Expliciting Semantic Relations Between Ontologies in Large Ontology Repositories

Carlo Allocca[⋆]

Knowledge Media Institute (KMi), The Open University, Walton Hall,
Milton Keynes MK7 6AA, United Kingdom
`c.allocca@open.ac.uk`

Abstract. Our goal is to develop a framework for detecting and managing semantic relations between ontologies for large ontology repositories. Making explicit implicit relations between ontologies provides meta-information that facilitates the development of Semantic Web Applications. The development of the framework relies on an ontology-based methodology. This approach offers several advantages, among which the possibility to reason upon ontologies and their relations.

1 Motivation

Semantic Web Applications (SWAs) use the Semantic Web (SW) as a large-scale knowledge source [1]; they achieve their tasks by automatically retrieving and exploiting knowledge from the SW as a whole using advanced Semantic Web Search Engines (SWSEs) such as WATSON [1]. On the other hand, because they are built and applied in different contexts, ontologies are not standalone information artifacts: they are linked to one another through semantic relations. This aspect has been totally ignored by current SWSEs, including WATSON. For example, the query "*student*" currently gives 1079 ontologies as a result in WATSON (valid on the 25/03/2009). However, already in the first page, at least 2 of the ontologies (`http://www.vistology.com/ont/tests/student1.owl` and `http://www.vistology.com/ont/tests/student2.owl`) represent, apart from their URIs and the base namespaces, exactly the same logical model, expressed in the same ontology language. Another common situation is when an ontology has been translated in different ontology languages, like it is the case of the first and second results of the query "*student, university, researcher*"(`http://reliant.teknowledge.com/DAML/Mid-level-ontology.owl` and `http://reliant.teknowledge.com/DAML/Mid-level-ontology.daml`). In that case, it is obvious that these two ontologies are in fact two different encodings of the same model. Inspecting the results of WATSON in the same way, it is not hard to find ontologies connected by other, more sophisticated semantic relations: *versioning*, *inclusion*, *similarity*, etc. Leaving implicit these relations between ontologies in SWSE's ontology repositories generates additional difficulties in exploiting their results, leaving to the users and the applications to find what is the right ontology, such as the latest version, to achieve their goal.

[⋆] **This work was funded by the EC IST-FF6-027595 NeOn Project**

2 Related Work

J. Heflin [5], was the first to study formally some of different types of link between ontologies, focusing on the crucial problems of versioning and evolution. Currently, there is no Ontology Management Systems that implements his framework. J. Hartmann [4] has proposed **OMV** (Ontoloy Metadata Vocabulary) as a standard that provide some external information/properties to the ontologies for supporting the reuse of ontologies. Among the expressed OMV properties there are some relations that might be useful to simplify the procedure of automatic discovering. The authors of [8] characterized, at a very abstract level, a number of relations between ontologies such as *sameConceptualization*, *Resemblance*, *Simplification* and *Composition*, without providing concrete elements for detecting them. Several approaches have been focusing on how to compare two different versions of ontologies in order to find out the differences. In particular, **PROMTDIFF** [9] compares the structure of ontologies and **OWLDiff** (`http://semanticweb.org/wiki/OWLDiff`) computes the differences by entailment checking of the two set of axioms. **SemVersion** [10] compares two ontologies and compute the differences at both the structural and the semantic levels. In addition, there exist many measures to compute the similarity of two ontologies [2]. While existing work provides partial answers to the problem of detecting (some of the) relations between ontologies, what is needed is a complete, homogeneous framework for detecting, managing, reasoning with and exploiting the links that implicitly relate ontologies on the Web.

3 Proposed Approach

Our framework is based on an ontology-based methodology. It means to build a semantic structure, called the *Ontology Semantic Relation (OSR) Ontology*, providing an explicit representation of the implicit relations between ontologies. There are several advantages in using this approach: 1) It is flexible enough to add a new relation at any time, without strong changes in the overall system; 2) We can apply reasoning to infer new relations from elementary detection mechanisms; 3) common ontology based infrastructures can be used to store, manipulate and query relations between ontologies.
We have designed an architecture for our framework, as depicted in Figure 1. The OSR-Ontology separates the on-line part of the architecture–providing APIs and Services that rely on a reasoner–from the off-line part–populating the OSR-Ontology. The population of the ontology is based on three component: the *Control Component (CC)*, the *Detecting Component (DC)* and the *Populating Component (PC)*. First, the CC selects from the Ontology Repository ontologies that need to be evaluated to establish potential relations. Second, the selected set of ontologies is processed by the DC, which contains the main mechanisms to discover the possible relations between the ontologies. Finally, the PC populates the semantic structure with the detected relations.

