

Exam Logical Verification

June 23, 2004

There are six (6) exercises.

Answers may be given in Dutch or English. Good luck!

Exercise 1. This exercise is concerned with first-order minimal propositional logic and simply typed λ -calculus.

- a. Show that the formula $((A \rightarrow B \rightarrow A) \rightarrow B) \rightarrow B$ is a tautology.
(5 points)
- b. Give the type derivation in simply typed λ -calculus corresponding to the proof of 1a.
(5 points)
- c. Replace in the following three terms the ?'s by types, such that we obtain correctly typed λ -terms. (NB: no typing derivation is asked.)
 $\lambda x :?. \lambda y :?. \lambda z :?. (x (y z))$
 $\lambda x :?. \lambda y :?. ((x y) y)$
 $\lambda x :?. \lambda y :?. (x (x y))$
(6 points)
- d. Give the derivation in first-order minimal propositional logic corresponding to the completed λ -term $\lambda x :?. \lambda y :?. (x (x y))$ (the last one from 1c).
(4 points)

Exercise 2. This exercise is concerned with first-order minimal propositional logic and simply typed λ -calculus.

- a. What is the provability question in first-order minimal propositional logic?
Is this decidable?
(NB: yes or no as answer suffices.)
(5 points)
- b. Give the correspondence between proofs in first-order minimal propositional logic and terms in simply typed λ -calculus in detail.
(5 points)

- c. Give the general form of a detour in first-order minimal propositional logic. Give also the corresponding part of a typing derivation in simply typed λ -calculus.
(5 points)

Exercise 3. This exercise is concerned with inductive types in Coq.

- a. Give the inductive definition of the datatype `natlist` of finite lists of natural numbers. The type `nat` for natural numbers may be used.
(5 points)
- b. Give the type of `natlist_ind` which is used to give proofs by induction on the finite lists of natural numbers.
(5 points)
- c. Give the definition in Coq of the function `length` that computes the length of a list in `natlist`.
(NB: small mistakes in the Coq syntax do not count.)
(5 points)
- d. Give the definition in Coq of the inductive predicate `evenlist` that holds exactly if the length of the list is even.
(NB: zero is even.)
(5 points)

Exercise 4.

- a. Give the two rules for detour-elimination in first-order minimal predicate logic.
(5 points)
- b. Show that $\forall x. (P(x) \rightarrow \neg \forall y. (\neg P(y)))$ is a tautology of first-order predicate logic.
(5 points)

Exercise 5. This exercise is concerned with λ -calculus with dependent types (λP).

- a. The application rule for λP is

$$\frac{\Gamma \vdash F : (x : A) B \quad \Gamma \vdash M : A}{\Gamma \vdash (F M) : B[x := M]}$$

Explain shortly the presence of the substitution $[x := M]$.

(5 points)

- b. First-order propositional logic can be encoded in Coq using dependent types as follows:

```
(* prop representing the propositions is declared as a Set *)
Variable prop:Set.
(* T expresses if a proposition in prop is valid
   if (T p) is inhabited then p is valid
   if (T p) is not inhabited then p is not valid *)
Variable T: prop -> Prop.
(* conjunction is a binary operator represented by conj *)
Variable conj : prop -> prop -> prop .
```

Give the types of the variables

```
conj_intro
conj_elim_l
conj_elim_r
```

modeling introduction of conjunction, and left- and right elimination of conjunction.

(5 points)

Exercise 6. This exercise is concerned with the polymorphic λ -calculus ($\lambda 2$) and second-order propositional logic.

- a. Give the polymorphic identity and its type.
Show how the polymorphic identity is instantiated in order to get the identity on the type `nat` of natural numbers.
(5 points)
- b. Give the encoding of the formulas of second-order minimal propositional logic in $\lambda 2$.
(5 points)
- c. Define the type `new_and` as follows:

$$(\text{new_and } A B) = \Pi c:\text{Prop}. (A \rightarrow B \rightarrow c) \rightarrow c$$

with $A : \text{Prop}$ and $B : \text{Prop}$.

Assume $\Gamma \vdash P : (\text{new_and } A B)$. Show how an inhabitant of A is obtained.

(NB: do not worry about the environments.)

(5 points)

The final note is (the total amount of points plus 10) divided by 10.