Problem Sheet 5

Questions 4 and 6 will be marked.

The **pure, untyped λ-calculus** is the subset of PCF that does not include *any* constants. That is, *pure* terms are defined by the following grammar:

$$M, N ::= x \mid (M\,N) \mid (\lambda x.\,M)$$

In this problem sheet we are going to show that we can actually *define* a version of each PCF constant: Z, S, pred, ifz and fix, rather than considering them to be "baked-in" to the language. Each will be just an abbreviation for a *pure* λ-term.

The **Church numeral** for the number n, abbreviated $\ulcorner n \urcorner$, is:

$$\lambda f x.\, \underbrace{f(\cdots(f\,x)\cdots)}_{n\text{-times}}$$

In other words, the Church numeral for n is the pure term that takes a function f and an argument x and iterates f n-times on x. Let's list the first natural numbers, using Church numerals:

$$
\begin{aligned}
\ulcorner 0 \urcorner &= \lambda f x.\, x \\
\ulcorner 1 \urcorner &= \lambda f x.\, f x \\
\ulcorner 2 \urcorner &= \lambda f x.\, f(f x) \\
\ulcorner 3 \urcorner &= \lambda f x.\, f(f(f x)) \\
\ulcorner 4 \urcorner &= \lambda f x.\, f(f(f(f x))) \\
&\vdots
\end{aligned}
$$

We can define this formally (without the need to resort to $\cdots$), if we first define an auxilliary function iter : $\mathsf{Var} \times \mathsf{Var} \to \mathbb{N} \to \Lambda$ which takes a pair of variables and a natural number and delivers a pure term.

$$\text{iter}(f, x)(0) = x \tag{1}$$

$$\text{iter}(f, x)(m + 1) = f\ (\text{iter}(f, x)(m)) \tag{2}$$

We have, for example, $\mathrm{iter}(f, x)(3) = f\,(f\,(f\,x))$. Then we can define Church Numerals by:

$$\ulcorner n \urcorner = \lambda f x.\,\mathrm{iter}(f, x)(n)$$

So that we get $\ulcorner 3 \urcorner = \lambda f x.\,\mathrm{iter}(f\,x)(3) = \lambda f x.\,f\,(f\,(f\,x))$, as expected.

We are going to replace our numerical constants by abbreviations using pure terms that manipulate Church numerals. To get started, let's define:

$$\underline{Z} = \ulcorner 0 \urcorner$$
$$\underline{S} = \lambda n.\,\lambda f x.\,f\,(n\,f\,x)$$

** 1.

    (a) Using induction on n, prove that, for all $n \in \mathbb{N}$, $\ulcorner n \urcorner g\,y \;\approx\; \mathrm{iter}(g, y)(n)$.

    (b) Use this to show that, for all $n \in \mathbb{N}$, $\underline{S}\ulcorner n \urcorner \approx \ulcorner n+1 \urcorner$.

Solution ─────────────────────────────

    (a) By induction on $n \in \mathbb{N}$.

- When $n = 0$, $\ulcorner 0 \urcorner g\,y = (\lambda f x.\,x)\,g\,y \;\triangleright^*\; y = \mathrm{iter}(g, y)(0)$.
- When n is of shape $k + 1$, assume the induction hypothesis:

$$\ulcorner k \urcorner g\,y \;\approx\; \mathrm{iter}(g, y)(k)$$

Then reason equationally:

$$
\begin{aligned}
\ulcorner k+1 \urcorner g\,y &= (\lambda f x.\,\mathrm{iter}(f, x)(k+1))\,g\,y \\
&= (\lambda f x.\,f\,(\mathrm{iter}(f, x)(k)))\,g\,y \\
&\approx (\lambda f x.\,f\,(\ulcorner k \urcorner f\,x))\,g\,y \\
&\approx g\,(\ulcorner k \urcorner g\,y) \\
&\approx g\,(\mathrm{iter}(g, y)(k)) \\
&= \mathrm{iter}(g, y)(k+1)
\end{aligned}
$$

Where we use the induction hypothesis twice: in different directions in the third and fifth lines.

