Type Theory and Coq 2020-2021

02-11-2020

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.

$\Gamma := x : a \to a \to b,\ y : a$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : a \to a \to b} \quad \dfrac{}{\Gamma \vdash y : a}}{\Gamma \vdash xy : a \to b} \qquad \dfrac{}{\Gamma \vdash y : a}}{\Gamma \vdash xyy : b}}{x : a \to a \to b \vdash \lambda y : a.\, xyy : a \to b}}{\vdash \lambda x : a \to a \to b.\, \lambda y : a.\, xyy : (a \to a \to b) \to a \to b}$$

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.

$$\dfrac{\dfrac{\dfrac{[a^x]}{a \vee a}\,Il\vee \qquad \dfrac{[a^{x_l}]}{a \to a}\,I[x_l]{\to} \qquad \dfrac{[a^{x_r}]}{a \to a}\,I[x_r]{\to}}{a}\,E\vee}{a \to a}\,I[x]{\to}$$

The top part of this has the shape:

$$\dfrac{\dfrac{\begin{array}{c} \vdots_1 \\ A \end{array}}{A \vee B}\,Il\vee \qquad \dfrac{\begin{array}{c} [A^{x_l}] \\ \vdots_2 \\ C \end{array}}{A \to C}\,I[x_l]{\to} \qquad \dfrac{\begin{array}{c} [B^{x_r}] \\ \vdots_3 \\ C \end{array}}{B \to C}\,I[x_r]{\to}}{C}\,E\vee$$

in which an introduction rule for disjunction is directly followed by a corresponding elimination rule, i.e., there is a detour for disjunction. Normalizing this proof consists of using the subproof 1 for all the assumptions $[A^{x_l}]$ in the subproof 2 of C:

$$\begin{array}{c} \vdots_1 \\ A \\ \vdots_2 \\ C \end{array}$$

When we do this for the example proof, we find the normal form for this proof to be:

$$\dfrac{[a^x]}{a \to a}\, I[x]{\to}$$

3. Give a proof in predicate logic of the formula

$$(\exists x.\, \forall y.\, q(x, y)) \to (\forall z.\, \exists w.\, q(w, z))$$

$$\dfrac{\dfrac{[\exists x.\, \forall y.\, q(x, y)^{H_1}] \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\forall y.\, q(x, y)^{H_2}]}{q(x, z)}\, E\forall}{\exists w.\, q(w, z)}\, I\exists}{\forall z.\, \exists w.\, q(w, z))}\, I\forall}{(\forall y.\, q(x, y)) \to \forall z.\, \exists w.\, q(w, z)}\, I[H_2]{\to}}{\forall x.\, ((\forall y.\, q(x, y)) \to \forall z.\, \exists w.\, q(w, z))}\, I\forall}{\forall z.\, \exists w.\, q(w, z))}\, E\exists}{(\exists x.\, \forall y.\, q(x, y)) \to (\forall z.\, \exists w.\, q(w, z))}\, I[H_1]{\to}$$

4. Give a proof in full intuitionistic second order propositional logic of the formula:

$$\exists a.\, (\forall b.\, a \to b)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\bot^x]}{b}\, E\bot}{\bot \to b}\, I[x]{\to}}{\forall b. \bot \to b}\, I\forall}{\exists a.\, (\forall b.\, a \to b)}\, E\exists$$

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.)

$$H := \Lambda a.\, \lambda y : b.\, xya$$

Or in the different notation:

$$H := \lambda a : *.\, \lambda y : b.\, xya$$

In this term we have the typings:

$$\begin{array}{rcl}
x & : & \neg b \equiv b \to \bot \\
y & : & b \\
xy & : & \bot \equiv \forall a.\, a \\
xya & : & a \\
\lambda y : b.\, xya & : & b \to a \\
\Lambda a.\, \lambda y : b.\, xya & : & \forall a.\, b \to a
\end{array}$$

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 10.

