Assignment 7
Introduction to Computational Logic, SS 2006

Prof. Dr. Gert Smolka, Dipl.-Inform. Mathias Möhl
www.ps.uni-sb.de/courses/cl-ss06/

Exercise 7.1 (3-Element Specification) Write a categorical specification A with the signature

$$
\begin{aligned}
0,\ 1,\ 2 &: V \\
\sigma &: V \to V \\
\to &: V \to V \to V
\end{aligned}
$$

such that a proper model $\mathcal{A}$ of A satisfies $\mathcal{A}V = \{\mathcal{A}0,\ \mathcal{A}1,\ \mathcal{A}2\}$ and $|\mathcal{A}V| = 3$. Write your specification such that you can prove $A, i = j \vdash x = 1$ for $i, j \in \{0, 1, 2\}$ and $i \neq j$.

Challenge: Can you find a similar specification for a sort with 4 elements?

Exercise 7.2 (Stability) A deduction rule is stable if for every instance (E, e) of the rule and every substitution θ the pair $(\mathbf{S}\theta E, \mathbf{S}\theta e)$ is an instance for the rule. For Ref, Sym, Trans, CL, CR it is easy to see that they are stable. For β and η one can prove that they are stable. You will show that ξ is not stable.

a) Find an instance (E, e) of ξ such that $(\mathbf{S}\theta E, \mathbf{S}\theta e)$ is not an instance of ξ for $\theta = \{x := a\}$.

b) Find an instance (E, e) of ξ such that $(\mathbf{S}\theta E, \mathbf{S}\theta e)$ is not an instance of ξ for $\theta = \{a := x\}$.

Exercise 7.3 (Deduction Rules) Consider the deduction rules in Figure 2 in §4. You will show that the rules Ref and CL are redundant.

a) Assume that t is a term. Give a derivation of $t = t$ from $\emptyset$ that employs only β, Sym and Trans.
Hint: Consider $(\lambda x.t)x$.

b) Assume that $st = s't$ is an equation. Give a derivation of $st = s't$ from $s = s'$ that employs only CR, β, Sym and Trans.
Hint: Consider $\hat{t} = \lambda f.ft$ and $\hat{t}s$, $\hat{t}s'$.

Exercise 7.4 (Conditionals) Prove $BA \vdash e$ for the following equations:

a) $(x, y, y) = y$

b) $(x, y, z) = (\bar{x} + z)(x + y)$ Hint: use resolution

c) $\overline{(x, y, z)} = (x, \bar{y}, \bar{z})$ Hint: use (b)

d) $(\bar{x}, y, z) = (x, z, y)$

e) $(x, y, z)u = (x, yu, zu)$

f) $(x, y, z) + u = (x, y + u, z + u)$

g) $(x, y, z)(x, u, v) = (x, yu, zv)$

Exercise 7.5 (Dualisation) Let s, t_0, t_1 be Boolean terms. Prove

$$BA \vdash \widehat{(s, t_0, t_1)} = (\hat{s}, \hat{t}_1, \hat{t}_0)$$

Exercise 7.6 (All Prime Trees for $x < y$) Let x, y be variables with $x < y$. Draw all prime trees that contain no other variables than x, y. For each prime tree give an equivalent Boolean term that is as simple as possible. Use also $\rightarrow$ and $\leftrightarrow$.

Exercise 7.7 Find a Boolean term t such that $FV(t) = \{x, y, z\}$ and $SV(t) = \{x\}$.

Exercise 7.8 (Order of Variables) Determine the prime trees for the Boolean term $\bar{x}yu + xzv$ for the following order of variables:

a) $x < y < u < v < z$

b) $u < v < x < y < z$

Exercise 7.9 (Computation of Prime Tree) Let x, y, z be variables with $x < y < z$, and let s be the term $x \leftrightarrow (y \leftrightarrow z)$. Give the prime trees for the following terms:

$$s, \quad \bar{s}, \quad \hat{s}, \quad s \cdot s, \quad s \rightarrow s, \quad s \leftrightarrow \hat{s}$$

Exercise 7.10 (Prime Tree Algorithms) We represent decision trees in Standard ML as follows:

```
type var = int
datatype dt = F | T | D of var * dt * dt
```

The constructors F and T represent 0 and 1, respectively. The order of the variables is the order on *int*.

a) Write a procedure *cond* : *var* → *dt* → *dt* → *dt* such that *cond x t t′* yields $\pi(x, t, t')$ if t and t' are prime trees containing only variables greater that x.
b) Write a procedure *neg* : *dt* → *dt*, such that *neg t* yields $\pi(\bar{t})$ if t is a prime tree.
c) Write a procedure *dual* : *dt* → *dt*, such that *dual t* yields $\pi(\hat{t})$ if t is a prime tree.
d) Write a procedure *andP* : *dt* → *dt* → *dt* such that *andP t t′* yields $\pi(t \cdot t')$ if t and t' are prime trees.
e) Write a procedure *imp* : *dt* → *dt* → *dt* such that *imp t t′* yields $\pi(t \to t')$ if t and t' are prime trees.
f) We represent boolean terms as follows:

```
datatype bt = FF | TT | V of var | NEG of bt
            | AND of bt * bt | OR of bt * bt
```

Write a procedure *pi* : *bt* → *dt* that yields the prime tree for a boolean term.

Exercise 7.11 (Prime Tree Algorithms) You are going to formulate an algorithm that for two prime trees s and t computes the prime tree for $s \leftrightarrow t$.

a) Complete the following equations so that they become tautologies on which the algorithm can be based.

$$(x, y, y) \quad =$$

$$x \leftrightarrow 1 \quad =$$

$$0 \leftrightarrow 0 \quad =$$

$$(x, y, z) \leftrightarrow (x, y', z') \quad =$$

$$(x, y, z) \leftrightarrow u \quad =$$

$$x \leftrightarrow y \quad =$$

b) Complete the declarations of *cond* and *equi* so that *equi* implements the algorithm mentioned above. Do not use other procedures.

```
type var = int
datatype dt = F | T | D of var * dt * dt


fun cond x s t =


fun equi T t =


  | equi t T =


  | equi F F =


  | equi F (D(y,t0,t1)) =


  | equi (D(x,s0,s1)) F =


  | equi (s as D(x, s0, s1)) (t as D(y, t0, t1)) =

    if x=y then


    else if x<y then


    else
```