CS 154 - Introduction to Automata and Complexity Theory
Spring Quarter, 2009
SOLUTIONS to Midterm

Grade Statistics: Out of a total of 150 points, the mean score was 108 and the standard deviation was 27.

Grading Queries: Use the following list to direct any questions on the gradings to the various TAs: Problem 1 [Nelson], Problem 2 [Nelson], Problem 3 [Gary], Problem 4 [Kostas], Problem 5 [Kostas], and Problem 6 [Gary].

Problem 1. [30 points] Classify the following languages into one of three categories: **(A)** regular; **(B)** context-free (but not regular); and, **(C)** not context-free.

You must give a brief justification of your choice in the space provided to receive full credit. The justification could be of the type: $L_{EQ} = \{0^n 1^n \mid n \geq 1\}$ *is context-free since a PDA can push X's onto stack for each 0 and match the X's against 1's.*

(a). $L = \{0^m 1^n \mid 2m - n = 20 \text{ where } m, n \geq 1\}$.

B – Since a PDA can push two X's for each 0, and then match X's with 1's, checking that there are exactly twenty X's left over at the end of input.

(b). $L = \{0^m 1^n \mid 2m + n = 20 \text{ where } m, n \geq 1\}$.

A – There are only finitely many choices of m and n possible, so the language is finite and hence regular.

(c). $L = L_1 \cap L_2$ where $L_1 = \{w \in \{a, b, c\}^* \mid w \text{ has an equal number of } a\text{'s, } b\text{'s and } c\text{'s}\}$ and L_2 is the language of the regular expression $a^*b^*c^*$.

C – Since $L = \{a^n b^n c^n\}$ which is known not to be context-free from the lecture notes.

(d). $L = \{a^i b^j c^k d^l \mid j = k \text{ and } i = l \text{ where } i, j, k, l \geq 0\}$.

B – Since a PDA can push A's for initial a's, followed by B's for b's, then match c's with B's and d's with A's.

(e). $L(G)$ where the grammar G's productions are $\{S \to 0S \mid 1S0 \mid 1S \mid \epsilon\}$.

A – Since the language contains all strings in Σ^*, which makes it regular.

Problem 2. [30 points] Provide regular expressions for the following languages defined over $\Sigma = \{0, 1\}$. You do not need to provide any justification or explanation.

a). The language consisting of all strings in which every third symbol is a 0. (That is, there should be a 0 at every position which is a multiple of 3.)

$$((0+1)(0+1)0)^*(\epsilon + (0+1) + (0+1)(0+1))$$

b). The language consisting of all strings with an even number of 0's.

$$(1^*01^*01^*)^* + 1^*$$

c). The language consisting of all strings where each 1 is immediately preceded by two consecutive 0s.

$$(0^*001)^*0^*$$

Problem 3. [30 points] Let $L \subseteq \{0,1\}^*$ be the language consisting of all strings whose every prefix has at least as many 0's as 1's. In other words, for a string $w = w_1w_2w_3 \ldots w_n$ in L, it must be the case that each of the n substrings w_1, w_1w_2, $w_1w_2w_3$, $\ldots$, $w_1w_2w_3 \ldots w_n$ has at least as many 0's as 1's.

Is L regular or not? Give a proof of your answer.

Solution: This language in non-regular. We can prove this using the Pumping Lemma.

Assume L is regular and apply the Pumping Lemma. Let n be the number in the Pumping Lemma. Choose $w = 0^n1^n$. It is clear that $|w| > n$. It is also easy to see that each prefix of w either does not contain any 1's, or contains n 0's but no more than n 1's. So, w must belong to the language.

We get a decomposition of $w = xyz$ from the Pumping Lemma, with $|xy| \leq n$ and $|y| \geq 0$. The latter two properties ensure that y only contains 0's and contains at least one 0. If the length of y is $r > 0$, we obtain that $y = 0^r$.

