CPSC 513 — Assignment #3
Solutions for Question #3

In this question you were reminded that each nonnegative integer k can be encoded by the string $1^k \in \Sigma^\star$, so that Post-Turing programs can also be considered as programs that compute functions $f : \mathbb{N}^\ell \to \mathbb{N}$, for $\ell \geq 1$.

For $\ell \geq 1$ let $\Phi_P^{(\ell)} : \mathbb{N}^{\ell+1} \to \mathbb{N}$ such that, for all $x_1, x_2, \ldots, x_\ell, p \in \mathbb{N}$,

$$\Phi_P^{(\ell)}(x_1, x_2, \ldots, x_\ell, p)$$

is the value (weakly) computed by the Post-Turing program $\mathcal{P}$, such that $\#(\mathcal{P}) = p$, when it is executed with inputs $x_1, x_2, \ldots, x_\ell$. Thus this value is defined if and only if the execution of $\mathcal{P}$ on the inputs $x_1, x_2, \ldots, x_\ell$ halts.

You were asked to prove the following

> ***Parameter Theorem for Post-Turing Programs:*** For all positive integers n and m there exists a primitive recursive function
>
> $$S_m^n : \mathbb{N}^{n+1} \to \mathbb{N}$$
>
> such that, for all $x_1, x_2, \ldots, x_m, u_1, u_2, \ldots, u_n, y \in \mathbb{N}$,
>
> $$\Phi_P^{(m+n)}(x_1, x_2, \ldots, x_m, u_1, u_2, \ldots, u_n, y) = \Phi_P^{(m)}(x_1, x_2, \ldots, x_m, S_m^n(u_1, u_2, \ldots, u_n, y)).$$

Solution: It will be useful to define several more pseudoinstructions, and short Post-Turing programs they correspond to, and to show that there are primitive recursive functions that map encodings of programs $\mathcal{P}$ to encodings of programs $\mathcal{Q}$ obtained by inserting these short programs at the beginning of $\mathcal{P}$.

Consider a pseudoinstruction “RIGHT TO NEXT BLANK” which corresponds to the pair of statements

$$\begin{array}{ll} [L] & \text{RIGHT} \\ & \text{IF } 1 \text{ GOTO } L \end{array}$$

where L is a label that is not used elsewhere.

If $\#(L) = \text{safeLabel}(x)$ for $x \in \mathbb{N}$ then the statement

$$[L] \quad \text{RIGHT}$$

has encoding

$$\langle \text{safeLabel}(x), 1 \rangle = 2^{\text{safeLabel}(x)} \times (2 \times 1 + 1) - 1 = 3 \times 2^{\text{safeLabel}(x)} - 1 = 3 \times 2^{\text{safeLabel}(x)} \dot{-} 1,$$

which is a primitive recursive function of x. As noted in the answer for Question #2, the statement

$$\text{IF } 1 \text{ GOTO } L$$

has encoding $10 + 4 \times \text{safeLabel}(x)$, and this is a primitive recursive function of x as well. Thus if moveRight : $\mathbb{N} \to \mathbb{N}$ such that, for all $x \in \mathbb{N}$,

$$\text{moveRight}(x) = \text{prepend}(3 \times 2^{\text{safeLabel}(x)} \dot{-} 1, \text{prepend}(10 + 4 \times \text{safeLabel}(x), x))$$

then moveRight is a primitive recursive function and, if $x = \#(\mathcal{P})$ for a Post-Turing program $\mathcal{P}$, then moveRight$(x) = \#(\mathcal{Q})$, where $\mathcal{Q}$ is the Post-Turing program obtained by inserting the statements

$$\begin{array}{ll} [L] & \text{RIGHT} \\ & \text{IF } 1 \text{ GOTO } L \end{array}$$

with $\#(L) = \text{safeLabel}(x)$ at the *beginning* of $\mathcal{P}$.

Next note that the instruction

$$\text{RIGHT}$$

has encoding

$$\langle 0, 1 \rangle = 2^0 \times (2^1 + 1) - 1 = 3 - 1 = 2$$

while the instruction

$$\text{PRINT } 1$$

has encoding

$$\langle 0, 3 \rangle = 2^0 \times (2^3 + 1) = 9 - 1 = 8.$$

Consider the functions $f : \mathbb{N} \to \mathbb{N}$ such that, for all $x \in \mathbb{N}$,

$$f(x) = \mathsf{prepend}(2, x)$$

and $g : \mathbb{N}^3 \to \mathbb{N}$ such that, for all $t, y, x \in \mathbb{N}$,

$$g(t, y, x) = \mathsf{prepend}(2, \mathsf{prepend}(8, y)).$$

These are both primitive recursive since they are obtained from other primitive recursive functions by a finite number of compositions. It follows that the function print : $\mathbb{N}^2 \to \mathbb{N}$ obtained from f and g by primitive recursion is a primitive recursive function as well. Now, since

$$\mathsf{print}(0, x) = f(x) = \mathsf{prepend}(2, x)$$

and, for $n \geq 0$,

$$\mathsf{print}(n + 1, x) = g(n, \mathsf{print}(n, x), x) = \mathsf{prepend}(2, \mathsf{prepend}(8, \mathsf{print}(n, x)))$$

it is not hard to show — by induction on n — that if $n \in \mathbb{N}$ and $x = \#(\mathcal{P})$ for a Post-Turing program $\mathcal{P}$, then $\mathsf{print}(n, x) = \#(\mathcal{Q})$ where $\mathcal{Q}$ is the Post-Turing program obtained by inserting the sequence

RIGHT
PRINT 1
RIGHT
PRINT 1
RIGHT
⋮
RIGHT
PRINT 1
RIGHT

with length $2n + 1$ (including $n + 1$ copies of "RIGHT" alternating with n copies of "PRINT 1") at the beginning of the program $\mathcal{P}$. Note that this corresponds to a pseudoinstruction

PRINT n TO THE RIGHT.

