Lambda-Calculus and Type Theory
ISR 2024
Obergurgl, Austria
Herman Geuvers & Niels van der Weide
Radboud University Nijmegen NL
Exercises Day 3

Lecture 6. Polymorphic types

1. Recall: $\bot := \forall\alpha.\, \alpha$, $\top := \forall\alpha.\, \alpha \to \alpha$.
   (a) Verify that in Church $\lambda 2$: $\lambda x : \top.\, x \top x : \top \to \top$.
   (b) Verify that in Curry $\lambda 2$: $\lambda x.\, x\, x : \top \to \top$
   (c) Find a type in Curry $\lambda 2$ for $\lambda x.\, x\, x\, x$
   (d) Find a type in Curry $\lambda 2$ for $\lambda x.\, (x\, x)(x\, x)$
   (e) Find a type in Curry $\lambda 2$ for $\lambda z.\, z(\lambda x.\, x\, x)$

2. Let $x : \top$ and remember that $\top := \forall\alpha : *.\, \alpha \to \alpha$.
   (a) Give a type to the term
   $$\lambda y.\, x\, y\ x(\lambda z.\, z\, x\, z)$$
   in $\lambda 2$ à la Curry and give the typing derivation of your result.
   (b) Give a type to the term
   $$\lambda y.\, x\, y\, (x(\lambda z.\, z\, z))$$
   in $\lambda 2$ à la Curry. Also give the typing derivation of your result.

3. Define:
   $$\begin{array}{rcl} \sigma \times \tau & := & \forall\alpha.\, (\sigma \to \tau \to \alpha) \to \alpha, \\ \sigma + \tau & := & \forall\alpha.\, (\sigma \to \alpha) \to (\tau \to \alpha) \to \alpha \end{array}$$
   (a) Define $\mathrm{inl} : \sigma \to \sigma + \tau$
   (b) Define pairing : $[-,-] : \sigma \to \tau \to \sigma \times \tau$
   (c) Define the first projection : $\pi_1 : \sigma \times \tau \to \sigma$ and show that $\pi_1[x, y] =_\beta x$.

4. Define the type of binary tress with leaves in B and node labels in A:
   $$\mathrm{Tree}_{A,B} \quad := \quad \forall\alpha.(B \to \alpha) \to (A \to \alpha \to \alpha \to \alpha) \to \alpha.$$
   (a) Define $\mathrm{leaf} : B \to \mathrm{Tree}_{A,B}$ and $\mathrm{join} : \mathrm{Tree}_{A,B} \to \mathrm{Tree}_{A,B} \to A \to \mathrm{Tree}_{A,B}$.
   (b) Give the Tree-iteration scheme for $\mathrm{Tree}_{A,B}$ and define $h : \mathrm{Tree}_{A,B} \to \mathrm{Nat}$ that counts the number of leaves of a tree.
   (c) Define $g : \mathrm{Tree}_{A,B} \to B$ that computes the left-most leaf of a tree.

Lecture 7. Higher order logic in the Calculus of constructions and in Coq

NB. The exercises can also be made with Coq. Please look at the web site for the `HOL_inductivetypes.v` file.

1. Definition in CC (for $t, q : A$):

$$t =_A q := \Pi P : A \to *. (Pt \to Pq)$$

   (a) (basic) Prove that this equality is reflexive by giving a term of type $\Pi x : A. x =_A x$.

   (b) (basic) Prove that this equality is transitive by giving a term of type $\Pi x, y, z : A. x =_A y \to y =_A z \to x =_A z$.

   (c) (advanced) Prove that this equality is symmetric by giving a term of the type $\Pi x, y : A. x =_A y \to y =_A x$.

2. The transitive closure of a binary relation R on A has been defined as follows.

$$\begin{aligned} \mathsf{trclos}\, R \quad := \quad & \lambda x, y : A. \\ & (\forall Q : A \to A \to *. (\mathsf{trans}\, Q \to (R \subseteq Q) \to (Q\, x\, y))). \end{aligned}$$

   (a) (basic) Prove – by giving a proof-term – that the transitive closure of R contains R.

   (b) (medium) Prove – by giving a proof-term – that the transitive closure is transitive.

   (c) (basic) Prove – by giving a proof-term – that, if P is transitive and P contains R, then P contains $\mathsf{trclos}\, R$.

3. The existential quantifier has been defined by

$$\exists x : \sigma. \phi := \forall \alpha : *. (\forall x : \sigma. \phi \to \alpha) \to \alpha$$

   (a) (medium) Given $t : \sigma$ and $q : P\, t$, give a term M such that $M : \exists x : \sigma. P\, x$

   (b) (medium) Given $q : \exists x : \sigma. P\, x$ and $h : \forall y : \sigma. P\, y \to C$ with $y \notin \mathrm{FV}(C)$, give a term N of type C.

4. For $D : *$, $A, B : D \to *$, we define $A \subseteq B$ as $\forall x : D. A\, x \to B\, x$. We now define

$$\begin{aligned} A \cap B \quad &:= \quad \lambda x : D. \forall P : D \to *. (\forall y : D. A\, y \to B\, y \to P\, y) \to P\, x \\ A \cup B \quad &:= \quad \lambda x : D. \forall P : D \to *. A \subseteq P \to B \subseteq P \to P\, x \end{aligned}$$

   Prove the following, by giving a (proof) term of the type. Remember that $X \vee Y$ is defined as $\forall \alpha : *. (X \to \alpha) \to (Y \to \alpha) \to \alpha$.

(a) $A \subseteq A \cup B$.

(b) (This is a hard question) $\forall x : D. (A \cup B)\, x \to A\, x \vee B\, x$.

(c) $A \cap B \subseteq A$.

(d) $\forall x : D. A\, x \to B\, x \to (A \cap B)\, x$.