Measuring Similarity of Elements in OWL DL Ontologies

Thanh-Le Bach, Rose Dieng-Kuntz

ACACIA Project, INRIA Sophia Antipolis
2004 route des Lucioles, BP 93, 06902 Sophia Antipolis, France
{Thanh-Le.Bach, Rose.Dieng}@sophia.inria.fr
http://www-sop.inria.fr/acacia/acacia.html

Abstract

OWL becomes nowadays a more and more widely-used language for representing ontologies. The number of OWL ontologies increasing in direct ratio to the development of the Semantic Web leads to the heterogeneity problem. The same concepts may be modeled differently, using different terms and different positions in concept hierarchy. The task of identifying similar entities (concepts, relations or individuals) in different ontologies becomes then crucial for the success of information integration systems, instance transformation… In this paper, we propose a new similarity measure for comparing entities in different OWL DL ontologies. This measure is designed so as to enable extraction of information encoded in OWL entity descriptions and to take into account the underlying meaning of OWL primitives. We propose a variable weighting scheme for combining more efficiently component similarities calculated from components in entity descriptions.

Introduction

Several researchers currently study thoroughly problems of comparison, alignment, matching, and integration of ontologies (Euzenat, 2004). The success of these tasks depends on the way how the similarity between entities of ontologies is defined. A good measure for the similarity between two entities in two ontologies will help to identify corresponding entities correctly.

In this article, we propose a new similarity measure for any two entities in two different OWL DL ontologies. This measure is based on the information extracted from the entity descriptions (definitions). An entity in OWL DL ontology can be a class, a relation, an instance or even the ontology itself. In our system, we try to extract as much as possible information encoded in the entity description. Extracted components are compared to produce partial component similarity values. They are then combined using prefixed weights under a variable weighting scheme (where weights can be changed during the calculation, depending on situation, for the better result). The similarity calculation takes into account the predefined meanings of OWL DL and RDF(S) primitives such as *rdf:type, owl:equivalentClass…* which we can extract from entity descriptions.

Copyright © 2002, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

A good similarity measure will be crucial for the success of several other emerging tasks in the context of semantic web such as comparing, mapping, aligning, merging or integrating ontologies as well as information.

Entity Similarity Measure

In our system, an entity in an OWL DL ontology can be a class which can have a name (an URI) or not (anonymous class), or be a relation. An entity is described in OWL DL ontology using RDF(S) or OWL DL primitives, such as *rdf:id, rdfs:range, owl:subClassOf…* Each of these primitives brings a piece of knowledge to the whole meaning of the entity. So, we can consider that the similarity between two entities in two ontologies is a combination of partial similarities which are similarities between pieces of descriptions using these primitives. The similarity combination is a variable-weighted sum calculated from partial similarities. An OWL DL ontology is an RDF document. We consider an OWL DL ontology as a set of RDF triples. Let O an ontology, let (s, p, o) a triple, where s, p, o are respectively subject, predicate, and object, O = { (s, p, o) }. Note that in OWL DL ontology, for every triple (s, p, o) $\subset$ O in entity descriptions, p is a predicate which is one of 33 RDF(S) and OWL primitives. Representation of OWL DL ontology in our system is lightly-modified RDF graph representation (Fig. 1). The labels on arcs starting from entities (classes, relations) are primitives in 33 RDF(S) or OWL primitives. For nodes which are instances, labels on arcs starting from them can be user-defined properties.

Let (s, p, o) a triple. We define:

$T(e) = \{ (e, p, o) \mid (e, p, o) \in O \}$, *the set of RDF triples having the entity e as their subject.* $P(e) = \{ p \mid \exists o, (e, p, o) \in T(e)\}$, *the set of predicates, which are parts of triples having entity e as their subject.* $O(e, p) = \{ o \mid (e, p, o) \in T(e) \}$, *the set of (RDF) objects, which are parts of triples having entity e as their subject and p as their predicate.* $E(e) = \{(p,o) \mid (e, p, o) \in T(e) \}$, *the set of predicate-object pairs, where the first is predicate and the second object in a triple having entity e as subject.*

The similarity measure between two entities e_1 in ontology O_1 and e_2 in ontology O_2, named $Sim(e_1,e_2)$ is based on two values: (1) similarity between their components and (2) similarity of their graph structure.