$\Gamma := a : *,\ v : a \to *,\ x : a$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{\vdash * : \square}}{a : * \vdash a : *} \qquad \dfrac{\dfrac{\overline{\vdash * : \square} \quad \overline{\vdash * : \square}}{a : * \vdash * : \square} \quad \dfrac{\vdots}{a : * \vdash a : *}}{a : *, x : a \vdash * : \square}}{a : * \vdash a \to * : \square}}{a : *,\ v : a \to * \vdash v : a \to *} \qquad \dfrac{\dfrac{\vdots}{a : * \vdash a : *} \quad \dfrac{\vdots}{a : * \vdash a \to * : \square}}{a : *,\ v : a \to * \vdash a : *}}{\Gamma \vdash v : a \to *} \qquad \dfrac{\dfrac{\vdots}{a : *, v : a \to * \vdash a : *}}{\Gamma \vdash x : a}}{\Gamma \vdash vx : *}$$

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

Take for example

$$X := \Pi a : *.\, a$$

the impredicative encoding of falsity. Then $X : *$, and if $x : X$ then x is a function that maps a type a to an inhabitant of a. This means that $xX : X$, and therefore xX is well typed.

Inductive types:

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

```
Inductive and (A B : Prop) : Prop :=
    conj : A -> B -> and A B
```

```
and_ind_dep
     : forall (A B : Prop) (P : and A B -> Prop),
       (forall (a : A) (b : B), P (conj A B a b)) -> forall a : and A B, P a
```

```
and_ind
     : forall A B P : Prop, (A -> B -> P) -> and A B -> P
```

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
```

```
pred = nat_rec (fun _ : nat => nat) 0 (fun n _ : nat => 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:

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

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
```

$$M' = \texttt{eq\_ind}\, D\, t\, (\lambda x : D.\, P(x))\, M\, s\, H$$

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.

