Principles of Knowledge Representation and Reasoning

B. Nebel, S. Wölfl, J. Hué
M. Westphal
Winter Semester 2012/2013

University of Freiburg
Department of Computer Science

Exercise Sheet 11

Due: January 23rd, 2013

Exercise 11.1 (Disjunctive Concepts in Inheritance Networks, 3)

In this exercise we consider strict inheritance networks that allow for expressing disjunctive concepts of the form C_1 **or** $\ldots$ **or** C_k. A formula of the form

$$\phi = C_1 \textbf{ or } \ldots \textbf{ or } C_k \textbf{ isa } C'_1 \textbf{ or } \ldots \textbf{ or } C'_l$$

is called a *disjunctive inheritance formula* and its intended interpretation is given by the first-order translation $\pi(\phi) = \forall x\big((C_1(x) \vee \cdots \vee C_k(x)) \rightarrow (C'_1(x) \vee \cdots \vee C'_l(x))\big)$, where the C_i and C'_i are atomic concept terms or unary relation symbols, respectively.

Show that the inheritance problem for disjunctive inheritance networks (given a set of disjunctive inheritance formulae Θ and a disjunctive inheritance formula ϕ, does $\pi(\phi)$ follow from $\{\pi(\vartheta) \mid \vartheta \in \Theta\}$?) can be solved in polynomial time.

Hint: Instead of the translation into first-order logic provide a translation of disjunctive inheritance formulae into propositional logic. You may use the deduction theorem for propositional logic and the fact that the validity of a given set of propositional Horn-formulae can be decided in polynomial time.

Exercise 11.2 (Strict Inheritance with Negation and Conjunction, 2)

The inheritance problem for strict inheritance networks with conjunction and negation is coNP-complete. Complete the proof sketch from the lecture.

Exercise 11.3 (Inheritance networks with negation, 5)

For this practical assignment, you are asked to implement reasoning in simple inheritance networks that feature negation but no conjunction. Your task is to implement the graph-based approach to decide the inheritance problem for inheritance networks with negation as presented in the lecture.

Input: Your program should be invoked the following way:

```
./inheritance-simple <knowledge base> <query>
```

Here, `<knowledge base>` is the name of a file that contains the knowledge base, and `<query>` is a string[1] that contains an **isa**-statement which is to be evaluated on the knowledge base. Each line in the knowledge base must also be an **isa**-statement. These statements are written as: `concept1 isa concept2`. Both `concept1` and `concept2` are concept terms in either one of the following forms:

- *atomic concepts* are terms consisting of lower case letters, numbers, and hyphens, beginning with a lower case letter. Not allowed are the keywords `isa`, `not`, and `and`.
- *negated concepts* are written as terms `not concept`, where `concept` is an atomic concept.

[1] Usually, when you are working on a shell you have to put the string in quotation marks to protect the spacing.

Output: Depending on whether the query follows from the knowledge base the output should be either `Yes.` or `No.` .

You can use any programming language you like (given that it is usable under Ubuntu 12). Source code **must be submitted** on time to: `westpham@informatik.uni-freiburg.de`.