COMP 9601

Assignment 1 Due midnight Oct 5, 2015

Warm-up. (I.) A language is said to be regular if it can be accepted by a DFA. Consider any two regular languages L_1 and L_2 over an alphabet Σ. Prove or disprove the following statements. (i) $L_1 \cap L_2$ is regular. (ii) L_1^r (which contains the reverse of all strings in L_1) is regular. (iii) $\bar{L_1} = \Sigma^* - L_1$ is regular. (iv) $L_1 - L_2$ is regular.

(II.) Let M be a nondeterministic finite automaton accepting a language L. Disprove that reversing the roles of the final and non-final states of M will give a nondeterministic finite automaton accepting the complement of L (i.e., $\{x \mid x \notin L\}$).

(III.) Let $L = \{w \mid w \in \{0,1\}^*$ and w contains an equal number of occurrences of the substrings 01 and 10 $\}$. For example, 101 is in L, and 1010 is not. Show that L is a regular language.

Each problem carries 25 points. Please submit answers to at least four problems. You can get at most 100 points. Make your answers **precise and concise**. (Marking criteria: 75% correctness, 25% preciseness & conciseness; Assignment box: A4, 3rd floor, CYC)

1. Let $L \subseteq \Sigma^*$ be a regular language. Are the following languages regular? Justify your answer.

   (a) $L_1 = \{xwy \mid x, w, y \in \Sigma^*$ and $xy \in L\}$

   (b) $L_2 = \{xy \mid x, w, y \in \Sigma^*$ and $xwy \in L\}$

2. Devise a recursive algorithm which, given a regular expression R, directly constructs a right linear grammar G such that $L(R) = L(G)$. Your algorithm is not allowed to construct a DFA or NFA as an intermediate step.

3. Constructing an NFA from a right linear grammar is straightforward. Intuitively, the rule $V \to aX$ would define the transition $f(V, a) = X$, and the rule $V \to a$ would give the transition $f(V, a) = Q$, where Q is the final state.

   Devise an algorithm which, given a left linear grammar G, directly constructs a right linear grammar generating the same language. Your algorithm is not allowed to construct a DFA or NFA as an intermediate step.

4. (Final Exam 2014) Let A be a one-state pushdown automaton that has two kinds of stack operations: (1) Pop: A reads and removes the top element z from the stack, and then depending on its current state, next input symbol a, and z, A makes a move which can push zero or more elements back to the stack. (2) Pop & peek: A reads and removes the top element z from the stack, and reads (without removing) the next top element z'; then A makes a move depending on its current state, next input symbol a, z and z', and may push zero or more elements back to the stack (on top of z').

   Show how to construct another one-state pushdown automaton B using only pop operations to simulate A. (Hint. Let Γ and δ be the stack alphabet and transition function of A, respectively. B could use a bigger stack alphabet $\Gamma' = \Gamma \times \Gamma$. Show how to define B's transition function δ' to simulate a move of A in each of the following four cases. Pop: (i) $\delta(q, a, z)$ contains $(q, z_1 z_2 \cdots z_k)$ for some $k \geq 1$; (ii) $\delta(q, a, z)$ contains (q, ϵ) where ϵ denotes the empty string; Pop and peek: (iii) $\delta(q, a, z', z)$

contains $(a, z_1 z_2 \cdots z_k)$ where z and z' denote the pop and peek symbol, respectively (i.e., z will be removed from the stack, and z' will stay); and (iv) $\delta(q, a, z', z)$ contains (q, ϵ).)

5. It is easy to derive a pushdown automaton to accept the language of all palindromes over $\{0, 1\}$. Use the pumping lemma to show that the language of all palindromes over $\{0, 1\}$ containing an equal number of 0s and 1s is not context free.