Finding equivalent ontologies in Watson

Conference or Workshop Item

Finding Equivalent Ontologies in Watson*

Carlo Allocca
Knowledge Media Institute,
Open University
Walton Hall, Milton Keynes
MK7 6AA
United Kingdom
c.allocca@open.ac.uk

Mathieu d'Aquin
Knowledge Media Institute,
Open University
Walton Hall, Milton Keynes
MK7 6AA
United Kingdom
m.daquin@open.ac.uk

Enrico Motta
Knowledge Media Institute,
Open University
Walton Hall, Milton Keynes
MK7 6AA
United Kingdom
e.motta@open.ac.uk

ABSTRACT

We present an efficient mechanism for finding equivalent ontologies motivated by the development of the Semantic Web search engine Watson. In principle, it computes a canonical form for the ontologies, which can then be compared syntactically to assess semantic equivalence. The advantage of using this method is that the canonical form can be indexed by the search engine, reducing the search for equivalent ontologies to a usual text search operation using the canonical form. This method is therefore more suitable for a search engine like Watson than the naive comparison of all possible candidate pairs of ontologies using a reasoner.

1. INTRODUCTION

In recent years, the Semantic Web (SW) community has been contributing to the authoring of a myriad of ontologies. A large number of them are made available on-line by specialized search engines, such as Watson. Watson[1] is a gateway for the SW that plays three main roles: 1) collecting the available semantic content on the Web, 2) analyzing it in order to extract useful metadata and indexes, and 3) providing efficient query facilities to access the data, both for human users and applications. A missing aspect in Watson concerns the detection and management of implicit semantic relations between ontologies, such as *semantic equivalence*.

Informally, we consider that two ontologies describing the same vocabulary are semantically equivalent if they express the same meaning, even if they may be written differently from a syntactic point of view. If we consider for example the two ontologies O_1 and O_2 described by the sets of axioms:

$$O_1\text{:}\quad C \sqsubseteq B, B \sqsubseteq A, C \sqsubseteq A, B \sqsubseteq \neg\exists R.D_1$$
$$O_2\text{:}\quad C \sqsubseteq B, B \sqsubseteq A, B \sqsubseteq \forall R.\neg D_1$$

a simple syntactic comparison would come up with the conclusion that these are different, while they are logically the same: they are semantically equivalent.

Although Watson has already indexed hundreds of thousands of ontologies, there is not yet an automatic mechanism to find/detect such equivalences. For example, the query "*student*" currently gives 1079 ontologies as a result[2]. However, already in the first page of results, at least 2 of the ontologies[3] can be shown to represent exactly the same logical model, even if they are written differently. 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*"[4]. In that case, it is obvious that these two ontologies are in fact two different encoding of the same model.

Finding equivalent ontologies is an important problem to tackle, as its solution would benefit both human users and applications. Indeed, simply being able to cluster the results of Watson according to equivalence would reduce the number of elements to be inspected by the user, and would provide alternative syntactic representations to chose from for each set of equivalent ontologies. For applications, obtaining non-redundant results is a good way to increase efficiency and being able to locate equivalent ontologies can improve robustness. For example, in case the currently used ontology is not available anymore, Watson could easily provide alternatives.

2. SEMANTIC EQUIVALENCE

Motivated by the concrete scenario of Watson, we want to add to it a service to find equivalent ontologies. The main issues we have been investigating are: 1) *How to keep track in Watson's repository of the ontologies which are semantically equivalent?* and 2) *How Watson recognizes that a newly added ontologies is semantically equivalent to others already indexed?* For both these issues, it is necessary to specify, formally, when two or more ontologies are semantically equivalent. Using the notions of axiom and entailment ($\models$) as considered in description logics [1], we define the equivalence relation between two ontologies as the following:

DEFINITION 1 (***Ontology Equivalence***). *Let O_1 and O_2 be two ontologies. We say that O_1 is semantically equivalent to O_2 ($O_1 \equiv O_2$) if and only if for every axiom α, $O_1 \models \alpha$ implies $O_2 \models \alpha$ and $O_2 \models \alpha$ implies $O_1 \models \alpha$ ($O_2 \models \alpha \Leftrightarrow O_1 \models \alpha$)*

A possible way to find out if two ontologies are equivalent according to this definition is to use a reasoner to check if

*This work was funded by the EC IST-FF6-027595 NeOn Project.

[1] `http://watson.kmi.open.ac.uk/WatsonWUI/`

[2] valid on the 24/07/2008

[3] `http://www.vistology.com/ont/tests/student1.owl` and `http://www.vistology.com/ont/tests/student2.owl`

[4] `http://reliant.teknowledge.com/DAML/Mid-level-ontology.owl` and `http://reliant.teknowledge.com/DAML/Mid-level-ontology.daml`

$\forall \alpha \in O2, O1 \models \alpha$ and $\forall \alpha \in O1, O2 \models \alpha$. However, this is a very complex procedure, which, if applied within Watson, would result in checking every candidate pair of ontologies through a reasoning mechanism. In addition, every time an ontology would be added to the repository, many candidate equivalent ontologies would have to be identified and tested, which seems technically unfeasible in a large (several hundred thousand ontologies), dynamic and distributed repository like the one of Watson.

3. METHOD

In order to avoid the situation described above, we employ a *Knowledge Compilation* technique [2]. The principle is to transform the ontologies into canonical forms, such that if the ontologies are semantically equivalent, the corresponding canonical forms have to be syntactically equal. Specifically, we employ the *Canonical Prenex Conjunctive Normal Form* (CPCNF) of a first order logic (FOL) formula corresponding to the ontology. Several steps are required to compute this particular canonical form. Each of these steps preserves the logical equivalence of formulas.

