Proving with Computer Assistance, 2IMF15

Herman Geuvers, TUE

Exercises on Polymorphic type Theory

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:

   $$\begin{array}{lcl} \sigma \times \tau & := & \forall\alpha.\,(\sigma{\to}\tau{\to}\alpha){\to}\alpha, \\ \sigma + \tau & := & \forall\alpha.\,(\sigma{\to}\alpha){\to}(\tau{\to}\alpha){\to}\alpha \end{array}$$

   (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:

   $$\mathbb{T}_{A,B} \quad := \quad \forall\alpha.(B{\to}\alpha){\to}(A{\to}\alpha{\to}\alpha{\to}\alpha){\to}\alpha.$$

   (a) Define $\mathsf{leaf} : \mathsf{B} \to \mathbb{T}_{\mathsf{A},\mathsf{B}}$ and $\mathsf{join} : \mathbb{T}_{\mathsf{A},\mathsf{B}} \to \mathbb{T}_{\mathsf{A},\mathsf{B}} \to \mathsf{A} \to \mathbb{T}_{\mathsf{A},\mathsf{B}}$.

   (b) Give the Tree-iteration scheme for $\mathbb{T}_{A,B}$ and define $h : \mathbb{T}_{A,B} \to \mathbb{N}$ that counts the number of leaves of a tree. (You may assume $\mathsf{Plus} : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$ to be defined.)

   (c) Define $g : \mathbb{T}_{A,B} \to B$ that computes the left-most leaf of a tree.

(d) Now take $A, B : \mathbb{N}$. Define $\mathsf{Sum} : \mathbb{T}_{\mathbb{N},\mathbb{N}} \to \mathbb{N}$ that takes the sum of all leaves and nodes in a tree. (You may assume $\mathsf{Plus} : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$ to be defined.)

5. The type of booleans $\mathbb{B}$ and the type of lists over A, $\mathbb{L}_A$ are defined by

$$\begin{aligned} \mathbb{B} &:= \forall\alpha{:}*.\alpha{\to}\alpha{\to}\alpha, \\ \mathbb{L}_A &:= \forall\alpha{:}*.\alpha{\to}(A{\to}\alpha{\to}\alpha){\to}\alpha. \end{aligned}$$

(a) Define true and false of type $\mathbb{B}$ and define the boolean connectives $\wedge$, $\vee$ and $\neg$ on $\mathbb{B}$.

(b) Define the function $E : \mathbb{L}_A{\to}\mathbb{B}$ that computes if a list is empty.

(c) Define the function $N : \mathbb{L}_{\mathbb{B}}{\to}\mathbb{L}_{\mathbb{B}}$ that negates all elements in the input list.