Theory of Computation

Spring 2020, Final Exam (Solutions)

June 23, 2020

1. (20 pts) True or False? No penalty for wrong answers. (Note that $\leq_P$ stands for polynomial-time many-one reduction and $\leq_m$ stands for many-one reduction.)

   (1) The language $\{a^i b^j c^i \mid i \leq j \leq 2i\}$ is not context-free.
   Sol. O

   (2) The class of non-context-free languages is closed under complementation.
   Sol. $\times$
   $\{ww \mid w \in \{0,1\}^*\}$ is non-context-free, but its complement is CFL

   (3) If A is Turing-recognizable and $A \leq_m \overline{A}$, then A is Turing-decidable.
   Sol. O

   (4) If L_1 and L_2 are NP-complete, then $L_1 \leq_P L_2$ and $L_2 \leq_P L_1$.
   Sol. O
   It follows directly from the definition of NP-completeness.

   (5) If L_1 and L_2 are NP-complete, then $L_1 \leq_m L_2$ and $L_2 \leq_m L_1$.
   Sol. O
   Surely, if $L_1 \leq_P L_2$ then also $L_1 \leq_m L_2$.

   (6) If $L_1 \leq_P L_2$, $L_2 \leq_P L_1$, and $L_1, L_2 \in NP$, then L_1 and L_2 are both NP-complete.
   Sol. $\times$
   Let $L_1 = L_2 = \emptyset$. Surely $\emptyset \leq_P \emptyset$ by a reduction that is e.g. the identity function but $\emptyset$ cannot be NP-complete (because none of the languages in NP, except for $\emptyset$ itself, are reducible to $\emptyset$).

   (7) If L_1 is NP-complete and $L_1 \leq_P L_2$, then L_2 is NP-complete.
   Sol. $\times$
   We only know that L_2 is NP-hard.

   (8) NP is the class of languages that cannot be decided in polynomial time using deterministic Turing machines.
   Sol. $\times$

   (9) co-NP $\subseteq$ EXPTIME.
   Sol. O

   (10) If $L \in$ P, then $L^* \in$ P as well.
   Sol. O

   (11) $L = \{\langle M, w\rangle \mid M$ accepts w in less than 100 steps$\}$ is decidable.
   Sol. O

   (12) If A is recursive and $A \leq_P B$, then B must be recursive.
   Sol. $\times$

   (13) The problem of determining if a context-free grammar generates the empty language is undecidable.
   Sol. $\times$

   (14) The class of Turing-recognizable languages is closed under intersection.
   Sol. O

   (15) The set of Turing-recognizable languages is a countably infinite set (i.e., there exists a one-to-one correspondence between the set and the set of natural numbers).
   Sol. O

   (16) Suppose L_1 is context-free and L_2 is regular, then the problem of deciding whether $L_1 \subseteq L_2$ is decidable.
   Sol. O

   (17) Primitive recursive functions are those that can be computed by Turing machines that always halt.
   Sol. $\times$ Ackermann function is a total recursive function which is not primitive recursive.

   (18) It is possible for some undecidable language to be NP-Complete.
   Sol. $\times$

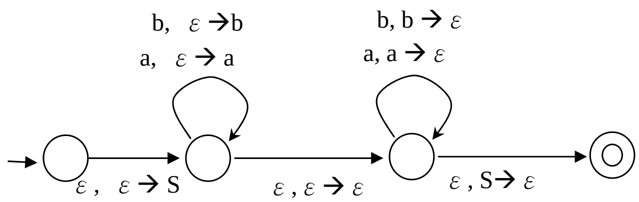
(19) The language $L = \{\langle M, w\rangle \mid$ TM M moves right exactly twice while operating on $w\}$ is decidable.
Sol. O
If it can move right only twice, then M can read only the first two input characters.

(20) $NSPACE(\log^2 n) \subseteq P$.
Sol. O

2. (10 pts) Let $\Sigma = \{a, b\}$, and consider the language $A = \{w \in \Sigma^* \mid w = w^R, |w| \text{ is even}\}$, where w^R denotes the reverse of w and $|w|$ denotes the length of w. For instance, $aabbaa \in A$.

   (a) Give a CFG G for A. Be sure to specify G as a 4-tuple $G = (V, \Sigma, R, S)$.
   Sol. $S \to aSa \mid bSb \mid \epsilon$

   (b) Give a PDA for A. You only need to give the drawing.
   Sol.



