Theory of Computation

Fall 2018, Final Exam. Solutions

Jan. 8, 2019

1. (20 pts) True or False? Justify your answer in a brief yet convincing way.

   (1) $\{\langle D_1, D_2\rangle \mid \text{DFAs } D_1 \text{ and } D_2 \text{ accept a string in common}\}$ is decidable.
   Solution: True. Decidable: create a DFA for $L(D_1) \cap L(D_2)$ using the Cartesian-product construction and check whether an accept state is reachable from the start state.

   (2) $\{\langle G, w\rangle \mid \text{CFG } G \text{ generates a string that starts with } w \text{ as its prefix}\}$ is decidable.
   Solution: True. Decidable. Construct a PDA for $L(G)$ and a DFA for the regular language $w\Sigma^*$, then combine them into a PDA for $L(G) \cap w\Sigma^*$ using the Cartesian-product construction; finally, convert the resulting PDA into a grammar and test whether the grammar generates any strings (using the algorithm from class).

   (3) $\{\langle M\rangle \mid L(M) \in PSPACE\} \in PSPACE$.
   Solution: False. Undecidable. Use Rice's theorem, which asserts that every nontrivial property of the language of a Turing machine is undecidable.

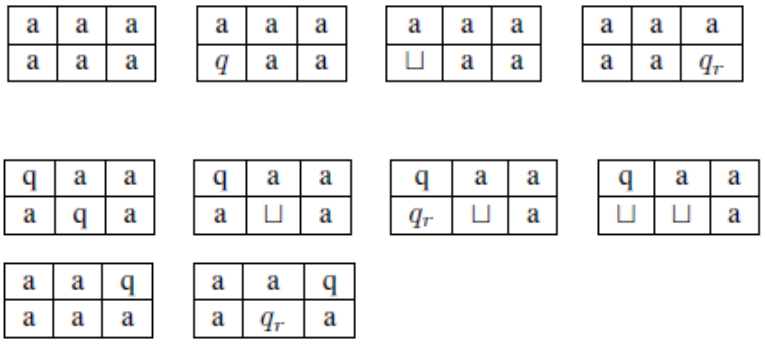
   (4) $\{\langle M, q\rangle \mid \text{the Turing machine } M \text{ enters the state } q \text{ on some input}\}$ is decidable.
   Solution: False. Undecidable. Let L denote the language in question. If L were decidable, then we would be able to check whether the language of any given Turing machine M is empty by running L's decider on $\langle M, q_{accept}\rangle$. This would contradict Rice's theorem, which asserts that every nontrivial property of the language of a Turing machine (such as emptiness) is undecidable.

   (5) $\{d \mid \text{the digit } d \text{ appears infinitely often in the decimal expansion of } \pi = 3.14159...\}$ is decidable.
   Solution: True. Decidable. The language is finite, and hence, regular.

2. (20 pts) Consider a (one-way infinite) nondeterministic Turing machine M over the input alphabet $\Sigma = \{a\}$, tape alphabet $\Gamma = \{a, \sqcup\}$, states $Q = \{q, q_{acc}, q_{rej}\}$ and the following transition function δ:

$$\delta(q, a) = \{(q, a, R), (q_{rej}, \sqcup, L)\}, \quad \delta(q, \sqcup) = \{(q_{acc}, a, R)\}.$$

   Fill in all legal windows with the following chosen first rows. The number of windows in every row indicates how many possible legal windows you should be able to find. Notice that if a left move is made at the leftmost tape symbol, the head does not move.
   Solution:



3. (10 pts) Prove formally that $L = \{\langle M\rangle \mid M \text{ is a Turing machine, and the language of } M \text{ is } \{(010)^n \mid n \geq 1\}\}$ is undecidable. Do not use Rice's theorem.
   Solution: Assume there is a TM M_L deciding the given language L. We use a reduction from A_{TM} to show a contradiction for the existence of such a M_L. On the input $\langle M, w\rangle$, the mapping (i.e., a TM M_f) does the following to construct a TM M_w:

   (1) On the input x does the following:
      (a) simulates the run of M on w;
      (b) if M rejects, - M_w rejects;

(c) if M accepts, - M_w checks whether x has the form $(010)^n$, $n \geq 1$. If yes, - accepts. If not, - rejects.

(2) Uses M_L to determine whether $L(M_w) = \{(010)^n \mid n \geq 1\}$. If yes, - accepts. If not, - rejects.

4. (5 pts) Show that every infinite Turing-recognizable language has an infinite Turing-decidable subset. (Hint: Recall that a langauge is Turing-recognizable iff there is a Turing machine that enumerates the language.)
Solution: It is known that every Turing-recognizable language has an enumerator that enumerates all the strings of the language. Let $W = w_1, w_2, \cdots w_n \cdots$ be the output sequence of the enumerator. We construct the following subsequence $X = x_1, x_2, \cdots$ iteratively: Initially, let $x_1 = w_1$. For every $i > 1$, if $|w_i|$ does not exceed the longest string in X, drop w_i; otherwise add w_i to X, and then repeat the above procedure by considering the next string w_{i+1}. It is clear that X is an infinite sequence of increasing length, which corresponds to a decidable language.

