M584 - Problem Set 2

Ambrose Bonnaire-Sergeant & Caner Derici

1) x1B.16 For *restriction*, let $F(u, n+1)$ be the function that constructs u up to the n^{th} factor.

$$\begin{aligned} F(u,0) &= \langle\rangle \\ F(u,n+1) &= append(F(u,n),(u)_n) \end{aligned}$$

So $F(u,0) = \langle\rangle$, $F(u,1) = \langle u_0\rangle$, $F(u,2) = \langle u_0, u_1\rangle$, and so on. Then $u \upharpoonright i$ is defined as follows.

$$u \upharpoonright i = \left\{ \begin{array}{ll} F(u,i), & \text{if } i \leq n \\ 0, & \text{otherwise} \end{array} \right\}$$

We could also define it just as a division as follows.

$$u \upharpoonright i = \left\{ \begin{array}{ll} quot(u, \Pi_{i\leq j\leq lh(u)} p_j^{(u)_j+1}), & \text{if } u = \langle u_0, \ldots, u_{n-1}\rangle with i \leq lh(u) \\ 0, & \text{otherwise} \end{array} \right\}$$

For *concatenation*, let $F(u, v)$ be the function that *appends* v to u one by one.

$$\begin{aligned} F(u,\langle\rangle) &= u \\ F(u,v) &= append(F(u, v \upharpoonright lh(v) \dot{-} 1), (v)_{lh(v)\dot{-}1}) \end{aligned}$$

So e.g.

$$\begin{aligned} F(\langle u_0\rangle, \langle v_0, v_1\rangle) &= append(F(\langle u_0\rangle, \langle v_0\rangle), v_1) \\ &= append(append(F(\langle u_0\rangle, \langle\rangle), v_0), v_1) \\ &= append(append(\langle u_0\rangle, v_0), v_1) \\ &= append(\langle u_0, v_0\rangle, v_1) \\ &= \langle u_0, v_0, v_1\rangle \end{aligned}$$

Then define $u * v$ as follows.

$$u * v = \left\{ \begin{array}{ll} F(u,v), & \text{if } u = \langle u_0, \ldots, u_{n-1}\rangle, v = \langle v_0, \ldots, v_{m-1}\rangle \\ 0, & \text{otherwise} \end{array} \right\}$$

Similar to the *restriction*, we could define the *concatenation* also as a multiplication as follows.

$$u * v = u.\Pi_{0\leq j<lh(v)} p_{n+j}^{(v)_j+1}$$

2) x1B.20

We know that

$$X_P(y, \overrightarrow{x}) = X_H(\langle X_P(0, \overrightarrow{x}), \ldots, X_P(y-1, \overrightarrow{x})\rangle, \overrightarrow{x})$$

In order to show that X_P is primitive recursive, instead of trying to define X_H by primitive recursion in terms of X_P (for which we try to define X_H by primitive recursion in the first place), let

$$\begin{aligned} F(0, \overrightarrow{x}) &= 1 \\ F(y+1, \overrightarrow{x}) &= append(F(y, \overrightarrow{x}), X_H(F(y, \overrightarrow{x}), y, \overrightarrow{x})) \end{aligned}$$

and let

$$\begin{aligned} X_P(y, \overrightarrow{x}) &= (F(y+1, \overrightarrow{x}))_y \\ &= \text{i.e. the last element of } F(y+1, \overrightarrow{x}) \end{aligned}$$

3) x1B.21 Given the function $g(x)$, $h(w, x, y)$, $\tau(x, y)$, show that there exists exactly one function $f(x, y)$ that satisfies the equations:

$$\begin{array}{rcl} f(0, y) & = & g(y) \\ f(x+1, y) & = & h(f(x, \tau(x, y)), x, y) \end{array}$$

In order to show that f is the one and only function that satisfies these equations, we need to carefully apply the Basic Recursion Lemma (BRL) (i.e. providing explicit definitions of functions and sets), and identify f with a function $(\mathbb{N} \to (\mathbb{N} \to \mathbb{N}))$ that is guaranteed to be unique by the BRL. That would establish a bijection between the solutions of the equations above and the equations for the function we define using BRL. And since that's a bijection, there's as many solutions/functions that satisfy one set of equations as ones that satisfy the other, namely only one.

Let

$$\begin{array}{rcl} f(0) & = & \lambda x.g(x) = g \\ f(x+1) & = & H(f(x), x) \text{ where } H(p, x) = \lambda y.h(p(\tau(x, y)), x, y) \end{array}$$

with $\mathcal{W} = \mathbb{N}^{\mathbb{N}}$ and $H : \mathbb{N}^{\mathbb{N}} \times \mathbb{N} \to \mathbb{N}^{\mathbb{N}}$.

We also show that $f(x, y)$ is primitive recursive if the given functions are primitive recursive. Let's first compute f with some initial inputs and try to identify the pattern in the computation.

$$\begin{array}{rcl} f(0, y) & = & g(y) \\ \\ f(1, y) & = & h(f(0, \tau(0, y)), 0, y) \\ & = & h(g(\tau(0, y)), 0, y) \\ \\ f(2, y) & = & h(f(1, \tau(1, y)), 1, y) \\ & = & h(h(g(\tau(0, \tau(1, y))), 0, \tau(1, y)), 1, y) \\ f(3, y) & = & h(f(2, \tau(2, y)), 2, y) \\ & = & h(h(h(g(\tau(0, \tau(1, \tau(2, y)))), 0, \tau(1, \tau(2, y))), 1, \tau(2, y)), 2, y) \end{array}$$

First, we define a function to compute $\tau(0, \tau(1, \tau(2, \ldots, \tau(n, y))) \ldots)$. To define a function for this by primitive recursion, we reverse the computation, and let

$$\begin{array}{rcl} T(0, m, y) & = & y \\ T(n+1, m, y) & = & \tau(m \dot{-} n, T(n, m, y)) \end{array}$$

The reason that we reverse it is that otherwise $T(n)$ would be defined in terms of $T(n+1)$.

So e.g. if $m = 3$, then

$$\begin{array}{rcl} T(0, 3, y) & = & y \\ T(1, 3, y) & = & \tau(3, y) \\ T(2, 3, y) & = & \tau(2, \tau(3, y)) \\ T(3, 3, y) & = & \tau(1, \tau(2, \tau(3, y))) \\ T(4, 3, y) & = & \tau(0, \tau(1, \tau(2, \tau(3, y)))) \end{array}$$

In order to define f by primitive recursion using T, we reverse it again, as follows.

$$S(n, m, y) = T(m + 1 \dot{-} n, m, y)$$

Finally we define f using S as follows.

$$\begin{array}{rcl} f(0, y) & = & g(y) \\ f(m+1, y) & = & h(f(m, y), m, S(m+1, m, y)) \end{array}$$