Ontology mapping for querying heterogeneous information sources

SIDI MOHAMED BENSLIMANE[1]
AFAF MERAZI[2]
MIMOUN MALKI[1]
DJAMEL AMAR BENSABER[1]

Evolutionary Engineering and Distributed Information Systems Laboratory
Computer science Department, University of Sidi Bel Abbes
B.P 89 Sidi Bel Abbes, 22000 Algeria
[1] (Benslimane,Malki,Amabensaber)@univ-sba.dz
[2] afaf.merazi@gmail.com

Abstract. Ontology mapping is becoming a crucial aspect in providing the background knowledge required for solving heterogeneity problems between semantically described data sources, and accessing distributed information repositories. Developing such ontology mapping has been a core issue of recent ontology research. In this paper we present a three-layer framework to (semi-)automatically discovering and using ontology mapping. We show how such resulting mapping is used for resolving semantic interrogation tasks, and enabling runtime semantic interoperability across heterogeneous information systems using semantic web technologies.

Keywords: Ontology mapping, Semantic Web, Semantic interoperability, Similarity measures.

(Received August 17, 2007 / Accepted December 17, 2007)

1 Introduction

Intuitively, ontologies can be seen as defining the basic terms and relations of a domain of interest, as well as the rules for combining these terms and relations [11]. They are considered to be an important technology for the semantic Web. Ontologies are used for communication between people and organizations by providing a common terminology over a domain. They provide the basis for interoperability between systems. They can be used for making the content in information sources explicit and serve as an index to a repository of information. Further, they can be used as a basis for integration of information sources and as a query model for information sources. They also support clearly separating domain knowledge from application-based knowledge as well as validation of data sources. Many onthologies have already been developed and many of these ontologies contain overlapping information. Often we would therefore want to be able to use multiple ontologies. For instance, companies may want to use community standard ontologies and use them together with company-specific ontologies. Applications may need to use ontologies from different areas or from different views on one area. Ontology builders may want to use already existing ontologies as the basis for the creation of new ontologies by extending the existing ontologies or by combining knowledge from different smaller ontologies. So to bring together these disparate source ontologies, two approaches are possible: (1) merging the ontologies to create a single coherent ontology, or (2) aligning the ontologies by establishing links between them and allowing the aligned ontologies to reuse information from one another. Ontology merging is the process of generating a single, coherent ontology from two or more existing and different ontologies related to the same subject [20]. A merged single coherent on-

tology includes information from all source ontologies but is more or less unchanged. The original ontologies have similar or overlapping domains but they are unique and not revisions of the same ontology [21]. Ontology alignment is the task of creating links between two original ontologies. Ontology alignment is made if the sources become consistent with each other but are kept separate [18]. Ontology alignment is made when they usually have complementary domains.

Until now, ontology designers have performed this complicated process of ontology merging and alignment mostly by hand, without any tools to automate the process or to provide a specialized interface. It is unrealistic to hope that merging or alignment at the semantic level could be performed completely automatically. It is, however, possible to use a hybrid approach where a system performs selected operations automatically and suggests other likely points of alignment or merging. Some researchers view the mapping process as an integral part of alignment or merging. Clearly, mapping is an essential aspect of alignment and could also be used to initiate merging.

In this paper we present a three-layer framework to (semi-)automatically discovering and using ontology mapping. We show how such resulting mapping is used for resolving semantic interrogation tasks, and enabling interoperability across heterogeneous information systems using semantic web technologies. The rest of the paper is organized as follows. Section 2 introduces the overall of our framework. Section 3 describes the mapping discovery and representation layer. Section 4 explains some features of the semantic query layer. Section 5 presents the implementation of our system and details some experimental result. Finally, Section 6 contains concluding remarks and suggests some future works.

2 The overall framework

The overall system architecture, as shown in figure 1, involves the following layers:

- The information source layer includes a set of data sources which typically store their data in relational databases. Each data source is related to a local ontology by semantic relationship. Data sources schemas are mapped into corresponding classes or properties defined in the local ontology.

