2.7.1 — Foundations of Proof Systems

Exam
Nov. 27th 2023

*Durée de l’épreuve : 2 heures.* Length of the exam : 2 hours.

1 HOL

For conciseness I write $\forall X^T$ ... instead of $(\forall_T \lambda X^T ...)$.

**Question 1** Given two propositions $A$ and $B$ in HOL, what do the following propositions correspond to? (in natural language)

1. $\forall X^o. A \implies X^o$
2. $\forall X^o. (A \implies B \implies X^o) \implies X^o$
3. $\forall X^o. (A \implies X^o) \implies X^o$
4. $\forall X^o. ((A \implies B) \implies X^o) \implies ((B \implies A) \implies X^o) \implies X^o$

**Solution.**

1. $\neg A$
2. $A \land B$
3. equivalent to $A$
4. $(A \implies B) \lor (B \implies A)$

**Question 2** Same question for the following constructions, given a property $P : i \to o$ and a relation $R : i \to i \to o$.

1. $\forall x^i. \forall y^i. (R x^i y^i) \implies (R y^i x^i) \implies \forall Q^{i\rightarrow o}. (Q x^i) \implies (Q y^i)$
2. $\forall X^o. (x^i. (P x^i) \implies X^o) \implies X^o$
3. $\lambda a^i. \lambda b^i. \forall X^{i\rightarrow o}. (X^{i\rightarrow o} a^i) \implies (\forall x^i. \forall y^i. (X^{i\rightarrow o} x^i) \implies (R x^i y^i) \implies (X^{i\rightarrow o} y^i)) \implies (X^{i\rightarrow o} b^i)$

**Solution.**

1. $R$ is anti-symmetrical
2. $\exists x. P(x)$
3. The transitive closure of $R$
2 System F

We use the usual encoding of natural numbers in System F as Church Numerals of the following type:

\[ \text{nat} \equiv \forall X. X \to (X \to X) \to X \]

**Question 3** Define the type \( NN \) which encodes the pairs of natural numbers, as well as the corresponding terms:

\[
\begin{align*}
pair : & \quad \text{nat} \to \text{nat} \to \text{NN} \\
\pi_1 : & \quad \text{NN} \to \text{nat} \\
\pi_2 : & \quad \text{NN} \to \text{nat}
\end{align*}
\]

**Solution.**

\[
\begin{align*}
\text{NN} & \equiv \forall X. (\text{nat} \to \text{nat} \to X) \to X \\
pair & \equiv \lambda a b : \text{nat}. \lambda f : \text{nat} \to \text{nat}. f a b
\end{align*}
\]

**Question 4** Define the term \( s : \text{NN} \to \text{NN} \) corresponding to the function \((n, m) \mapsto (S n, n)\).

**Solution.**

\[ s \equiv \lambda c : \text{NN}. \text{pair} (S (\pi_1 c)) (\pi_1 c) \]

**Question 5** Use this to define a predecessor function over \( \text{nat} \).

**Solution.**

\[ \text{pred} \equiv \lambda n. \pi_1 (n \text{ NN} (\text{pair} 0 0) pp) \]

3 Lists in Type Theory

We start not in Type Theory, but in System T, that is simply-typed \( \lambda \)-calculus with the constants:

\[
\begin{align*}
0 : & \quad N \\
S : & \quad N \to N \\
R_T : & \quad T \to (N \to T \to T) \to N \to T \quad \text{(for any type} \ T) \\
\end{align*}
\]

and the usual reduction rules for \( R_T \).

**Question 6** Extend this with corresponding constructions for a type list\(_T\) of lists whose elements are of type \( T \) with constants \( \text{nil}_T \) and \( \text{const}_T \). You can call \( RL_T \) the recursion operator over these lists. Give the corresponding reduction rules.
Solution.

\[ \text{nil}_T : \text{list}_T \]  \hspace{1cm} (1)
\[ \text{cons}_T : T \rightarrow \text{list}_T \rightarrow \text{list}_T \]  \hspace{1cm} (2)
\[ \text{RL}_{T,U} : U \rightarrow (T \rightarrow \text{list}_T \rightarrow U \rightarrow U) \rightarrow \text{list}_T \rightarrow U \]  \hspace{1cm} (3)

and the reductions:

\[ (\text{RL}_{T,U} t_0 t_e \text{nil}_T) \triangleright t_0 \]  \hspace{1cm} (4)
\[ (\text{RL}_{T,U} t_0 t_e (\text{cons}_T u l)) \triangleright (t_e u l (\text{RL}_{T,U} t_0 t_e l)) \]  \hspace{1cm} (5)

