PRFSYS — Foundations of Proof Systems

Exam

Nov. 25^{th} 2025

1 Trees in system F

We suppose we have already defined the type nat of natural numbers together with 0 and S and the addition function over them.

Question 1 Define a type Tree representing binary trees with natural numbers at the nodes, and nothing at the leaves.

That is, define Tree and two terms :

— Leaf : Tree

— Node : nat → Tree → Tree → Tree. ◇

Solution.

$$\mathsf{Tree} \equiv \forall X.X \to (\mathsf{nat} \to X \to X \to X) \to X.$$

$$\mathsf{Leaf} \equiv \Lambda X.\lambda x : X.\lambda f : \mathsf{nat} \to X \to X \to X.x.$$

$$\mathsf{Node} \equiv \lambda n : \mathsf{nat}.\lambda t_1 : \mathsf{Tree}.\lambda t_2 : \mathsf{Tree}.\Lambda X.\lambda x : X.\lambda f : f\ n\ (t_1\ X\ x\ f)\ (t_2\ X\ x\ f).$$

Question 2 Construct a function count : Tree → nat which counts the number of nodes of a tree (without taking the leaves into account). ◇

Solution.

$$\mathsf{count} \equiv \lambda t : \mathsf{Tree}.t\ \mathsf{nat}\ 0\ (\lambda v\ n_1\ n_2.S(n_1 + n_2)).$$

Question 3 Construct function tsum : Tree → nat which returns the sum of the values at the nodes. ◇

Solution.

$$\mathsf{tsum} \equiv \lambda t : \mathsf{Tree}.t\ \mathsf{nat}\ 0\ \lambda v : \mathsf{nat}.\lambda n_1 : \mathsf{nat}.\lambda n_2 : \mathsf{nat}.v + n_1 + n_2.$$

2 Impredicative Definitions of Properties

We are in HOL. We consider as given the type nat with 0 and S, as well as a type list and two constants $\mathsf{nil} : \mathsf{list}$ and $\mathsf{cons} : \mathsf{nat} \to \mathsf{list} \to \mathsf{list}$.

Question 4 Define $\leq : \mathsf{nat} \to \mathsf{nat} \to o$ which states that a number is less or equal than another. $\diamond$

Solution.

$$(\leq\ a\ b) \equiv \forall P : \mathsf{nat} \to o.(P\ a) \to (\forall x : \mathsf{nat}.(P\ x) \to (P\ (Sx))) \to (P\ b).$$

One can also use a quantification over $R : \mathsf{nat} \to \mathsf{nat} \to o$ (it is a little longer but definitively ok). $\square$

Question 5 Define $\mathsf{low} : \mathsf{nat} \to \mathsf{list} \to o$ which states that either a number is less or equal than the first element of a list, or the list is empty. $\diamond$

Solution.

$$\begin{array}{l}(\mathsf{low}\ a\ l) \equiv \\ \quad \forall R : \mathsf{nat} \to \mathsf{list} \to o. \\ \qquad (\forall x : \mathsf{nat}.(R\ a\ \mathsf{nil})) \to \\ \qquad\qquad (\forall x\ y : \mathsf{nat}.\forall m : \mathsf{list}.(\leq\ x\ y) \to (R\ x\ (\mathsf{cons}\ y\ m))) \to \\ \qquad\qquad (R\ a\ l).\end{array}$$

Question 6 Define $\mathsf{sorted} : \mathsf{list} \to o$ which states that a list is sorted. $\diamond$

Solution.

$$\begin{array}{l}(\mathsf{sorted}\ l) \equiv \\ \quad \forall P : \mathsf{list} \to o. \\ \qquad (P\ \mathsf{nil}) \to \\ \qquad\quad (\forall x : \mathsf{nat}.\forall m : \mathsf{list}.(P\ m) \to (\mathsf{low}\ x\ m) \to (P\ (\mathsf{cons}\ x\ m))) \to \\ \qquad\qquad (P\ l).\end{array}$$

3 Exercises in Type Theory

Question 7 We are in Martin-Löf's Type Theory.

Fill in the blanks, so that the following statements ought to hold ; or say when they cannot hold.

