Theory of Computation

Fall 2017, Final Exam. Solutions (January 9, 2018)

1. (10 pts) Give an example that witnesses each of the following. No need to give the proof.

   (a) L_1 and L_2 both are *r.e.* but not *recursive*, $L_1 \cap L_2$ is recursive.
   Solution: Pick two non-recursive r.e. languages over disjoint alphabets. For instance, PCP (over Σ) $\cap$ A_{TM} (over Σ') $= \emptyset$, when $\Sigma \cap \Sigma' = \emptyset$.

   (b) L_1 and L_2 are context free, $L_1 \cap L_2$ is not context free.
   Solution: $\{a^n b^n c^m \mid n, m \geq 0\} \cap \{a^n b^m c^m \mid n, m \geq 0\} = \{a^n b^n c^n \mid n \geq 0\}$

   (c) A language L which is context-free but not deterministic context-free.
   Solution: $\{ww^R \mid w \in \Sigma^*\}$

   (d) Give an example to witness that *r.e.* languages are not closed under complement.
   Solution: A_{TM}

   (e) Define $md(L) = \{x \mid w \in L \ (w = sxt, s \in \Sigma^*, x \in \Sigma^*, t \in \Sigma^*)\}$. If L is context free but not regular, $md(L)$ may be regular.
   Solution: $md(\{a^n b^n \mid n \geq 0\} = \{a^i b^j \mid i, j \geq 0\}$.

2. (15 pts) Use the CYK algorithm on the strings 010010 for the grammar below to determine whether it belongs to the language or not. You must exhibit the following table (i.e., fill out those blanks) correctly to receive credit. Recall that the CYK algorithm is based on dynamic programming computing $X_{i,j} = \{A \mid A \overset{*}{\Rightarrow} a_i \cdots a_j\}$ in a bottom up fashion. Assume $010010 = a_1 a_2 a_3 a_4 a_5 a_6$ and $X_{1,1}, X_{2,2}, X_{3,3}, X_{4,4}, X_{5,5}, X_{6,6}$ are given in the table.

$S \to AB \mid BC$
$A \to BA \mid 0$
$B \to CC \mid 1$
$C \to AB \mid 0$

$\{B\}$	X	X	X	X	X
$\{S, C\}$	$\{S, C, A\}$	X	X	X	X
$\{A, S\}$	$\emptyset$	$\{A, S, C\}$	X	X	X
$\{B\}$	$\emptyset$	$\{B\}$	$\{B\}$	X	X
$\{S, C\}$	$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$	X
$\{A, C\}$	$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
0	1	0	0	1	0

3. (5 pts) A pushdown automaton M is $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \to 2^{Q \times (\Gamma \cup \{\epsilon\})}$ is the transition function. Note that a PDA is a nondeterministic device. To make M *deterministic*, what are the additional constraints needed to be imposed on δ? That is, define the transition function for a deterministic PDA.
Solution: $|\delta(q, x, z)| + |\delta(q, \epsilon, z)| \leq 1, x \in \Sigma$

4. (10 pts) Let $L = \{0^i 1^j 2^k \mid i \leq k \text{ or } j \leq k\}$.

   (a) Is L a CFL? Explain why?
   Solution: Yes. A PDA nondeterministically checks whether $i \leq k$ or $j \leq k$ can be designed.

   (b) What is $min(L)$? Recall that $min(L) = \{x \in L \mid \text{ no proper prefix of } x \text{ is in } L\}$.
   Solution: $min(L) = \{0^i 1^j 2^k \mid k = min(i, j)\}$.
   If you answer $\{\epsilon\}$, you will also receive full credit.

5. (20 pts) Regarding whether a language class is closed with respect to an operation, fill in O (for "yes") or $\times$ (for "no") for each of the blanks in the following table.