It will also be useful to have primitive recursive functions corresponding to the insertion of a fixed number of insertions of these small programs at the beginning of a given one. With that noted, let

$$\mathsf{moveLeft}_1 = \mathsf{moveLeft} : \mathbb{N} \to \mathbb{N}$$

and, for $k \geq 2$, let

$$\mathsf{moveLeft}_k = \mathsf{moveLeft} \circ \mathsf{moveLeft}_{k-1} : \mathbb{N} \to \mathbb{N},$$

so that $\text{moveLeft}_k(x) = \text{moveLeft}(\text{moveLeft}_{k-1}(x))$ for all $x \in \mathbb{N}$. Similarly, let

$$\text{moveRight}_1 = \text{moveRight} : \mathbb{N} \to \mathbb{N}$$

and, for $k \geq 2$, let

$$\text{moveRight}_k = \text{moveRight} \circ \text{moveRight}_{k-1} : \mathbb{N} \to \mathbb{N},$$

so that $\text{moveRight}_k(x) = \text{moveRight}(\text{moveRight}_{k-1}(x))$ for all $x \in \mathbb{N}$. Finally, let

$$\text{print}_1 = \text{print} : \mathbb{N}^2 \to \mathbb{N}$$

and, for $k \geq 2$, let

$$\text{print}_k : \mathbb{N}^{k+1} \to \mathbb{N}$$

such that, for all $n_1, n_2, \ldots, n_k, x \in \mathbb{N}$,

$$\text{print}_k(n_1, n_2, \ldots, n_k, x) = \text{print}_{k-1}(n_1, n_2, \ldots, n_{k-1}, \text{print}(n_k, x)).$$

It is not hard to show the following by induction on k: Each of the following properties is satisfied for every number $k \geq 1$.

- The functions $\text{moveLeft}_k : \mathbb{N} \to \mathbb{N}$, $\text{moveRight}_k : \mathbb{N} \to \mathbb{N}$, and $\text{print}_k : \mathbb{N}^{k+1} \to \mathbb{N}$ are all primitive recursive.

- If $x = \#(\mathcal{P})$ for a Post-Turing program $\mathcal{P}$ then $\text{moveLeft}_k(x) = \#(Q)$ for a Post-Turing program $\mathcal{Q}$ such that $\mathcal{Q}$ is obtained by inserting k copies of the program corresponding to the pseudoinstruction

  LEFT TO NEXT BLANK

  at the beginning of $\mathcal{P}$.

- If $x = \#(\mathcal{P})$ for a Post-Turing program $\mathcal{P}$ then $\text{moveRight}_k(x) = \#(\mathcal{Q})$ for a Post-Turing program $\mathcal{Q}$ such that $\mathcal{Q}$ is obtained by inserting k copies of the program corresponding to the pseudoinstruction

  RIGHT TO NEXT BLANK

  at the beginning of $\mathcal{P}$.

- If $x = \#(\mathcal{P})$ for a Post-Turing program $\mathcal{P}$ and $n_1, n_2, \ldots, n_k \in \mathbb{N}$ then

  $$\text{print}_k(n_1, n_2, \ldots, n_k, x) = \#(\mathcal{Q})$$

  for a Post-Turing program $\mathcal{Q}$ such that $\mathcal{Q}$ is obtained by inserting a Post-Turing program corresponding to the pseudoinstructions

PRINT n_1 TO THE RIGHT
PRINT n_2 TO THE RIGHT
$\vdots$
PRINT n_k TO THE RIGHT

at the beginning of $\mathcal{P}$.

It remains only to notice that if $S_m^n : \mathbb{N}^{n+1} \to \mathbb{N}$ such that, for all $u_1, u_2, \ldots, u_n, y \in \mathbb{N}$,

$$S_m^n(u_1, u_2, \ldots, u_n, y) = \text{moveRight}_m(\text{print}_n(u_1, u_2, \ldots, u_n, \text{moveLeft}_{m+n}(y)))$$

then S_m^n is a primitive recursive function such that, for all $x_1, x_2, \ldots, x_m, u_1, u_2, \ldots, u_n, y$,

$$\Phi_P^{(m+n)}(x_1, x_2, \ldots, x_m, u_1, u_2, \ldots, u_n, y) = \Phi_P^{(m)}(x_1, x_2, \ldots, x_m, S_m^n(u_1, u_2, \ldots, u_n, y))$$

as required to complete the proof of the “Parameter Theorem for Post-Turing Programs.”