Foundations of Semantic Web Technologies

Tutorial 2

Sebastian Rudolph

SS 2013

Exercise 2.1. Recap the notions "theory", "logical consequence" and "equivalence" and decide if the following claims are true or false for FOL. Give an informal justification for your answer.
For arbitrary theories $\mathcal{T}$ and $\mathcal{S}$ holds:

(a) If a formula (axiom) F is generally valid, then $\mathcal{T} \models F$, i.e., every theory has at least all tautologies as consequence.

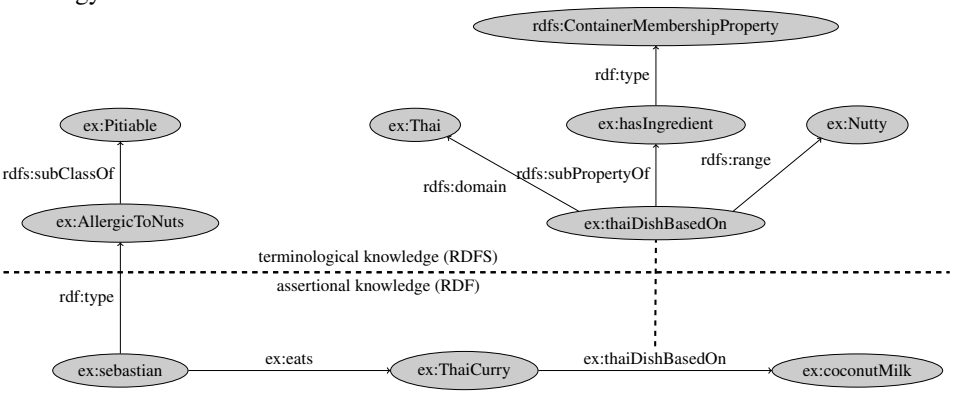
(b) The more axioms a theory contains the more models it has. More precisely: if $\mathcal{T} \subseteq \mathcal{S}$, then every model of $\mathcal{T}$ is a model of $\mathcal{S}$.

(c) The more axioms a theory contains, the more logical consequences it has. More precisely, if $\mathcal{T} \subseteq \mathcal{S}$, then every logical consequence from $\mathcal{T}$ is also a consequence from $\mathcal{S}$.

(d) If $\neg F \in \mathcal{T}$, then $\mathcal{T} \models F$ can never hold(F being an arbitrary formula).

(e) If two theories differ syntactically ($\mathcal{T} \neq \mathcal{S}$), then they differ in at least one logical consequence (e.g., through the existence of a formula F with $\mathcal{T} \models F$ but $\mathcal{S} \not\models F$).

Exercise 2.2. Describe an RDFS interpretation that is a model of the example ontology from Exercise 1.4. For reference, find here the RDFS graph representation of this ontology:



Exercise 2.3. For the ontology from Exercise 2.3, find

- a triple that is simply entailed,
- a triple that is RDF-entailed but not simply entailed,
- a triple that is RDFS-entailed but not RDF-entailed.

Exercise 2.4. As you know, the unique name assumption does not hold in RDF(S), i.e. in a model, several URIs might be assigned to the same resource. Contemplate whether (and if so, how) it is possible to specify in RDFS that two given URIs refer to the same resource.

Exercise 2.5. The empty graph does not contain any triples (i.e. it corresponds to the empty set). Give derivations showing that the empty graph RDFS-entails the following triples:

(a) `rdfs:Resource rdf:type rdfs:Class .`

(b) `rdfs:Class rdf:type rdfs:Class .`

(c) `rdfs:Literal rdf:type rdfs:Class .`

(d) `rdf:XMLLiteral rdf:type rdfs:Class .`

(e) `rdfs:Datatype rdf:type rdfs:Class .`

(f) `rdf:Seq rdf:type rdfs:Class .`

(g) `rdf:Bag rdf:type rdfs:Class .`

(h) `rdf:Alt rdf:type rdfs:Class .`

(i) `rdfs:Container rdf:type rdfs:Class .`

(j) `rdf:List rdf:type rdfs:Class .`

(k) `rdfs:ContainerMembershipProperty rdf:type rdfs:Class .`

(l) `rdf:Property rdf:type rdfs:Class .`

(m) `rdf:Statement rdf:type rdfs:Class .`

(n) `rdfs:domain rdf:type rdf:Property .`

(o) `rdfs:range rdf:type rdf:Property .`

(p) `rdfs:subPropertyOf rdf:type rdf:Property .`

(q) `rdfs:subClassOf rdf:type rdf:Property .`

(r) `rdfs:member rdf:type rdf:Property .`

(s) `rdfs:seeAlso rdf:type rdf:Property .`

(t) `rdfs:isDefinedBy rdf:type rdf:Property .`

(u) `rdfs:comment rdf:type rdf:Property .`

(v) `rdfs:label rdf:type rdf:Property .`

Exercise 2.6. Let the instance I be given as follows (find a visualization in Fig. 1):

$$u(00,01), r(01,11), u(11,12), r(12,22), u(22,23), r(23,33), r(11,31), u(31,33)$$

and the Datalog program

$$d(x,z) \leftarrow u(x,y) \wedge r(y,z)$$
$$d(x,z) \leftarrow r(x,y) \wedge u(y,z)$$
$$d(x,z) \leftarrow d(x,y) \wedge d(y,z)$$

Provide a naïve algorithm for computing all instances of the relation (the predicate) d, based on the fixpoint semantics. For each iteration step, note down which tuple belong to the (intermediate) relation.

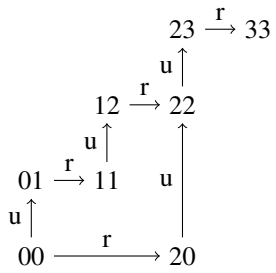


Figure 1: Visualization of the instance from Exercise 2.6

Exercise 2.7. Note that the theorem linking the RDFS entailment with the presented deduction calculus just guarantees soundness of the latter. When the calculus was provided in the RDF semantics specification, it was also considered complete, but a bit later that turned out not to be the case.

As an example of the calculus' incompleteness, consider the following set of triples:

```
ex:isHappilyMarriedTo  rdfs:subPropertyOf     _:bnode .
_:bnode                rdfs:domain            ex:Person .
ex:markus              ex:isHappilyMarriedTo  ex:anja .
```

It is not hard to show that the triple

```
ex:markus rdf:type ex:Person .
```

is a semantic consequence of the above. However, it cannot be derived by means of the given deduction calculus. Make a suggestion how the calculus could be "repaired".