3. (10 pts) Consider the following context-free grammar G in Chomsky normal form:

$$S \to AA \mid \epsilon$$
$$A \to BB \mid AB \mid a$$
$$B \to BA \mid b$$

In CYK parsing algorithm, given a $w = a_1 \dots a_n$, we define $t_{ij} = \{A \mid A \overset{*}{\Rightarrow} a_i \dots a_j\}$. Fill in the blanks in the following table in the process of parsing $w = abba$.

t_{ij}	1	2	3	4
1				
2	-			
3	-	-		
4	-	-	-	
	a	b	b	a

Sol.

t_{ij}	1	2	3	4
1	A	A	S, A	S, A
2	-	B	A	S, A
3	-	-	B	B
4	-	-	-	A

4. (10 pts) Show that if P=NP then every language $B \in P$, except for $\emptyset$ and Σ^*, is NP-complete. (Hint: Show that for every language $A \in$ P=NP, $A \leq_P B$.)
Sol. Assume a polynomial-time TM M_A decides A.

Assume that P=NP. Let $B \in$ P such that $B \neq \emptyset$ and $B \neq \Sigma^*$. This means that there is a string $w_{in} \in B$ and a string $w_{out} \notin B$. We want to show that B is NP-complete.

Surely, the language B is in NP=P (by our assumption). We have to show that B is NP-complete. Let A be an arbitrary language from NP=P. Hence A has a polynomial time decider M_A. We need to argue that $A \leq_P B$. Here is a poly-time reduction f from A to B:

"On input w:
1. Run M_A (decider for A) on w.
2. If M_A accepted then output w_{in}. If M_A rejected then output w_{out}."

This is a poly-time reduction from A to B, and hence B is NP-complete.

5. (12 pts) Consider the following classes of languages as (1)-(7).
(1) Finite, (2) Regular, (3) Context-free, (4) Context-sensitive, (5) Recursive, (6) Recursively enumerable, (7) All possible languages.
For each of the following languages, specify the *lowest-numbered class* to which it surely belongs. For example, for a context-free language L that is not regular, the right answer is (3), although L clearly belongs to all classes of languages larger than (3). Similarly, suppose L is recursively enumerable, the right answer is (6), although L could possibly be recursive but the available information does not guarantee that.

(a) .... The complement of an undecidable language.
Sol. 7 Take an r.e. but not recursive language, whose complement is not r.e.

(b) .... The complement of a language in NP.
Sol. 5; $NP \subseteq PSPACE$. $PSPACE$ is recursive.

(c) .... The intersection of two context-free languages.
Sol. 4 Context-free languages are also context-sensitive. Context-sensitive languages are closed under intersection.

(d) .... The complement of a context-sensitive language.
Sol. 4 Context-sensitive language are closed under complementation (due to Immerman theorem).

(e) .... The intersection of a recursive language and a language that is not recursively enumerable.
Sol. 7 Let the recursive language be Σ^*.

(f) .... The intersection of a recursive language and a recursively enumerable language.
Sol. 6 r.e. $\cap$ recursive is r.e.

6. (10 pts) Consider $E_{TM} = \{\langle M \rangle \mid M$ is a TM with $L(M) = \emptyset\}$. It is known that E_{TM} is not recursive. Answer the following question:

(a) Is E_{TM} co-Turing-recognizable? Why?
Sol. Yes. Design a TM M' that nondeterministically guesses an input x and simulates M on x, accepts if M accepts x. Clearly, M' accepts $\overline{E_{TM}}$; hence, E_{TM} co-Turing-recognizable.

(b) Does $E_{TM} \leq_m A_{TM}$ hold? Why? Recall that $A_{TM} = \{\langle M, w\rangle \mid M$ is a TM that accepts $w\}$.
Sol. No. If $E_{TM} \leq_m A_{TM}$, then E_{TM} is Turing-recognizable. This, together with (a) above, implies Tring-decidability of E_{TM}, which is known to be false.

7. (10 pts) Define a *two-headed finite automaton* (2DFA) to be a deterministic finite automaton that has two read-only, bidirectional (i.e., two-way) heads that start at the left-hand end of the input tape and can be independently controlled to move in either direction. The tape of a 2DFA is finite and is just large enough to contain the input plus two additional blank tape cells, one on the left-hand end and one on the right-hand end, that serve as delimiters (i.e., end-markers). A 2DFA accepts its input by entering a special accept state.

   (a) Explain in a convincing way how a 2DFA can recognize the language $\{a^n b^n c^n \mid n \geq 0\}$.
   Sol. Let the two heads be h_1 and h_2. Assume that initially both are scanning the first input symbol.

      i. Move h_2 to the beginning of b, while keeping h_1 intact;
      ii. Move both h_1 and h_2 to the right if h_1 reads an a and h_2 reads a b; repeat until h_1 sees a b and h_2 sees a c simultaneously, then go to the next step. (This step is to compare the number of as with the number of bs.)
      iii. Move both h_1 and h_2 to the right if h_1 reads an b and h_2 reads a c; repeat until h_1 sees a c and h_2 sees the right endmarker simultaneously, then accepts. (This step is to compare the number of bs with the number of cs.)