Further, the pumping lemma guarantees that for all $k \geq 0$, the string $xy^kz \in L$. We choose $k = 0$ and consider the string $s = xy^0z = xz$. It is clear that $s = 0^{n-r}1^n$. But this string cannot belong to L since s itself is a prefix which has fewer 0's than 1's.

This gives a contradiction and implies that L is non-regular.

Problem 4. [20 points] Consider a language L over the alphabet $\Sigma = \{a, b, c\}$. For a string $w \in \Sigma^*$, define sort(w) to be the string obtained by rearranging the symbols in w such that all the a's appear before all the b's, and all the b's appear before all the c's. That is, we sort the symbols in w in the lexicographic order. We then define the language:

$$\text{sort}(L) = \{\text{sort}(w) \mid w \in L\}$$

as the set of "sorted" versions of the strings in L.

a). Let R be a regular language over the two-letter alphabet $\{a, b\}$. Then, is sort(R) always a regular language? (In other words, are regular languages closed under the sort operation?) Justify your answer.

Solution: Let R be the regular language defined by the regular expression $(ab)^*$. Clearly, every string in R has an equal number of a's and b's. So, sort(R) must be the language $\{a^nb^n \mid n \geq 0\}$. This is essentially the same language as L_{EQ} which was shown to be non-regular (via Pumping Lemma) in the lecture notes.

b). Let R be a regular language over the three-letter alphabet $\{a, b, c\}$. Then, is sort(R) always a context-free language? Justify your answer.

Solution: Let R be the regular language defined by the regular expression $(abc)^*$. Clearly, every string in R has an equal number of a's, b's and c's. So, sort(R) must be the language $\{a^nb^nc^n \mid n \geq 0\}$. This language was shown to be non-context-free (via Pumping Lemma) in the lecture notes.

Problem 5. [10 points] Let $\Sigma = \{0, 1\}$. For a string $x \in \Sigma^*$, let $\overline{x}$ denote the boolean complement of x; that is, the string obtained from x by converting all the 0's to 1's and all the 1's to 0's. For instance, if $x = 010$, then $\overline{x} = 101$. Recall that x^R denotes the reversal of the string x.

Provide a context-free grammar for the language $L = \{w \in \Sigma^* \mid w^R = \overline{w}\}$. For instance, 011001 and 010101 are in L but 101101 is not in L.

Solution:
The context-free grammar $G = (V, T, P, S)$ is defined as follows. We let $T = \{0, 1\}$ and $V = \{S\}$. Then, the productions are

$$S \to 0S1 \mid 1S0 \mid \epsilon$$

.

Problem 6. [30 points] Suppose that L_1 is a regular language. Show that the following language must be a context-free language:

$$L_2 = \{ww^R \mid w \in L_1\}.$$

Observe that L_2 is **not** the same as $L_1 L_1^R$; in fact, it contains all strings that are formed by taking a string w from L_1 followed by a *reversal* of w *itself.*
(**Hint:** You could describe the construction of a PDA for the language. Note that you do not need to give all the fine details, just explain the construction at a high-level and justify its correctness.)

Solution: Suppose we are given a DFA M for the language L_1. We will show how to construct a PDA P for the language L_2. The PDA P starts by behaving like the machine M, scanning the input string w, and making the same state transitions as M. In each of these transitions, it stores each symbol of the input w on the stack. We also add new ϵ-transitions to each state that was a final state in M, allowing the machine P to "guess" that the first part of the input (the string w) is over and the second part (the string w^R) is about to begin. These ϵ-transitions do not change the stack which now contains w^R on top of Z_0; further, these ϵ-transitions move to a new set of states which just scan the rest of the input, comparing them to symbols popping off the stack, to ensure that the second half of the input is exactly the same as w^R. When the symbol Z_0 appears on the stack, the PDA can empty the stack and accept by empty stack (assuming that the input is over).

The first simulation of M reaches a state that was final in M if and only if the first part of the input string belongs to L_1, and then allows P to move to the second stage. The second stage verifies that the second part of the input strng is the reversal of the first part, and then allows P to accept by final stack.