	DCFL	CFL	Recursive	r.e.
Complementation	O	X	O	X
Intersection	X	X	O	O
Union	X	O	O	O
Reversal	X	O	O	O
Concatenation	X	O	O	O

NOTE:

- (DCFL not closed under union) $\{a^i b^j c^k \mid i < j\} \cup \{a^i b^j c^k \mid i < k\}$ is NOT a DCFL.
- (DCFL not closed under concatenation) $\{a^m b^n \mid m < n\} \cdot \{wcw^R \mid w \in \{a,b\}^*\}$ is NOT a DCFL. This string $abbb \cdot bbbacabbb$ is accepted, but how would you know on which b you should "jump across" from one language to the other and stop consuming the b and start stacking it?
- (DCFL not closed under reversal) $L = 0 \cdot \{a^i b^j c^k \mid i < j\} \cup 1 \cdot \{a^i b^j c^k \mid i < k\}$ is a DCFL, for 0 and 1 can be used to guide the DPDA to check $i < j$ and $i < k$, resp. However, for L^R, the 0 and 1 won't be seen until reaching the end of the string. L^R is NOT a DCFL.

6. (10 pts) Prove that PCP is undecidable even if we restrict its alphabet to two symbols, for example $\{0, 1\}$. To this end, suppose PCP_r is such a restricted PCP. Show PCP $\leq_m$ PCP_r in detail. You may further assume the alphabet of an PCP instance to be $\Sigma = \{a_1, a_2, ..., a_k\}$, for some k.
Solution: Map a_i to $0^i 1$, $1 \leq i \leq k$.

7. (10 pts) Prove the following results:

   (a) There is a regular language L such that for every language A, A is recursive $\Leftrightarrow$ $A \leq_m L$.
   Solution: $f(w) = 1$ if $w \in A$; else $f(w) = 0$. Hence $A \leq_m \{1\}$.

   (b) If $L \in NP$ (nondeterministic polynomial time), then $L^* \in NP$.
   Solution: Let M be a polynomial-time TM accepting L. If $w \in L^*$, $w = w_1 w_2 \cdots w_k, w_i \in L$, run M on $w_1, w_2, ..., w_k$ sequentially, and the running time is $P(|w_1|) + P(|w_2|) + ... + P(|w_k|)$, which is $\leq |w| P(|w|)$

8. (10 pts) Prove the following two statements:

   (a) Suppose A and B are recursively enumerable languages such that $A \cup B$ and $A \cap B$ are both decidable. Prove that A is decidable.
   Solution: Given a w,
   - if $w \in A \cap B$, then accept ($w \in A$); otherwise,
   - if $w \in A \cup B$, run the TMs M_A and M_B recognizing A and B, resp., in parallel. If M_B accepts, then reject; if M_A accepts, then accept.

   (b) Suppose A is recursively enumerable and $A \leq_m \overline{A}$. Prove that A is decidable.
   Solution: $A \leq_m \overline{A} \Rightarrow \overline{A} \leq_m \overline{\overline{A}} = A$; hence, $\overline{A}$ is also r.e., which implies A is recursive.

9. (10 pts) A log space computable function f is a DTM M_f with a read-only input tape, a write-only output tape, and a read-write work tape. The work tape may contain $O(\log n)$ symbols, and the TM output $f(w)$ on input w. A language A is log space reducible to a language B (written $A \leq_L B$) if there is a log space computable function f such that $w \in A$ if and only if $f(w) \in B$ for every w. Suppose we want to prove the following theorem:

$$\text{If } A \leq_L B \text{ and } B \in L \text{ (i.e., } B \text{ is in } DSPACE(O(\log n))\text{), then } A \in L.$$

Consider the following "proof":

Since $B \in L$, there exists a DTM M_B running in $DSPACE(O(\log n))$ to accept B. To show $A \in L$, we create a new TM M_A that given an input w, first runs the TM M_f guaranteed by the log-space reduction implied by $\leq_L$ to get $f(w)$, and then run M_B on $f(w)$. M_A accepts w if and only if M_B accepts $f(w)$.

   (a) Explain why the above "proof" is NOT correct.
   Solution: The length of $f(w)$ could be $2^{\log n}$, which is no longer logarithmic. Hence, M_A, which combines M_f and M_B, may require $O(n)$ space.

(b) Can you come up with a fix for the proof? An informal explanation of your fix is sufficient.
Solution: M_A uses a tape of length $O(n)$ to keep track of the position of $f(w)$ at which M_B is reading. To find what the symbol is, M_A always asks M_f to re-compute $f(w)$ from the beginning until the symbol is output.