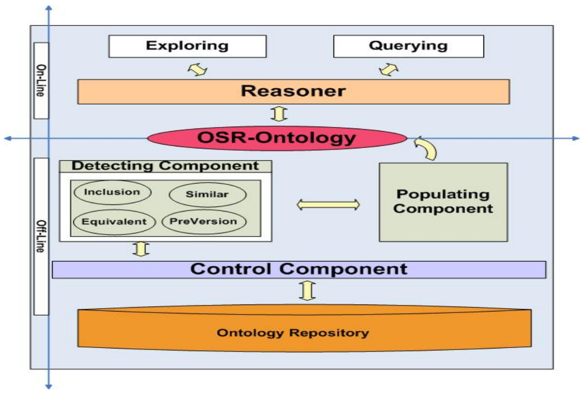

Fig. 1. Architecture of the framework.

4 Result

The first results concerning this work are: 1) The OSR-Ontology; 2) The Detection Component; and 3) A first evaluation.

4.1 OSR-Ontology

A first analysis of the implicit relations between ontologies has led us to distinguish two sets of relations, here called atomic and complex. Both are formalized in the OSR-Ontology structure downloadable at `http://kannel.open.ac.uk/ontology`. The first set contains relations, which do not depend on other relations. The second set, instead, contains relations which are defined using atomic relations or other complex relations. In particular we discuss the following relations:

Semantic Relation: Let OR the collection of ontologies on the Semantic Web. An Semantic Relation is any binary relation defined over OR.

Inclusion and Equivalence: Inclusion, also known as *Extention* in the literature [5], is an example of atomic relation. It links together two ontologies if one is *contained* in the other. We distinguish two kinds of inclusions: syntactic (considering the set of axioms asserted in the ontologies) and semantic (also considering the entailments of the ontologies). The *Equivalence* complex relation can be defined through a rule of the form $Equivalence(O1, O2) :- Inclusion(O_1, O_2), Inclusion(O_2, O_1)$. Detecting equivalence (syntactic and semantic) is particularly important when ranking ontologies: the same ontology should not been ranked at several different places in the result set [3].

Similar: Informally, *similarity* is a measure that expresses how close two ontologies are. [2] describes various ways to compute the similarity between two ontologies, relevant in various application contexts. We included in the OSR-Ontology several notions of similarity, starting from the three: *SyntacticSimilarity*, *SemanticSimilarity* and *VocabularySimilarity*, working at the axiom, entailment and lexical levels respectively.

PrevVersion: Linking an ontology to its previous version is of particular importance [7]. In accordance with the definition in [7] we distinguish: 1) **PrevVersionConceptualChange**: a change in the way a domain is interpreted; 2) **PrevVersionExplicationChange**: a change in the way the conceptualization is specified, without changing the conceptualization itself.

4.2 Detecting Component

Below, we report on some semantic relations for which we have implemented detection mechanisms, and that are evaluated in the next section[1 2]

SyntacticInclusion compares the ontologies at the structural level, checking if the set of axioms in O_1 is included in the set of axioms in O_2.

SemanticInclusion compares the ontologies at the semantic level, checking if the set of axioms in O_1 is entailed by the set of axioms in O_2.

SyntacticSimilar: The method is based on the metric (1), where T is a given parameter.

$$\frac{|setAxioms(O_1) \bigcap setAxioms(O_2)|}{max(|setAxioms(O_1)|, |setAxioms(O_2)|)} \geq T \qquad \textbf{(1)}$$

VocabularySimilar:The method is based on the metric (2).

$$\frac{|Vocabulary(O_1) \bigcap Vocabulary(O_2)|}{max(|Vocabulary(O_1)|, |Vocabulary(O_2)|)} \geq T \qquad \textbf{(2)}$$

SemanticSimilar: We note LC(O_1, O_2) the set of axioms of O_1 that are logical consequences of O_2. The method is based on the metric (3).

$$\frac{|LC(O_1, O_2) \bigcap LC(O_2, O_1)|}{max(|setAxioms(O_1)|, |setAxioms(O_2)|)} \geq T \qquad \textbf{(3)}$$

4.3 Initial Evaluation

To provide an initial evaluation of the current implementation of the detection mechanisms, we collected 20 pairs of ontologies from WATSON's repository and we applied the above detection methods on them. In Table 1 the results for each evaluated method are showed.