   (b) Let $E_{2DFA} = \{\langle M \rangle \mid M \text{ is a 2DFA and } L(M) = \emptyset\}$. Explain in a convincing way how to use the undecidability of PCP to show that E_{2DFA} is not decidable. (Hint: suppose $P = \{(x_1, y_1), ..., (x_n, y_n)\}$ is an instance of a PCP. Can you design a 2DFA M such that $L(M) \neq \emptyset$ iff P has a match?)
   Sol. Suppose the alphabet of $P = \{(x_0, y_0), ..., (x_n, y_n)\}$ is $\Sigma = \{a, b\}$. Design a E_{2DFA} with alphabet $\{0, 1, \#, a, b\}$ which operate in the following way:

      i. Check if the input w is of the form $(\{0,1\}^+ \cdot \#)^* \{a,b\}^*$; reject if otherwise. Reset the two heads to the leftmost position.
      ii. For an input, e.g., $011\#101\#0\#abbaaabab$, the E_{2DFA} accepts if $abbaaabab = x_3 x_5 x_0 = y_3 y_5 y_0$, which can be done by
         A. using the first head h_1 to read 011, find x_3 (kept in the finite state control of the 2DFA), compare x_3 with the prefix of *abbaaabab* scanned by the second head h_2. If successful, repeat the above by letting h_1 read the second index, i.e., 101 in our case, and h_2 compare x_5 with the remainder of the input, and so on ... until the input is completely read. Then reset both heads to the leftmost position, and go to the next step.
         B. As in the previous step, use h_1 to find the index i and h_2 to check whether y_i matches the corresponding part in *abbaaabab*.

   Clearly, the 2DFA accepts iff the PCP has a match.

8. (10 pts) True or False? Justify your answers.

   (a) Suppose L is TM-recognizable but not TM-decidable. Then any TM that recognizes L must fail to halt on an infinite number of strings.
   Sol. True. If not we can imagine a decider for L: first compare input to the elements of that finite set and if it is there reject. Otherwise simulate "recognizer" of L on the input and return what it returns. Note that it just shows that such a decider exists, it is not a recipe of how to construct it since we don't have a representation of that finite set.

   (b) Suppose A and B are recursively enumerable languages such that $A \cup B$ and $A \cap B$ are both decidable (i.e., recursive). Then A is decidable.
   Sol. True.

Let M_A and M_B be TMs recognizing A and B, respectively. Let $M_{A\cup B}$ and $M_{A\cap B}$ be decision procedures for $A \cup B$ and $A \cap B$, respectively. An algorithm for A is as follows.

```
On input x
Run M_{A∪B} on x
If M_{A∪B} rejects then reject (and halt)
else /*** x is in A ∪ B ***/
   Run M_{A∩B} on x
   If M_{A∩B} accepts then accept (and halt)
   else /*** x ∈ (A ∪ B) \ (A ∩ B) ***/
      Run M_A and M_B in parallel (using dovetailing) on x
      If M_A accepts then accept (and halt)
      else if M_B accepts then reject (and halt)
```

The main observation is that if on x, $M_{A\cup B}$ accepts and $M_{A\cap B}$ rejects, then x belongs to exactly one out of A and B. Thus, exactly one of the simulations of M_A and M_B will accept, and whichever one terminates first, we know whether x belongs to A or not.

9. (8 pts) A set L is r.e. iff there is a recursive predicate (computable by a TM that always halts) R such that $L = \{x \mid \exists y : R(x, y)\}$. L is co-r.e. iff there is a recursive predicate R such that $L = \{x \mid \forall y : R(x, y)\}$. By counting the number of alternating quantifiers, you actually get a measure of difficulty.

   Consider $HALT_{TM} = \{\langle M, x\rangle \mid M \text{ halts on } x\}$. As $HALT_{TM}$ can be rewritten as $\{\langle M, x\rangle \mid \exists i : M \text{ halts in } i \text{ steps on } x\}$, the corresponding R can be defined as $R(M, x, i)=$ *true* if M (on x) halts in i steps; *false*, otherwise. Likewise, $E_{TM} = \{\langle M\rangle \mid L(M) = \emptyset\}$ can be written as $\{\langle M\rangle \mid \forall x \forall i : M \text{ does not accept in } i \text{ steps on } x\}$.

   (a) Consider $ALL_{TM} = \{\langle M\rangle \mid L(M) = \Sigma^*\}$. Show that $L(M) = \Sigma^*$ can be expressed as $\forall..\exists..R(..)$. Complete the detail of the above logical formula.
   Sol. $\forall x, \exists i, M$ accepts x in i steps.

   (b) Consider $FINITE_{TM} = \{\langle M\rangle \mid L(M) \text{ is finite}\}$. Express the condition "$L(M)$ is finite" using a logical formula.
   Sol. $\exists i \forall j \forall y, |y| > i, M$ does not accept y in j steps.