CPSC 513 — Assignment #3

Solutions for Question #2

In this question it was noted that the pseudoinstruction

LEFT TO NEXT BLANK

corresponds to a pair of statements

$[L]$ LEFT
IF 1 GOTO L

where L is a label that does not appear anywhere else in the Post-Turing program where this pseudoinstruction is used.

You were asked to prove that there is a primitive recursive function moveLeft : $\mathbb{N} \to \mathbb{N}$ such that, for all $n \in \mathbb{N}$, if $\mathcal{P}$ is the Post-Turing program such that $\#(P) = n$, moveLeft(n) is the encoding of a Post-Turing program $\mathcal{Q}$ that begins with the above pair of statements (with L as described above) and continues with the statements in $\mathcal{P}$.

Solution: It will be useful — both for the question and the question that follows it, to define two additional primitive recursive functions:

- Consider the function prepend : $\mathbb{N}^2 \to \mathbb{N}$ such that, for $x, y \in \mathbb{N}$,

$$\text{prepend}(x, y) = \begin{cases} 2^{x+1} & \text{if } y \leq 1, \\ 2^x \times \prod_{i \leq \text{Lt}(y)} p_{i+1}^{(y)_i} & \text{otherwise.} \end{cases}$$

  This is a composition of a finite number of functions that have already been proved to be primitive recursive in lectures, along with an application of definition by cases; it should not be hard to see that it is primitive recursive.

  It should not be hard to see — after inspection of the definition of the encoding of a Post-Turing program — that whenever $x = \#(\mathcal{S})$ for a statement $\mathcal{S}$, and y is an encoding of a (possibly) empty program $\mathcal{P}$, including statements

$$\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k$$

— prepend(x, y) is the encoding of the program

$$\mathcal{S}, \mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k$$

obtained by inserting $\mathcal{S}$ at the *beginning* of $\mathcal{P}$.

- Consider, as well, the function safeLabel : $\mathbb{N} \to \mathbb{N}$ such that, for all $x \in \mathbb{N}$,

  $$\text{safeLabel}(x) = \max(\text{maxLabel}, \text{maxJump}) + 1$$

  Since the functions $\max$, maxLabel and maxJump were all proved to be primitive recursive in the answer for Question #1, safeLabel is a primitive recursive function as well.

  Furthermore, it should not be hard to see — after a consideration of the definitions of both maxLabel and maxJump — that safeLabel$(x) = \#(L)$ for some label L that does not appear anywhere in the Post-Turing program encoded by x.

Now consider the pair of statements

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

— supposing that L is safeLabel(x) for some natural number x.

Then the encoding of

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

is

$$\langle \text{safeLabel}(x), 0 \rangle = 2^{\text{safeLabel}(x)} \times 1 - 1 = 2^{\text{safeLabel}(x)} - 1 = 2^{\text{safeLabel}(x)} \dot{-} 1,$$

which is a primitive recursive function of x.

The encoding of

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

is

$$\begin{aligned} \langle 0, 5 + 2 \times \text{safeLabel}(x) \rangle &= 1 \times (2 \times (5 + 2 \times \text{safeLabel}(x)) + 1) - 1 \\ &= 10 + 4 \times \text{safeLabel}(x), \end{aligned}$$

which is also a primitive recursive function of x.

It remains only to notice that it follows from the above that, for all $x \in \mathbb{N}$,

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

and that this is primitive recursive, as claimed, because it has been produced from primitive recursive functions using a finite number of compositions.