Limits of Computation (CS:4340:0001 or 22C:131:001)
Homework 2

The homework is due in class on Thursday, Februrary 19th. If you can't make it to class, drop it in my mailbox in the MacLean Hall mailroom.

1. Consider Claim 1.6, which states that for any k-tape Turing Machine M that computes a function $f : \{0,1\}^* \rightarrow \{0,1\}$, there is a corresponding one-tape Turing Machine $\tilde{M}$ that computes f with a quadratic slowdown in running time. Describe in some detail how one step of the Turing machine M is "simulated" by $\tilde{M}$. For simplicity, assume that M uses $k = 2$ tapes and the alphabet $\{0, 1, \triangleright, \square\}$. You don't have to give a full description of the transition function of $\tilde{M}$ in terms of that of M. It suffices to describe the sequence of states used by $\tilde{M}$ to simulate one step of M, in a manner similar to what we did in class to substantiate Claim 1.5. (3 points)

2. Let $\texttt{AEL} : \{0,1\}^* \rightarrow \{0,1\}$ be the function defined as follows: $\texttt{AEL}(\alpha) = 1$ if M_α outputs 1 on any string $x \in \{0,1\}^*$ such that $|x|$ is even, and 0 on any string $x \in \{0,1\}^*$ such that $|x|$ is odd; $\texttt{AEL}(\alpha) = 0$ otherwise.[1] Recall that M_α is the Turing machine encoded by α.

   Show that there is no Turing machine for computing `AEL`. **Hint:** Do a reduction from the HALT function. This reduction is quite similar to the reductions we did in class, for the `Hello-World` and `Accepts-All-Strings` functions. (3 points)

3. This problem is a slight variant of Exercise 1.10 in the text. Conisder the following simple programming language. It has a single infinite array A of elements in $\{0, 1, \square\}$ (initialized to $\square$) and a single integer variable i. A program in this language contains a sequence of lines of the following form:

   $$\textit{label}: \texttt{ If A[i] equals } \sigma \texttt{ then } \textit{cmds}$$

   where $\sigma \in \{0, 1, \square\}$ and *cmds* is a list of one or more of the following commands: (1) `Set A[i] to` τ where $\tau \in \{0, 1, \square\}$, (2) `Goto` *label*, (3) `Increment i by 1`, (4) `Decrement i by 1`, and (5) `Output` b `and halt`, where $b \in \{0, 1\}$.

   A program is executed on an input $x \in \{0,1\}^n$ by placing the i'th bit of x in $A[i]$, initializing i to 1 (the first index in array A), and then running the program using the obvious semantics.[2]

   Prove that if a function $f : \{0,1\}^* \rightarrow \{0,1\}$ is computable by a program in this language, then f is also computable by a Turing Machine. You should do this by describing a Turing Machine that simulates the program. It is enough to do so at a high level, like in the proofs of Claims 1.5 and 1.6.

---

[1] `AEL` abbreviates `Accepts-Even-Lengths`.

[2] The programming language is a bit like assembly language.

If the program computes f in $T(n)$ time, how can we bound the running time of the corresponding TM? (4 points)