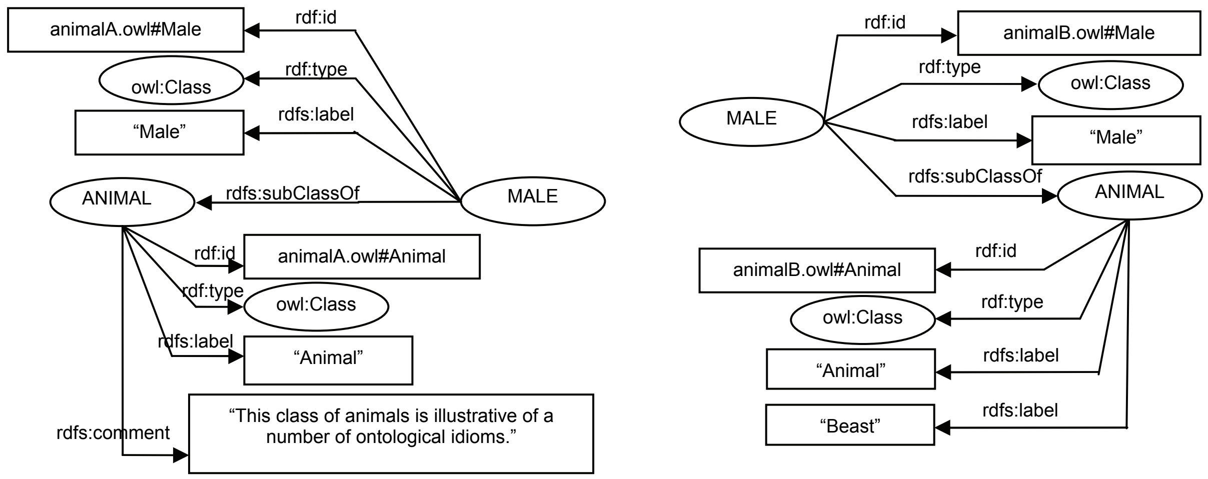

Fig. 1. Extract of modified RDF graph representations for two different ontologies: animalA.owl and animalB.owl

Similarity between Components

The components in an entity description are triples. Similarity between components of two entities is similarity between two sets of pairs $E(e_1)$ and $E(e_2)$. For comparing these two sets of pairs in order to produce a similarity value between [0,1], we propose the following steps:

1. Identify set of predicates P_{c1} which contain predicates in pairs in $E(e_1)$ having a similar predicate in a pair in $E(e_2)$ and similarly, P_{c2}. Let P_c the union of these two sets.

$P_{c1} = \{ p \mid p \in P(s_1) \wedge (\exists q \in P(s_2), SimPred(p,q) > 0) \}$

$P_{c2} = \{ p \mid p \in P(s_2) \wedge (\exists q \in P(s_1), SimPred(p,q) > 0) \}$

$P_c = P_{c1} \cup P_{c2}$

where SimPred(p,q) is a similarity function between two predicates, which are RDF(S) or OWL properties. SimPred is defined as follows: (i) SimPred(p,p) = 1; (ii) SimPred(p,q) is a predefined value in [0,1] if $p \neq q$. Some OWL properties can be considered as similar semantically, e.g. *owl:cardinality* and *owl:maxCardinality*.

2. To be able to calculate partial similarities over predicates which can appear several times in description of an entity, such as *rdfs:label, rdfs:subPropertyOf…*, we firstly collect objects in triples having this same predicate p. We thus obtain two sets of objects for two entities in two ontologies: $O(s_1,p)$ and $O(s_2,p)$. Secondly, objects in these two sets are paired, in order to calculate the similarity between object sets. The object pairing is done by the following algorithm: Let o_{1i} and o_{2j} two objects in these sets, $o_{1i} \subset O(s_1,p)$, $o_{2j} \subset O(s_2,p)$, (a) Find o_{1i} and o_{2j} whose $Sim_{total}(o_{1i},o_{2j})$ is maximal. (o_{1i},o_{2j}) is an object pair; (b) Remove o_{1i} from the set $O(s_1,p)$ and remove o_{2j} from the set $O(s_2,p)$; (c) Repeat step (a) until no more (o_{1i},o_{2j}) is found.

