Theory of Computation

Spring 2021, Final Exam (Solutions)

Due: June 22, 2021

1. (10 pts) Decide if the following language is decidable or not with a formal proof.
$L = \{\langle M \rangle \mid \text{Turing machine } M \text{ accepts some } w \in \Sigma^* \text{ with more than one million } (i.e., 10^6) \text{ steps.}\}$
Sol.: Undecidable. Assume there exists a decider R for L. Construct the following S:

   $S =$ On $\langle M \rangle$, a TM
   1. Construct $\langle M' \rangle$ from $\langle M \rangle$:
      $M' =$ On x, an input string
      a. Run 10^6 useless steps;
      b. Simulate M on x and return the result of the simulation.
   2. If $R(\langle M' \rangle)$ accepts, then reject; otherwise accept.

   The above S is a decider for E_{TM}, solving the emptiness problem of TMs, which is known to be undecidable.

2. (10 pts) Show that if $P = PSPACE$, then every language $A \in P$, except $A = \emptyset$ and $A = \Sigma^*$, is $PSPACE$-complete under the polynomial-time many-one reduction.
Sol. Let B be any language in PSPACE and let $A \in P(= PSPACE)$ be another language not equal to $\emptyset$ or Σ^*. Then there exist strings $x \in A$ and $y \notin A$. To reduce an instance w of B to that of A, we just check in polynomial time if $w \in B$. If yes, we output x; output y when $w \notin B$. That is, $f(w) = x$ if $w \in B$, and $f(w) = y$ if $w \notin B$. So $w \in B$ iff $f(w) \in A$. Therefore $B \leq_m^p A$ holds. Hence A is PSPACE-hard.

3. (10 pts) True or False? Justify your answers.
   (a) If $A, B \in NP$, then $(A \cup B) \in NP$ and $(A \cap B) \in NP$.
   Sol.: True. Let M_A and M_B be polynomial-time NTM accepting A and B, resp., construct M whose initial state has ϵ transitions going to the initial states of M_A and M_B. M accepts $A \cup B$. For $A \cap B$, construct the product of M_A and M_B.
   (b) There exist NP-complete languages A and B such that $A \cap B$ is not NP-complete.
   Sol.: True. Suppose A and B have their alphabet disjoint. The $A \cap B = \emptyset$, which is clearly not NP-hard.

4. (10 pts) Consider the following two languages over the alphabet $\Sigma = \{0, 1\}$.
   - $L_u = \{\langle M, w \rangle : \text{ TM M accepts input } w\}$
   - $L_{15} = \{\langle M' \rangle : |L(M')| = 15\}$, i.e., the set of TMs whose languages have exactly 15 strings.

   Prove that L_{15} is undecidable via a reduction from L_u. Do not use the Rice's theorem.
   Sol.

   Given a Turing machine M and input w, construct a Turing machine $\widehat{M}$ which behaves as follows on being given input $\widehat{w}$.

   1. $\widehat{M}$ simulates the behavior of M on input w;
   2. if M halts on w and accepts, then $\widehat{M}$ examines its own input $\widehat{w}$, halting in a final state if $|\widehat{w}| \leq 3$ and halting in a non-final state otherwise;
   3. if M halts on w and rejects, then $\widehat{M}$ rejects its own input $\widehat{w}$.

5. (10 pts) Let $\langle M_1\rangle, \langle M_2\rangle, \langle M_3\rangle, ...$ be an enumeration of all Turing machines over alphabet Σ. Let $w_1, w_2, w_3, ...$ be an enumeration of all words over Σ (i.e., $w_1, w_2, w_3 ... \in \Sigma^*$). We consider the following language $L = \{w \in \Sigma^* \mid w = w_i$, for some i, and M_i does not accept $w_i\}$. Prove that L is not Turing-recognizable.
Sol. Proof by contradiction. Assume L is TM-recognizable. Then there is some TM M that recognizes L. Let i be such that $M = M_i$. Consider the word w_i. If w_i is accepted by M_i, then by definition of L, $w_i \in \Sigma^* \backslash L$; hence M_i cannot be accepting L. If w_i is not accepted by M_i, then by definition of L, $w_i \in L$; hence again M_i must accept w_i. But one of these must be true; hence M cannot be a recognizer for L. Contradiction proves that there is no TM that recognizes L.

6. (10 pts) A *finite automaton with outputs* (FAO) $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ is a device which is just like a finite automaton except that at each step, when M reads an input symbol in Σ, it advances its input head, enters a new state, and outputs a string in Γ^*. That is, M's transition is of the form $(q', w) \in \delta(q, a)$, where q, q' are states, a is an input symbol, and $w \in \Gamma^*$ is an output string. Suppose $(q_1, 000) \in \delta(q_0, a)$ and $(q_2, 1111) \in \delta(q_1, b)$, where q_0 and q_2 are the initial and final states, respectively. Then M outputs 0001111 upon accepting ab, and we say the pair $(ab, 0001111) \in R(M)$. Formally, $R(M) = \{(x, y) \mid x \in \Sigma^*, y \in \Gamma^*, M$ outputs y upon accepting $x\}$.

