CSC4010: Introduction to Theoretical Computer Science
Problem Set 3

---

1. Consider the language $L = \{\langle M \rangle \mid L(M) = \overline{A}_{TM}\}$. Is this language decidable or undecidable? Justify your answer by giving a decider if it is decidable, otherwise formally prove it is undecidable (via one of the methods described in class).

2. Consider the language $L = \{\langle M \rangle \mid M \text{ is a Turing machine and } L(M) \text{ is decidable}\}$. Is this language decidable or undecidable? Justify your answer by giving a decider if it is decidable, otherwise formally prove that it is undecidable (via one of the methods described in class).

3. Rice's Theorem states that it is impossible to decide any non-trivial property of recognizable languages. More concretely, a property P is a subset of recognizable languages, and it is said to be non-trivial if there exist a Turing Machine M_1 such that $L(M_1) \in P$, and there exists M_2 such that $L(M_2) \notin P$. Let $L_P = \{\langle M \rangle \mid L(M) \in P\}$. Show that it is undecidable.

   For each language L below, determine whether it is decidable. If it is undecidable, prove it. In your proof, clearly state whether you are using Rice's Theorem or a direct reduction. If you use Rice's Theorem, explain how the property you use is non-trivial. If you do not use Rice's Theorem, explain why it does not apply.

   (a) The language $L = \{\langle M \rangle \mid L(M) \text{ is recognizable}\}$.

   (b) The language $L = \{\langle M \rangle \mid M \text{ has at most 30 states}\}$.

   (c) The language $L = \{\langle M \rangle \mid M \text{ halts on } \epsilon\}$.

   (d) The language $L = \{\langle M \rangle \mid L(M) \text{ recognized by some Turing machine having an even number of states}\}$.

   (e) The language $L = \{\langle M \rangle \mid L(M) \text{ is recognized by a TM with at most 30 states and a tape alphabet with at most 30 symbols}\}$.

4. Consider the language $L = \{\langle M \rangle \mid L(M) \text{ contains a palindrome}\}$. Is this language decidable or undecidable? Is it recognizable or unrecognizable? Justify your answer.

5. Consider the language $L = \{\langle M \rangle \mid M \text{ accepts string } 01 \text{ in perfect-square number of steps}\}$. Is this language decidable or undecidable? Justify your answer.

6. Consider the language $L = \{\langle M \rangle \mid \forall w \in \{0,1\}^*, M \text{ accepts } w \text{ iff } M \text{ accepts } w^R\}$. Is this language decidable or undecidable? Justify your answer.

7. Consider the problem of determining whether a two-tape Turing machine ever writes a nonblank symbol on its second tape during the course of its computation on some input string. Formulate this problem as a language and show that it is undecidable.

Solutions For Problem Set 3

1. The language is decidable. Note that $\bar{A}_{TM}$ is unrecognizable. Therefore, there is no Turing machine M such that $L(M) = \bar{A}_{TM}$. Thus, $L = \emptyset$ and is decidable, as the Turing machine which rejects all inputs decides it.

2. Suppose, for contradiction, that D decides $L = \{\langle N \rangle \mid L(N) \text{ is decidable}\}$. We use D to construct a decider for A_{TM}. Given $\langle M, w \rangle$, construct a Turing machine N_w that behaves as follows on any input x:

   (a) Simulate M on w.

   (b) If M accepts, simulate a fixed recognizer for A_{TM} on x, accepting exactly when it accepts.

   (c) If M rejects, reject. If M loops, keep looping.

   Therefore,

   $$L(N_w) = \begin{cases} A_{\text{TM}}, & \text{if } M \text{ accepts } w, \\ \varnothing, & \text{if } M \text{ does not accept } w. \end{cases}$$

   Since A_{TM} is undecidable and $\varnothing$ is decidable, $\langle M, w \rangle \in A_{\text{TM}} \iff \langle N_w \rangle \notin L$. On input $\langle M, w \rangle$, we can construct $\langle N_w \rangle$, run D on it, and reverse D's answer. This procedure always halts and correctly decides A_{TM}, contradicting its undecidability.

3. We reduce $L_{\text{self}} = \{\langle M \rangle \mid M \text{ halts and rejects } \langle M \rangle\}$ to L_P. Recall that L_{self} is undecidable.

   Assume first that $\varnothing \notin P$. If $\varnothing \in P$, then replace P by its complement $\overline{P}$. Since $L_{\overline{P}} = \{\langle M \rangle \mid L(M) \notin P\} = \overline{L_P}$, the language L_P is decidable iff $L_{\overline{P}}$ is decidable. Thus it suffices to prove that $L_{\overline{P}}$ is undecidable.

   Choose a TM T such that $L(T) \in P$. Given $\langle M \rangle$, construct a TM N that behaves as follows on input x:

   (a) Simulate M on $\langle M \rangle$.

   (b) If this simulation rejects, simulate T on x, and accept exactly when T accepts.

   (c) Otherwise, reject all inputs.

   Therefore,

   $$L(N) = \begin{cases} L(T), & \text{if } \langle M \rangle \in L_{\text{self}}, \\ \varnothing, & \text{otherwise.} \end{cases}$$

   Since $\varnothing \notin P$, we have $\langle M \rangle \in L_{\text{self}} \iff L(N) \in P$. Hence a decider for L_P would yield a decider for L_{self}, contradicting the undecidability of L_{self}. Therefore L_P is undecidable.

   (a) L is decidable. A decider for L just has to check if the input string is a valid Turing machine description. It does not satisfy the premise of Rice's theorem as there exists no M_2 such that $L(M_2)$ is not recognizable.