- At the mapping discovery layer, a mapping generation process produces semantic correspondences between the local ontologies. The discovered mapping is encoded using standardized mapping representation languages.

- At the semantic query layer a form-based query interface is offered to construct semantic queries over the set of ontology mapping. A semantic query is automatically generated at runtime, and submitted to the semantic query engine, where the query will be rewritten into a set of SQL queries using mapping contained in the ontology mapping registry. Finally, the results of SQL queries will be merged and forwarded back to the end-user.

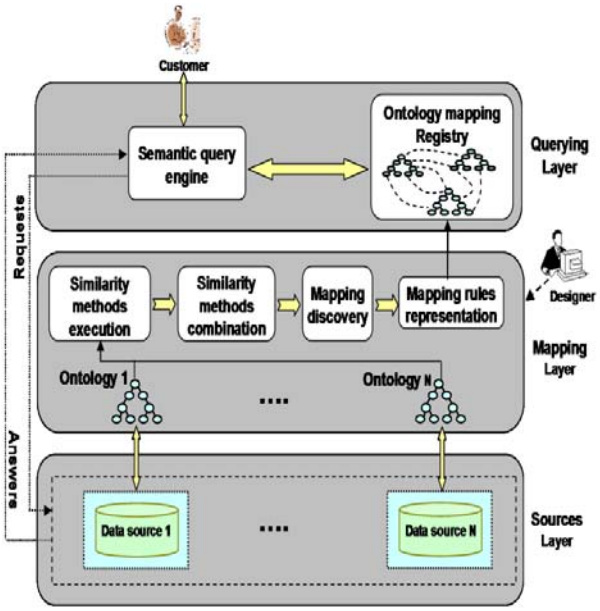

Figure 1: Ontology mapping and semantic querying framework

3 Ontology mapping layer

Ontology mapping is one of the core tasks for ontology interoperability. A systematic and epistemological account of ontology mapping and its related definitions is provided in the surveys of [13] for ontology mapping, [23] for database schema matching, and [17] for combined views of these two mapping regimes. Due to the wide range of expressions used in this area (merging, alignment, integration etc.), we adopt the following definition for the term mapping: "Given two ontologies O_1 and O_2, mapping one ontology onto another means that for each entity (concept C, relation R, or instance I) in the source ontology O_1, we find a corresponding entity, which has the same intended meaning, in the target ontology O_2" [8].

The mapping generation methods basically operate in three phases. Firstly, given two ontologies, how do we find the similarities between them and how do we

determine which concepts and properties represent similar notions. Second, according to the similarity measures obtained in the first step, the methods specify how to represent the mapping between entities. Finally the resulting mapping is used for various interrogation tasks (e.g. query answering, web-service composition, etc.)

3.1 Mapping discovery

The task of finding mapping has been an active area of research in both database and ontology communities [13, 23]. We identify two major architectures for mapping discovery between ontologies. For the first approach, finding correspondences between two extensions can be facilitated by a general upper ontology [16]. The second set of approaches comprises heuristics-based or machine learning techniques that use various characteristics of ontologies, such as their structure, definitions of concepts, and instances of classes, to find mapping. To compare two entities from two different ontologies, one considers their characteristics, i.e. their features [8]. The features of ontological entities are extracted from extensional and intentional ontology definitions [8]. Each of the features can be used to calculate ontology mapping function between the entities of the ontologies to be mapped based on similarities measures.

Definition 1. A similarity measure is a real-valued function $Sim(x, y) : O \times O \rightarrow [0, 1]$ measuring the degree of similarity between the two entities x and y.
$Sim(x, y) = 1 \rightarrow x = y$: two entities are identical.
$Sim(x, y) = 0$: two entities are different and have no common characteristics.

