CPSC 513 — Assignment #3

Solutions for Question #1

Following a definition of encodings for Post-Turing programs you were asked, in this question, to prove that a variety of functions are primitive-recursive.

(a) length : $\mathbb{N} \to \mathbb{N}$ such that, for all $n \in \mathbb{N}$, length(n) is the length of (that is, the number of instructions in) the Post-Turing program $\mathcal{P}$ encoded by n.

Solution: This function is identical to the function Lt that was introduced during the lecture on "encodings," as part of the discussion of Gödel numbers.

Since that function was shown to be primitive recursive during that lecture (and a proof of this is included in the lecture notes) it was sufficient to point that out in order to solve this problem.

(b) maxLabel : $\mathbb{N} \to \mathbb{N}$ such that, for all $n \in \mathbb{N}$, if $\mathcal{P}$ is the Post-Turing program that is encoded by n, then

- if none of the instructions in $\mathcal{P}$ are labelled then maxLabel$(n) = 0$, and
- if at least one instruction in $\mathcal{P}$ has a label then maxLabel(n) is the largest integer m such that $m = \#(L)$ for a label L that labels one or more instructions in $\mathcal{P}$.

Note: The beginning of this solution also provides material that makes it easier to answer part (c). This will be separated from material that is specific to this question in the following solution.

Solution (Common Material): To begin, recall that the total predicate whose inputs are a pair of natural numbers $x, y \in \mathbb{N}$ and value is 1 if and only if $y \geq x$ is a primitive recursive predicate. It follows (using "definition by cases") that the function $\max : \mathbb{N}^2 \to \mathbb{N}$ such that, for all $x, y \in \mathbb{N}$,

$$\max(x, y) = \begin{cases} y & \text{if } y \geq x \\ x & \text{otherwise} \end{cases}$$

is a primitive recursive function.

Now let $\varphi : \mathbb{N} \to \mathbb{N}$ be any primitive recursive function.

Next consider the function $f : \mathbb{N} \to \mathbb{N}$ such that $f(n) = 0$ for all $n \in \mathbb{N}$ as well as the function $g : \mathbb{N}^3 \to \mathbb{N}$ such that, for all $t, y, x \in \mathbb{N}$,

$$g(t, y, x) = \max(y, \varphi((x)_{t+1}))$$

where, as usual $(x)_{t+1}$ is one of the "inverses" of the Gödel number function. It should not be hard to see that f and g are both primitive recursive. It follows that the function $h : \mathbb{N}^2 \to \mathbb{N}$ defined from f and g using primitive recursion is primitive recursive as well.

Now, for all $x \in \mathbb{N}$,

$$h(0, x) = f(x) = 0$$

and, for $t \in \mathbb{N}$,

$$h(t + 1, x) = g(t, h(t, x), x) = \max(h(t), \varphi((x)_{t+1})).$$

It is easy to show, by induction on n, that for every number $n \geq 1$,

$$h(n, x) = \max(\varphi((x)_1), \varphi((x)_2), \ldots, \varphi((x)_n)).$$

Finally, let $\max_\varphi : \mathbb{N} \to \mathbb{N}$ be the function such that, for all $x \in \mathbb{N}$,

$$\max_\varphi(x) = \begin{cases} 0 & \text{if } x = 0 \\ \varphi((x)_1 - 1) & \text{if } \mathsf{Lt}(x) = 1 \\ \max(h(\mathsf{Lt}(x) - 1, x), \varphi((x)_{\mathsf{Lt}(x)} - 1)) & \text{if } \mathsf{Lt}(x) > 1. \end{cases}$$

Since $\max_\varphi$ has been obtained from primitive recursive functions by composition, and definition by cases, $\max_\varphi$ is a primitive recursive function, for any primitive recursive function $\varphi : \mathbb{N} \to \mathbb{N}$, as well.

Note that

$$\max_\varphi(x) = \max(\varphi((x)_1), \varphi((x)_2), \ldots, \varphi((x)_{\mathsf{Lt}(x)-1}), , \varphi((x)_{\mathsf{Lt}(x)} - 1))$$

for every number $n \geq 1$. Consequently if x is the encoding of a Post-Turing program including instructions

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

then

$$\max_\varphi(x) = \max(\varphi(\#(\mathcal{S}_1)), \varphi(\#(\mathcal{S}_2)), \ldots, \varphi(\#(\mathcal{S}_k))).$$

Solution (Material Only Needed for Part (b)): Consider the function label : $\mathbb{N} \to \mathbb{N}$ such that, for all $x \in \mathbb{N}$,

$$\text{label}(x) = \ell(x),$$

one of the inverses of the pairing function. As shown in Lecture #5, this function is primitive recursive. Furthermore, if x is the encoding of a statement $\mathcal{S}$ in a Post-Turing program, as described in the assignment, then label$(x) = 0$ if $\mathcal{S}$ does not have a label, and label(x) is the encoding of the label of this statement otherwise.

It follows that maxLabel $= \max_{\varphi}$ where $\varphi : \mathbb{N} \to \mathbb{N}$ is the above primitive recursive function label — implying that maxLabel is primitive recursive, as claimed.

(c) maxJump : $\mathbb{N} \to \mathbb{N}$ such that, for all $n \in \mathbb{N}$, if $\mathcal{P}$ is the Post-Turing program that is encoded by n, then

- if no instructions in $\mathcal{P}$ are conditional branches (i.e., either "IF $\sqcup$ GOTO L" or "IF 1 GOTO L" for some label L) then maxJump$(n) = 0$.
- Otherwise maxJump(n) is the largest integer m encoding a label L such that either the instruction "IF $\sqcup$ GOTO L" or the instruction "IF 1 GOTO L" (or both) is included in $\mathcal{P}$.

Solution: Consider the function jump : $\mathbb{N} \to \mathbb{N}$ such that, for all $x \in \mathbb{N}$,

$$\text{jump}(x) = \lfloor (r(x) \dot{-} 4)/2 \rfloor.$$

where r is the "right" inverse of the pairing function. It should be reasonably clear that jump is a primitive recursive function.

Now note that if $\mathcal{I}$ is any of "LEFT," "RIGHT," "PRINT $\sqcup$" or "PRINT 1" then $\#(\mathcal{I}) \leq 3$ so that, for all $n \in \mathbb{N}$,

$$\text{jump}(\langle n, \#(\mathcal{I}) \rangle) = 0.$$

If $\mathcal{I}$ is an instruction "IF $\sqcup$ GOTO L" for some label L then $\#(\mathcal{I}) = 4 + 2 \times \#(L)$ so that, for all $n \in \mathbb{N}$,

$$\text{jump}(\langle n, \#(\mathcal{I}) \rangle) = \lfloor (2 \times \#(L))/2 \rfloor = \#(L).$$

Finally, if $\mathcal{I}$ is an instruction "IF 1 GOTO L" for some label L then $\#(\mathcal{I}) = 5 + 2 \times \#(L)$ so that, for all $n \in \mathbb{N}$,

$$\text{jump}(\langle n, \#(\mathcal{I}) \rangle) = \lfloor (2 \times \#(L) + 1)/2 \rfloor = \#(L).$$

Thus maxJump is just the function $\max_{\varphi}$ where φ is the primitive recursive function jump, described above. It follows that maxJump is also primitive recursive, as claimed.