   (b) L is decidable, as the decider just has to check if M has at most 30 states based on its description. It does not satisfy the premise of Rice's theorem as Rice's theorem is a statement about non-trivial properties of languages whereas, L is based on the properties of Turing machines.

   (c) We use a direct reduction from HALT_{TM} to $L = \{\langle M \rangle \mid M \text{ halts on } \epsilon\}$. Given $\langle M, w \rangle$, construct a Turing machine N that ignores its input and simulates M on w. If the simulation halts, N halts; otherwise, N continues simulating forever. Thus, $\langle M, w \rangle \in \text{HALT}_{\text{TM}} \iff N \text{ halts on } \epsilon \iff \langle N \rangle \in L$. A decider for L would therefore decide HALT_{TM}, which is impossible. Hence L is undecidable.

   Rice's theorem does not apply, since halting on ϵ is a property of the machine's computation, not of its recognized language: two machines can recognize the same language $\varnothing$ (one rejecting every input, one looping on every input) while differing on whether they halt on ϵ.

   (d) L is decidable, L consists of all valid Turing machine descriptions. Indeed, for any Turing machine M, the language $L(M)$ is recognizable by M itself. If M has an even number of states, we are done. If M has an odd number of states, add one new unreachable state to obtain a Turing machine M' with an even number of states and $L(M') = L(M)$. Hence every recognizable language has an even-state recognizer, so every valid Turing machine description belongs to L.

(e) The property P is non-trivial. It is nonempty because $\varnothing \in P$: the empty language is recognized by a Turing machine with very few states and a small tape alphabet. It is not the set of all recognizable languages because there are only finitely many Turing machines with at most 30 states and at most 30 tape symbols, hence only finitely many languages in P. But there are infinitely many recognizable languages. Therefore P is a proper, nonempty subset of the recognizable languages.

Since P is a non-trivial property of recognizable languages, Rice's Theorem implies that $L_P = \{\langle M\rangle \mid L(M) \in P\}$ is undecidable.

4. The language is **recognizable**. Let $p_1, p_2, \cdots$ be an ordering of the palindromes. The recognizer R will just take as input $\langle M\rangle$ and it will run in phases wherein phase i, M runs on $p_1, p_2, \cdots, p_i$ for one additional step. That is, in phase one, M runs on p_1 for one step. Then, in phase two it runs on p_1, p_2 for one additional step. In phase 3 on p_1, p_2, p_3 for one additional step and so on and so forth. If a palindrome $p_i \in L(M)$ and suppose $M(p_i)$ halts in t steps, then R will halt at the end of phase $i + t$ and each phase consists of finitely many steps. (This technique is referred to as dovetailing.)

   To show **undecidability**, we reduce A_{TM} to L. Given $\langle M, w\rangle$, construct M'_w that rejects every input except ww^R. On input ww^R, it simulates M on w and accepts exactly when M accepts. Thus,

$$L(M'_w) = \begin{cases} \{ww^R\}, & \text{if } M \text{ accepts } w, \\ \emptyset, & \text{otherwise.} \end{cases}$$

   Since ww^R is a palindrome, $\langle M, w\rangle \in A_{\text{TM}} \iff \langle M'_w\rangle \in L$. A decider for L would therefore decide A_{TM}, which is impossible. Hence L is undecidable.

5. We show that $L = \{\langle M\rangle \mid M \text{ accepts } 01 \text{ in a perfect-square number of steps}\}$ is undecidable by reducing A_{TM} to L. Given $\langle M, u\rangle$, effectively construct a machine N that rejects inputs other than 01. On input 01, it simulates M on u. If the simulation accepts, N performs redundant computation and then accepts, so that N's total number of steps, including all overhead, is a perfect square. If the simulation rejects or runs forever, N never accepts.

   While simulating M on u, N keeps an exact count of its own steps by advancing a unary tally on a separate track once per step, so when the simulation accepts, the tally equals the exact number T of steps elapsed so far. N then copies this T-mark block once on another tape (T steps) and makes T further passes over it: each pass sweeps the block (alternating direction, so no reset is needed), detects the blank at its end, and erases one mark of the copy, so each pass takes exactly $T + O(1)$ steps, and the copy, used as a countdown, guarantees that exactly T passes occur. Finally, N adds a fixed number of dummy steps, chosen so that its total running time is exactly $(T + k)^2$ for a fixed constant k, and then accepts. Thus, if M accepts u then N accepts 01 after a perfect-square number of steps, and if M does not accept u then N never accepts 01.

   Hence $\langle M, u\rangle \in A_{\text{TM}} \iff \langle N\rangle \in L$. A decider for L would therefore decide A_{TM}, which is impossible.

6. We show that L is undecidable by reducing A_{TM} to L. Given $\langle M, u\rangle$, construct N that accepts 01, simulates M on u when its input is 10, and rejects all other inputs. On input 10, it accepts iff M accepts u. Thus,

$$L(N) = \begin{cases} \{01, 10\}, & \text{if } M \text{ accepts } u, \\ \{01\}, & \text{otherwise.} \end{cases}$$

   Only the first language is closed under reversal, so $\langle M, u\rangle \in A_{\text{TM}} \iff \langle N\rangle \in L$. A decider for L would decide A_{TM}, a contradiction. Therefore, L is undecidable.

7. Define $L = \{\langle M\rangle \mid M \text{ is a two-tape TM that writes a non-blank symbol on its second tape on some input}\}$. We reduce A_{TM} to L as follows: On input $\langle M, w\rangle$ construct a Turing machine N_w that does the following: it is a two-tape Turing machine that on any input simulates the behavior of M on w on the first tape. If M reaches accept state, then it writes a non-blank symbol on the second tape. Observe that $\langle N_w\rangle \in L \iff \langle M, w\rangle \in A_{TM}$.