```
le_ind_dep
     : forall (n : nat) (P : forall m : nat, le n m -> Prop),
       P n (le_n n) ->
       (forall (m : nat) (H : le n m), P m H -> P (S m) (le_S n m H)) ->
       forall (m : nat) (H : le n m), P m H
```

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.

It clearly is sufficient if

$$M =_\beta \lambda x.M$$

which follows from

$$M =_\beta (\lambda mx.m)M$$

This is a fixed point equation solved by:

$$M := Y(\lambda mx.m)$$

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.

The term with full parentheses is:

$$(\lambda x.(\lambda y.((xy)((xy)y))))$$

When we annotate the term with type variables, we get

$$\lambda x^{\alpha}.\lambda y^{\beta}.\ \underbrace{\underbrace{xy}_{\gamma}\,(\overbrace{\overbrace{xy}^{\epsilon}\ y}^{\eta})}_{\delta}$$

with equations:

$$\begin{aligned}
\alpha &= \beta \to \gamma \\
\gamma &= \eta \to \delta \\
\alpha &= \beta \to \epsilon \\
\epsilon &= \beta \to \eta
\end{aligned}$$

If we do step I on the first equation, we get:

$$\begin{aligned}
\alpha &= \beta \to \gamma \\
\gamma &= \eta \to \delta \\
\beta \to \gamma &= \beta \to \epsilon \\
\epsilon &= \beta \to \eta
\end{aligned}$$

Then step I on the second equation, substituting in the first equation as well, gives:

$$\begin{aligned}
\alpha &= \beta \to \eta \to \delta \\
\gamma &= \eta \to \delta \\
\beta \to (\eta \to \delta) &= \beta \to \epsilon \\
\epsilon &= \beta \to \eta
\end{aligned}$$

Step II on the third equation, and then removing $\beta = \beta$ gives:

$$\begin{aligned}
\alpha &= \beta \to \eta \to \delta \\
\gamma &= \eta \to \delta \\
\eta \to \delta = {}& \epsilon \\
\epsilon &= \beta \to \eta
\end{aligned}$$

Then another step I on the third equation, after first reversing it, gives:

$$\begin{aligned} \alpha &= \beta \to \eta \to \delta \\ \gamma &= \eta \to \delta \\ \epsilon &= \eta \to \delta \\ \eta \to \delta &= \beta \to \eta \end{aligned}$$

Another step II on the fourth equation:

$$\begin{aligned} \alpha &= \beta \to \eta \to \delta \\ \gamma &= \eta \to \delta \\ \epsilon &= \eta \to \delta \\ \eta &= \beta \\ \delta &= \eta \end{aligned}$$

A step I on the fourth equation (substituing this in the earlier equations as well):

$$\begin{aligned} \alpha &= \beta \to \beta \to \delta \\ \gamma &= \beta \to \delta \\ \epsilon &= \beta \to \delta \\ \eta &= \beta \\ \delta &= \beta \end{aligned}$$

Finally, step I on the last equation, substituting δ everywhere else:

$$\begin{aligned} \alpha &= \beta \to \beta \to \beta \\ \gamma &= \beta \to \beta \\ \epsilon &= \beta \to \beta \\ \eta &= \beta \\ \delta &= \beta \end{aligned}$$

Now the type of the term was $\alpha \to \beta \to \delta$, which becomes under these substitutions:

$$(\beta \to \beta \to \beta) \to \beta \to \beta$$

Which therefore is the principal type of this term.

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.

Take for example:

$$A := \{a, b, c\}$$

with

$$\begin{aligned} a &\to_R b \\ a &\to_R c \\ c &\to_R c \end{aligned}$$

Or, in a picture:

$$\begin{tikzcd}
 & a \arrow[dl] \arrow[dr] & \\
b & & c \arrow[loop below]
\end{tikzcd}$$

The only normal form is b, so Uniqueness of Normal forms clearly holds. But b and c are both reducts of a, and do not have a common reduct, so the system is not Church-Rosser.

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:

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

$$\begin{tikzcd}
 & (\lambda f.\, f(II))\, I \arrow[dl] \arrow[d] \\
(\lambda f.\, fI)\, I \arrow[dr] & I(II) \arrow[d] \arrow[d, bend left] \\
 & II \arrow[d] \\
 & I
\end{tikzcd}$$

$$\begin{aligned}
((\lambda f.\, f(II))\, I)^* &= II \\
((\lambda f.\, fI)\, I)^* &= II \\
(I(II))^* &= I \\
(II)^* &= I \\
I^* &= I
\end{aligned}$$

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

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

We prove by induction on the derivation of $\Gamma \vdash M : A$ that for all $\Delta \supseteq \Gamma$ we have $\Delta \vdash M : A$. Because there are three derivation rules, the induction proof has three cases:

- If the last rule was the variable rule

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

  then we have $(x : A) \in \Gamma$, so certainly $(x : A) \in \Delta$.

- If the last rule was the application rule

  $$\frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

  then with induction we know that $\Delta \vdash M : A \to B$ and $\Delta \vdash N : A$ are derivable, which gives us the derivation

  $$\frac{\Delta \vdash M : A \to B \quad \Delta \vdash N : A}{\Delta \vdash MN : B}$$

  of $\Delta \vdash MN : B$.

- If the last rule was the abstraction rule

  $$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.\, M : A \to B}$$

  and $\Gamma \subseteq \Delta$, then also $\Gamma, x : A \subseteq \Delta, x : A$, and therefore we have by induction that $\Delta, x : A \vdash M : B$ is derivable. This gives us with

  $$\frac{\Delta, x : A \vdash M : B}{\Delta \vdash \lambda x : A.\, M : A \to B}$$

  a derivation of $\Delta \vdash \lambda x : A.\, M : A \to B$.

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.)

$$\lambda x : \bot .\, x\,(\bot \to \bot)\,(x\bot)$$

We have:

$$\begin{array}{rcl} x & : & \bot \equiv \forall a.\, a \\ x\,(\bot \to \bot) & : & \bot \to \bot \\ x\bot & : & \bot \\ x\,(\bot \to \bot)\,(x\bot) \equiv (x\,(\bot \to \bot))\,(x\bot) & : & \bot \\ \lambda x : \bot .\, x\,(\bot \to \bot)\,(x\bot) & : & \bot \to \bot \end{array}$$

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.

We have:

$$[\![\forall a.\, a]\!]_\rho = \bigcap_{X \in \mathsf{SAT}} [\![a]\!]_{\rho, \alpha := X} = \bigcap_{X \in \mathsf{SAT}} X$$

So the set $[\![\forall a.\, a]\!]_\rho$ is the intersection of all saturated sets. By definition of 'saturated set' we have that $xN_1 \ldots N_k$ is an element of each saturated set for all $k \geq 0$ and $N_1, \ldots, N_k \in \mathsf{SN}$. And therefore this also holds for the intersection of all saturated sets.

In fact it is easy to prove that the intersection of saturated sets is always saturated. This implies that the interpretation of a type is always a saturated set, including the one from the exercise.

Therefore $[\![\forall a.\, a]\!]_\rho$ is not empty, as it contains all variables.