    (b) Let n be a natural number. We reason as follows:

$$
\begin{aligned}
\underline{S}\ulcorner n \urcorner &\approx \lambda f x.\,f\,(\ulcorner n \urcorner f\,x) \\
&\approx \lambda f x.\,f\,(\mathrm{iter}(f, x)(n)) \\
&= \lambda f x.\,\mathrm{iter}(f, x)(n+1) \\
&= \ulcorner n+1 \urcorner
\end{aligned}
$$

** 2. Construct a *pure* term $\underline{\mathsf{ifz}}$ satisfying, for all natural numbers $n \in \mathbb{N}$, all M, N:

$$\underline{\mathsf{ifz}}\ulcorner 0 \urcorner M N \approx M$$
$$\underline{\mathsf{ifz}}\ulcorner n+1 \urcorner M N \approx N$$

Solution ────────────────────────────────

$\underline{\mathsf{ifz}} = \lambda xyz. x (\lambda z_1. z) y$

** 3. The predecessor is somewhat more difficult to define as a pure term. We will need to recall that tuples and their projections, as we defined them, are themselves pure terms.

Consider the following Haskell program pred$'$ on natural numbers.

$$\mathsf{pred}'\ n = \mathsf{fst}\ (\mathsf{foldn}\ n\ \mathsf{incr}\ (0,0))$$

where

$$\begin{aligned}
\mathsf{incr}\ (n,0) &= (n,1) \\
\mathsf{incr}\ (n,1) &= (n+1,1)
\end{aligned}$$

$$\begin{aligned}
\mathsf{foldn}\ 0\ f\ x &= x \\
\mathsf{foldn}\ n\ f\ x &= f\ (\mathsf{foldn}\ (n-1)\ f\ x)
\end{aligned}$$

(a) What is the result of computing foldn 3 incr $(0,0)$?
(b) What is the result of computing foldn 4 incr $(0,0)$
(c) Following the same strategy as the Haskell program, construct a *pure* term $\underline{\mathsf{pred}}$ that satisfies:

$$\underline{\mathsf{pred}}\ulcorner 0 \urcorner \approx \ulcorner 0 \urcorner \quad \text{and} \quad \underline{\mathsf{pred}}\ulcorner k+1 \urcorner \approx \ulcorner k \urcorner$$

Hint: Compare the behaviour of foldn n and $\ulcorner n \urcorner$.

Solution ────────────────────────────────

(a) $(2,1)$
(b) $(3,1)$
(c) Let $\underline{\mathsf{incr}}$ be the term:

$$\lambda p. \underline{\mathsf{ifz}}\ (\underline{\mathsf{proj}}_2^2\ p)\ (\underline{\mathsf{proj}}_1^2\ p, \ulcorner 1 \urcorner)\ (\underline{\mathsf{S}}\ (\underline{\mathsf{proj}}_1^2\ p), \ulcorner 1 \urcorner)$$

Then we can define $\underline{\mathsf{pred}} = \lambda n. \underline{\mathsf{proj}}_1^2\ (n\ \underline{\mathsf{incr}}\ (\ulcorner 0 \urcorner, \ulcorner 0 \urcorner))$.

4. We have already seen in the previous problem sheet the pure term $\underline{Y}$ which can be used as a fixed point combinator in place of fix.