Finally, after the pairing process, similarities between paired objects are summed up, and then the summed value is divided by the maximal cardinality of both object sets. The obtained value is the partial similarity $Sim_{partial}$ over the considered predicate.

$$Sim_{partial}(s_1,s_2,p) = \frac{\sum_{(o_1,o_2)\in Pairing(O(s_1,p),O(s_2,p))} Sim_{total}(o_1,o_2)}{max(|O(s_1,p)|,|O(s_2,p)|)}$$

Using example in Fig. 1, O(ANIMAL$_1$, rdfs:label) will result {"Animal"}, O(ANIMAL$_2$, rdfs:label) will give {"Animal", "Beast"}. Pairing two sets gives {("Animal", "Animal")}. So the $Sim_{partial}$ over predicate *rdfs:label* for two classes ANIMAL will be 0.5.

Some predicates, such as *owl:equivalentProperty, owl: sameAs, owl:equivalentClass* which can appear several times in the description of an entity, require a different processing. Instead of pairing objects in triples having the same considered predicate in order to compute the partial similarity from the similarity between two sets of objects, the partial similarity is calculated based on the meaning of the predicate, and in this case, is the maximal similarity value of objects in two sets.

$$Sim_{partial}(s_1,s_2,p) = \max_{o_1\in O(s_1,p),o_2\in O(s_2,p)} (Sim_{total}(o_1,o_2), Sim_{total}(s_1,o_2), Sim_{total}(o_1,s_2))$$

3. For predicates which can only appear at most once in any entity description, such as *rdf:id, owl:complementOf, owl:inverseOf…*, the partial similarity is only based on two objects and the underlying meaning of the predicate. Function θ, depending on meaning of given predicate and objects, returns similarity value of two entities.

$$Sim_{partial}(s_1,s_2,p) = \theta(p, Sim_{total}(o_1,o_2))$$

As example, if the predicate is *owl:complementOf*, the partial similarity of two entities is based on the similarity between two objects of two triples: if two objects are similar classes, their complementary classes are also similar. The $Sim_{partial}$ of two entities over the predicate *owl: complementOf* will then be the Sim_{total} of the two objects.

4. Depending on the ontology modeling, description of an entity (class, relation) can consist of one or several RDF(S) /OWL properties. A given property (predicate) can appear in the description, others not. The total similarity between two entities is a combination of their partial similarities calculated as described above: it is a weighted sum from the obtained partial similarities. As discussed, the entity description components are not fixed, they vary from entity to entity. So, applying a fixed weighting scheme over partial similarities might be not efficient. Let us use a simple example: suppose that we have a simple ontology language which supports only three property primitives: *ol:id, ol:label* and *ol:comment*. An entity description is an arbitrary combination of these primitives: one can be defined using only *ol:id* for its identification, another one can be described by all three primitives for its identification, its human-readable name and comment. For two entities, our method will produce partial similarities from these components. In related research for combining partial similarities, each component is assigned a pre-fixed weight (e.g. 0.5, 0.35 and 0.15 respectively), and then the final result is the weighted sum of products of partial similarities and these weights. If the descriptions for both entities being compared contain only one primitive *ol:id*, the maximal similarity of two entities is only 0.5 (in the case that both identifications are same). To solve this problem, we propose a variable weighting scheme where predefined weights can be modified automatically in the calculation depending on descriptions of entities being compared. Using previous example, notifying that both entities have only one primitive *ol:id* in their descriptions, the corresponding weight for this primitive changes from 0.5 to 1.0, thus the obtained total similarity value may reach the value of 1.0.

$$Sim_{component}(s_1,s_2)=\sum_{p_i\in P_c}\phi(w_{p_i})*Sim_{partial}(s_1,s_2,p_i)$$

$\phi(w)$ is an adaptation function for modifying weights.

