Midterm 2
CS 373: Theory of Computation

Date: Thursday, March 28, 2013.

Instructions:

- This is a closed book exam. No notes, cheat sheets, textbook, or printed material allowed.
- You have 90 minutes to solve this exam.
- This exam has 4 problems. Problems 1,2 and 4 are worth 10 points, while problem 3 is worth 20 points. The points are not a measure of the relative difficulty of the problems.
- Please write your name on the top of *every* page in the space provided.
- If your solution does not fit in the space provided, and continues onto one of the back sheets, please indicate clearly where we should look for the solution.
- Unless otherwise stated, "prove that", "show that" for a problem means you need to formally prove what you are claiming.
- You may use, without proof, any result that you were asked to prove in the homework or was proved in the lecture. If you use such a result, please explicitly state the result (like " 'Reverse of regular languages is regular' was proved in a homework", instead of "this was shown in a homework").

Name	SOLUTIONS
Netid	solutions

Discussion:	W 10:00–10:50	W 11:00–11:50	W 12:00–12:50
	W 2:00–2:50	W 3:00–3:50	W 4:00–4:50

Problem	Maximum Points	Points Earned	Grader
1	10		
2	10		
3	20		
4	10		
Total	50		

You may use without proof that the following languages are not context-free

- $L_{anbncn} = \{a^n b^n c^n \mid n \geq 0\}$; the alphabet for this language is $\{a, b, c\}$.
- $L_{a=c \wedge b=d} = \{a^i b^j c^i d^j \mid i, j \geq 0\}$; the alphabet for this language is $\{a, b, c, d\}$.
- $E = \{ww \mid w \in \{0,1\}^*\}$; the alphabet for this language is $\{0, 1\}$.
- $A_1 = \{0^n 1^{2n} 0^{2n} 1^n \mid n \geq 0\}$; the alphabet for this language is $\{0, 1\}$.
- $L_{wwrw} = \{ww^R w \mid w \in \{0,1\}^*\}$; the alphabet for this language is $\{0, 1\}$.
- $S = \{a^{n^2} \mid n \geq 0\}$; the alphabet for this is $\{a\}$.

Problem 1. [Category: Comprehension] **True/False.** Decide for each statement whether it is true or false. Circle **T** if the statement is *necessarily true*; circle **F** if it it is not necessarily true. Each correct answer is worth **1 point**.

(a) Let $G = (V, \Sigma, R, S)$ be a CFG. If $\alpha \in (V \cup \Sigma)^*$ with $\alpha \neq S$, then there is no β such that $\alpha \Rightarrow_G \beta$.

False. Derivations are defined from any string of variables and terminal symbols.

(b) The language $L_1 = \{wxw \mid w, x \in \{0, 1\}^*\}$ is context-free.

True. For any string $u \in \{0, 1\}^*$, take $w = \epsilon$, $x = u$, then $u = wxw$ and so $u \in L_1$. Thus, $L_1 = \{0, 1\}^*$.

(c) The language $L_2 = \{a^n b^n a^n \mid n \geq 0\}$ is context-free.

False. Consider the homomorphism $h : \{a, b, c\}^* \to \{a, b\}^*$, with $h(a) = h(c) = a$ and $h(b) = b$. Then $h^{-1}(L_2) \cap \mathbf{L}(a^* b^* c^*) = \{a^n b^n c^n \mid n \geq 0\}$ which is non-CFL. Thus, L_2 is not context-free.

(d) Let $G = (V, \Sigma, R, S)$ be an unambiguous CFG. Every $w \in \mathbf{L}(G)$ has exactly one derivation from S.

False. A grammar is unambiguous if every string has exactly one *parse tree*; strings could have multiple derivations. For example consider the grammar $S \to AB$, $A \to aA \mid \epsilon$, and $B \to bB \mid \epsilon$ is an umbiguous grammar describing $\mathbf{L}(a^* b^*)$ but strings have multiple derivations.

(e) Any PDA must push a bottom of stack symbol before it starts reading any input symbols because no transitions are enabled when the stack is empty.

False. A PDA accepts a string when it reaches an accepting state, *no matter* what its stack is.

