Course Lambda calculus: Answers exam 4.6.2012.

1. Let A be a Π^0_2 statement of arithmetic, i.e. A is of the form

$$\forall x \exists y. P(x, y)$$

with x, y ranging over $\mathbb{N}$ and P (primitive) recursive. For example Goldbach's conjecture

"Every even number > 2 is the sum of two primes"

is such a statement. Purpose of this exercise is to construct λ-terms M_A, M_{true} such that

$$\text{BT}(M_A) = \text{BT}(M_{\text{true}}) \iff A \text{ is valid} \tag{1}$$

(a) Let K_P be the characteristic function of P. Construct a closed λ-term H such that

$$\begin{array}{rcll} H\mathbf{c}_n\mathbf{c}_m & = & \mathbf{c}_0 & \text{if } K_P(n,m) = 1 \\ & = & H\mathbf{c}_n(\mathbf{c}_{m+1}) & \text{else.} \end{array}$$

What is $\text{BT}(H\mathbf{c}_n\mathbf{c}_0)$, depending on n?

Answer. As P is a recursive predicate, its characteristic function K_P is λ-definable, by say F_P.

$$\begin{array}{rcll} F_P\mathbf{c}_n\mathbf{c}_m & = & \mathbf{c}_1, & \text{if } P(n,m), \\ & = & \mathbf{c}_0, & \text{else.} \end{array}$$

Taking $F'_P = \lambda nm.F_P nm(\mathsf{KK})(\mathsf{KI})$ we get

$$\begin{array}{rcll} F'_P\mathbf{c}_n\mathbf{c}_m & = & \mathsf{K}, & \text{if } P(n,m), \\ & = & \mathsf{KI}, & \text{else.} \end{array}$$

By the fixed point theorem there exists a term H such that

$$H = \lambda nm.F'_P nm\mathbf{c}_0(Hn(m+1)),$$

where $m + 1 = \lambda fx.f(mfx)$. Indeed we kan take

$$H = \mathsf{Y}(\lambda hnm.F'_P nm\mathbf{c}_0(hn(m+1)).$$

Then

$$\begin{array}{rcll} H\mathbf{c}_n\mathbf{c}_0 & = & \mathbf{c}_0, & \text{if } P(n,0), \\ & = & H\mathbf{c}_n\mathbf{c}_1, & \text{else}, \\ & = & \mathbf{c}_0, & \text{if } P(n,1), \\ & = & H\mathbf{c}_n\mathbf{c}_2, & \text{else}, \\ & = & \ldots & \end{array}$$

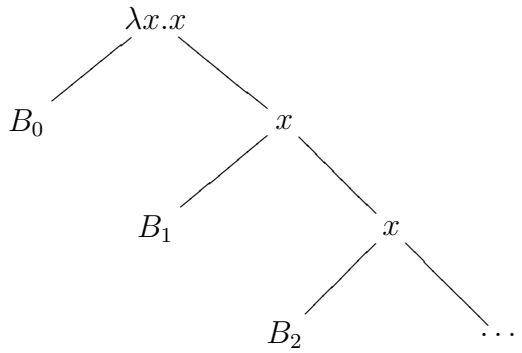
Therefore

$$\begin{array}{rcll} \mathrm{BT}(H\mathbf{c}_n\mathbf{c}_0) & = & \mathrm{BT}(\mathbf{c}_0), & \text{if } \exists m.P(n,m), \\ & = & \bot & \text{if } \neg\exists m.P(n,m). \end{array}$$

(b) Construct a closed λ-term G

$$\begin{array}{rcl} G\mathbf{c}_n x & = & x(H\mathbf{c}_n\mathbf{c}_0)(G\mathbf{c}_{n+1}x) \\ M_A & = & G\mathbf{c}_0 \end{array}$$

Answer. The term G can be constructed by the fixed point theorem. Note that $\mathrm{BT}(M_A)$ is