Definition 2. Formally we define an ontology mapping function, $Map : O_1 \rightarrow O_2$
$Map(e_{O_1}) = e_{O_2}$, if $Sim(e_{O_1}, e_{O_2}) > threshold$.
where: O_i is an ontology; e_{O_i} are entities of O_i and $Sim(e_{O_1}, e_{O_2})$ is a similarity function between two entities e_{O_1} and e_{O_2}.

In what follow, we will introduce a set of rules for lexical, structural, and taxonomic based mapping discovery between ontologies.

3.1.1 Lexical-based mapping discovery

The more evident method for relating two entities is to look at the label describing them. This label generally consists of one term that has to be compared with the labels of the other ontology. Several ideas have already been created to compare tow strings.

String distance String distance or string similarity algorithms take as an input two strings and return a value indicating the distance or similarity between them.

Rule 1. *Two entities are identical if they have the same identifier (e.g. URIs) or the same name (e.g. RDF label).*

For two concepts e_1 and e_2, the string similarity measure Sim_{stri} can be given based on edit distance, which is shown in equation (1).

$$Sim_{stri}(e_1, e_2) = max(0, \frac{min(|e_1,e_2| - ed(e_1,e_2))}{min(|e_1,e_2|}) \in [0, 1] \quad (1)$$

Where ed is the edit distance formulated by Levenshtein [14]. It measures the minimum number of token insertions, deletions, and substitutions required to transform one string into another.

Lexical semantics The technique reviewed above is efficient but need to be completed. Two terms may be similar even if they are completely differently spelt. This is the example of synonyms. More generally, two terms having a related sense deserve to be somehow related. In order to be able to capture these relations between the terms, it is necessary to get their semantics. The lexical semantics similarity measure explores the semantic meanings of the word constituents by using external resources, like user-defined lexica and/or dictionaries (e.g. Wordnet, [10]) to help identify synonyms in matching.

Rule 2. *Two entities are identical if they are synonyms*.

For two concepts e_1 and e_2, the lexical semantics similarity measure Sim_{lsem} can be given using the WordNet synsets (i.e. term for a sense or a meaning by a group of synonyms) based on the equation (2).

$$Sim_{lsem}(e_1, e_2) = 1/length(e_1, e_2) \quad (2)$$

Where *length* is the length of the shortest path between two entities using node-counting.
Based on equation (1) and (2), we give the lexical similarity equation as (3).

$$Sim_{lexi}(e_1, e_2) = max[Sim_{stri}(e_1, e_2), Sim_{lsem}(e_1, e_2)] \quad (3)$$

3.1.2 Structure-based mapping discovery

Lexical similarity measure is not sufficient. This section presents how the concept's structure and property's

structure, helps in finding similarities between ontologies. A concept's structure includes its properties and relations. A property's structure includes its domain, range, and constraints. The structure based similarity of two concepts is defined by the combined similarity between the two concept's structures.

Rule 3. *Two concepts are equal if their properties are equal*.

$$Sim_{stru}(e_1, e_2) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(Sim_{prop}(p_i,p_j))}{(n+m)/2} \quad (4)$$

Where p_i and p_j are the properties of e_1 and e_2 respectively. n and m are the properties number of e_1, e_2 respectively.

Rule 4. *Two properties are equal if one of the following conditions can be satisfied:*
(i) They have the same name (using Rule 1).
(ii) The domain and the range of the two properties are equal.

$$Sim_{prop}(p_i, p_j) = max(Sim_{lexi}(p_i, p_j), \frac{Sim(p_i^{dom},p_j^{dom})+Sim(p_i^{ran},p_j^{ran})}{2}) \quad (5)$$

where p_i^{dom}, p_i^{ran} and p_j^{dom}, p_j^{ran} are the domain and the range of the proprieties p_i and p_j respectively.

3.1.3 Taxonomy-based mapping discovery

It is tempting to use back-propagation in the inheritance hierarchy, as we did for structure matching. That is, if two concepts have similar specializations (generalization), the concepts are probably related. Given two concepts, we estimate their similarity by comparing the similarities of their ascendants (descendants).

Rule 5. *Two concepts are similar if their sub-concepts are the same*.