**Question 7** Transpose this to Martin-Löf’s Type Theory (MLTT) by giving a dependent typing for this \( \text{RL}_T \) operator, so that it becomes an extension of MLTT.

\[ \text{Solution.} \] With \( P : N \rightarrow \text{Type} \),

\[ \text{RL}_{T,P} : (P \text{nil}_T) \rightarrow (\forall x : T. \forall l : \text{list}_T. (P (\text{cons}_T x l)) \rightarrow \forall l : \text{list}_T \rightarrow (P l). \]

Indepedently, we extend MLTT with an operator \( D : N \rightarrow \text{Type} \) with two reduction rules:

\[ (D 0) \triangleright \top \]
\[ (D (S t)) \triangleright \bot \]

**Question 8** Use this new operator to prove \( 0 =_N (S 0) \rightarrow \bot \) in this extension of MLTT.

\[ \text{Solution.} \] You can prove \( 0 =_N (S 0) \rightarrow (D 0) \rightarrow (D 1) \) and since \( (D 0) \) is provable, you get \( 0 =_N (S 0) \rightarrow (D 1) \) which is identical to \( 0 =_N (S 0) \rightarrow \bot \).

**Question 9** We now want to prove \( \forall x : T. \forall l : \text{list}_T. \text{nil}_T =_{\text{list}_T} (\text{cons}_T x l) \rightarrow \bot. \)

Do you need additional operator to prove this or can you do with the operator \( D \) of the previous question? How do you proceed?

\[ \text{Solution.} \] You do not need any new operator or extension. Just translate lists to numbers with

\[ tr \equiv \lambda l : \text{list}_T. \text{RL}_{T,N} 0 \lambda_\_ \lambda_\_ 1 l \]

and then use the previous question to prove \( tr \text{nil}_T \neq \text{cons}_T n l. \)

\[ \square \]

### 4 Surjective Pairing

One considers the following additional reduction rule for Martin-Löf’s Type Theory:

\[ (\pi_1(t), \pi_2(t)) \triangleright_{SR} t \]

This reduction rule is known as the *surjective pairing* reduction. Note that the rule is not linear (the two occurrences of \( t \) in the left hand part need to be identical).
**Question 10** Show that this rule enjoys the subject reduction property. That is, if $\Gamma \vdash (\pi_1(t), \pi_2(t)) : U$, then $\Gamma \vdash t : U$.

**Solution.** We remember that we have uniqueness of typing modulo conversion : if $\Gamma \vdash u : U_1$ and $\Gamma \vdash u : U_2$, then $U_1 = \beta U_2$.

If $(\pi_1(t), \pi_2(t)) : \Sigma x : A.B$ is well typed in $\Gamma$, then so is $\pi_1(t)$ and thus there exists $A$ and $B$ such that $\Gamma \vdash t : \Sigma x : A.B$.

Thus $\Gamma \vdash \pi_1(t) : A$ and $\Gamma \vdash \pi_2(t) : B[x \ \pi_1(t)]$.

Thus $\Gamma \vdash (\pi_1(t), \pi_2(t)) : \Sigma x : A.B$.

Thus $\Gamma \vdash t : \text{Type}$ and $T = \beta \Sigma x : A.B$, and thus $\Gamma \vdash t : T$.

## 5 Markov’s Principle

In this section, we work in Martin-Löf’s Type Theory (MLTT). We consider that $P$ is a predicate over natural numbers, that is an object of type $N \rightarrow \text{Type}$.

**Question 11** Show that, for at least some values of $P$, the proposition $\neg \neg (\Sigma n : N.P n) \rightarrow \Sigma n : N.P n$ is not provable in MLTT.

**Solution.** Take a variable $X : \text{Type}$ and $P \equiv \lambda x : N.X$. Then $\Sigma n : N.P n$ is equivalent to $X$ and the principle would entail $X + \neg X$ thus giving full classical logic.

The soviet mathematician Andrei Markov proposed a version of this proposition, weakened in order to preserve constructivity. He suggested to admit the axiom $\neg \neg (\Sigma n : N.P n) \rightarrow \Sigma n : N.P n$ but only for decidable properties, that is provided the following is provable : $\forall n : N.P n + \neg (P n)$. (Here + denotes the sum type operator in MLTT).

