Type Theory and Coq 2020-2021
02-11-2020

This exam consists of 9 out of 18 exercises (therefore, the numbering is not consecutive). Every exercise is worth 10 points, the first 10 points are free, and the final mark is the number of points divided by 10. The mark for the exam will be rounded to an integral mark afterwards. The oral part of this exam determines in which direction we will round.

To hand in your work, you have to submit photos or scans of your *handwritten* answers in the Brightspace assignment where you got this exam PDF. You can submit multiple times, but only the last submission will be looked at. Write your name, study ('MFoCS', 'software science' or 'mathematics') and student number on your first answer sheet for reference.

Write proofs, terms and types according to the conventions of Femke's course notes. Obviously this exam is open book, and you can also use your computer if you like. However, communication among students is not allowed, see the notice at the end of the exam.

If a technical problem occurs during the exam, send mail to `freek@cs.ru.nl` and we will call you back as soon as possible, to see what we can do.

Good luck!

Logics and systems of the lambda cube:

1. Give a closed inhabitant in simple type theory of the type
$$(a \to a \to b) \to a \to b$$
and give the corresponding full type derivation.

2. Give an example of a proof in propositional logic that contains a detour for disjunction, and also give the normal form of that proof.

3. Give a proof in predicate logic of the formula
$$(\exists x.\, \forall y.\, q(x, y)) \to (\forall z.\, \exists w.\, q(w, z))$$

4. Give a proof in full intuitionistic second order propositional logic of the formula:
$$\exists a.\, (\forall b.\, a \to b)$$

5. Give a term H such that
$$b : *,\, x : \neg b \vdash H : \forall a.\, b \to a$$
is derivable in (Church-style) $\lambda 2$. *(Note: you should* ***not*** *give the derivation!)* In this we use the abbreviations:
$$\bot := \forall a.\, a$$
$$\neg b := b \to \bot$$
You can write either $\forall a.\, A$ or $\Pi a : *.\, A$, and $\Lambda a.\, M$ or $\lambda a : *.\, M$. (In this exercise we use the first notation.)

6. Give a derivation of the λP judgment

$$a : *,\ v : a \to *,\ x : a \vdash vx : *$$

Duplicate subderivations can be replaced with dots, and the rules of λP are on page 5.

7. Give a (Church-style) $\lambda 2$ type X such that for all $x : X$, the term xX is well typed. Explain your answer.

Inductive types:

8. Give an inductive definition of conjunction, together with both the dependent and non-dependent induction principles.

9. Define the (truncated) predecessor function on the natural numbers, i.e., the function defined as

$$\texttt{pred}\ n = \begin{cases} 0 & \text{if } n = 0 \\ n - 1 & \text{if } n > 0 \end{cases}$$

using the recursor:

```
nat_rec
     : forall P : nat -> Set,
       P 0 -> (forall n : nat, P n -> P (S n)) -> forall n : nat, P n
```

10. Suppose we have a goal $P(s)$ and we rewrite with an assumption $H : t = s$, using `rewrite <- H` to get a new goal $P(t)$. In this s and t have the type D.

The final proof of $P(s)$ will be a term M' that contains a subterm M of type $P(t)$, and the derivation of the typing of M' will look like:

$$\dfrac{\vdots \quad \dfrac{\dfrac{\vdots}{\Gamma,\ H : t = s \vdash M : P(t)} \quad \vdots \quad \vdots}{\Gamma,\ H : t = s \vdash M' : P(s)}}{}$$

The exercise is to give the term M' in terms of M and H, using the induction principle for equality:

```
eq_ind
     : forall (A : Type) (x : A) (P : A -> Prop),
       P x -> forall y : A, x = y -> P y
```

11. In Coq, the relation $\leq$ is defined by:

```
Inductive le (n : nat) : nat -> Prop :=
    le_n : le n n | le_S : forall m : nat, le n m -> le n (S m)
```

The corresponding non-dependent induction principle that is generated by Coq is:

```
le_ind
     : forall (n : nat) (P : nat -> Prop),
       P n ->
       (forall m : nat, le n m -> P m -> P (S m)) ->
       forall n0 : nat, le n n0 -> P n0
```

The exercise is to write down the type of the corresponding *dependent* induction principle `le_ind_dep` (which Coq does not generate), where `P` is a predicate on the type `le n m`, and which involves the constructors `le_n` and `le_S` explicitly.