There are several weight-changing strategies. We propose the following one: (1) Initiate the 33 pre-fixed weights corresponding to the 33 RDF(S)/OWL primitives, so that their sum is equal to 1.0. These weights are firstly assigned manually to different values to give different emphasis to different components (corresponding to primitives) in entity descriptions. The fact that for entity similarity calculation, the information getting from *rdf:id* is more important than from *owl:versionInfo*, so the weight assigned to the former is set higher than one assigned to the latter; (2) When comparing two entities, for each RDF(S)/OWL primitive p_i which does not appear in the descriptions of both entities, set (automatically) its corresponding weight from w_i to 0.0 and increase (automatically) all other non-zero weights by an amount being equal to w_i/[number of non-zero weights]. This guarantees that their sum is always equal to 1.0.

Similarity of Entity Graph Structures

An entity description is represented by an RDF graph. Similarity between two entities can be derived not only from similarities between their description components, but also from the similarity between the structures of the RDF graphs representing them. In our system, the entity graph similarity is formulated from the ratio between the number of similar predicates over the maximal number of triples of both entities. Taking into account the underlying meaning of RDF(S)/OWL properties, some properties are considered as similar, e.g. *owl:cardinality, owl:maxCardinality* and *owl:minCardinality*. The formulation does not regard the similarity of objects, but only the similarity of predicates of two entities.

$$Sim_{graph}(s_1,s_2)=\frac{|P_c|}{max(|P(s_1)|,|P(s_2)|)}$$

Total Similarity

The total similarity between two entities is the combination of two similarity values: their component similarity $Sim_{component}$ and their graph structure similarity Sim_{graph}. Here, a fixed weighting scheme is applied for the combination. The reason is that these two similarity components are not optional, they exist for every pair of entities. The weights can be chosen following experimental results.

$$Sim_{total}=w_{component}*Sim_{component}+w_{graph}*Sim_{graph}$$
$$\text{where } w_{component}+w_{graph}=1$$

Implementation and Results

Our entity similarity measure was implemented in Java. Jena API [1] was used for loading OWL DL ontologies into memory. The OWL DL graph representation is derived from the Jena RDF model.

Ontologies used for evaluation are ontologies proposed in the context of the I^{3}CON conference[2] even though we did not compete in 2004. To evaluate our similarity measure, a very simple algorithm is installed: for each entity (class, property) in an ontology, the similarity value with all other entities in the other ontology is calculated using our measure. The maximal similarity value will be shown out. Note that our goal is to test the measure, not the efficiency of the matching algorithm.

Order	Entity in ontology animalA.owl	Entity in ontology animalB.owl	Similarity value
1	Shoesize	shoesize	1.0
2	Shirtsize	Shirtsize	1.0
3	Animal	Animal	1.0
…			
23	hasFemaleParent	hasWife	0.037
24	hasMaleParent	hasHusband	0.034

Table 1: compare ontology animalA.owl with animalB.owl

We calculated similarity values (see table 1) between entities in ontology *animalA.owl* (having 35 entities) and entities in ontology *animalB.owl* (having 24 entities). The latter is a modified version of the former. It has no instance and it has reduced entity descriptions. Note that this simple non-optimized algorithm is not recursive and has no similarity matrix. Its total averaged running time for the test is about 2.204 second. As result, we have 3 entity pairs with similarity value of 1.0. The worst value is 0.034 for the (incorrect) pair *hasMaleParent-hasHusband*. All we know about the entity *hasMaleParent* is information extracted from its description. This informs us that the entity type is *owl:ObjectProperty* and that it is *owl:equivalentProperty* with *hasFather*. As the algorithm

[1] http://jena.sourceforge.net/
[2] http://www.atl.external.lmco.com/projects/ontology/i3con.html

does not store temporary similarity values, it does not know that *hasFather* in *animalA.owl* ontology is exactly similar (similarity value of 1.0) to the entity *hasFather* in *animalB.owl* ontology. That explains why *hasMaleParent* is not matched with *hasParent* but *hasHusband* in the second ontology. But in the future work, we will adapt an incremental algorithm which can overcome these limits.