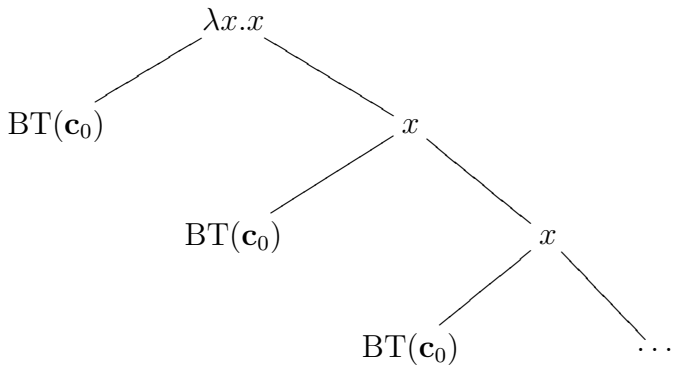
where $B_n = \mathrm{BT}(H\mathbf{c}_n\mathbf{c}_0)$.

(c) Construct a λ-term M_{true} such that (1) holds and prove this.

Answer. Using the fixed point theorem there exists a term

$$M_{\mathrm{true}} = \lambda x.x\mathbf{c}_0(M_{\mathrm{true}}x).$$

Then $\mathrm{BT}(M_{\mathrm{true}})$ is



By (a) and (b) one has

$$\mathrm{BT}(M_A) = \mathrm{BT}(M_{\mathrm{true}}) \iff A \text{ is valid}$$

2. For simple types A define $[[A]]$ as follows.

$$\begin{array}{rcl} [[0]] & = & SN = \{M \in \Lambda \mid M \text{ is strongly normalizing}\} \\ [[A \to B]] & = & \{F \in \Lambda \mid \forall N \in [[A]].FN \in [[B]]\} \end{array}$$

(a) Do we have for arbitrary $M \in \Lambda$ and simple types A

$$(\forall N.M \to_\beta N \Rightarrow N \in [[A]]) \Leftarrow M \in [[A]]?$$

Answer. Yes. We prove this by induction on A.

Case $A = 0$. Then $[[A]] = SN$. If $M \in [[A]]$, then M is SN. Let $M \to_\beta N$. Suppose N has an infinite reduction path; then so has M, quod non.

Case $A = B \to C$. Let $M \in [[A]]$. Then by definition of $[[A]]$

$$\forall P \in [[B]].MP \in [[B]].$$

Suppose $M \to_\beta N$. Then also $MP \to_\beta NP$. Hence

$$\forall P \in [[B]].NP \in [[B]],$$

by the Induction Hypothesis on B. Therefore $N \in [[A]]$.

(b) Do we have for arbitrary $M \in \Lambda$ and simple types A

$$(\forall N.M \to_\beta N \Rightarrow N \in [[A]]) \Rightarrow M \in [[A]]?$$

[Hint. Consider $M = \lambda x.xx$.]

Answer. Suppose the implication is correct. Then $M \in [[0 \to 0]]$, as M is in nf and therefore we have $(\forall N.M \to_\beta N \Rightarrow N \in [[0 \to 0]])$. Also $M \in [[0]]$ as it is SN. Therefore by definition $MM \in [[0]]$, quod non as it isn't normalizing: $M \to_\beta M$

3. We want to prove in Coq that $0 \neq 1$. In Coq we already have the following constants:

$$\begin{array}{rcl}
\mathsf{Prop} & : & \mathsf{Type} \\
\mathsf{False} & : & \mathsf{Prop} \\
\mathsf{True} & : & \mathsf{Prop} \\
\mathsf{I} & : & \mathsf{True} \\
\mathsf{not} & : & \mathsf{Prop} \to \mathsf{Prop} \\
\mathsf{eq} & : & \forall A : \mathsf{Type}.\, A \to A \to \mathsf{Prop} \\
\mathsf{refl_equal} & : & \forall A : \mathsf{Type}.\, \forall x : A.\, \mathsf{eq}\, A\, xx \\
\mathsf{eq_ind} & : & \forall A : \mathsf{Type}.\, \forall x : A.\, \forall P : A \to \mathsf{Prop}.\, Px \to \forall y : A.\, \mathsf{eq}\, A\, xy \to Py \\
\mathsf{nat} & : & \mathsf{Type} \\
\mathsf{O} & : & \mathsf{nat} \\
\mathsf{S} & : & \mathsf{nat} \to \mathsf{nat} \\
\mathsf{nat_rect} & : & \forall P : \mathsf{nat} \to \mathsf{Type}.\, P\,\mathsf{O} \to (\forall n : \mathsf{nat}.\, Pn \to P(\mathsf{S}\, n)) \to \forall n : \mathsf{nat}.\, Pn
\end{array}$$