3.1 Transformations

The first step is about transforming the ontology O into an equivalent FOL formula F. For example, with O_1 an ontology containing the axioms $C \sqsubseteq B$, $B \sqsubseteq A$, and $C \sqsubseteq A$ and O_2 another ontology containing $C \sqsubseteq B$ and $B \sqsubseteq A$, we obtain the following FOL formulas:

$$F_1 = \forall x_1 \quad [C(x_1) \rightarrow B(x_1)] \wedge \forall x_1[B(x_1) \rightarrow A(x_1)] \wedge \forall x_1[C(x_1) \rightarrow A(x_1)]$$

$$F_2 = \forall x_1 \quad [C(x_1) \rightarrow B(x_1)] \wedge \forall x_1[B(x_1) \rightarrow A(x_1)]$$

The second step is about transforming the formula F into its Prenex Conjunctive Normal Form (PCNF), obtaining PCNF(F). Let L be a FOL language[5]:

DEFINITION 2 (***PCNF***). *A formula $F \in L$ is in a prenex conjuntive normal form if it is of the form: $F = Q_1 v_1 Q_2 v_2 ... Q_n v_n.M$ where each Q_i is a quantifier, i.e. $Q_i \in \{\exists, \forall\}$, $v_1 ... v_1$ are variables and M (called matrix) is a quantifier free formula in Conjunctive Normal Form.*

Continuing the example above, the PCNFs of the previous F_1 and F_2 formulas are:

$$PCNF(F_1) = \forall x_1 \quad [\neg C(x_1) \vee B(x_1)] \wedge [\neg B(x_1) \vee A(x_1)] \wedge [\neg C(x_1) \vee A(x_1)]$$

$$PCNF(F_2) = \forall x_1 \quad [\neg C(x_1) \vee B(x_1)] \wedge [\neg B(x_1) \vee A(x_1)]$$

An essential property of PCNF is that if F_1 and F_2 are FOL formulas then if PCNF(F_1) = PCNF(F_2) then $F_1 \equiv F_2$. The transformation of a formula into PCNF eliminates some syntactic differences. For example, formulas $\forall x \forall y \neg(P(x) \wedge Q(y))$ and $\forall x \forall y(\neg P(x) \vee \neg Q(y))$ have the same PCNF. However, it is not necessarily true that if $F_1 \equiv F_2$ then PCNF(F_1) = PCNF(F_2). To obtain such a property we need to introduce a canonical form.

DEFINITION 3 (***CPCNF***). *A PCNF formula F is in CPCNF if all the clauses of the matrix are maximized following the rule: $G_i \longmapsto G_i \vee (P \wedge \neg P) \longmapsto (G_i \vee P) \wedge (G_i \vee \neg P)$, for each P belonging to the set of positive literals in F and missing in the clause G_i.*

Applying this final step on the previously obtained formulas gives:

$$CPCNF(F_1) = \forall x_1[A(x_1) \vee B(x_1) \vee \neg C(x_1)] \wedge [\neg A(x_1) \vee B(x_1) \vee \neg C(x_1)] \wedge [A(x_1) \vee \neg B(x_1) \vee C(x_1)] \wedge [A(x_1) \vee \neg B(x_1) \vee \neg C(x_1)] \wedge [A(x_1) \vee B(x_1) \vee \neg C(x_1)] \wedge [A(x_1) \vee \neg B(x_1) \vee \neg C(x_1)]$$

$$CPCNF(F_2) = \forall x_1[A(x_1) \vee B(x_1) \vee \neg C(x_1)] \wedge [\neg A(x_1) \vee B(x_1) \vee \neg C(x_1)] \wedge [A(x_1) \vee \neg B(x_1) \vee C(x_1)] \wedge [A(x_1) \vee \neg B(x_1) \vee \neg C(x_1)]$$

By eliminating duplicated clauses in CPCNF(F_1), we can see that we obtain twice exactly the same formula. We can therefore conclude that the original ontologies were semantically equivalent.

4. APPLICATION IN WATSON

The main advantage of this method is that the computed canonical form of an ontology can simply be indexed as text within the Watson search engine, hence acting as a sort of *signature* for this ontology. In this case, the task of finding ontologies that are equivalent to a given ontology O is reduced to a search using the canonical form of O as a query. This method not only benefits from existing, very efficient search mechanisms, but also simplifies the offline treatment of equivalence in Watson. Indeed, whenever a new ontology is added to the repository, it is only needed to compute and index its canonical form so that it can be searched for equivalence. Hence, the process of adding this new ontology in the collection is independent from any other ontology already indexed, as it does not require any direct comparison.

5. CONCLUSIONS AND FUTURE WORK

In this document we have presented, briefly, a mechanism for finding equivalent ontologies in a large-scale ontology repository, such as the one of Watson. For that, we described a method based on a Knowledge Compilation technique that transform the ontologies in a canonical form: CPCNF. The correctness and completeness of our method have been proved for OWL-DL with some restrictions on the axioms that can be employed in the ontology[6]. A prototype working for OWL-DL ontologies has been implemented and tested. In accordance with initial theoretical and experimental results, we anticipate that a practical evaluation of the tool will show that it is effective for a large majority of the ontologies made available on the Semantic Web through Watson.

6. REFERENCES

[1] F. Baader. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[2] A. Darwiche and P. Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[3] R. M. Smullyan. *First-Order Logic.* Dover Publications, 1995.

[5] In the definitions of this section, we use notions from FOL such as formula, literal and Conjunctive Normal Form as defined in [3].

[6] i.e. axioms that leads to CPCNF formulas containing only universal quantifiers