  .
  ?p  people:name  "Einstein"  .}
```

(c) SPARQL query

Fig. 1. Example ontologies and SPARQL query.

S:Teacher} or {T:People = S:Student}, the mapping system can detect "Teacher is the People who teaches" (similar for Student). However, to the best of our knowledge, there is no automatic system that can detect this kind of complex mapping.

Another approach may consider an entire workload of queries, as a batch or as a continual pay-as-you-go refinement. In other words, a complete mapping is determined, but the information in a set of queries is used to bias the choices made. As many applications comprise a set of dynamic web pages, their query set is easily identified. Consider the example and a course selection application. Since students are often interested in who is teaching a class, (and their grading policy), and privacy laws disallow revealing their fellow student's enrollment, the mapping {T:People = S:Teacher} would always be correct. Incremental, pay-as-you go, solutions could integrate crowd-sourcing.

The pedagogical example's brevity shouldn't be used to diminish the problem's importance. Comparing to Clio's[1] algorithms our system demonstrates favorable results [1, 3]. Inspection of individual results suggests that resolving ambiguity is the primary source of improvement, and can be significant. However measuring the quality of the solutions, as a whole, and quantifying the frequency of ambiguity poses its own set of problems. Gold standard baselines must include queries and correct mappings. OAEI benchmarks cannot be used directly. Correct query reformulation may not require a unique mapping. Entity level ambiguity may not manifest wrt query reformulation, making it hard to identify through manual curation. To date, we have created three such test cases[2]. The test suite accommodates the unique mapping problem by including additional partial mappings and including test data corresponding query results. Not all ambiguity may be revealed. Our inspection of individual results looked at the discrepancies between the two systems. False negatives are not quantifiable.

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*, 2009.
2. P. Shvaiko and J. Euzenat. Ontology matching: state of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering*, 2012.
3. A. Tian, J. F. Sequeda, and D. P. Miranker. Query specific ontology matching. Technical report, Department of Computer Science, University of Texas, 2012.

[1] Clio is an automatic relational schema mapping system. However, the algorithms are applicable to ontologies.

[2] The test cases are available, see http://www.cs.utexas.edu/~atian/page/dataset.html