# Tentamen Toegepaste Logica

12 januari 2000

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

**Exercise 1.** This question concerns minimal logic and simply typed lambda calculus.

a. Give a proof of the formula

$$(A \to (B \to C)) \to B \to A \to C$$

in minimal logic such that the proof contains a detour and doesn't contain open assumptions.

(5 points)

b. Give the typing derivation corresponding to the proof given as anwer to question 1a.

(5 points)

c. Reduce the term of the answer to question 1b to beta-normal form (it is not necessary to give the typing derivation).

(3 points)

d. Give the proof corresponding to (the typing derivation) of the term found as answer to question 1c.

(5 points)

**Exercise 2.**

a. What concept in lambda calculus does a formula correspond to? Explain this correspondence in detail.

(3 points)

b. What concept in lambda calculus does a proof correspond to? Explain this correspondence in detail.

(3 points)

c. What concept in lambda calculus does a detour correspond to? Explain this correspondence in detail.

   (3 points)

d. What is inhabitation? To which concept in minimal logic does it correspond?

   (3 points)

e. What is type checking? To which concept in minimal logic does it correspond?

   (3 points)

**Exercise 3.**

a. Give and explain the definition of an inductive type `nat` of natural numbers in Coq.

   (5 points)

b. Give the type of the term `nat_ind` that is automatically generated by Coq once the inductive definition of `nat` is given.

   (5 points)

c. Explain how the term `nat_ind` is used for induction on natural numbers.

   (5 points)

d. Explain what program extraction is (this is done in Coq using the tactic `Extraction`).

   (5 points)

**Exercise 4.** Consider the term

$$\mathsf{plus} = \lambda m{:}\mathsf{Nat}.\, \lambda n{:}\mathsf{Nat}.\, \mathsf{r}_{\mathsf{Nat}}(n, \lambda x{:}\mathsf{Nat}.\, \lambda y{:}\mathsf{Nat}.\, \mathsf{s}(x), m).$$

in Gödel's system **T**. Recall that the rewrite rules for the recursor $\mathsf{r}_{\mathsf{Nat}}$ are as follows, with $N : \mathsf{Nat}$, $F : \mathsf{Nat}{\to}\mathsf{Nat}{\to}\mathsf{Nat}$, $\mathsf{z} : \mathsf{Nat}$, and $\mathsf{s} : \mathsf{Nat}{\to}\mathsf{Nat}$:

$$\begin{array}{lcl} \mathsf{r}_{\mathsf{Nat}}(N, F, \mathsf{z}) & \to & N \\ \mathsf{r}_{\mathsf{Nat}}(N, F, \mathsf{s}(P)) & \to & F\, \mathsf{r}_{\mathsf{Nat}}(N, F, P)\, P \end{array}$$

a. Suppose we want to show that the term plus represents addition. Which two things need to be shown?

   (4 points)

b. Show that the term plus indeed represents addition, that is, show that the two requirements formulated in the answer to question 4a hold.

   (8 points)

**Exercise 5.** This question concerns first-order predicate logic and lambda calculus with dependent types ($\lambda P$).

a. Show that the following formula is a tautology of first-order predicate logic:

$$(\forall x.A(x) \to B(x)) \to (\forall x.A(x)) \to \forall y.B(y)$$

(5 points)

b. Explain the correpondence between proofs in first-order predicate logic of the answer to question 5a and terms (typing rules) in $\lambda P$.

(5 points)

c. The typing system for $\lambda P$ (lambda calculus with dependent types) contains the conversion rule:

$$\frac{\Gamma \vdash A : B \qquad \Gamma \vdash B' : s}{\Gamma \vdash A : B'}$$

with $B =_\beta B'$. Explain why this rule is necessary.

(5 points)

**Exercise 6.** This question concerns lambda calculus with polymorphic types.

a. The function map takes as input a function from natural numers to natural numbers and a list of natural numbers, and gives as output a list of natural numbers.

We wish to consider a polymorphic version of the map-function, denoted by pmap, where the type of the elements of the list is a parameter. What is the type of the function pmap?

(5 points)

b. Let the type of natural numbers be written as Nat and the type of booleans as Bool. Explain how pmap can be used to obtain a map-function for lists of natural numbers, and how it can be used to obtain a map-function for lists of booleans.

(5 points)

*Het tentamencijfer is (het totaal aantal punten plus 10) gedeeld door 10.*