Method	Nb. Relations	Average Time (ms)	Manual eval.
SyntacticInclusion	8	2	8
SemanticInclusion	9	80	9
SyntacticEquivalent	4	5	4
SemanticEquivalent	6	180	6
SyntacticSimilar	15	3	15
VocabularySimilar	17	1	17

Table 1. Evaluation results for the detection mechanisms.

The second column contains the number of pairs of ontologies related by the corresponding relation according to the detection mechanisms (similarity is computed with a threshold T=0.5). The last column represents the result of a manual evaluation of the relations between the pairs of ontologies. Although, we have considered only a few pairs of ontologies, containing on average 1000 axioms, the analysis of the result in

[1] Note that when comparing two ontologies, to avoid computing all consequences (which is not always possible [6]) our methods only check if the set of axioms of the first ontology is entailed by the second ontology and vice-versa.

[2] Note that these comparisons ignore the URIs and base namespace of the ontologies, as they do not have any influence when establishing the considered ontology relations.

Table 1 shows that there is no discrepancy between the results produced from the automatic method and the manual evaluation. Moreover, this result validates some of our hypothesis. In particular, having two methods for detecting syntactic and semantic relations appears relevant as not all the semantic relations have been discovered through the syntactic techniques. For example, it means that two pairs of ontologies where semantically equivalent, while syntactically different. Furthermore, we can observe an important difference, in terms of time, between syntactic and semantic mechanisms. The formers are very fast and can be easily applied over large ontology repositories. The latter are a lot more complex, and will require more efficient techniques to scale to repositories like the one of WATSON.

5 Conclusion and Future Work

An first analysis of implicit relations between ontologies is presented. Given their importance for supporting the development of the SWAs, our goal is to build a framework for detecting and managing relations between ontologies. To do this, an ontology based approach is taken and first parts of our framework, focusing on the OSR-Ontology and on the detection mechanisms, is discussed. There are several directions for future work. First, we need to finalize the framework to make it completely automatic, robust and integrated with WATSON. We also plan to produce a first release of the OSR-Ontology as well as a more complete evaluation of the detection methods. The versioning problem will require some deeper studies to provide a model and a method to keep track of versions of the ontologies. Other kinds of ontology relations should also be considered such as incompatibility. Finally, an important aspect is to formalize the model underlying the linked ontologies in order to provide reasoning and querying mechanisms, as well as visualizations for networks of ontologies.

References

1. M. d'Aquin, E. Motta, and M. Sabou et al. Towards a new generation of semantic web applications. *IEEE Intell. Sys.*, 23(3), 2008.
2. J. David and J. Euzenat. Comparison between ontology distances (preliminary results). *7th Int. Semantic Web Conference, ISWC*, 2008.
3. N. Shadbolt H. Alani, C. Brewster. Ranking ontologies with aktiverank. *Proc of the 4th Int Sem Web Conf (ISWC2006), 2006, LNCS, Springer*, pages 1–15., 2006.
4. J. Hartmann, R. Palma, and et al. Ontology metadata vocabulary and applications. pages 906–915, OCT 2005.
5. J. Heflin and Z. Pan. A model theoretic semantics for ontology versioning. *3th Intern Sem Web Conf, Hiroshima, Japan, LNCS 3298 Springer*, pages 62–76., 2004.
6. Z. Huang and H. Stuckenschmidt. Reasoning with multi-version ontologies: a temporal logic approach. *Proc of the Fourth Intern Sem Web Conf (ISWC2005),3729, LNCS, Springer*, pages 62–76., 2005.
7. M. Klein, D. Fensel, A. Kiryakov, and D. Ognyanov. Ontology versioning and change detection on the web. *13th Intern Conf on Know. Engineering and Know. Management (EKAW02)*, pages 197–212, 2002.
8. A. Kleshchev and I. Artemjeva. An analysis of some relations among domain ontologies. *Int. Journal on Inf. Theories and Appl*, 12:85–93, 2005.
9. N. F. Noy and M. A. Musen. Promptdiff: A fixed-point algorithm for comparing ontology versions. *18th National Conf. on Artificial Intelligence (AAAI)*, 2002.
10. M. Volkel. *D2.3.3.v2 SemVersion Versioning RDF and Ontologies. EU-IST Network of Excellence (NoE) IST-2004-507482 KWEB.*