Problem Set 2

CSE 531 - Computational Complexity

Winter 2024

Exercise 1 (not in the A&B book; 10pts)
Recall that in the lecture we have proven that the function \( \text{HALT} \) given by
\[
\text{HALT}(x, \alpha) := \begin{cases} 
1 & \text{if } M_\alpha \text{ halts on input } x \\
0 & \text{otherwise}
\end{cases}
\]
is undecidable (meaning there is no Turing machine \( M \) computing \( \text{HALT} \)). Now consider the problem \( \text{HALT}_{\text{EMPTY}} \) where
\[
\text{HALT}_{\text{EMPTY}}(\alpha) := \begin{cases} 
1 & \text{if } M_\alpha \text{ halts on input of the empty string} \\
0 & \text{otherwise.}
\end{cases}
\]
Prove that \( \text{HALT}_{\text{EMPTY}} \) is undecidable.

Exercise 2 (slightly modified from Ex 1.11 in A&B book; 10pts)
We say that a function \( f : \{0,1\}^* \rightarrow (\{0,1\}^* \cup \{\text{UNDEF}\}) \) is a\ partial function where \( f(x) = \text{UNDEF} \) means that the function is not defined on input \( x \). We say that a Turing machine \( M \) computes the partial function \( f \) with
\[
f(x) = \begin{cases} 
M(x) & \text{if } M \text{ halts on } x \\
\text{UNDEF} & \text{if } M \text{ does not halt on } x
\end{cases}
\]
For a set \( S \) of partial functions, we define \( f_S : \{0,1\}^* \rightarrow \{0,1\} \) by letting
\[
f_S(\alpha) := \begin{cases} 
1 & \text{if the partial function computed by } M_\alpha \text{ is in } S \\
0 & \text{otherwise}
\end{cases}
\]
We say that such a set \( S \) is \textit{non-trivial} if there are Turing machines \( M_0 \) and \( M_1 \) so that the partial function computed by \( M_0 \) is not in \( S \) and the partial function computed by \( M_1 \) is in \( S \). Prove the following:

\textbf{Rice’s Theorem.} For any non-trivial set \( S \) of partial functions, there is no Turing machine computing \( f_S \).

\textbf{Hint.} By possibly replacing from \( S \) by its complement (which doesn’t affect the claim), we may assume that \( f_0 \in S \), where \( f_0 : \{0,1\}^* \rightarrow \{0,1\}^* \cup \{\text{UNDEF}\} \) is the partial function with \( f_0(\alpha) = \text{UNDEF} \) for all \( \alpha \). Use this to show that an algorithm to compute \( f_S \) can compute the function \( \text{HALT} \) considered in the first homework problem.

\textbf{Comment.} One can interpret this problem as proving that testing \textit{any} kind of output behavior of Turing machines is undecidable.