and (among others) the following reduction rules:

$$\begin{array}{rcl}
\mathsf{nat_rect}\, PGH\mathsf{O} & \twoheadrightarrow_{\beta\delta\iota} & G \\
\mathsf{nat_rect}\, PGH(\mathsf{S}\, n) & \twoheadrightarrow_{\beta\delta\iota} & Hn\,(\mathsf{nat_rect}\, PGHn)
\end{array}$$

Finally not is defined as:

$$\mathsf{not} \;:=\; \lambda A : \mathsf{Prop}.\, A \to \mathsf{False}$$

Now define four Coq terms:

(a) A term $\mathsf{f\_equal}$ with type:

$$\mathsf{f\_equal} \;:\; \forall A\,B : \mathsf{Type}.\,\forall f : A \to B.\,\forall x\,y : A.\,\mathsf{eq}\,A\,x\,y \to \mathsf{eq}\,B\,(f\,x)(f\,y)$$

This says that if $x = y$ then also $fx = fy$.
(Hint: use $\mathsf{eq\_ind}$ with a well-chosen P.)

Answer. $\mathsf{f\_equal} := \lambda A\,B : \mathsf{Type}.\,\lambda f : A \to B.\,\lambda x : A.\;\mathsf{eq\_ind}\,A\,x\,(\lambda y : A.\,\mathsf{eq}\,B\,(f\,x)\,(f\,y))\,(\mathsf{refl\_equal}\,B\,(f\,x))$.

(b) A term g with type

$$g \;:\; \mathsf{nat} \to \mathsf{Prop}$$

that satisfies:

$$g\,(\mathsf{S}^n\,\mathsf{0}) \;=_{\beta\delta\iota}\; \begin{cases} \mathsf{True} & \text{if } n = 0 \\ \mathsf{False} & \text{otherwise} \end{cases}$$

Show that this term indeed satisfies the given equations.
(Hint: use $\mathsf{nat\_rect}$ with a well-chosen P.)

Answer. $g := \mathsf{nat\_rect}\,(\lambda n : \mathsf{nat}.\,\mathsf{Prop})\,\mathsf{True}\,(\lambda n : \mathsf{nat}.\,\lambda r : \mathsf{Prop}.\,\mathsf{False})$.

(c) A term h with type

$$h \;:\; \mathsf{eq}\,\mathsf{Prop}\,\mathsf{True}\,\mathsf{False} \to \mathsf{False}$$

This says that from $\mathsf{True} = \mathsf{False}$ it follows that False is inhabited.
(Hint: again use $\mathsf{eq\_ind}$, and use that I inhabits True.)

Answer. $h := \mathsf{eq\_ind}\,\mathsf{Prop}\,\mathsf{True}\,(\lambda x : \mathsf{Prop}.\,x)\,\mathsf{I}\,\mathsf{False}$.

(d) A term $\mathsf{neq\_0\_1}$ with type

$$\mathsf{neq\_0\_1} \;:\; \mathsf{not}\,(\mathsf{eq}\,\mathsf{nat}\,\mathsf{O}\,(\mathsf{S}\,\mathsf{O}))$$

This says that $0 \neq 1$.
(Hint: use the previous three terms. You do not need to spell them out again, but can just refer to them by name. If you didn't succeed in defining some of the previous terms, you are still allowed to use them in this final term.)

Answer. $\mathsf{neq\_0\_1} := \lambda H : \mathsf{eq}\,\mathsf{nat}\,\mathsf{O}\,(\mathsf{S}\,\mathsf{O}).\,h\,(\mathsf{f\_equal}\,\mathsf{nat}\,\mathsf{Prop}\,g\,\mathsf{O}\,(\mathsf{S}\,\mathsf{O})\,H)$.