(f) Let $G = (V, \Sigma, R, S)$ be a CFG without ϵ-productions, unit productions, and useless variables, where the length of the right-hand-side of any rule in R be at most k. Suppose $G' = (V', \Sigma, R', S')$ is the grammar in Chomsky Normal form constructed by the algorithm discussed in class. The $|R'| = O(k|R| + |\Sigma|)$.

True. To convert G into Chomsky Normal form, we introduce rules of the form $X_a \to a$ (for each $a \in \Sigma$) to ensure that rules whose RHS has length > 1 consist only of variables; this is $O(|\Sigma|)$ rules. Then, to convert a rule whose RHS has length > 2, we introduce additional rules, but the number of rules is bounded by the length of the RHS. Thus, each rule in G introduces at most $O(k)$ new rules in G'. Putting it together we get the result.

(g) Let $G = (V, \Sigma, R, S)$ be a grammar in Chomsky normal form. Suppose $w_1, w_2 \in \mathbf{L}(G)$. Let T_1 be a parse tree with root labelled S and yield w_1, and let T_2 be a parse tree with root S and yield w_2. Then T_1 and T_2 have the same number of vertices.

False. Consider G with rules $S \to a \mid AB$, $A \to a$ and $B \to b$. The strings a and ab have parse trees with different number of vertices. This result would be true if we consider w_1, w_2 such that $|w_1| = |w_2|$.

(h) Suppose L_1 is a finite language and L_2 is not context-free. Then $L_1 L_2$ is not context-free.

False. Take $L_1 = \emptyset$. Then $L_1 L_2 = \emptyset$ (which is regular) no matter what L_2 is.

(i) Since context-free languages are not closed under intersection, if $L_1, L_2 \subseteq \Sigma^*$ are context-free then $L_1 \cap L_2$ cannot be context-free.

False. Non-closure under intersection only implies that there is some CFL languages L_1 and L_2 such that $L_1 \cap L_2$ is non-CFL. We can easily see that the statement does not hold by considering $L_1 = \emptyset$ and L_2 to be any CFL; then $L_1 \cap L_2 = \emptyset$ which is regular and CFL.

(j) Since every regular language is also a context-free language, context-free languages are closed under every operation that regular languages are closed under.

False. Regular languages are closed under complementation/intersection but CFLs are not.

Problem 2. [Category: Comprehension+Proof] Consider the context-free grammar $G = (V = \{S, A\}, \Sigma = \{0, 1\}, R, S)$ where the rules are given by

$$S \to 0S0 \mid 1S1 \mid 0A1 \mid 1A0 \qquad A \to 0A0 \mid 0A1 \mid 1A0 \mid 1A1 \mid 0 \mid 1 \mid \epsilon$$

(a) For each of the following strings, answer whether or not they belong to the language $\mathbf{L}(G)$, and if they do then give a derivation: 101, 100, 1011. **[3 points]**

$101 \notin \mathbf{L}(G)$.
$100 \in \mathbf{L}(G)$: Consider $S \Rightarrow 1A0 \Rightarrow 100$.
$1011 \in \mathbf{L}(G)$: Consider $S \Rightarrow 1S1 \Rightarrow 10A11 \Rightarrow 1011$.

(b) For a variable $C \in V$, define $\mathbf{L}_G(C) = \{w \in \Sigma^* \mid C \stackrel{*}{\Rightarrow} w\}$. Fill in the blanks for each of the variables in G. **[2 points]**

$\mathbf{L}_G(S) = \underline{\{w \in \{0,1\}^* \mid w \neq w^R\}}$
$\mathbf{L}_G(A) = \underline{\{0,1\}^*}$

(c) Prove that your answer for $\mathbf{L}_G(A)$ given in part (b) is correct. **[5 points]**

Observe that $\mathbf{L}_G(A) \subseteq \{0,1\}^*$ because $\{0,1\}^*$ is the set of all strings. We will show that if $w \in \{0,1\}^*$ then $A \stackrel{*}{\Rightarrow} w$ by induction on $|w|$.

Base Case: Consider w such that $|w|$ is 0 or 1. Since $A \Rightarrow \epsilon$, $A \Rightarrow 0$, and $A \Rightarrow 1$, we have established the base case.

Ind. Hyp.: Let us assume that for any w with $|w| < n$, we have $A \stackrel{*}{\Rightarrow} w$.