For instance, for $[] \vdash 0 : \boxed{}$, you should answer N (it is not necessary to mention well-formed types convertible to N). You may want to add some remarks or side conditions, but do not go into long explanations.

$$
\begin{aligned}
[] &\vdash (a,b) : \boxed{} && (1)\\
\Gamma &\vdash \lambda x : A.t : \boxed{} && (2)\\
[] &\vdash \boxed{} : A + B && (3)\\
\Gamma &\vdash (\mathsf{refl}_A\ t) : \boxed{} && (4)\\
[] &\vdash \boxed{} : a =_A b && (5)\\
[(x : N)] &\vdash \boxed{} : N && (6)\\
[] &\vdash \boxed{} : \bot && (7)
\end{aligned}
$$

◇

Solution.

$$
\begin{aligned}
[] &\vdash (a,b) : \Sigma x : A.B && (8)\\
\Gamma &\vdash \lambda x : A.t : \Pi x : A\ .\ B && (9)\\
[] &\vdash i(a)\ \text{ or }\ j(b) : A + B && (10)\\
\Gamma &\vdash (\mathsf{refl}_A\ t) : t =_A t && (11)\\
[] &\vdash \mathsf{refl}_A(c) : a =_A b && (12)\\
[(x : N)] &\vdash S^{(i)}O\ \text{ or }\ S^{(i)}x : N && (13)\\
[] &\vdash \quad \text{no possible term}\quad : \bot && (14)
\end{aligned}
$$

□

4 A paradox in Type : Type

We consider the pure type system with a unique sort Type such that $\mathsf{Type} : \mathsf{Type}$. That is Martin-Löf's original, paradoxical type theory of 1971.

$$
\frac{}{[]\ \mathsf{wf}} \qquad \frac{\Gamma \vdash T : \mathsf{Type}}{\Gamma(x : T)\ \mathsf{wf}} \qquad \frac{\Gamma\ \mathsf{wf}}{\Gamma \vdash \mathsf{Type} : \mathsf{Type}} \qquad \frac{\Gamma\ \mathsf{wf}}{\Gamma \vdash x : T}\ (\text{If } (x : T) \in \Gamma)
$$

$$
\frac{\Gamma \vdash T_1 : \mathsf{Type} \quad \Gamma(x : T_1) \vdash T_2 : \mathsf{Type}}{\Gamma \vdash \Pi x : T_1.T_2 : \mathsf{Type}} \qquad \frac{\Gamma(x : U) \vdash t : T}{\Gamma \vdash \lambda x : U.t : \Pi x : U\ .\ T}
$$

$$
\frac{\Gamma \vdash t : \Pi x : U\ .\ T \quad \Gamma \vdash u : U}{\Gamma \vdash (t\ u) : T[x \setminus u]} \qquad \frac{\Gamma \vdash t : T \quad \Gamma \vdash U : \mathsf{Type}}{\Gamma \vdash t : U}\ (\text{if } T =_\beta U)
$$

Question 8 Describe a (very) simple transformation f over terms of PTSs, such that :

— if $\Gamma \vdash t : T$ in some PTS,

— then $f(\Gamma) \vdash f(t) : f(T)$ in the PTS above.

— With the requirement that if $t \rhd_\beta t'$ then $f(t) \rhd_\beta f(t')$. ◇

Solution. Replace all sorts in t by Type. □

Question 9 We define
$$\mathsf{U} \equiv \Pi X : \mathsf{Type}.X \to \mathsf{Type}.$$

What is the type of U? ◇

Solution. It is Type. □

Question 10 Given $u : \mathsf{U}$, how do you, simply, turn u into a property over U, that is a term of type $\mathsf{U} \to \mathsf{Type}$? ◇

Solution. Take $(u\ \mathsf{U})$. □

Question 11 Construct a relation $\in: \mathsf{U} \to \mathsf{U} \to \mathsf{Type}$.
From now on, we write $u \in v$ for $(\in\ u\ v)$. ◇

Solution.
$$\in\ \equiv\ \lambda u1 : \mathsf{U}\ .\ \lambda u_2 : \mathsf{U}\ .\ (u_2\ U\ u_1).$$

We also add to the type system the, usual, primitive equality, together with the additional axiom K which states unicity of equality proofs :

$$\begin{array}{rcll} = & : & \Pi X : \mathsf{Type}\ .\ X \to X \to X \to \mathsf{Type} & \text{we write } t =_T u \text{ for } (=\ T\ t\ u). \\ \mathsf{refl} & : & \Pi X : \mathsf{Type}\ .\ \Pi x : X\ .\ x =_X x & \\ L & : & \Pi X : \mathsf{Type}\ .\ \Pi x : X\ .\ \Pi y : X\ .\ \Pi P : X \to \mathsf{Type}\ .\ (P\ x) \to x =_X y \to (P\ y) & \\ K & : & \Pi X : \mathsf{Type}\ .\ \Pi x : X\ .\ \Pi P : x =_X x \to \mathsf{Type}\ .\ \Pi e : x =_X x\ .\ (P\ (\mathsf{refl}\ X\ x)) \to (P\ e) & \end{array}$$

with the (usual) reduction rules :

$$\begin{array}{rcl} (L\ T\ t\ t'\ P\ p\ (\mathsf{refl}\ T'\ t'')) & \rhd & p \\ (K\ T\ t\ P\ (\mathsf{refl}\ T'\ t')\ p) & \rhd & p \end{array}$$

Remark : you will not need the second reduction rule.

