

# Exam Logical Verification

January 14, 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  $\lambda$ -calculus.

- a. Show that the formula  $((A \rightarrow B) \rightarrow C) \rightarrow B \rightarrow C$  is a tautology.  
(That is, give a proof in which all assumptions are cancelled.)  
(5 points)
- b. Give the type derivation in simply typed  $\lambda$ -calculus corresponding to the proof of 1a.  
(5 points)
- c. Replace in the following three terms the ?'s by simple types, such that we obtain typable  $\lambda$ -terms. (NB: it is not asked to give the type derivations.)  
 $\lambda x : ?. \lambda y : ?. \lambda z : ?. (x z) (y z)$   
 $\lambda x : ?. \lambda y : ?. x (x y)$   
 $\lambda x : ?. \lambda y : ?. \lambda z : ?. (y x) (x z)$   
(6 points)
- d. Give a proof of  $A \rightarrow B \rightarrow A$  with a detour, and in which all assumptions are cancelled.  
(4 points)

**Exercise 2.** This exercise is concerned with first-order minimal propositional logic and simply typed  $\lambda$ -calculus, and consistency.

- a. What is the inhabitation problem in simply typed  $\lambda$ -calculus?  
(3 points)
- b. The inhabitation problem corresponds via the Curry-Howard-De Bruijn isomorphism to a problem in first-order minimal propositional logic.  
What problem?  
(2 points)

- c. Give the correspondence between proofs in first-order minimal propositional logic and terms in simply typed  $\lambda$ -calculus in detail.  
(5 points)
- d. If a type is inhabited by a closed term, then it is inhabited by a closed term in  $\beta$ -normal form. A closed term in  $\beta$ -normal form is of the form  $\lambda x : A. M$ .  
Does a closed inhabitant of the type  $\perp$  exist? Why (not)?  
(5 points)
- e. Now consider  $\perp$  as a formula in first-order minimal propositional logic. What can we conclude using 2abcd?  
(2 points)

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

- a. Give the inductive definition of the datatype `nat` of natural numbers.  
(5 points)
- b. Give the type of `nat_ind` which is used to give proofs by induction on the natural numbers.  
(5 points)
- c. Give the definition of the inductive type `bintree` of binary trees with natural numbers both on the *leaves* and on the *nodes*.  
(5 points)

**Exercise 4.** This exercise is concerned with first-order predicate logic.

- a. Give the two detours of first-order minimal predicate logic.  
(5 points)
- b. Show that  $\forall x. (P(x) \rightarrow \neg(\forall x. \neg P(x)))$  is a tautology of first-order minimal predicate logic with  $\perp$ .  
(5 points)

**Exercise 5.** This exercise is concerned with  $\lambda$ -calculus with dependent types ( $\lambda P$ ).

- a. We assume a constructor `natlist_dep` that is used to build the type ‘lists of natural numbers of length  $n$ ’ for every  $n \in \{0, 1, 2, \dots\}$ .  
What is the type of `natlist_dep`?  
(3 points)
- b. A typing rule that is characteristic for  $\lambda P$  is the following:

$$\frac{\Gamma \vdash A : \text{Set} \quad \Gamma, x : A \vdash B : \text{Type}}{\Gamma \vdash \Pi x:A. B : \text{Type}}$$

Explain the use of this rule in the `natlist_dep` example. (Hint: think of 5a.)  
(5 points)

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

```
(* prop representing the propositions is a Set *)
Variable prop:Set.
(* implication on prop is a binary operator *)
Variable imp: prop -> prop -> prop.
(* 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.
```

Give the types of the variables `imp_introduction` and `imp_elimination` modelling the introduction- and elimination rule of implication.

(5 points)

**Exercise 6.** This exercise is concerned with polymorphic  $\lambda$ -calculus ( $\lambda 2$ ).

- a. Define the type `new_or`

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

Assume  $\Gamma \vdash a : A$ . Give an inhabitant of `(new_or A B)`.

(NB: it is not asked to give the type derivation.)

(5 points)

- b. Assume `new_or` as in 6a, and in addition  $\Gamma \vdash f : A \rightarrow D$ , and  $\Gamma \vdash g : B \rightarrow D$ , and  $\Gamma \vdash M : (\text{new\_or } A B)$ . Give an inhabitant of `D`.

(NB: it is not asked to give the type derivation.)

(5 points)

- c. We define the booleans `B` and `true` (`T`) and `false` (`F`) as follows:

```
B =  $\Pi a:\text{Set}. a \rightarrow a \rightarrow a$ 
T =  $\lambda a:\text{Set}. \lambda x:a. \lambda y:a. x$ 
F =  $\lambda a:\text{Set}. \lambda x:a. \lambda y:a. y$ 
```

Give a definition of negation in  $\lambda 2$ .

(5 points)

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