The similarity about sub-concepts of two concepts e_1 and e_2 is denoted as Sim_{subc} which is shown as equation (6).

$$Sim_{subc}(e_1, e_2) = \frac{\sum_{i=1}^{l}\sum_{j=1}^{k}(Sim(e_{1_{d_i}},e_{2_{d_j}}))}{(l+k)/2} \quad (6)$$

Where $e_{1_{d_i}}$ and $e_{2_{d_j}}$ are the sub-concepts of e_1 and e_2 respectively. l and k are the sub-concepts number of e_1, e_2 respectively.

Rule 6. *Two concepts are similar if their super-concepts are the same.*

The similarity about super-concepts of two concepts e_1 and e_2 is denoted as Sim_{supc} which is shown as equation (7).

$$Sim_{subp}(e_1, e_2) = \frac{\sum_{i=1}^{t}\sum_{j=1}^{v}(Sim(e_{1_{a_i}},e_{2_{a_j}}))}{(t+v)/2} \quad (7)$$

Where $e_{1_{a_i}}$ and $e_{2_{a_j}}$ are the super-concepts of e_1 and e_2 respectively. t and v are the sub-concepts number of e_1, e_2 respectively.

Based on equation (6) and (7), we give the conceptual similarity equation as (8).

$$Sim_{taxo}(e_1, e_2) = [W_{subc} * Sim_{subc}(e_1, e_2) + W_{supc} * Sim_{supc}(e_1, e_2)]/(W_{subc} + W_{supc}) \quad (8)$$

With W_{subc} and W_{supc} being the weights which indicates the importance of the similarity methods Sim_{subc}, and Sim_{supc} respectively.

3.1.4 Similarities aggregation

To achieve high accuracy for a large variety of ontologies, a single similarity method may be unlikely to be successful. Hence, combining different similarity methods is an effective way. For this purpose, many approaches combining the results of several independently executed mapping algorithms are proposed [5], [6], [7], [15], [19]. There are two kinds of approaches to combine multiple similarity methods: hybrid and composite combination [5].
Hybrid approach is the most common where different mapping criteria (e.g. name, structure) are used within a single algorithm. Typically these criteria are fixed and used in a specific way. By contrast, a composite mapping approach combines the results of several independently executed mapping algorithms, which can be simple of hybrid. This allows for a high flexibility, as there is the potential for selecting the mapping algorithms to be executed based on the mapping task at hand. In this paper, we exploit the later to combine multiple similarity methods in ontology mapping. A combination of the so far presented rules leads to better mapping results compared to using only one at a time. A general equation for this integration task can be given by the following average over the weighted similarity methods.

$$Sim(e_1, e_2) = [W_l * Sim_{lexi}(e_1, e_2) + W_s * Sim_{stru}(e_1, e_2) + W_t * Sim_{taxo}(e_1, e_2)]/(W_l + W_s + W_t) \quad (9)$$

With W_l, W_s, W_t being the weights which indicates the importance of the similarity methods Sim_{lexi}, Sim_{stru}, and Sim_{taxo} respectively. The weights could be as-

signed manually or learned, e.g. through maximization of the f-measure of a training set.

3.1.5 Process

We briefly introduce a canonical process that subsumes all the mapping discovery steps illustrated in figure 1. It is started with two ontologies, which are going to be mapped onto one another, as its input.

Similarity methods execution. A main step during mapping discovery is the execution of multiple independent mapping algorithms based on all the introduced similarity measures (equation 1 through equation 9). Each algorithm determines similarity values between a candidate mappings (e_1, e_2) based on their definitions in O_1 and O_2, respectively. Each mapping algorithm determines an intermediate mapping result according to the similarity value between 0 and 1 for each possible candidate mapping. The result of the mapping execution phase with k algorithms, m entities in O_1 and n entities in O_2 is a $k \times m \times n$ cube of similarity values, which is stored in the repository for later strategies detection and combination steps.