Metatheory:

12. Give a term M of untyped lambda calculus that satisfies:

$$Mx =_\beta M$$

In other words, we want a term that 'eats' its arguments. In your answer you do not need to write out the fixed point combinator Y.

13. Write the untyped lambda term

$$\lambda xy.xy(xyy)$$

with parentheses for *each* abstraction and application, and compute its principal type using algorithm W which was described in Herman's lecture.

14. Give a rewrite system $(A, \to_R)$, i.e., with $\to_R \subseteq A \times A$, for which $\mathrm{UN}(\to_R)$ holds, but $\mathrm{CR}(\to_R)$ does not hold.

15. Give the full reduction graph of the untyped lambda term:

$$(\lambda f.\, f(II))\, I$$

where $I := \lambda x.\, x$. For each of the terms M in this graph compute the associated term M^*, defined by:

$$\begin{aligned} x^* &:= x \\ (\lambda x.\, M)^* &:= \lambda x.\, M^* \\ (MN)^* &:= \begin{cases} P^*[x := N^*] & \text{if } M = \lambda x.\, P \\ M^* N^* & \text{otherwise} \end{cases} \end{aligned}$$

16. Prove the *thinning lemma* for simple type theory. This lemma says that:

$$\Gamma \vdash M : A,\ \ \Gamma \subseteq \Delta \ \Rightarrow\ \Delta \vdash M : A$$

17. The term $\lambda x.\, xx$ is typable in Curry-style $\lambda 2$, with type $\bot \to \bot$, where we use the abbreviation $\bot := \forall a.\, a$. The exercise is to give the corresponding term for Church-style $\lambda 2$.

    Some relevant typing rules are, Curry-style:

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M : \forall a.\, A}\ a \notin \mathsf{FV}(\Gamma) \qquad \frac{\Gamma \vdash M : \forall a.\, A}{\Gamma \vdash M : A[a := B]}$$

    and Church-style:

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \Lambda a.\, M : \forall a.\, A}\ a \notin \mathsf{FV}(\Gamma) \qquad \frac{\Gamma \vdash M : \forall a.\, A}{\Gamma \vdash MB : A[a := B]}$$

    You can write either $\forall a.\, A$ or $\Pi a : *.\, A$, and $\Lambda a.\, M$ or $\lambda a : *.\, M$. (In this exercise we use the first notation.)

18. The type $\forall a.\, a$ represents falsity in $\lambda 2$. In the saturated set semantics used to prove that $\lambda 2$ is SN, the set of untyped lambda terms

$$[\![\forall a.\, a]\!]_\rho$$

    does not depend on ρ, as there are no free type variables. Is this set empty or not? Explain your answer.

And finally: *By taking the exam, the student declares that no plagiarism is or will be committed. If the lecturer has the suspicion that fraud has been committed, the student will be contacted. If needed, the case will be redirected to the Examination Board.*

Typing rules of λP

axiom

$$\frac{}{\vdash * : \Box}$$

variable

$$\frac{\Gamma \vdash A : s}{\Gamma,\, x : A \vdash x : A}$$

weakening

$$\frac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s}{\Gamma,\, x : C \vdash A : B}$$

application

$$\frac{\Gamma \vdash M : \Pi x : A.\, B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B[x := N]}$$

abstraction

$$\frac{\Gamma,\, x : A \vdash M : B \qquad \Gamma \vdash \Pi x : A.\, B : s}{\Gamma \vdash \lambda x : A.\, M : \Pi x : A.\, B}$$

product

$$\frac{\Gamma \vdash A : * \qquad \Gamma,\, x : A \vdash B : s}{\Gamma \vdash \Pi x : A.\, B : s}$$

conversion

$$\frac{\Gamma \vdash A : B \qquad \Gamma \vdash B' : s}{\Gamma \vdash A : B'} \quad \text{when } B =_\beta B'$$