Using this equality, we will now build the construction dual to $\in$. That is construct an term $\mathsf{comp} : (\mathsf{U} \to \mathsf{Type}) \to \mathsf{U}$, such that $u \in (\mathsf{comp}\ P)$ will be equivalent to $P\ u$.

Question 12 Construct

$$\mathsf{tr} : \Pi X : \mathsf{Type}.\mathsf{U} =_{\mathsf{Type}} X \to (\mathsf{U} \to \mathsf{Type}) \to (X \to \mathsf{Type}).$$

What is a notable reduct of $(\mathsf{tr}\ \mathsf{U}\ (\mathsf{refl}\ \mathsf{U})\ P)$? ◇

Solution.

$$\mathsf{tr} \equiv \lambda X : \mathsf{Type}.\lambda e : \mathsf{U} =_{\mathsf{Type}} X\ .\ L\ (\lambda Y : \mathsf{Type}.Y \to \mathsf{Type})\ X\ \mathsf{U}\ P\ e.$$

We have :
$$(\mathsf{tr}\ \mathsf{U}\ (\mathsf{refl}\ \mathsf{U})\ P) \rhd P.$$

We can now define

$$\mathsf{comp} \equiv \lambda P : \mathsf{U} \to \mathsf{Type} \,.\, \lambda X : \mathsf{Type} \,.\, \lambda x : X \,.\, \Pi e : \mathsf{U} =_{\mathsf{Type}} X \,.\, \mathsf{tr}\ X\ e\ P\ x.$$

Question 13 Give a proof $\mathsf{trid} : \Pi e : \mathsf{U} =_{\mathsf{Type}} \mathsf{U} \,.\, \mathsf{tr}\ \mathsf{U}\ e\ P = P.$ ◇

Solution.

$$\lambda e : \mathsf{U} =_{\mathsf{Type}} \mathsf{U} \,.\, K\ (\lambda e : \mathsf{U} =_{\mathsf{Type}} \mathsf{U} \,.\, \mathsf{tr}\ \mathsf{U}\ e\ P = P)\ (refl\ P).$$

Question 14 Give a term :

$$g : \Pi P : \mathsf{U} \to \mathsf{Type}.\Pi u : \mathsf{U} \,.\, P\ u \to (u \in (\mathsf{comp}\ P))$$

Solution.

$$\begin{array}{l} g \equiv \\ \ \lambda P : \mathsf{U} \to \mathsf{Type}.\lambda u : \mathsf{U}.\lambda p : P\ u.\lambda e : U = U. \\ L\ (U \to \mathsf{Type})\ (P)\ (\mathsf{tr}\ U\ e\ P)\ (\lambda Q : \mathsf{U} \to \mathsf{Type}.Q\ u)\ p\ (\mathsf{trid}\ e) \end{array}$$

(actually we should insert a use of symmetry of equality, to turn it into a proof of $P = (\mathsf{tr}\ \mathsf{U}\ e\ P)$; the blame of this mishap is on me, I should have typed trid the other way around)

Question 15 Given $u : \mathsf{U}$, $P : \mathsf{U} \to \mathsf{Type}$ and $i : (In\ u\ (\mathsf{comp}\ P))$, give a term $t : (P\ u)$. ◇

Solution. We have :

$$(In\ u\ (\mathsf{comp}\ P)) \rhd \Pi e : \mathsf{U} =_{\mathsf{Type}} .\mathsf{tr}\ \mathsf{U}\ e\ P\ u$$

so

$$i\ (\mathsf{refl}\ \mathsf{U}) : \mathsf{tr}\ \mathsf{U}\ (\mathsf{refl}\ U)\ P\ u$$

and thus also :

$$i\ (\mathsf{refl}\ \mathsf{U}) : P\ u.$$

We now give ourselves a variable $\alpha : \mathsf{Type}$. We write $\neg^{\alpha} T$ for $T \to \alpha$.

Question 16 Define R of type U which corresponds to Russell's paradoxical set $\{x | \neg^{\alpha}(x \in x)\}$.

Show $\neg^{\alpha}(\mathsf{R} \in \mathsf{R})$ ◇

Solution. Take

$$\mathsf{R} \equiv \mathsf{comp}\ \lambda x : \mathsf{U}.\neg^{\alpha}(x \in x).$$

Question 15 gives us a term $h : (\mathsf{R} \in \mathsf{R}) \to ((\lambda : x : \mathsf{U}.\neg^{\alpha}(x \in x))\ \mathsf{R})$
that is $h : (\mathsf{R} \in \mathsf{R}) \to \neg^{\alpha}(\mathsf{R} \in \mathsf{R})$.
So we have :

$$h_0 \equiv \lambda p : \mathsf{R} \in \mathsf{R} \,.\, (h\ p\ p) : (\mathsf{R} \in \mathsf{R}) \to \alpha.$$

Question 17 Show $(\mathsf{R} \in \mathsf{R})$. Deduce α. $\diamond$

Solution. We use the term g of question 14.