Similarity methods combination. By multiple mapping algorithms, there are several similarity values for a candidate mapping (e_1, e_2). For example, one is the similarity of their name and another one is the similarity of their structure. This step is to derive the combined mapping result from the individual algorithm results stored in the similarity cube. For each combination of ontology entities the algorithm-specific similarity values are aggregated into a combined similarity value, e.g. by taking the average or maximum value.

Mapping discovery. This step uses the individual or combined similarity values to derive mappings between entities from O_1 to O_2. Some mechanisms here are, using thresholds or maximum values for similarity mappings, performing relaxation labeling [2], or other criteria. The mapping process supports an optional designer interaction phase for mapping correction. The designer's specified mappings will influence the similarity computations for the neighborhood of the respective entities and thus can improve the mapping accuracy of structural algorithms. Eventually, the output is a mapping table including multiple entries of $Map(e_1, e_2)$ from O_1 to O_2.

3.2 Mapping representation

The discovered mapping is encoded using standardized mapping representation languages [24]. The language provide a formalism for describing a mapping element as a 5-uple: <id, e_1, e_2, n, R>, where id is a unique identifier of the given mapping element; e_1 and e_2 are the entities (Concepts, properties, relations) of the first and the second ontology respectively; n is a similarity measure in some mathematical structure (typically in the [0,1] range) holding for the correspondence between the entities e_1 and e_2; R is a relation holding between the entities e_1 and e_2.
The relation R is defined based on the confidence measure. If $Sim(e_1, e_2) = 1$ then R is equivalence $(=)$ relation. If $Sim(e_1, e_2) = 0$ then R is disjointness $(\perp)$ relation. If $Sim(e_1, e_2) > t$ (threshold) then R is subsume $(\supseteq)$ relation.

To illustrate our mapping discovery and representation process, we take as input a set of simplified ontologies concerning travel, accommodation and attraction sites. The ontologies share some related concepts and relations. The execution of multiple mapping discovery algorithms described above, will generate the e-Tourism ontology describing tourism domain described by a set of ontology mapping (see Figure 2). The mapping rules are stored in RDF format. For example, cost of the accommodation ontology is equivalent to price of the attraction ontology. This mapping rule is formulated as follow:

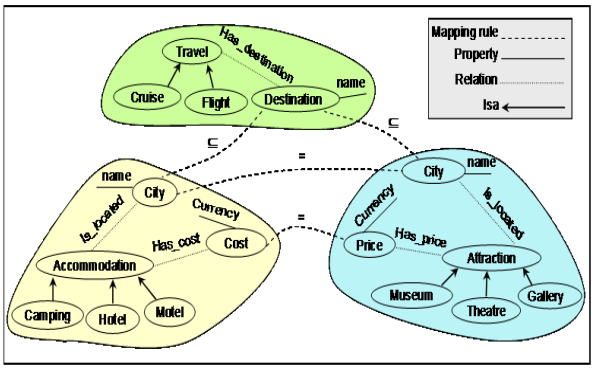

Figure 2: Example of simplified e-tourism ontology

```
<map>
<entity1 rdf:resource='URL1'/>
<entity2 rdf:resource='URL2'/>
<measure rdf:datatype='&xsd;float'> Value </measure>
<relation> relation </relation>
</map>
```

4 The semantic query layer

We select SPARQL [22] as our query language which is declarative, expressive and efficient for repositories

based on RDF graph (compatible with OWL). In SPARQL, an SQL-like query describes desired graph patterns with some variables and the selected boundable variables are selected as query result.

At the query layer, semantic query engine is used to process semantic SPARQL queries. Firstly, it gets mapping information from the semantic ontology mapping registry to decompose the semantic SPARQL query into a set of SPARQL sub-queries and to translate them to SQL sub-queries. Afterward, the SQL sub-queries are dispatched toward the specific data sources. Finally, the results of SQL sub-queries will be recomposed and transformed back to semantically-enriched format in a way that the user indicates. Figure 3 illustrates a query expressed in SPARQL. The query allows selecting the hotels in 'Paris' that have a cost lower than 60 euros, and displaying all its museums. The original query is translated into a set of SPARQL sub-queries (One query about accommodation, and one query about attraction).