Related Work

The closest related work with our proposition is the work of (Euzenat and Valtchev, 2004). Similarity between two nodes depends on categories which they belong to and relations between categories. It is a fixed weighted sum of partial similarities while we apply a variable weighting scheme. (Weinstein & Birmingham, 1999) proposes several similarity measures for comparing concepts in differentiated ontologies. Their measures are mainly based on the compatibility comparison of structural descriptions and do not rely on the underlying meaning of relations between concepts. (Maedche & Staab, 2002) presents another work for measuring the global similarity between the whole two ontologies. Contrarily to theirs, our measure is for the similarity between entities in two ontologies.
Other researchers (Doan *et al*., 2002), (Melnik *et al*., 2002), (Noy and Musen, 2000) propose algorithms for finding entity mappings between two schemas (simple ontologies). They do not focus specially on constructing a similarity measure as we do. For lack of room, we don't detail all related work but good surveys can be found in (Rahm and Bernstein, 2001), in (Kalfoglou and Schorlemmer, 2003) and in (Euzenat, 2004).

Conclusions and Future Work

In this paper, we presented our new measure for calculating similarity of two entities (classes, relations) from two different OWL DL ontologies. These ontologies are represented in RDF-like graphs and the similarity measure is formulated based on two parts: (1) similarity between the components of the entity descriptions and (2) similarity between graphs representing entities. For the first part, we proposed methods for dealing with the similarity of two sets and for combining component partial similarities in a variable weighting scheme. Our similarity measure also takes into account the underlying meanings of RDF(S) and OWL properties, which are used in entity descriptions. The measure was implemented in Java and tested with entities in OWL ontologies for the validation. In addition to I^{3}CON[1], we will test our measure on EON[2] and on two real-world ontologies O'COMMA and O'Aprobatiom compared with our previous algorithm (Bach *et al*, 2004).

Many algorithms based on similarity measures to discover mappings can be developed using our measure. A mapping will be created for two entities in two different ontologies when these entities are considered as (semantically) similar, i.e. when their similarity value is higher a certain threshold. For future work, we will focus on the integration of our proposed measure in an ontology matching algorithm or in a merging algorithm and evaluate its performance and efficiency in these real and crucial tasks.

1 http://www.atl.external.lmco.com/projects/ontology/i3con.html
2 http://co4.inrialpes.fr/align/Contest/

References

Bach, T.L., Dieng-Kuntz, R., Gandon, F. 2004: On *Ontology Matching Problems - for Building a Corporate Semantic Web in a Multi-Communities Organization*. In Proc. of ICEIS'04, Porto, Portugal, 2004, p. 236-243.
Cohen, W. W., Ravikumar, P., and Fienberg, S. 2003. *A Comparison of String Distance Metrics for Name-Matching Tasks*. IJCAI'2003 Workshop on Information Integration on the Web.
Deborah, L. M., Fikes, R., Rice, J., and Wilder, S. 2000. *An Environment for Merging and Testing Large Ontologies*. In Proc. of KR'00. Breckenridge, Colorado, USA. April 12-15, 2000.
Dieng, R. and Hug, S. 1998. *Comparison of "Personal Ontologies" Represented through Conceptual Graphs*. In Proc. of ECAI'98, p. 341-345, Brighton, UK.
Doan, A., Madhavan, J., Domingos, P., and Halevy, A. 2002. *Learning to Map between Ontologies on the Semantic Web*. In Proc. of WWW'02, Hawaii, USA.
Euzenat, J., Valtchev, P.: *Similarity-based ontology alignment in OWL-Lite*. ECAI'04, Valencia, pp333-337
Euzenat, J.: *State of the art on current alignment techniques*, 12/2004, IST Knowledge Web Network of Excellence no FP6-507482, #KWEB/2004/ D2.2.3.
Kalfoglou, Y. and Schorlemmer, M. 2003. *Ontology mapping: the state of the art*. The Knowledge Engineering Review Journal, (18(1)):1-31, 2003.
Maedche, A. and Staab, S. 2002: *Measuring Similarity between Ontologies*. In Proc. of EKAW'02. Madrid, ES.
Melnik, S., Garcia-Molina, H., and Rahm, E. 2002. *Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching*. In Proc. 18th ICDE'02, San Jose CA.
Noy, N. F. and Musen, M. A. 2000. *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment*. In Proc. of AAAI'00, Austin, TX.
Rahm, E. and Bernstein, P. A. 2001. *A survey of approaches to automatic schema matching*. In The VLDB Journal: Volume 10 Issue, pages 334-350.
Weinstein, P., and Birmingham, W.: *Comparing concepts in differentiated ontologies*. In Proc. of KAW'99, 1999.