Take

$$g_0 \equiv (g\ (\lambda : x : \mathsf{U}.\neg^{\alpha}(x \in x))\ \mathsf{R}\ h_0) : \mathsf{R} \in \mathsf{R}.$$

So $(h_0\ g_0)$ is of type α.

Question 18 Explain, from this, why the type system cannot enjoy normalization. $\diamond$

Solution. We have a closed proof of α. If we have normalization, we can have a closed normal proof of α, or to be precise, a term of type α whose only free variable s α (or a closed normal proof of $\Pi x : \mathsf{Type}\ .\ X$. By enumerating the possible closed normal forms, we see non can be a proof of (that is a term of type) $\Pi x : \mathsf{Type}\ .\ X$ (or of type α).

The term cannot be of the forms :

- $\lambda x : T.u$ (it would be a function type)
- $\Pi y : A.B$ (it would be of type Type)
- $(x\ u_1\ \ldots\ u_n)$ (x would need to be α, which would lead to the correct type),
- Type
- same arguments with a little more reasoning for the operators of the primitive equality. $\square$

Fixed-Point Operator

In this last part, we want to go a little further and build a real fixed-point operator. For that, for any $x : \mathsf{U}$ and $P : \mathsf{U} \to \mathsf{Type}$, we admit that we can construct two terms :

$$\begin{array}{rcl} F_P^x & : & P\ x \to x\ \in\ (\mathsf{comp}\ P) \\ G_P^x & : & x\ \in\ (\mathsf{comp}\ P) \to P\ x \end{array}$$

with the property that $(G_P^x\ (F_P^x\ p)) \triangleright_\beta^* p$.

We write :

$$\begin{array}{rcl} P & \equiv & \lambda x : \mathsf{U}\ .\ \neg^{\alpha}(x \in x) \\ F_0 & \equiv & F_P^{\mathsf{R}} : \neg^{\alpha}(\mathsf{R} \in \mathsf{R}) \to \mathsf{R} \in \mathsf{R} \\ G_0 & \equiv & G_P^{\mathsf{R}} : \mathsf{R} \in \mathsf{R} \to \neg^{\alpha}(\mathsf{R} \in \mathsf{R}) \end{array}$$

Question 19 Using G_0 and F_0, construct two terms of type :

$$\begin{array}{rcl} H_1 & : & \neg^{\alpha}(\mathsf{R} \in \mathsf{R}) \\ H_2 & : & \mathsf{R} \in \mathsf{R} \end{array}$$

What does $(H_1\ H_2)$ reduce to ? $\diamond$

Solution. Take :

$$\begin{array}{rcl} H_1 & \equiv & \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p) \\ H_2 & \equiv & (F_0\; H_1) \end{array}$$

We see that

$$\begin{array}{rcl} H_1\; H_2 & = & \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p)\;\; (F_0\; \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p)) \\ & \rhd & (G_0\; (F_0 \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p))\; (F_0 \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p))) \\ & \rhd^* & (\lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p)(F_0\; \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p))) \\ & = & H_1\; H_2 \end{array}$$

□

Question 20 We give ourselves a variable $f : \alpha \to \alpha$. By modifying the terms of the previous question, build a term $Y : \alpha$, such that $Y \rhd^*_\beta f\; Y$. ◇

Solution. Take $H_1' \equiv \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (f\; (G_0\; p\; p))$.

We have

$$\begin{array}{rcl} H_1'\; H_2 & = & \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (f\; (G_0\; p\; p))\;\; (F_0\; H_1') \\ & \rhd & f\; (G_0\; (F_0 \lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (f\; (G_0\; p\; p)))\; (F_0\; H_1')) \\ & \rhd^* & f\; (\lambda p : (\mathsf{R} \in \mathsf{R}) \,.\, (G_0\; p\; p)(F_0\; H_1')) \\ & = & f\; (H_1'\; H_2) \end{array}$$

□

Question 21 Show that all λ-terms are typable in this type system. ◇

Solution. The construction above is for any typeα and any f.

So we can instantiate α with Type (since $\mathsf{Type} : \mathsf{Type}$) and f by $\lambda X : \mathsf{Type}.X \to X$.

So take $\omega \equiv (H_1'\; H_2)$. We have $\omega : \mathsf{Type}$ and $\omega =_\beta \omega \to \omega$.

This allows to type any λ-term. □