Semantic SPARQL Query

```
PREFIX  trv: <owlmapping.example.com/Travel/ontology#>
PREFIX  acc: <owlmapping.example.com/Accommodation/ontology#>
PREFIX  att: <owlmapping.example.com/Attraction/ontology#>
SELECT ?myDest ?hotel ?hprice ?museum
FROM <owlmapping.example.com/Travel/ontology>
FROM <owlmapping.example.com/Accommodation/ontology>
FROM <owlmapping.example.com/Attraction/ontology>
WHERE {
       _:anyDest trv:name  ?myDest .
   FILTER ( ?myDest = "Paris") .
   ?hotel a acc:Hotel .
   ?hcity acc:name ?myDest .
   ?hotel acc:is_located ?hcity .
   ?hotel acc:hasCost ?hcost .
   FILTER ( ?hcost < 60 ) .
   ?museum a att:Museum .
   ?mcity att:name ?myDest .
   ?museum att :is_located ?mcity }
```

Query to Attraction Ontology

```
PREFIX att:<owlmapping.example.com
          /Attraction/ontology#>
SELECT ?myDest ?museum
FROM <owlmapping.example.com
          /Attraction/ontology>
WHERE {
 ? mycity att:name ? myDest
FILTER (? myDest = "Paris")
? museum a att:Museum
? mcity att:name ?myDest
? museum att :is_located ? mcity }
```

Query to Accommodation Ontology

```
PREFIX acc:<owlmapping.example.com
          Accommodation/ontology#>
SELECT ?myDest ?hotel ?hprice
FROM <owlmapping.example.com
          /Accommodation/ontology>
WHERE {
? Hcity acc: name ?myDest
FILTER ( ?myDest = "Paris")
?hotel a acc:Hotel .
?hotel acc:is_located ?hcity
?hotel acc:hasCost ?hcost
FILTER ( ?hcost < 60 )}
```

Figure 3: Semantic query expressed in SPARQL and its decomposition

5 Implementation and experimental results

In this section, we present some experiments we performed to assess the effectiveness of the ontology mapping development rules, and to verify that the proposed approach can contribute to help users resolving semantic interrogation tasks.

5.1 Prototype

A prototype is developed using Java (j2sdk 1.4.2), OWL-API [1], and Java WordNet Library (jwnl) for ontology mapping discovery and semantic query processing. The prototype that has a user-friendly GUI (Figure 4), has been implemented in order to experiment and verify that the proposed approach is doable.

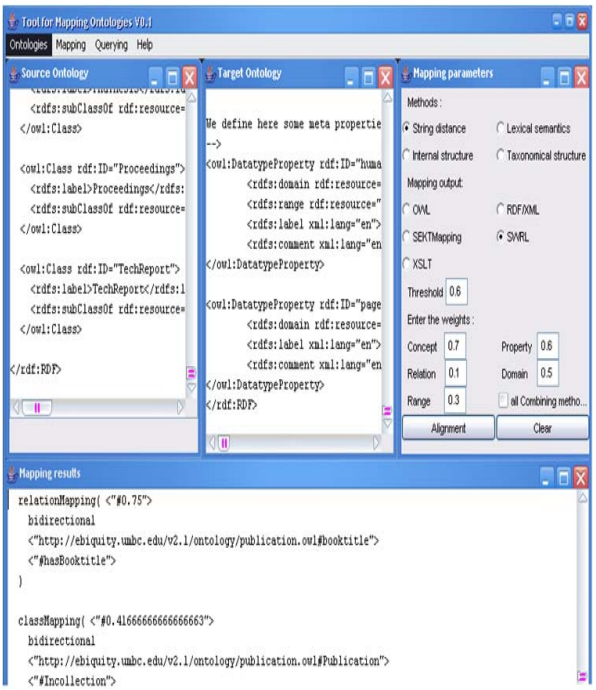

Figure 4: Snapshot of the ontology mapping discovery tool

To perform the ontology mapping discovery process, some parameters such as the mapped ontologies, information for the similarity methods combination, threshold, and weights, are given in an input configuration. The output ontology mapping can be formalized in the following standard formats: OWL, RDF/XML, SWRL [12], and XSLT [3].