   - Given FAOs M_1 and M_2, is it decidable whether $R(M_1) \cap R(M_2) = \emptyset$? Justify your answer.

   Sol. Given an instance of PCP $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where $x_i, y_i \in \Sigma^*$, we construct M_1 and M_2 over input alphabet $\{\sigma_i \mid 1 \le i \le n\}$, in the following way:

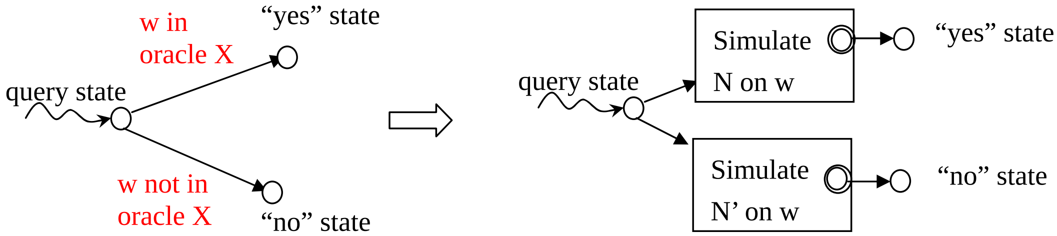
   (a) M_1: read σ_i, write x_i

   (b) M_2: read σ_i, write y_i

   It is not hard to see that the PCP has a match iff $R(M_1) \cap R(M_2) \neq \emptyset$.

7. (10 pts) Give a convincing argument to show that $BPP \subseteq PSPACE$, where BPP is the class of *Bounded-Error Polynomial Probabilistic Time.*
Sol. BPP is contained within PSPACE, because a deterministic poly-space machine can simulate a probabilistic poly-time machine on all possible random sequences, calculate the probability that the probabilistic machine will accept the input, and give its output based on whether this is $\geq 2/3$ or $\leq 1/3$.

8. (10 pts) Give a convincing argument to show that if $NP = co\text{-}NP$, then $NP^{NP} = NP$. Recall that NP^{NP} is the class of languages that can be accepted by nondeterministic polynomial-time oracle Turing machines using languages in NP as oracles. (Hint: can you replace a query to the oracle by simulating the computation of a nondeterministic Turing machine operating in polynomial time?)
Sol. Consider a language in NP^{NP} accepted by a nondeterministic polynomial-time OTM M using oracle set O (in NP). As $NP = co - NP$, O (resp., $\Sigma^* \backslash O$, i.e., the complement of O) can be accepted by a polynomial-time NTM N (resp., N'). Whenever M encounters a query state with w on its query tape, instead of inquiring oracle O, M triggers N and N' using w as their inputs. If N accepts, enters "yes" state of M; if N' accepts, enters "no" state of M. By doing so, there is no need to ask oracle O, as the inquiry can be simulated faithfully as shown in the following figure.



9. (10 pts) Prove that the following language is NP-complete.
$L = \{(\langle M \rangle, x, 1^t) : \exists y \in \{0,1\}^*, |y| \leq t, M(x,y) = 1, M \text{ halts after} \leq t \text{ steps}\}$.
To this end, you must show $L \in NP$ and L is NP-hard. In the definition of L, M is a deterministic Turing machine (i.e., a "verifier") treating y as a "certificate". You may think of $M(x,y) = 1$ as M accepts given x, y.
Sol.

- ($\in NP$) Consider the following NTM N, which on input $(\langle M \rangle, x, 1^t)$, nondeterministically chooses a $y \in \{0,1\}^*$, $|y| \leq t$, simulates M on (x,y) for at most t steps. If M halts, accepts; otherwise, reject. Then clearly $L(N) = L$.
- ($NP-hard$) Given an arbitrary language $A \in NP$ which is accepted by an NTM M in $p(n)$ time, for some polynomial $p(n)$. Given an $w \in \Sigma^*$, we define the following mapping f such that $f(w) = (\langle M \rangle, w, 1^{p(|w|)})$. Clearly, $w \in A$ iff $(\langle M \rangle, w, 1^{p(|w|)}) \in L$. Hence, $A \leq_m^p L$.

10. (10 pts) Suppose a language $L \in NP$ is proved to be $EXPTIME$-complete, answer the following two questions. Here $EXPTIME$ stands for deterministic exponential time.

(a) Is it necessary that L is NP complete? Why?
Sol.: Yes. If an NP-complete problem is EXP-complete, then $NP = PSPACE = EXP$ and every EXP-complete problem is NP-complete as well (and PSPACE which was between the two classes get sandwiched).

(b) Can we conclude that $P = PSPACE$ or $P \neq PSPACE$? Why?
Sol.: $P \neq PSPACE$. Since $P \neq EXPTIME$ by time hierarchy theorem, and the hypothesis implies $NP = PSPACE = EXPTIME$, it follows that $P \neq PSPACE$.