CS/ECE 374 ✧ Fall 2016

Homework 11

"Due" Tuesday, December, 2016

This homework is only for practice; it will not be graded. However, **similar questions may appear on the final exam,** so we still strongly recommend treating this as a regular homework. Solutions will be released next Tuesday as usual.

---

1. Recall that w^R denotes the reversal of string w; for example, $\texttt{TURING}^R = \texttt{GNIRUT}$. Prove that the following language is undecidable.

$$\textsc{RevAccept} := \left\{ \langle M \rangle \;\middle|\; M \text{ accepts } \langle M \rangle^R \right\}$$

Note that Rice's theorem does *not* apply to this language.

2. Let M be a Turing machine, let w be an arbitrary input string, and let s be an integer. We say that ***M accepts w in space s*** if, given w as input, M accesses only the first s (or fewer) cells on its tape and eventually accepts.

   (a) Sketch a Turing machine/algorithm that correctly decides the following language:

$$\left\{ \langle M, w \rangle \;\middle|\; M \text{ accepts } w \text{ in space } |w|^2 \right\}$$

   (b) Prove that the following language is undecidable:

$$\left\{ \langle M \rangle \;\middle|\; M \text{ accepts at least one string } w \text{ in space } |w|^2 \right\}$$

3. Consider the language $\textsc{SometimesHalt} = \{\langle M \rangle \mid M \text{ halts on at least one input string}\}$. Note that $\langle M \rangle \in \textsc{SometimesHalt}$ does not imply that M *accepts* any strings; it is enough that M *halts* on (and possibly rejects) some string.

   (a) Prove that SometimesHalt is undecidable.

   (b) Sketch a Turing machine/algorithm that *accepts* SometimesHalt.

Solved Problem

4. For each of the following languages, either prove that the language is decidable, or prove that the language is undecidable.

   (a) $L_0 = \{\langle M \rangle \mid \text{given any input string, } M \text{ eventually leaves its start state}\}$

   Solution: We can determine whether a given Turing machine M always leaves its start state by careful analysis of its transition function δ. As a technical point, I will assume that crashing on the first transition does *not* count as leaving the start state.

   - If $\delta(\text{start}, a) = (\cdot, \cdot, -1)$ for any input symbol $a \in \Sigma$, then M crashes on input a without leaving the start state.
   - If $\delta(\text{start}, \square) = (\cdot, \cdot, -1)$, then M crashes on the empty input without leaving the start state.
   - Otherwise, M moves to the right until it leaves the start state. There are two subcases to consider:
     – If $\delta(\text{start}, \square) = (\text{start}, \cdot, +1)$, then M loops forever on the empty input without leaving the start state.
     – Otherwise, for any input string, M must eventually leave the start state, either when reading some input symbol or when reading the first blank.

   It is straightforward (but tedious) to perform this case analysis with a Turing machine that receives the encoding $\langle M \rangle$ as input. We conclude that L_0 is ***decidable***. ■

   (b) $L_1 = \{\langle M \rangle \mid M \text{ decides } L_0\}$

   Solution:

   - By part (a), there is a Turing machine that decides L_0.
   - Let M_{reject} be a Turing machine that immediately rejects its input, by defining $\delta(\text{start}, a) = \text{reject}$ for all $a \in \Sigma \cup \{\square\}$. Then M_{reject} decides the language $\varnothing \neq L_0$. ■

   Thus, Rice's Decision Theorem implies that L_1 is ***undecidable***.

   (c) $L_2 = \{\langle M \rangle \mid M \text{ decides } L_1\}$

   Solution: By part (b), no Turing machine decides L_1, which implies that $L_2 = \varnothing$. Thus, M_{reject} correctly decides L_2. We conclude that L_2 is ***decidable***. ■

   (d) $L_3 = \{\langle M \rangle \mid M \text{ decides } L_2\}$

   Solution: Because $L_2 = \varnothing$, we have

   $$L_3 = \{\langle M \rangle \mid M \text{ decides } \varnothing\} = \{\langle M \rangle \mid \text{REJECT}(M) = \Sigma^*\}$$

   - We have already seen a Turing machine M_{reject} such that $\text{REJECT}(M_{\text{reject}}) = \Sigma^*$.
   - Let M_{accept} be a Turing machine that immediately accepts its input, by defining $\delta(\text{start}, a) = \text{accept}$ for all $a \in \Sigma \cup \{\square\}$. Then $\text{REJECT}(M_{\text{accept}}) = \varnothing \neq \Sigma^*$. ■

   Thus, Rice's Rejection Theorem implies that L_1 is ***undecidable***.

(e) $L_4 = \left\{ \langle M \rangle \mid M \text{ decides } L_3 \right\}$

Solution: By part (b), no Turing machine decides L_3, which implies that $L_4 = \varnothing$. Thus, M_{reject} correctly decides L_4. We conclude that L_4 is ***decidable***.

At this point, we have fallen into a loop. For any $k > 4$, define

$$L_k = \left\{ \langle M \rangle \mid M \text{ decides } L_{k-1} \right\}.$$

Then L_k is decidable (because $L_k = \varnothing$) if and only if k is even. ■

Rubric: 10 points: 4 for part (a) + 1½ for each other part.

Rubric (for all undecidability proofs, out of 10 points):
Diagonalization:
- \+ 4 for correct wrapper Turing machine
- \+ 6 for self-contradiction proof (= 3 for $\Leftarrow$ + 3 for $\Rightarrow$)

Reduction:
- \+ 4 for correct reduction
- \+ 3 for "if" proof
- \+ 3 for "only if" proof

Rice's Theorem:
- \+ 4 for positive Turing machine
- \+ 4 for negative Turing machine
- \+ 2 for other details (including using the correct variant of Rice's Theorem)