Ind. Step: Consider w such that $|w| = n$ (with $n > 1$). Without loss of generality, $w = 0u0$, or $w = 0u1$, or $w = 1u0$, or $w = 1u1$, where $|u| = n - 2$. Observe that by induction hypothesis, we have $A \stackrel{*}{\Rightarrow} u$. Consider the case when $w = 0u0$. Then we have $A \Rightarrow 0A0 \stackrel{*}{\Rightarrow} 0u0$. At this point, you could say that the "other cases are similar" and stop. But for completeness, we have the remaining cases as well. When $w = 0u1$, we have $A \Rightarrow 0A1 \stackrel{*}{\Rightarrow} 0u1$. When $w = 1u0$, we have $A \Rightarrow 1A0 \stackrel{*}{\Rightarrow} 1u0$. When $w = 1u1$, we have $A \Rightarrow 1A1 \stackrel{*}{\Rightarrow} 1u1$.

Problem 3. [Category: Design+Comprehension+Proof] Given $L \subseteq \Sigma^*$, define an operation $\mathsf{REFLECT}$ as follows:

$$\mathsf{REFLECT}(L) = \{ww^R \mid w \in L\}$$

where w^R stands for the reverse of the string w. For example, $\mathsf{REFLECT}(\{01, 10\}) = \{0110, 1001\}$ because $0110 = 01(01)^R$ and $1001 = 10(10)^R$.

(a) For $L_1 = \mathbf{L}(0^*1^*)$, what is $\mathsf{REFLECT}(L_1)$? **[1 point]**

Observe that any string $w \in \mathbf{L}(0^*1^*)$ is of the form 0^i1^j, where $i, j \geq 0$. Then for such a string, $ww^R = 0^i1^{2j}0^i$. Thus, $\mathsf{REFLECT}(L_1) = \{0^i1^{2j}0^i \mid i, j \geq 0\}$.

(b) For $L_2 = \{0^n1^n \mid n \geq 0\}$, what is $\mathsf{REFLECT}(L_2)$? **[1 point]**

$\mathsf{REFLECT}(L_2) = \{0^n1^{2n}0^n \mid n \geq 0\}$.

(c) We will show that if L is regular then $\mathsf{REFLECT}(L)$ is context-free. Consider a regular language L recognized by DFA $M = (Q, \Sigma, \delta, q_0, F)$. We will construct a PDA P to recognize $\mathsf{REFLECT}(L)$ as follows. Observe that on input u, P needs to check if there is a w such that $u = ww^R$ and w is accepted by M. P will start in a new initial state p_0, push a bottom of stack symbol \$, and move to the initial state of M without reading any input symbols. After that, P will simulate M on the input, while pushing every symbol it reads onto its stack. It will nondeterministically guess when it has read half of the input. If P is in the final state of M when it guesses that half the input has been read (i.e., the first half is accepted by M), then it will move to a new state p_1, where it will check to see if the second half is the reverse of the first half by popping symbols from the stack as it reads the input. Finally, it will move from p_1 to the new final state p_F (which has no transitions), by popping the bottom of the stack symbol.

Complete the formal description of the PDA P based on the above outline. Let $P = (Q', \Sigma, \Gamma, \delta', q_0', F')$ where

(i) $Q' = \underline{Q \cup \{p_0, p_1, p_F\}}$ **[1 point]**

(ii) $\Gamma = \underline{\Sigma}$ **[1 point]**

(iii) $q_0' = \underline{p_0}$ **[1 point]**

(iv) $F' = \underline{\{p_F\}}$ **[1 point]**

(v) δ' is given by (fill in the 7 blanks to complete the description) **[7 points]**

$$\delta'(q,a,b) = \begin{cases} \underline{\{(q_0,\$)\}} & \text{if } q = p_0,\ a = \epsilon,\ b = \epsilon \\ \underline{\{(q',a)\}} & \text{if } q \in Q,\ a \in \Sigma,\ b = \epsilon, \text{ and } q' = \delta(q,a) \\ \underline{\{(p_1,\epsilon)\}} & \text{if } q \in F,\ a = \epsilon,\ b = \epsilon \\ \underline{\{(p_1,\epsilon)\}} & \text{if } q = p_1,\ a \in \Sigma,\ b = \underline{a} \\ \underline{\{(p_F,\epsilon)\}} & \text{if } q = p_1,\ a = \epsilon,\ b = \$ \\ \underline{\emptyset} & \text{otherwise} \end{cases}$$

(d) Context free languages are not closed under the operation $\mathsf{REFLECT}$. Consider the language $L_2 = \{0^n 1^n \mid n \geq 0\}$ (from part (b)).

