Proofs and Programs

TD 10 - Revisions

Philippe Audebaud, Aurore Alcolei

12 april 2018

Exercice 1. (Warming up)

1. Inhabit the following types in $\lambda_{\to,\times}$:

$$X \to (X \to R) \to R \qquad A \times ((B \to R) \to R) \to (A \times B \to R) \to R$$

2. Recall the encoding of binary trees with element of type A in system F. How could you generalize it to n-ary trees? to infinite branching trees?

Exercice 2 (Typing with type algebra)**.** In this exercise we give more power to the simple type system by considering types up to some congruence $\equiv$. For example, we can have equivalence of the form $A \equiv A \to B$. Typing rules remains unchanged but a type can be replaced by an equivalent type at any point of the derivation. In this system, type derivations are denoted $\Delta \vdash_{\equiv} t : A$.

- Show that if $A \equiv A \to B$ then $\vdash_{\equiv} \Omega : B$. (Hint : first show that $\vdash_{\equiv} \lambda\, x.xx : A$)
- Show that if $\equiv$ is a congruence for $\to$ then the subject reduction property holds. You can use the generation lemma associated to simply typed lambda-calculus.

Exercice 3 (Existential in System F)**.** In propositional second order intuitionistic logic the existential quantifier is introduced and destructed via the following (annotated) rules:

$$\frac{\Delta \vdash t : T\langle S/X\rangle}{\Delta \vdash [S,t]_{\exists X.T} : \exists X.T}\ (\exists I) \qquad \frac{\Delta \vdash t : \exists X.T \quad \Delta, x : T \vdash s : B \quad X \notin \mathrm{FV}(\Delta, B)}{\Delta \vdash \mathrm{let}\ [X, x : T] = t\ \mathrm{in}\ s : B}\ (\exists E)$$

From a logical point of view existentials can be seen as infinite disjunction, from a programming point of view they can be interpreted as an encapsulation mechanism.

1. Find an appropriate type representation for the existential in System F, with an encoding for $[\_,\_]_\_$ and let ... in .... Check that it validates the corresponding β rule.
2. Recall the encoding of NJ in System F and deduce that second order propositional intuituinistic logic is representable in System F.
3. In programming, streams are co-inductive datatypes with two accessors:

$$\mathrm{hd}\ : \mathrm{Str}_A \to A \qquad \mathrm{tl}\ : \mathrm{Str}_A \to \mathrm{Str}_A$$

   and a building function $\mathrm{build} : (A \to B) \to \mathrm{Str}_A \to \mathrm{Str}_B$ such that

$$\mathrm{hd}\ (\mathrm{build}\ f\ s) = f(\mathrm{hd} s) \qquad \mathrm{tl}\ (\mathrm{build}\ f\ s) = \mathrm{build}\ f\ (\mathrm{tl} s)$$

   What could be an encoding of Str_A in System F?

4. Define the function $\mathrm{nth} : \mathrm{Nat} \to \mathrm{Str}_A \to A$ that returns the n^{th} element of a stream.

Exercice 4 (Final 2017 – Equivalence lifting)**.** In HoTT, proofs can become involved. The goal is to prove that if $f : A \to B$ is an equivalence between A and B, then for each pair of elements $a, a' : A$, the map $\mathbf{ap}_{f,a,a'} : a =_A a' \to f(a) =_B f(a')$ is an equivalence as well. Let $g : B \to A$ being an of f meaning that there are witnesses $\alpha : \Pi_{x:A} g(f\ x) =_A \mathrm{id}_A\ x$ and $\beta : \Pi_{b:B} f\ (g\ x) =_B \mathrm{id}_B\ x$ In the sequell we will left the subscript a, a' in $\mathbf{ap}_f$ implicit.

Question 1 As a quasi-inverse candidate for ap_f, let us consider $G(\cdot)$, defined by

$$G(q) \equiv \alpha(a)^{-1} \cdot \mathsf{ap}_g(q) \cdot \alpha(a') \tag{1}$$

To satisfy the requirement, we have to exhibit homotopies γ (as left inverse) and δ (as right inverse):

$$\gamma : \prod_{p:J} G(\mathsf{ap}_f(p)) =_J p \quad \text{et} \quad \delta : \prod_{q:K} \mathsf{ap}_f(G(q)) =_K q$$

a) What are the types J and K? What is the type of the candidate $G(\cdot)$?

b) Prove the existence of a witness γ.

c) Why is that not possible to use a similar approach to prove the existence of δ?

Question 2. Let $T : \mathcal{U}$ and $\varphi : T \to T$ such that $\varepsilon : \prod_{x:T} \varphi(x) =_T \mathsf{id}_T(x)$.

a) Given $x, x' : T$ and $r : x =_T x'$, prove that $\varepsilon(x)^{-1} \cdot \mathsf{ap}_\varphi(r) \cdot \varepsilon(x') =_S r$, where the type S will be made explicit.

b) Conclude that, for all $x : T$, $\varepsilon(\varphi(x)) = \mathsf{ap}_\varphi(\varepsilon(x))$.

Question 3. Given $x : A$, let us note $\nu(x) \equiv \beta(f(x))^{-1} \cdot \beta(f(x))$.

a) State the type of $\nu(\cdot)$.

b) Prove that $\beta(f(a))^{-1} \cdot \mathsf{ap}_f(\mathsf{ap}_g(q)) \cdot \beta(f(a')) =_K q$.

c) Simplify the path $\nu(a) \cdot \mathsf{ap}_f(G(q)) \cdot \nu(a')$.

d) Conclude for the existence of an homotopy proof δ such that

$$\delta \quad : \quad \prod_{q:K} \mathsf{ap}_f(G(q)) =_K q$$

Exercice 5 (More on equivalences)**.** Prove the following statements:

1. (identity) For all $A : \mathcal{U}$, $\mathsf{quasi\text{-}inverse}\,(\mathsf{id}_A)$;
2. (between identity types) For all $A : \mathcal{U}$, $x, y : A$ and $p : x =_A y$,
   - $(p \cdot -) : y = z \to x = z$ and $(p^{-1} \cdot -)$ are quasi-inverse one of the other ;
   - $(- \cdot p) : z = x \to z = y$ et $(- \cdot p^{-1})$ are quasi-inverse one of the other.
3. (transport) If $P : A \to \mathcal{U}$, then $\mathsf{tr}^P(p, -) : P(x) \to P(y)$ has $\mathsf{tr}^P(p^{-1}, -)$ for a quasi-inverse.

Exercice 6 (Barendregt natural numbers)**.** Back to pure λ-calculus The Barendregt natural numbers $\lceil n \rceil$ ($n \in \mathbb{N}$) are defined by:

$$\lceil 0 \rceil \equiv \mathsf{I} \qquad \lceil n+1 \rceil \equiv (\mathsf{pair}\ \mathsf{F}\ \lceil n \rceil)$$

a) Using the alternative representation, code the successor, predecessor and test-to-zero functions.

b) Implement the addition.

c) In your understanding, how to the two natural numbers encodings (Church vs Barendregt) compare ?