5. (5 pts) Show that if $P = NP$ then the language $L = \{0^n 1^n \mid n \geq 0\}$ is NP-complete (under $\leq_p$) . The $L \in NP$ is simple. You only have to show L to be NP-hard, i.e., for every language $A \in NP(= P)$, $A \leq_p L$.
Solution: For every language $A \in NP(= P)$ over alphabet Σ, there is a DTM M_A accepting A in polynomial time. We consider the following mapping (i.e., a DTM M_f) from Σ^* to $\{0,1\}^*$. If $w \in L(M_A)$, then $f(w) = 01 \in L$; otherwise, $f(w) = 10 \notin L$. The DTM M_f runs in polynomial time as it can be constructed by slightly modifying M_A. Hence, we have $A \leq_p L$.

6. (10 pts) Prove that the class P is closed under Kleene star, i.e., if $A \in P$, then $A^* \in P$. (Hint: use dynamic programming. Given a string w, we let $w_{i,j} = w_i \cdots w_j$ denote the substring of $w = w_1 w_2 \cdots w_n$ starting with w_i and ending with w_j. Let $table(i,j) = true$ if $w_{i,j} \in A^*$.) Be sure to analyze the running time.
Solution:
Let $A \in$ P. We want to show that $A^* \in$ P. Since $A \in$ P there exists a deterministic Turing machine M_A with time complexity $O(n^k)$ for some $k \geq 0$.

We now build, using M_A, a deterministic decider for A^* and show that its time complexity is bounded by a polynomial. The central observation in our construction is that $w \in A^*$ if and only if one of the following conditions is true

- $w = \varepsilon$, or
- $w \in A$, or
- $\exists u, v : w = uv$ and $u \in A^*$ and $v \in A^*$.

In the decider described below we let $w_{i,j}$ denote the substring of $w = w_1 w_2 \ldots w_n$ starting with w_i and ending with w_j. The decider builds a table where $table(i,j) = true$ if $w_{i,j} \in A^*$. We do this by considering all substrings of w starting with substrings of length 1 and ending with the substring of length n.

"On input $w = w_1 w_2 \ldots w_n$:
1. If $w = \varepsilon$ then accept, else
2. For $\ell := 1$ to n
3. For $i := 1$ to $n - (\ell - 1)$
4. $j := i + \ell - 1$
5. Run M_A on $w_{i,j}$
6. If M_A accepts $w_{i,j}$ then $table(i,j) := true$
7. Else
8. For $k := i$ to $j - 1$
9. If $table(i,k) = true$ and $table(k+1,j) = true$
10. then $table(i,j) := true$
11. If $table(1,n) = true$ then accept, else reject."

We now analyze the complexity of our decider. The algorithm uses three nested loops, each of which can be traversed at most $O(n)$ times. In the second loop we run M_A on an input of length at most n, so the total time is at most $O(n) \cdot O(n) \cdot (O(n^k) + O(n)) = O(n^{2+(\max(k,1))})$ steps, which is polynomial in n.

7. (18 pts) For each of the following languages, decide whether it is **(1) recursive, (2) recursively enumerable but not recursive, (3) not recursively enumerable**. Justify your answer formally.

- $L_1 = \{\langle M \rangle \mid M$ is a TM and $|L(M)| \leq 3\}$.
  Solution: (3) Not recursively enumerable.
  We prove this by a reduction from the complement of the halting problem $\overline{HP}$. $f(\langle M, x\rangle) = \langle M'\rangle$, where M' on input w: it erases its input, copies M and x to its tape, and runs M on x; it accepts if M halts on x. We now prove the validity of reduction:
  - $\langle M, x\rangle \in \overline{HP} \Rightarrow M$ does not halt on $x \Rightarrow M'$ does not accept any input $\Rightarrow |L(M') \leq 3 \Rightarrow M' \in L_1$.
  - $\langle M, x\rangle \notin \overline{HP} \Rightarrow M$ halts on $x \Rightarrow M'$ accepts all input $\Rightarrow |L(M') > 3 \Rightarrow M' \notin L_1$.
- $L_2 = \{\langle M \rangle \mid M$ is a TM and $|L(M)| > 3\}$.
  Solution: (2) Recursively enumerable but not recursive
  - Recursively enumerable. The acceptor runs M on all inputs in an interleaved mode, and halts whenever 3 inputs have been accepted. Notice that the acceptor generates the input strings for M one by one as they are needed. Another way to think of the acceptor is to nondeterministically generate 3 input strings and simulate the 3 strings one by one and accepts if all 3 inputs are accepted.
  - Not recursive. Follows from Rice's theorem.
- $L_3 = \{\langle M \rangle \mid M$ is a TM and there exists an input on which M halts in less than $|\langle M\rangle|$ steps$\}$.
  Solution: (1) Recursive.
  The TM that decides the language works as follows on input $\langle M\rangle$. It first finds the length of $\langle M\rangle$, and stores it. Then, it runs M on all inputs of length at most $\langle M\rangle$, for at most $\langle M\rangle$ steps, and accepts if M accepts at least one of the strings within the specified number of steps.

8. (7 pts) Prove formally that $f(n) = 1 + 2 + \cdots + n$ is primitive recursive. You may assume that functions $x + y,\ x \cdot y,\ x^y,\ x - y,\ sign(x),\ compare_{\leq}(x, y)$ are p.r.
   Solution:
   $$f(n) = f'(0, n) \qquad f'(x, 0) = g(x) \qquad f'(x, S(n)) = h(x, f'(x, n), n)$$
   Here the function $g(x) = 0$, $h(x, y, z) = \pi_2(x, y, z) + S(\pi_3(x, y, z))$.

9. (5 pts) In the *silly Post Correspondence Problem*, SPCP, in each pair the top string has the same length as the bottom string. Is the SPCP decidable? Why?
   Solution: Decidable. SPCP has a solution iff there is a pair (x, y) such that $x = y$.