Solutions for Problem Set 7
CS 373: Theory of Computation

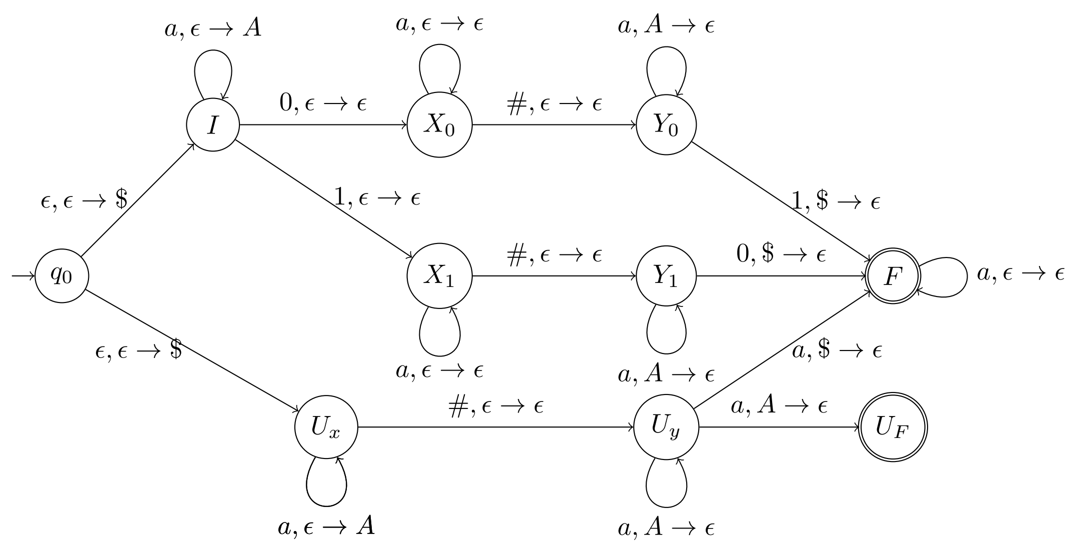
Assigned: March 7, 2013 Due on: March 14, 2013

Problem 1. [Category: Design] Design a PDA to recognize the language

$$C = \{x\#y \mid x, y \in \{0,1\}^*,\ x \neq y\}$$

You need not prove the correctness of your construction but you should provide the intuition behind the states and stack symbols used, that makes your construction clear and understandable. **[10 points]**

Solution: Observe that a string $x\#y \in C$ if either $|x| \neq |y|$ or there is a position i such that $x_i \neq y_i$ (where x_i, y_i denote the ith symbol of x and y, respectively). The PDA will nondeterministically guess which of the above reasons helps show $x\#y \in C$. To check $|x| \neq |y|$ the PDA will count $|x|$ by pushing a symbol for each symbol of x, and then compare $|y|$ by popping from the stack for each symbol of y read. To check if $x_i \neq y_i$, the PDA will "guess" i, push a symbol for the first i symbols, remember the ith symbol of x in the control state, and then pop the i symbols from the stack as y is read, and then check that y_i is different from the symbol remembered in the control state. In the PDA diagram, a stands for either 0 or 1. That is,



a transition $a, X \to Y$ means that we have both transitions $0, X \to Y$ and $1, X \to Y$. The stack alphabet of the above PDA is $\{\$, A\}$. ■

Problem 2. [Category: Comprehension+Design] A CFG G will be said to *right-linear* if every rule in G is either of the form $A \to \epsilon$ or $A \to aB$, where a is a terminal symbol, and B is a variable. Prove that if G is right-linear then $\mathbf{L}(G)$ is regular. *Hint:* Construct an NFA that accepts a string iff it is generated by G.**[10 points]**

Solution: Let $G = (V, \Sigma, R, S)$ be a right-linear grammar. Consider the NFA $M = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = V$, where $q_F \notin V$
- $q_0 = S$
- $F = \{A \in V \mid \text{there is a rule } A \to \epsilon \in R\}$
- $\delta(A, a) = \{B \mid \text{if } A \to aB \in R\}$.

$\mathbf{L}(M) = \mathbf{L}(G)$ as $\forall A \in V$, $\forall w \in \Sigma^*$, $A \overset{*}{\Rightarrow}_G w$ iff $\hat{\delta}_M(A, w) \cap F \neq \emptyset$. Before proving this observation (which you were not required to do), observe that it proves the correctness because taking $A = S$, we have $w \in \mathbf{L}(G)$ iff $S \overset{*}{\Rightarrow}_G w$ iff $\hat{\delta}_M(S, w) \cap F \neq \emptyset$ iff $\hat{\delta}_M(q_0, w) \cap F \neq \emptyset$ iff $w \in \mathbf{L}(M)$.

We will now argue that for any variable A, $A \overset{*}{\Rightarrow}_G w$ iff $\hat{\delta}_M(A, w) \cap F \neq \emptyset$. You were not required to prove this but we do it here for completeness. We will prove this by induction on the length of the string w. For the base case, $w = \epsilon$, observe that $A \overset{*}{\Rightarrow}_G \epsilon$ iff $A \to \epsilon \in R$ because any derivation using a rule of the form $B \to cC$ cannot produce ϵ. By definition, that means $A \in F$. Moreover, since M has no ϵ-transitions, $\hat{\delta}_M(A, \epsilon) = \{A\}$. Thus, $\hat{\delta}_M(A, \epsilon) \cap F \neq \emptyset$ iff $A \in F$, which establishes the base case. For the induction step, consider $w = au$, where $|u| = n$. Now, if $A \overset{*}{\Rightarrow}_G w$ then the derivation has the form $A \Rightarrow_G aB \overset{*}{\Rightarrow}_G au$, where $B \overset{*}{\Rightarrow}_G u$. By induction hypothesis, $B \overset{*}{\Rightarrow}_G u$ iff $\hat{\delta}_M(B, u) \cap F \neq \emptyset$. Moreover from the definition of δ we have $B \in \delta(A, a)$. Thus, $\hat{\delta}_M(A, au = w) \cap F \neq \emptyset$ iff $A \overset{*}{\Rightarrow}_G w$.

The converse of this homework problem is also true. That is, L is regular iff there is a right-linear grammar G such that $\mathbf{L}(G) = L$. ■

Problem 3. [Category: Proof] Let G be a CFG in Chomsky normal form that contains b variables. Show that if G generates some string with a derivation having at least 2^b steps, $\mathbf{L}(B)$ is infinite. **[10 points]**

Solution: The solution rests on the observation that in a binary tree of height h, the number of leaves is 2^h and the number of internal vertices is $2^h - 1$. Thus, if a string w has a derivation of at least 2^b steps, then a parse tree for w has at least 2^b internal vertices; the reason is because in each step of the derivation one of the variables is replaced. Thus, there is a parse tree (say) T for w of height $b + 1$. The rest of the proof is like the proof for the pumping lemma. Now T has a path π of length at least $b + 1$, which has at least $b + 2$ vertices. Hence, π has at least $b + 1$ vertices labeled by variables, and which by pigeon hole principle, implies that there is some variable that appears at least twice in π. Thus, just like in the case of the pumping lemma, we can identify subtrees that can be pumped to get infinitely many trees in the language. ■