CSE 322 Winter 2007
Assignment #7

Due: Friday, March 2, 2007

Reading assignment: Read the CKY algorithm handouts and Section 2.3 of Sipser's book.

Problems:

1. Apply the Cocke-Kasami-Younger algorithm to the following Chomsky Normal Form grammar to show that string $babbaa$ is accepted (show the tableau):

$$\begin{array}{lcl} S & \rightarrow & AB \mid BA \mid AT \mid BU \mid SS \\ T & \rightarrow & SB \mid SU \\ U & \rightarrow & SA \\ A & \rightarrow & a \\ B & \rightarrow & b \end{array}$$

2. Convert the PDA given in the diagram in Sipser's text 2nd edition Figure 2.19, page 114 (1st edition Figure 2.8, page 106) into an equivalent CFG using the general construction of CFGs from PDAs given in Sipser's text 2nd edition Lemma 2.27 (1st edition Lemma 2.15). Note that without loss of generality you can restrict the rules to those only involving variables of the form A_{pq} such that state q is reachable from state p in the PDA diagram. Show your work.

3. Sipser's text 2nd edition, Problem 2.30 (a), (d) (1st edition, Problem 2.18 (a), (d)).

4. Let B be the language of all palindromes (strings w such that $w = w^R$) over $\{0, 1\}$ containing an equal number of 0s and 1s. Show that B is not context-free.

5. We saw that allowing nondeterminism did not allow finite automata to recognize any additional languages. In this problem you will show that this is not the case for PDAs.
We define Deterministic Pushdown Automata (DPDA) below. DPDAs are similar to PDAs with a few important differences.

   - Most importantly, they are deterministic: any configuration has at most one next configuration, and any configuration that has not read the entire input has a unique next configuration.
   - Instead of beginning with an empty stack, DPDAs start with a special symbol \$ as the lone symbol on the stack. A DPDA can always know when it has reached the bottom of the stack. The \$ symbol cannot be erased; i.e., every transition reading \$ must push it back on the stack.
   - DPDAs are allowed to push more than one symbol in a single step. (With PDAs we already have seen that we can simulate this by adding extra states. With DPDAs we must include this power in the base model so that we can require that a DPDA reads a symbol of the input at every step.)

   Formally, a DPDA is a 7-tuple $M = (Q, \Sigma, \Gamma, \$, \delta, q_0, F)$ where

   - Q, Σ, Γ, q_0, and F are defined as in an ordinary PDA except that we require that $\$ \in \Gamma$.

- $\delta : Q \times \Sigma \times \Gamma \to Q \times \Gamma^*$. Thus a DPDA reads precisely one symbol per step and always examines the top symbol on the stack. Given these, there is precisely one next state that can be reached and one option for how the stack changes. Moreover, for all $q \in Q$, and $a \in \Sigma$ we have $\delta(q, a, \$) \in Q \times \Gamma^*\$$; i.e., the only allowable stack operation when the bottom-of-stack symbol is read is to push some string on top of it.
- The starting configuration on input x is $(q_0, x, \$)$. The DPDA accepts iff the computation ends in a final state (independent of what is on the stack).

(a) Show that there is a DPDA that recognizes a non-regular language.

(b) Show that if A is recognized by a DPDA then so is the complement of A.

(c) Use the fact shown in (b) to show that there is a CFL that cannot be recognized by a DPDA.

6. (Extra Credit) Show that any grammar for the language $A = \{a^i b^j c^k \mid i = j \text{ or } j = k\}$ must be ambiguous.

7. (Extra Credit) For a PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ we say that a string $\alpha \in \Gamma^*$ is a *possible stack of* M if there is some input and some choice of moves of M such that α appears on the stack at some point during its computation. Prove that the language $L \subseteq \Gamma^*$ of possible stacks of M is regular.