

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.

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

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.

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

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

$$u \upharpoonright i = \begin{cases} \text{quot}(u, \prod_{i \leq j \leq lh(u)} p_j^{(u)_j+1}), & \text{if } u = \langle u_0, \dots, u_{n-1} \rangle \text{ with } i \leq lh(u) \\ 0, & \text{otherwise} \end{cases}$$

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

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

So e.g.

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

Then define $u * v$ as follows.

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

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

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

2) **x1B.20**

We know that

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

So 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

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

and let

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

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:

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

In order to show that f is the one and only function that satisfies those, we need to carefully apply the Basic Recursion Lemma (BRL) (with explicit definitions of functions and sets), and identify f with a function $(\mathbb{N} \rightarrow (\mathbb{N} \rightarrow \mathbb{N}))$ that is guaranteed to be unique by the BRL. That would establish a bijection between the solutions of the equations above and the equations that we're going to show below. 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 one.

Let

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

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

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.

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

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

$$\begin{aligned} T(0, m, y) &= y \\ T(n+1, m, y) &= \tau(m \div n, T(n, m, y)) \end{aligned}$$

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

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

To define f by primitive recursion using T , we reverse it again, as follows.

$$S(n, m, y) = T(m+1 \div n, m, y)$$

Finally we define f using S as follows.

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