5.2 Empirical Experiment

We evaluated our algorithm on the benchmark ontologies from the Ontology Alignment Evaluation Initiative (OAEI 2005, [9]). The benchmark ontologies consist of versions of a base ontology, where different aspects

have been changed. For most of the following tests, we focus on six interesting ontologies: In two cases (ontologies 10 and 227 from the test suite), all names and labels have been replaced with synonyms or foreign words, and in four cases, independently developed "real-world" ontologies that describe the same domain have been used (304, 615, 997, 1072).

We tested various techniques to evaluate the effectiveness of our algorithm using information retrieval metrics such as precision, recall and F-measure [4]. The results in figure 5 show that the configuration that combines different similarity methods performs better or equal than the configuration with only one similarity method.

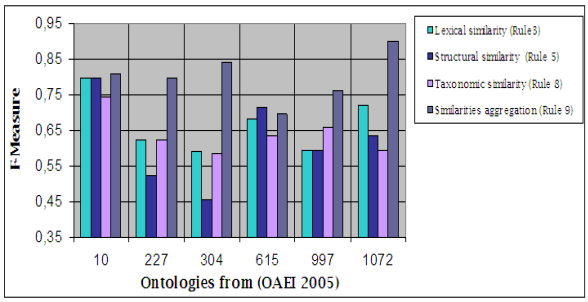

Figure 5: Comparison of different configurations of our algorithm

To find out what value for the threshold is best, we took a closer look at ontologies 615. Figure 6 shows the relation between the threshold and the precision, recall and F-measure on ontologies 615.

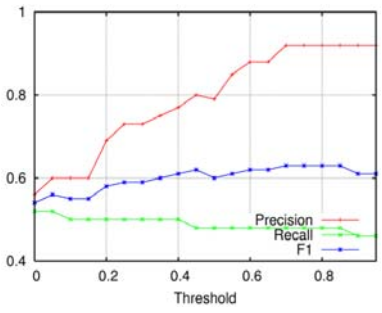

Figure 6: Relation between threshold and precision, recall and F1 for ontologies 615

In this ontology, precision and recall at threshold 0 are lower. When raising the threshold, recall drops only slightly while precision increases rather quickly. Maximum F-measure is reached at a threshold between 0.7 and 0.85. We have to conclude that the best cut-off value for our mapping algorithm depends strongly on the dataset.

6 Conclusions

The increasing popularity of the semantic Web poses new challenges for ontology mapping. If we accept that mapping ontologies can provide a better knowledge management of the heterogeneous sources on the semantic Web, then issues of inconsistency and incompleteness need to be addressed. Therefore ontology mapping systems that operate in this environment should have the appropriate mechanisms to cope with these issues. In this paper, we have presented a three-layer framework to (semi-)automatically resolving semantic interrogation tasks, and enabling runtime semantic interoperability across heterogeneous information systems using semantic web technologies. We have implemented a tool to support this task and presented some experimental results.

As a perspective of our work, we plain to treat the maintenance of the mappings: if some changes occur in the local ontologies, some of the mappings may become inconsistent; the problem is therefore to detect the inconsistent mappings and to propagate the changes into the mapping definitions.

References

[1] Bechhofer, S., Volz, R., and Lord, P. Cooking the semantic web with the owl api. In *Proc. of the First International Semantic Web Conference 2003 (ISWC 2003), October 21-23, 2003, Sanibel Island, Florida*, pages 659–675, 2003.

[2] Calvanese, D., Giacomo, G. D., and Lenzerini, M. A framework for ontology integration. *The Emerging Semantic Web*, pages 201–214, 2002.