(i) Prove that L_2 is context-free by either constructing a grammar or a PDA describing/recognizing L_2. You need not prove that your construction is correct. **[2 points]**

The grammar $G_2 = (\{S\}, \{0,1\}, S, R)$ where R is given by $S \to 0S1 \mid \epsilon$ describes the language L_2.

(ii) Prove that $\mathsf{REFLECT}(L)$ is not context-free. **[5 points]**

Recall that $\mathsf{REFLECT}(L_2) = \{0^n 1^{2n} 0^n \mid n \geq 0\}$. We could prove this is not context-free by using either closure properties or the pumping lemma, the former being much shorter.

Closure Properties: Consider the homomorphism $h : \{a,b,c\}^* \to \{0,1\}^*$, where $h(a) = h(c) = 0$ and $h(b) = 11$. Then $h^{-1}(\mathsf{REFLECT}(L_2)) \cap \mathbf{L}(a^*b^*c^*) = \{a^n b^n c^n \mid n \geq 0\}$. Thus, $\mathsf{REFLECT}(L_2)$ is not context-free.

Pumping Lemma: Proof is similar to showing $\{a^n b^n c^n \mid n \geq 0\}$ does not satisfy the pumping lemma.

Problem 4. [Category: Proof] Consider the language $B \subseteq \{a, b\}^*$ defined as

$$B = \{(a^n b^n)^n \mid n \geq 0\}$$

Prove that B is not context-free. **[10 points]**

Closure Properties: This is by far the shortest proof. Take $h : \{a, b\}^* \to \{a\}^*$ where $h(a) = a$ and $h(b) = \epsilon$. Then $h(B) = \{a^{n^2} \mid n \geq 0\}$ which is non-CFL. Since CFLs are closed under homomorphic images, B is not context-free.

Pumping Lemma Proof: Let p be the pumping length. Pick $z = (a^p b^p)^p \in L$. Let u, v, w, x, y be some division of z such that $z = uvwxy$, $|vwx| \leq p$, and $|vx| > 0$. Now $uv^2wx^2y \notin L$; this can be proved in two ways.

- **Combinatorial Argument:** Notice that any string in L is of the form $(a^n b^n)^n$ and so has length $2n^2$. Now $|uv^2wx^2y| > 2p^2$, because $|vx| > 0$. Further $|uv^2wx^2y| \leq 2p^2 + p$ since $|vwx| \leq p$. Thus, $2p^2 < 2p^2 + 1 \leq |uv^2wx^2y| \leq 2p^2 + p \leq 2p^2 + 2p + 2 = 2(p + 1)^2$ and so $uv^2wx^2y \notin L$.
- **Standard Case-by-Case Analysis:** We consider 4 cases based on what form vwx takes.
  - $vwx \in L(a^*)$, i.e., consists only of as. Then, $u = (a^p b^p)^s a^i$, $v = a^{j_1}$, $w = a^{j_2}$, $x = a^{j_3}$, and $y = a^k b^p (a^p b^p)^t$, where $s + t = p - 1$, $i + j_1 + j_2 + j_3 + k = p$ and $j_1 + j_2 > 0$. Now, $uv^2wx^2y = (a^p b^p)^s a^i a^{2j_1} a^{j_2} a^{2j_3} a^k b^p (a^p b^p)^t = (a^p b^p)^s (a^{p+|vx|} b^p)(a^p b^p)^t \notin L$.
  - $vwx \in L(a^* b^*)$, i.e., begins with as and ends with bs. Then $u = (a^p b^p)^s a^i$, $vwx = a^{j_1} b^{j_2}$, and $y = b^k (a^p b^p)^t$, where $s + t = p - 1$, and $j_1 + j_2 \neq p$ and $|vx| > 0$. Now in uv^2wx^2y, if we consider the block of $a^* b^*$ of which vwx is a substring, will contain either more than p as or bs (or both). Thus, $uv^2wx^2y \notin L$.
  - $vwx \in L(b^*)$. This case is similar to case 1.
  - $vwx \in L(b^* a^*)$. This case is similar to case 2.