OPPA European Social Fund
Prague & EU: We invest in your future.

Assignment 1

1 Rules of the Game

- You work on this assignment alone, no groups of students are allowed.
- Your solution to the assignment will be evaluated with points ranging from 0 to 15.
- You have to upload your solution to this assignment by 14.10.2012. After this date, **you lose 3 points for each started week of delay**. In exceptional and justified cases (e.g. long-term disease) we decide how to proceed on individual basis. In that case write me an email at petr.kremen@fel.cvut.cz.
- The solution of the assignment is uploaded through the **web application** `http://cw.felk.cvut.cz/upload`. Please, upload the ZIP archive containing:
  - one file `.pdf` – answers to the questions in the Section 2.1
  - one or more file(s) `.owl` – final ontology developed by you in Section 2.2.
  - more file(s) `.rq` – SPARQL queries developed by you in Section 2.3.

2 Assignment

Explore the ontologies at the following URLs:

- `http://krizik.felk.cvut.cz/ontologies/2011/general-family.owl`.
- `http://krizik.felk.cvut.cz/ontologies/2011/father-without-children.owl`.

2.1 Analysis of an Existing Ontology

If not stated otherwise, please use description logic notation.

1. What is the problem with the definition of the class $SomeOneWithBrotherAndSister$? Why this class is not unsatisfiable ? Correct all modeling problems related to this issue.

2. From the semantic point of view the definition of the class $Parent$ is redundant. Which axioms (their parts) can be safely removed from the ontology, without affecting its semantics (i.e. preserving the set of logical consequences) ?

3. Ontology `father-without-children.owl` is of expressiveness $\mathcal{ALC}$. Making use of the tableau algoritgm and some of the error-explanation algorithm according to your choice (Reiter algorithm, CS-tree algorithm) find all minimal unsatisfiability preserving sets for the unsatisfiable class $FatherWithoutChildren$. Describe in detail and visualize the run of both algorithms. Check the correctness of your results using the OWL reasoner Pellet.

4. Why is the ontology `father-without-children.owl` consistent, although it contains unsatisfiable class $FatherWithoutChildren$? How to change (add/remove axioms) the ontology in order to ensure its inconsistency.

5. Explain, why $JIRI$ is an (inferred) instance of the class $ParentOfAtLeastOneChild$, although there is no axiom of the form $hasChild(JIRI, \bullet)$?

6. Explain, why $PETR$ is not an (inferred) instance of the class $ParentOfAtLeastTwoChildren$, although it occurs in **two** axioms of the form $hasChild(PETR, \bullet)$, i.e. $hasChild(PETR, OLGA)$ and $hasChild(PETR, JIRI)$. Find at least two ways how to adjust the ontology so that $PETR$ becomes an instance of $ParentOfAtLeastTwoChildren$.

2.2 Synthesis of Own Ontology – Genealogical Tree of a Well-Known (e.g. Aristocratic) Family

Implement tasks in this part as a new OWL ontology that imports (`owl:imports`) the ontology `http://krizik.felk.cvut.cz/ontologies/2011/general-family.owl`. The resulting ontology must be consistent.

1. Specify characteristics (reflexivity, asymmetry, etc.) and define inverses of the object properties $hasChild$ and $hasSibling$.

2. Formalize the object properties $hasDescendant$ and $hasAncestor$ that will be used for inferring descendants/ancestors into arbitrary depth. E.g. it will be possible to infer $hasAncestor(JIRI, MIRKO)$.

3. Define the class of "all parents, that have at least 5 children, but at most 1 daughter that has exactly two sons.".

4. Finalize the ontology for a genealogical information system – add and axiomatize at least 10 more classes and 5 more properties (both object and data properties are required). In particular pay attention to:
   a) marriage – relationships of being spouse, etc.
   b) complex family relationships – relationship of being an uncle of someone, brother-in-law, stepson, etc.

c) genealogical data – date, place of birth, etc.

Define classes and relationships in such a way that you can easily use them for query formulation in section 2.3.

5. Develop a genealogical tree (at least 3 generations) of a known historical family (see e.g. `http://www.burkespeerage.com/articles/scotland/page31d.aspx`) and check adequacy of the ontology you developed in the previous point.

2.3 Querying the Ontology

For each query you developed in this part (i) write its SPARQL form into a separate .rq file, (ii) test on the developed ontology using the Pellet inference engine of version 2.3 (http://pellet.owldl.com), (iii) write its results into a comment (#) of the .rq file. Next identify queries that can be answered by using the DL query tab in Protégé, and those for which full conjunctive query engine is necessary.

1. Create a query that finds all pairs of persons being in brother-in-law/sister-in-low relationship, and, at the same time, each having at least one sibling.

2. Create a query that finds all pairs of stepsiblings. Explain, where to use an undistinguished variable and where to use a distinguished variable.

3. Create a query that finds out whether there exists (or can be inferred) at least one person, at least one son of which has a daughter. We are interested just in the existence, not in their identity.

4. Show, how the previous query could be evaluated only by means of the standard tableau algorithm for consistency checking, i.e. if there is not inference engine for conjunctive queries available.

OPPA European Social Fund
Prague & EU: We invest in your future.