[3] Clark, J. Xsl transformations (xslt). http://www.w3.org/TR/xslt, 1999. World Wide Web Consortium (W3C).

[4] Do, H. H., Melnik, S., and Rahm, E. Comparison of schema matching evaluations. in proceedings of the 2nd int. workshop on web databases 2002.

[5] Do, H. H. and Rahm, E. Coma - a system for flexible combination of schema matching approaches. In *Proceedings of 28th International Conference on Very Large Data Bases (VLDB)*, pages 610–621, Hong Kong,China, 2002.

[6] Doan, A., Madhavan, J., Domingos, P., and Halevy, A. Y. Learning to map between ontologies on the semantic web. In *World Wide Web*

Conference, pages 662–673, Hawaii, USA, 7-11 May 2002.

[7] Ehrig, M. and Staab, S. Qom - quick ontology mapping. In *GI Jahrestagung (1)*, pages 356–361, September 2004.

[8] Ehrig, M. and Sure, Y. Ontology mapping - an integrated approach. In *First European Semantic Web Symposium, ESWS*, Lecture Notes in Computer Science, pages 76–91, Crete, Greece, May 10-12 2004. Springer.

[9] Euzenat, J., Stuckenschmidt, H., and Yatskevich, M. Introduction to the ontology alignment evaluation 2005. In *Integrating Ontologies '05, Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, Banff, Canada, October 2, 2005*.

[10] Fellbaum, C. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[11] Gomez-Pérez, A. Ontological engineering: A state of the art,expert update. *British Computer Society*, 2(3):33–43, 1999.

[12] Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., and Dean, M. Swrl: A semantic web rule language combining owl and ruleml. http://www.w3.org/Submission/SWRL/, 2004. World Wide Web Consortium (W3C).

[13] Kalfoglou, Y. and Schorlemmer, W. M. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18(1):1–31, 2003.

[14] Levenshtein, V. I. Binary codes capable of correcting deletions, insertions, and reversals. In *Cybernetics and Control Theory 10*, pages 707–710, 1966.

[15] Maedche, A., Motik, B., Silva, N., and Volz, R. Mafra - a mapping framework for distributed ontologies. In *13th International Conference, EKAW*, pages 235–250, Siguenza, Spain, October 1-4 2002.

[16] Niles, I. and Pease, A. Towards a standard upper ontology. In *In Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS).*, pages 2–9, Ogunquit, Maine, USA, 2001.

[17] Noy, N. F. and Klein, M. C. A. Ontology evolution: Not the same as schema evolution. *Knowl. Inf. Syst.*, 6(4):428–440, 2004.

[18] Noy, N. F. and Musen, M. Prompt: Algorithm and tool for automated ontology merging and alignment. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 450–455. AAAI Press / The MIT Press, 2000.

[19] Noy, N. F. and Musen, M. The prompt suite: Interactive tools for ontology merging and mapping, 2002.

[20] Pinto, H. S., Gomez-Perez, A., and Martins, J. P. Some issues on ontology integration. In *n Proc. of IJCAI99's Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends*, pages 7–12, Stockholm, Sweden, August 1999.

[21] Pinto, H. S. and Jo a. P. M. A methodology for ontology integration. In *K-CAP '01: Proceedings of the 1st international conference on Knowledge capture*, pages 131–138, New York, NY, USA, 2001. ACM Press.

[22] Prudhommeaux, E. and Seaborne, A. Sparql query language for rdf. technical report, world wide web consortium (w3c), april 2006. url http://www.w3.org/tr/rdf-sparql-query/.

[23] Rahm, E. and Bernstein, P. A. A survey of approaches to automatic schema matching. *VLDB J.*, 10(4):334–350, 2001.

[24] Zhdanova, A. V. and Shvaiko, P. Community-driven ontology matching. In *3rd European Semantic Web Conference, ESWC*, pages 34–49, Budva, Montenegro, June 11-14 2006.