In other words, Markov proposed to accept the following axiom scheme, which is thus known as Markov’s principle :

$$(\forall n : N.P n + \neg (P n)) \rightarrow \neg \neg (\Sigma n : N.P n) \rightarrow \Sigma n : N.P n.$$ **Question 12** Explain informally why Markov’s principle can be constructive; that is how one could give evidence for Markov’s principle in Heyting’s semantics.

**Solution.** Evidence for (1) $\forall n : N.P n + \neg(P n)$ is a function giving evidence for $P n + \neg(P n)$ for any $n$.

Evidence for $\neg \neg (\Sigma n : N.P n)$ entails, classically, that there exists a number $\alpha$ for which $P \alpha$ is true.

So enumerating all natural numbers and checking (1) one will find $\alpha$ and a proof of $P \alpha$.

(For the record, it is possible, but difficult, to show that Markov’s principle is not provable in MLTT (or in Heyting’s arithmetic).)

One proposes to extend MLTT with a specific term corresponding to Markov’s principle in the Curry-Howard setting.

Given terms $P,d,p,n$, one has a new term $MP(d,p,n)$. One adds the following typing rule :
\[ \Gamma \vdash P : N \rightarrow \text{Type} \quad \Gamma \vdash d : \forall n : N. P n + \lnot(P n) \quad \Gamma \vdash p : \lnot\lnot(\exists n : N. P n) \]

\[ \Gamma \vdash \text{MP}_p(d, p, 0) : \exists n : N. P n \]

One suggests the following reduction rule:

\[ (R_{MP}) \quad \text{MP}_p(d, p, n) \quad \Rightarrow \quad \delta(d, n, x.\langle n, x \rangle, y.\text{MP}_p(d, p, \langle S n \rangle)) \]

Remember \( \delta \) is the elimination operator for sum types, that is logical disjunction.

**Question 13** Explain the idea behind this MP operator and this reduction rule. 

**Solution.** It is precisely what is described in the response to the previous question.

**Question 14** Show that this \( R_{MP} \) reduction rule is not strongly normalizable (or in other words, that MLTT with this reduction rule in not strongly normalizable). *This should be very short.*

**Solution.** The reduction rule can obviously be repeated infinitely:

\[
\begin{align*}
\text{MP}_p(d, p, 0) & \Rightarrow \delta(d, n, x.\langle n, x \rangle, y.\text{MP}_p(d, p, \langle S n \rangle)) \\
& \Rightarrow \delta(d, 0, x.\langle n, x \rangle, y.\delta(d, 1, x.\langle n, x \rangle, y.\text{MP}_p(d, p, \langle S n \rangle))) \\
& \Rightarrow \ldots
\end{align*}
\]

**Question 15** Show that the system with the \( R_{MP} \) reduction rule is not weakly normalizable either. *Hint: you may look at the next question to find the idea.*

**Solution.** If we are in an incoherent context with \( b : \bot \), then one can use \( b \) to prove \( \lnot\lnot\exists x : N. \bot \). Using the simple proof of \( \forall x : N. \bot + \lnot\bot \) which always returns the proof of \( \lnot\bot \), the operator will never find a witness and loop forever.

One therefore suggests the following restriction: *the \( R_{MP} \) reduction can only be performed when the terms \( d \) and \( p \) are closed (that is they contain no free variable).*

**Question 16** Sketch a proof of weak normalization for MLTT extended by this restricted \( R_{MP} \) reduction rule.

**Solution.** For any well-typed term \( t \), we call \( \#(t) \) the size of its normal form (for conventional reduction, that is without considering \( R_{MP} \)).

Suppose \( \Gamma \vdash t : T \). We show by induction over \( \#(t) \) that \( t \) has a normal form for the extended reduction.

We take the conventional normal form of \( t \). Suppose it contains a \( R_{MP} \) redex \( \text{MP}_p(d, p, 0) \) (the third argument of \( \text{MP} \) must be convertible to 0 because of the typing rule and because we have not performed any \( R_{MP} \) reduction). By induction hypothesis, we can normalize \( d \) and \( p \). We suppose thus that \( d \) and \( p \) are closed and normal.

We can thus argue that there must exists a closed term \( S^{(i)} 0 \) such that \( d \ S^{(i)} 0 \) reduces to some \( i(q) \) (because \( p \) is closed). Thus \( \text{MP}_p(d, p, 0) \) reduces to \( (S^{(i)} 0, q) \).