Lecture #9: Emulation of Computations — and a "Universal" $\mathcal{S}$-Program

Exercises and Review

Questions for Review

1. *Briefly* describe how $\mathcal{S}$-programs can be encoded as natural numbers.
2. Unfortunately, multiple $\mathcal{S}$-programs are encoded by the same natural number, when the encoding scheme for $\mathcal{S}$-programs from the lecture notes is used. Explain this, and explain why this encoding scheme can still be used to develop a "universal $\mathcal{S}$-program".
3. Describe each of the following predicates or functions. What do their inputs and outputs represent?

   Say whether each is primitive recursive. μ-recursive (but not primitive recursive), Turing-computable (but not shown, yet, to be μ-recursive) or not Turing-computable.

   (a) The function start$_k$: $\mathbb{N}^k \to \mathbb{N}$ (for some positive number k
   (b) The function next : $\mathbb{N}^2 \to \mathbb{N}$
   (c) The function snapshot$_k$: $\mathbb{N}^{k+2} \to \mathbb{N}$, for a positive integer k.
   (d) The predicate halted : $\mathbb{N}^2 \to \{0, 1\}$
   (e) The function output : $\mathbb{N}^2 \to \mathbb{N}$
   (f) The predicate Halt : $\mathbb{N}^{k+2} \to \{0, 1\}$, for a positive integer k.
   (g) The function Output : $\mathbb{N}^{k+2} \to \mathbb{N}$, for a positive integer k.

4. In fact, the predicate halted : $\mathbb{N}^2 \to \{0, 1\}$, the function started$_k$: $\mathbb{N}^k \to \mathbb{N}$ (for a positive integer k) and functions next : $\mathbb{N}^2 \to \mathbb{N}$ and output : $\mathbb{N}^2 \to \mathbb{N}$ are all primitive recursive.

   Thus there exist $\mathcal{S}$ programs that compute these — and these can be used in macros when showing that other partial and total functions are computable by writing programs for them.

With that noted, consider the following program — which makes use of macros to compute each of the above primitive recursive predicates and functions:

$$
\begin{array}{ll}
 & Z_1 \leftarrow \text{start}_k(X_1, X_2, \ldots, X_k) \\
A_1 & Z_2 \leftarrow \text{halted}(X_{k+1}, Z_1) \\
 & \text{IF } Z_2 \neq 0 \text{ GOTO } A_2 \\
 & Z_3 \leftarrow Z_1 \\
 & Z_1 \leftarrow \text{next}(X_{k+1}, Z_3) \\
 & \text{GOTO } A_1 \\
A_2 & Y \leftarrow \text{output}(X_{k+1}, Z_1)
\end{array}
$$

(a) Describe, in simple English the partial function that is computed as this function —- when it is executed with $k + 1$ inputs, $x_1, x_2, \ldots, x_{k+1}$.

(b) Why is it reasonable to think of this as an "$\mathcal{S}$-program emulator," or a ***universal $\mathcal{S}$-program***?

(c) A ***universal Turing machine*** was (probably) presented and discussed during CPSC 351 (or 313). How are universal $\mathcal{S}$-programs and universal Turing machines similar?

5. Define (using reasonably simple English, and a bit of mathematical notation) the "halting predicate," HALT : $\mathbb{N}^2 \rightarrow \{0, 1\}$.
6. State the ***Normal Form Theorem*** and *briefly* explain how this is proved, using results that were established in the previous lecture.
7. Describe what this implies about the relationship between μ-recursive functions and Turing-computable functions.

Practice Problem

This problem — which continues activities in the lecture presentation — will not be discussed during the lecture, and solutions for these problems will not generally be made available. It can be used as a "practice" problem that can help you to practice skill considered in the lecture presentation for Lecture #9.

1. Consider the partial function stops : $\mathbb{N} \rightarrow \mathbb{N}$ such that, for all $x_1, x_2 \in \mathbb{N}$,

   - if the program $\mathcal{P}$ such that $\#(\mathcal{P}) = x_2$ halts on input x_1, then stops$(x_1, x_2) = 1$, and
   - if the above program $\mathcal{P}$ does not halt on input x_1, then stops(x_1, x_2) is undefined.

   Prove that stops is a computable partial function.