Technische Universität München (I7) | **Summer term 2011**
Prof. Dr. Dr. h.c J. Esparza / M. Luttenberger / R. Neumann | **Wednesday 6th July, 2011**

Logic – Homework 8

Discussed on 11.07.2011.

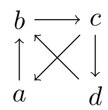
Exercise 8.1 Resolution restrictions

Resolution w.r.t. a set of support: Let F be a formula in clause form. A subset T of F is called a *set of support (sos)* for F if $F \setminus T$ is satisfiable. Given a sos T of F, a *sos-deduction of the empty clause from* (F, T) is any deduction of the empty clause from F where never two clauses from $F \setminus T$ are resolved.

The idea is that as $F \setminus T$ is satisfiable, we can only deduce the empty clause by using at least one clause which is directly or indirectly obtained from the sos T via resolution ("T supports the deduction of the empty clause."). The sos-restriction therefore allows us to guide the resolution such that it does not "get stuck" within the resolvents of $F \setminus T$.

(a) Formally prove that sos-resolution is complete.

(b) Consider the following directed graph:



We can represent the transitive closure of the graph as the following "knowledge base" H in clause form where a, b, c, d are constants, and X, Y, Z are variables (implicitly universally quantified):

$$H = \{\{E(a,b)\}, \{E(b,c)\}, \{E(c,d)\}, \{E(d,b)\}, \{E(c,a)\}\} \cup \{\{\neg T(X,Y), \neg T(Y,Z), T(X,Z)\}, \{\neg E(X,Y), T(X,Y)\}\}.$$

Use sos-resolution to show that a is transitively reachable from d.

Exercise 8.2

(a) On the webpage, you can find the otter-input files for the examples "ape" and "towers of hanoi" discussed in the lecture.

Try to extend the "ape" example analogously to the "towers of hanoi" example such that otter tells you how the ape can reach the banana. See also the remarks in the "ape" input file.

(b) A farmer accompanied by a large radish, a sheep, and a dog needs to cross a river. Unfortunately, he only has a raft which can accommodate at most two of the four objects mentioned (recall, it's a large radish). To state the obvious, only the farmer can steer the raft. The problem, of course, is that the dog is quite hungry and would love to eat the sheep, while the sheep is more than willing to feast on the radish. Thus, he needs to find a way to cross the river in such a way that at no point of time the dog is left alone with the sheep, and analogously for the sheep and the radish.

Analogously to the "ape" example, use otter to find a sequence of commands to guide the farmer and his belongings safely across the river.

Remark: You probably are going to need to use a set of support. See the "ape" file for instructions how to distinguish between a sos T and the clauses $F \setminus \{T\}$.

(c) Otter is accompanied by mace, a tool which enumerates all finites structures for a given formula until a model (or counterexample) is found.

Try to use mace in order to find a solution to the problem described in (b).

Remark: Note that you cannot make use of the Herbrand universe underlying otter/resolution anymore. Your formula therefore should contain additional constraints which guarantee that a model indeed encodes a meaningful solution. Use `mace2 -N 16 -m 1 -p < input.in` to enumerate all models whose universe consists of at most 16 elements and to ouput the first model found.

Exercise 8.3

Express the join $R_1 \bowtie_{i=j} R_2$ (as defined in the slides) by means of selection σ and projection π.