(a) Use this to define a *pure* term $\underline{\text{addcn}}$ which satisfies:

$$\underline{\text{addcn}} \ulcorner 0 \urcorner \ulcorner m \urcorner \approx \ulcorner m \urcorner$$

$$\underline{\text{addcn}} \ulcorner n+1 \urcorner \ulcorner m \urcorner \approx \underline{S} \left(\underline{\text{addcn}} \ulcorner n \urcorner \ulcorner m \urcorner \right)$$

(b) Prove, by induction on $n \in \mathbb{N}$ that, therefore, $\underline{\text{addcn}}$ satisfies:

$$\underline{\text{addcn}} \ulcorner n \urcorner \ulcorner m \urcorner \approx \ulcorner n+m \urcorner$$

Do not unpack your definition of $\underline{\text{addcn}}$. Instead, just use the two equations from part (a). As long as you used the recipe correctly, your definition will be guaranteed to satisfy these two equations - that is the whole point of following the recipe!

Solution ───

(a) $\underline{Y} (\lambda f \, x \, y. \underline{\text{ifz}} \, x \, y \, (\underline{S} (f \, (\underline{\text{pred}} \, x) \, y)))$

(b) By induction on n:

- When $n = 0$, our goal is to show $\underline{\text{addcn}} \ulcorner 0 \urcorner \ulcorner m \urcorner \approx \ulcorner 0+m \urcorner = \ulcorner m \urcorner$, and this is just the first equation given in (a).

- When n is of shape $k+1$, our goal is to show that

$$\underline{\text{addcn}} \ulcorner k+1 \urcorner \ulcorner m \urcorner \approx \ulcorner k+1+m \urcorner$$

Assume the induction hypothesis:

$$\underline{\text{addcn}} \ulcorner k \urcorner \ulcorner m \urcorner \approx \ulcorner k+m \urcorner$$

We reason equationally, using the second of the two equations given in (a), the induction hypothesis and the equation from Q1:

$$\begin{aligned} \underline{\text{addcn}} \ulcorner k+1 \urcorner \ulcorner m \urcorner &\approx \underline{S} \left(\underline{\text{addcn}} \ulcorner k \urcorner \ulcorner m \urcorner \right) \\ &\approx \underline{S} \ulcorner m+k \urcorner \\ &\approx \ulcorner m+k+1 \urcorner \end{aligned}$$

Equipped with our pure abbreviations for all of the constants, we are now in a position to define also a version of $\underline{\text{mult}}$ that works on Church numerals, a version of $\underline{\text{leq}}$ and so on. By doing so, we can build a translation from While programs to *pure* terms, following the same pattern as we have done for PCF terms, except that *λ-definability* will use Church numerals rather than PCF numerals. Then we will have shown that the pure λ-calculus is also Turing-complete.

The remaining two exercises are about undecidability.

4

5. Show that the set $\{\, M \in \Lambda \mid M \rhd^* \underline{1} \,\}$ is undecidable.

<hr/>

Solution

Let us call this set X. This set is non-trivial, since it contains $\underline{1}$ but it does not contain $\underline{2}$. It is also closed under convertibility, by the following reasoning. Suppose $M \in X$ and suppose $M \approx N$. Since $M \in X$, it must be that $M \rhd^* \underline{1}$ and hence $M \approx \underline{1}$. Hence, $N \approx \underline{1}$ too (by transitivity). So $N \in X$ too. Therefore, undecidability follows from the Scott-Curry Theorem.

6. Is the set $\{\, M \in \Lambda \mid \exists N.\ M\,N \approx \underline{0} \,\}$ decidable? Justify your answer.

<hr/>

Solution

No. Let us call this set Φ. Then Φ is non-trivial because:

- It contains $\lambda x.\underline{0}$.

- It does not contain $\lambda x.\underline{1}$: suppose that it did, then we would be able to deduce $\underline{0} \approx \underline{1}$, which is impossible.

Also, Φ is behavioural: suppose $P \in \Phi$ and (*) $Q \approx P$. We aim to show that $Q \in \Phi$ by finding a witnessing term N' (such that $Q\,N' \approx \underline{0}$). By definition of Φ, there is an N such that $P\,N \approx \underline{0}$ so we can take as our N' the very same N because, by equational reasoning using (*), $Q\,N \approx P\,N \approx \underline{0}$. Therefore, undecidability follows from the Scott-Curry Theorem.