Semantic Web: Lab 3

OWL Ontology Reasoning

This lab is to use OWL API and optimized DL reasoners to reason over ontologies. OWL API is a high level Application Programming Interface (API) for working with OWL ontologies. It is closely aligned with the OWL 2 structural specification. It supports parsing and rendering in the syntaxes defined in the W3C specification (Functional Syntax, RDF/XML, OWL/XML and the Manchester OWL Syntax); manipulation of ontological structures; and the use of reasoning engines. The reference implementation of the OWL API, written in Java, includes validators for the various OWL 2 profiles - OWL 2 QL, OWL 2 EL and OWL 2 RL. The OWL API has widespread usage in a variety of tools and applications. See the following paper for details of its design (downloadable from `www.lri.fr/~ma/M2Dk`):

```
Horridge, M., and Bechhofer, S. (2011). The OWL API: A java API for OWL ontologies.
Semantic Web, 2(1), 11-21.
```

In OWL API, the *OWLReasoner* interface provides access to functionality relating to the process of reasoning with OWL ontologies, supporting tasks such as consistency checking, computation of class or property hierarchies and axiom entailment. A number of existing implementations including the reasoners: CEL , FaCT++ , HermiT , Pellet , ELK, Racer Pro. In this lab, you will practice to use the OWL API together with the HermiT reasoner (optionally with ELK reasoner). HermiT can handle the full language of OWL 2 standard, whilst the ELK reasoner is an implementation of highly optimized reasoner for OWL 2 EL .

You are asked first to get familiar the usage of OWL API to manipulate OWL ontologies, following the next steps

- Download the HermiT reasoner from `http://hermit-reasoner.com`
- Import the HermiT java project in Eclipse (or other java IDE that you prefer).
- Study the example code using OWL API and a reasoner over an ontology, starting with EntailmentChecking.java for example.

Now you are ready to complete the following exercises.

Excise 1 (OWL API). Please answer the following questions:

- How can you load an existing ontology into Java by OWL API?

- How can you get the numbers of concepts, relations (roles), instances, and logical axioms contained in an ontology? Note that a logical axiom is either a TBox or an ABox axiom that has description logic based semantics; In OWL 2 specification, ontologies and axioms can have annotations (see Fig 1 in `http://www.w3.org/TR/owl2-syntax`) that has no logical semantics.

- How can you save an ontology to a file in a particular syntax?

Excise 2 (Ontology Creation). Please create your own ontology from Lab 2 by OWL API. For this, you can follow the steps below:

- Create an empty ontology.

- Create atomic concepts, roles, individuals, and complex concepts.

- Create different logical axioms (concept subsumptions, concept instance, etc.)

- Add the logical axioms into the original ontology and then save it to a file.

Excise 3. (Ontology Reasoning by HermiT)

- Is the ontology created in Excise 2 consistent? Use HermiT to check it.

- Can you define some reasoning tasks over your ontology and ask HermiT to achieve them? For example, is a certain concept A subsumed by another concept B (i.e. is $O \models A \sqsubseteq B$) or by a more complex concept such as $\exists r.C$ (resp. $O \models A \sqsubseteq \exists r.C$)?

- What does the classification mean as a reasoning task? Classify your ontology and save it. What are and how many new axioms you obtain after the classification by HermiT?

Excise 4. (Testing Additional Functionality of OWL API) As specified in the OWL API paper mentioned above, its current version is enhanced with functionality supporting tasks such as profile validation, explanation and modularity. So please test such functionalists over your ontology or another ontology such as bigontology.owl downloadable from `www.lri.fr/~ma/M2Dk`.

Excise 5. (Optional: Ontology Reasoning by ELK)

- Download the ELK reasoner from `https://github.com/liveontologies/elk-reasoner`

- Get working with the ELK reasoner with your ontology. Can you observe the differences between ELK and HermiT reasoners?

- Compare these two reasoners over a large bio-medical OWL 2 EL ontology (the .owl file) downloadable from `www.lri.fr/~ma/M2Dk`. Does ELK perform faster than HermiT on this ontology? Why?