Assignment 1

1 Rules of the Game

- You work on this assignment alone, no groups of students are allowed.
- Your solution will be evaluated with points ranging from 0 to 15.
- You have to upload your solution to this assignment by 26.10.2013. 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 has to be 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

2.1 Analysis

2.1.1 General Questions

Consider a description logic theory $\mathcal{K} = \mathcal{T} \cup \mathcal{A}$ with a TBox containing three axioms:

$$\begin{aligned} \mathcal{T} \quad = \quad & \{Man \sqcup Woman \sqsubseteq Person, \\ & Man \sqsubseteq \neg Woman \sqcap \exists hasFather \cdot Man\}, \end{aligned}$$

and $\mathcal{A} = \emptyset$. For each question, you can use Protégé to verify your findings.

1. Construct any model $\mathcal{I} = (\Delta^{\mathcal{I}}, \bullet^{\mathcal{I}})$ of $\mathcal{K}$ for which $|\Delta^{\mathcal{I}}| < 4$, $|Woman^{\mathcal{I}}| > 1$ and $|Man^{\mathcal{I}}| > 0$. If no such model exists, explain why.
2. Reduce the subsumption problem $\mathcal{K} \models (\exists hasFather \cdot \top \sqsubseteq Man)$ into the consistency checking problem, i.e. construct a description logic theory, inconsistency of which indicates that the subsumption holds.

3. Using a tableau algorithm, check, whether the concept $Man \sqcap \neg Woman$ is satisfiable with respect to $\mathcal{K}$. Describe and depict the algorithm run in detail, including algorithm states, state transitions, and inference rule applications.

4. Both the "tree model property" (at least one model of the concept is tree-shaped) and the "finite model property" (at least one model of the concept is finite) allows tableau algorithm implementations to use efficient optimizations and speed-up the reasoning process. Unfortunately, these two properties do not hold for all description logics.

   For each of the three theories:

$$\begin{aligned} &\mathcal{K} \\ &\mathcal{K}_1 \quad = \mathcal{K} \cup \{Woman \sqsubseteq \left(\leq 1\, hasFather^{-}\right)\} \\ &\mathcal{K}_2 \quad = \mathcal{K}_1 \cup \{hasFather \sqsubseteq hasParent\} \end{aligned}$$

   decide, whether it employs the *tree model property* and whether it employs the *finite model property*. You can use the Description Logic Complexity Navigator at `http://www.cs.man.ac.uk/~ezolin/dl` to answer this question.

2.1.2 Analysis of Existing Ontology

We will abuse the terminology and use description logic terms and OWL terms interchangably (see `https://cw.felk.cvut.cz/wiki/_media/courses/ae4m33rzn/protege-crash-course.pdf` for more details). To keep the description compact, please use the description logic notation when solving the following tasks.

Explore the OWL document at the URL `http://onto.fel.cvut.cz/ontologies/2014/geneaology.owl`.

1. The class *StrangeSister* is unsatisfiable. Which **minimal** sets of axioms should be removed to make it satisfiable ? You can use Protégé explanation feature when solving this task.

2. The OWL document contains one SWRL rule. Describe what is the purpose of such rule and which inferences would be lost in case we delete the rule.

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

When implementing tasks in this part, use the ontology `http://onto.fel.cvut.cz/ontologies/2014/geneaology.owl`, as a starting point (do not use `owl:imports`). Rename the ontology to the $geneaology- < FEL-username > .owl$. Using Protégé refactoring options, move all axioms, except those causing unsatisfiability of *StrangeSister* class, from the imported `geneaology.owl` ontology into your ontology.

1. Specify suitablecharacteristics (reflexivity, asymmetry, etc.) of the object properties *has_parent* and *has_sibling*.

2. Formalize the object properties *has_descendant* and *has_ancestor* that will be used for inferring descendants/ancestors into arbitrary depth. E.g. it will be possible to infer $has\_ancestor(cyril, alice)$.

3. Define the class of "all fathers, that have at least three daugthers, one of which must be a mother of at least two boys.".

4. Finalize the ontology for a genealogical information system – using OWL+SWRL axiomatize at least 5 more classes and 5 more properties (both object and data properties are required), including

   a) family relationships – relationships of being spouse, being an uncle of someone, being brother-in-law, etc.

   b) genealogical data – date of birth/death, etc.

   Define classes and relationships in such a way that you can reuse them for queries in section 2.3.

5. Develop a genealogical tree (min 3 generations with min 10 individuals in total) of a known historical family (see e.g. `http://www.historickaslechta.cz/slechticke-rodokmeny`) and check adequacy of the ontology you developed in the previous point. The resulting OWL document must be consistent.

2.3 Querying the Ontology

For each query you develop in this part (i) encode it in the SPARQL syntax and put it 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.

1. Create a query that finds all parents having at least one son and one daugther that have a common descendant.

2. Create a conjunctive query that finds out whether there necessarily exists some person having at least one grandson. We are interested just in the existence, not in its identity.

3. Show, how the previous query could be evaluated only by means of the standard tableau algorithm for consistency checking, if there is not inference engine for conjunctive queries available (e.g. in the DL Query tab in Protégé).