Prof. Dr. K. Madlener
Dipl.-Inf. P. Michel
Dipl.-Inf. C. Feller

University of Kaiserslautern
Department of Computer Science
AG Grundlagen der Informatik

Exercise Sheet 4: Specification and Verification with Higher-Order Logic (Summer Term 2011)

Date: 10.05.2011

Exercise 1 Conservative Extensions

a) (Prepare!) Let $T = (\chi, \Sigma, A)$ be the core HOL theory as defined in the lecture. Consider the following extension of T:

$$T' = (\chi, \Sigma, A \cup \{(\neg P \Longrightarrow P) \Longrightarrow P\})$$

Is T' a conservative extension of T?

b) (Prepare!) In the lecture we defined the type set of typed sets (slide 276), using the conservative extension schema for type definitions (slide 273).

Based on the types of core HOL and nat, define the type $mset$ of typed multisets in the same style.

Hint: Multisets are sets where the same element can appear more than once.

c) (Prepare!) Based on the types of core HOL and nat, define the type $list$ of typed lists.

d) Define both types in Isabelle/HOL using `typedef` and define additional helpful functions on the types.

e) Define simple generic properties of the newly defined functions and prove them (e.g. the empty list does not contain any elements, formulated on the two constants `empty` and `contains`).

Handling (type-)definitions: Functions on newly defined types are likely defined as `definition`s and involve applications of `Rep_t` and `Abs_t`. Isabelle/HOL does **not** automatically use definitions for simplification. As definitions define equalities, however, you can use the proof command `apply (subst myfunction_def)` to unfold them. Using the same command you can unfold the definition of the type (`t_def`) and the two axioms `Rep_t_inverse` and `Abs_t_inverse`.

Exercise 2 Methods and Rules in Isabelle/HOL

In this exercise we want to practice the use of different methods (like `rule`, `erule` or `frule`) to prove properties in propositional and predicate logic.

You should only use the rules of the first exercise sheet, together with the following additional rules: `conjE`, `impE`, `iffI`, `iffE` and `classical`.

Hint: You can always invoke `C-c C-v` *to enter a command like* `thm impI` *and see the concrete definition of the rule in Isabelle/HOL.*

a) (Prepare!) Apply the rule

$$[\![(?a, ?b) \in ?r^*; \bigwedge x.\ ?P\ x\ x; \bigwedge x\ y\ z.\ [\![(x, y) \in ?r^*;\ ?P\ x\ y;\ (y, z) \in ?r]\!] \Longrightarrow ?P\ x\ z]\!] \Longrightarrow ?P\ ?a\ ?b$$

with the method `erule` to the following subgoal by hand (i.e. on paper):

$$(i, j) \in s^* \Longrightarrow 0 \leq (dist\ i\ j)$$

Hint: Don't be distracted by unknown function names; you don't have to know anything about their meaning. Just apply the rule syntactically.

b) Prove or disprove the following theorems.

- $A \longrightarrow A$
- $A \wedge B \longrightarrow B \wedge A$
- $(A \wedge B) \longrightarrow (A \vee B)$
- $((A \vee B) \vee C) \longrightarrow A \vee (B \vee C)$
- $A \longrightarrow B \longrightarrow A$
- $(A \vee A) = (A \wedge A)$
- $(A \longrightarrow B \longrightarrow C) \longrightarrow (A \longrightarrow B) \longrightarrow A \longrightarrow C$
- $(A \longrightarrow B) \longrightarrow (B \longrightarrow C) \longrightarrow A \longrightarrow C$
- $\neg\neg A \longrightarrow A$
- $A \longrightarrow \neg\neg A$
- $(\neg A \longrightarrow B) \longrightarrow (\neg B \longrightarrow A)$
- $((A \longrightarrow B) \longrightarrow A) \longrightarrow A$
- $A \vee \neg A$
- $(\neg(A \wedge B)) = (\neg A \vee \neg B)$
- $(\exists x.\ \forall y.\ P\ x\ y) \longrightarrow (\forall y.\ \exists x.\ P\ x\ y)$
- $(\forall x.\ P\ x \longrightarrow Q) = ((\exists x.\ P\ x) \longrightarrow Q)$
- $((\forall x.\ P\ x) \wedge (\forall x.\ Q\ x)) = (\forall x.\ (P\ x \wedge Q\ x))$
- $((\forall x.\ P\ x) \vee (\forall x.\ Q\ x)) = (\forall x.\ (P\ x \vee Q\ x))$
- $((\exists x.\ P\ x) \vee (\exists x.\ Q\ x)) = (\exists x.\ (P\ x \vee Q\ x))$
- $(\forall x.\ \exists y.\ P\ x\ y) \longrightarrow (\exists y.\ \forall x.\ P\ x\ y)$
- $(\neg(\forall x.\ P\ x)) = (\exists x.\ \neg P\ x)$