CSE 531: Computability and Complexity | Autumn 2002
Problem Set #2 | Instructor: Venkatesan Guruswami
Due on Thursday, **October 31, 2002** in class.

Instructions: You are allowed to collaborate with fellow students taking the class in solving problem sets. If you do so, please indicate for each problem the people you worked with on that problem. Note that you must write down solutions on your own and collaboration must be restricted to a discussion of solution ideas. You are expected to refrain from looking up solutions or solution ideas from websites or other literature.

---

1. Which of the following problems about Turing machines are decidable and which are not? Briefly justify your answers. (20 points)
   (a) To determine, given a Turing machine M, whether M has the property that it accepts a string $w \in \{0,1\}^*$ if and only if it accepts the sring $\overline{w}$ (here $\overline{w}$ denotes the bitwise complement of w; eg. $\overline{100110} = 011001$).
   (b) To determine, given a Turing machine M and a string w, whether M ever moves it head to the left when it is run on input w.
   (c) To determine, given a Turing machine M and a string w, whether M on input w ever tries to move its head left when its head is on the left-most tape cell.
   (d) To determine, given a Turing machine M, whether the tape ever contains four consecutive 1's during the course of M's computation when it is run on input 01.
2. (a) Problem 5.19, Sipser's book (Ambiguity of CFGs is undecidable)
   (b) Use the approach used in part (a) above to give a proof different from the one given in class of the undecidability of $COMMON_{\mathrm{CFG}}$ defined as:

   $$COMMON_{\mathrm{CFG}} = \{\langle G_1, G_2\rangle \,|\, G_1, G_2 \text{ are context-free grammars and } L(G_1) \cap L(G_2) = \emptyset\} .$$
3. Prove that a language L is Turing recognizable if and only if L is mapping reducible to A_{TM}.
4. Problem 5.20, Sipser's book (Acceptance and emptiness problems for two headed finite automata)
   - Suggestion: Reading Theorems 5.8 and 5.9 on linear bounded automata will help.
5. Define the language

   $$REGULAR_{\mathrm{CFG}} = \{\langle G\rangle \,|\, G \text{ is a context-free grammar and } L(G) \text{ is regular } \}$$

   that consists of grammars which generate regular languages. Prove that $REGULAR_{\mathrm{CFG}}$ is undecidable. (Suggestion: Use an approach based on computation histories of Turing machines similar to the proof that ALL_{CFG} is undecidable.)
6. For a pushdown automaton (PDA) or a nondeterministic Turing machine (NTM), we say that it has a useless state if there exists a state in its finite control which is *never* reached, on any input or any non-deterministic branch. Define the languages $USELESS_{\mathcal{C}} = \{\langle M\rangle \,|\, M \in \mathcal{C}$ and M has a useless state$\}$, where $\mathcal{C} = \mathrm{PDA}, \mathrm{NTM}$.

(a) Prove that $USELESS_{\text{PDA}}$ is decidable.

(b) Prove that $USELESS_{\text{NTM}}$ is not Turing-recognizable.

7. * **(Optional Problem)** We have seen in class that the languages

$$E_{\text{TM}} = \{\langle M\rangle \mid M \text{ is a Turing machine and } L(M) = \emptyset\}$$

as well as

$$ALL_{\text{TM}} = \{\langle M\rangle \mid M \text{ is a Turing machine and } L(M) = \Sigma^*\}$$

are both undecidable. There is, however, a sense in which ALL_{TM} is actually harder than E_{TM}. Specifically, even if we were miraculously given access to a decider for E_{TM}, it is still not clear how one could use that to decide ALL_{TM} (contrast this with the fact that it is easy to decide A_{TM} using a miracle box that can decide E_{TM}). On the other hand, one can construct a decider for E_{TM} if there were a miracle black box that could decide ALL_{TM} (think about how).

Of course E_{TM} is provably undecidable, so by assuming that it is decidable, we are assuming a false statement and can then logically deduce anything. One must therefore be careful in trying to formalize a statement like "ALL_{TM} is harder than E_{TM}". This exercise defines the *arithmetical hierarchy* which gives us such a formalism and asks you to prove some basic facts concerning the hierarchy.

Define a relation $R \subseteq (\Sigma^*)^k$ to be decidable if the language

$$L_R = \{\langle x_1, x_2, \ldots, x_k\rangle \mid (x_1, x_2, \ldots, x_k) \in R\}$$

is decidable. Define Σ_k, for $k \geq 0$, to be the class of all languages L for which there is a decidable $(k+1)$-ary relation R such that

$$L = \{x \mid \exists x_1 \forall x_2 \cdots Q_k x_k \; R(x_1, x_2, \ldots, x_k, x)\} \; ,$$

where the quantifier Q_k is $\exists$ if k is odd and $\forall$ if k is even. We define $\Pi_k = \text{co}\Sigma_k$, i.e. Π_k is the set of all complements of languages in Σ_k.

In this notation, clearly Σ_0 and Π_0 equal the set of decidable languages, and in Problem 6 of Problem set 1, you showed that Σ_1 equals the class of Turing-recognizable languages.

Now to your exercises:

(a) Show that, for $k \geq 0$, Π_k is the class of all languages L such that there is a decidable relation R for which

$$L = \{x \mid \forall x_1 \exists x_2 \cdots Q_k x_k \; R(x_1, x_2, \ldots, x_k, x)\} \; ,$$

where the quantifier Q_k is $\forall$ if k is odd and $\exists$ if k is even.

(b) Show that for all $k \geq 0$, $\Sigma_k \subseteq \Sigma_{k+1}$, and $\Pi_k \subseteq \Sigma_{k+1}$.

(c) Show that Σ_2 is the class of languages that can be recognized (**not** decided) by Turing machines that are equipped with the following extra power: At any point the machine may write any string z onto a special tape and enter a special state $q_?$, and the next state will one of two dedicated states q_Y or q_N depending on whether or not $z \in A_{\text{TM}}$ (do not confuse q_Y and q_N with the accept and reject states of the Turing machine, these states are just used to find out the answer to the question "Does $z \in A_{\text{TM}}$?").

(d) Prove that for all $k \geq 0$, $\Sigma_{k+1} \neq \Sigma_k$. (Hint: Generalize the proof of the case $k = 0$.)

(e) Prove that $ALL_{\mathrm{TM}} \in \Pi_2 \setminus (\Sigma_1 \cup \Pi_1)$. (Hint: Use parts (c) and (d) above.)
(**Comment:** Note that E_{TM} is co-Turing-recognizable and thus $E_{\mathrm{TM}} \in \Pi_1$, so ALL_{TM} is harder than E_{TM} in terms of the lowest level of the arithmetical hierarchy in which it lies.)

(f) Place as low in the arithmetical hierarchy as possible the language:

$$INFINITE_{\mathrm{TM}} = \{\langle M \rangle \mid M \text{ is a Turing machine and } L(M) \text{ is infinite}\} .$$