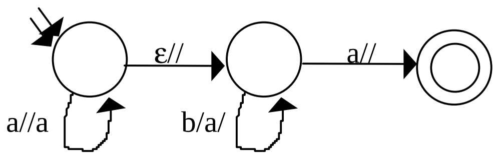
CS 341 Homework 14
Pushdown Automata and Context-Free Grammars

1. In class, we described an algorithm for constructing a PDA to accept a language L, given a context free grammar for L. Let L be the balanced brackets language defined by the grammar G = ({S, [,]}, {[,]}, R, S), where R =

$S \rightarrow \varepsilon$, $S \rightarrow SS$, $S \rightarrow [S]$

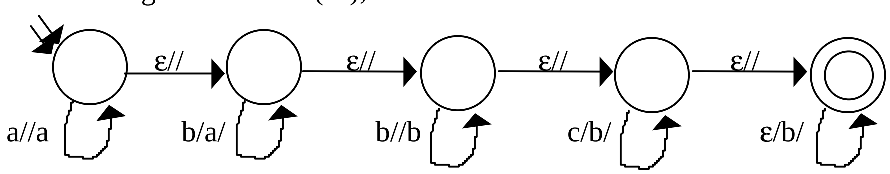
Apply the construction algorithm to this grammar to derive a PDA that accepts L. Trace the operation of the PDA you have constructed on the input string [[][]].

2. Consider the following PDA M:



(a) What is L(M)?
(b) Give a deterministic PDA that accepts L(M) (***not*** L(M)$).

3. Write a context-free grammar for L(M), where M is



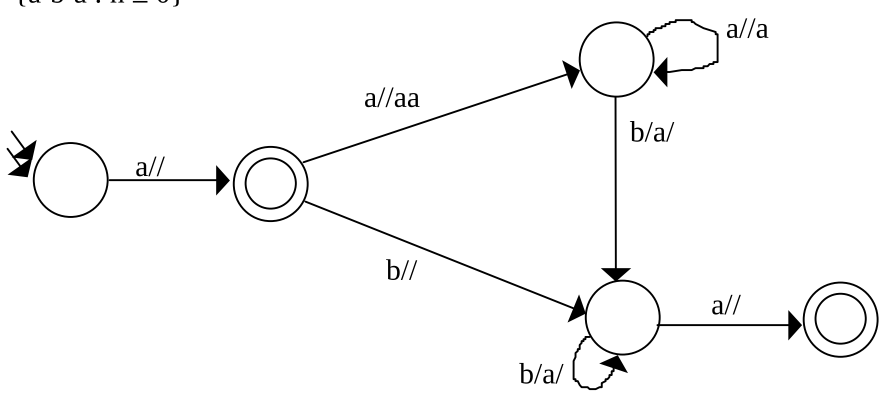
4. Consider the language L = $\{ba^{m1}ba^{m2}b\ldots ba^{mn} : n \geq 2, m1, \ldots, mn \geq 0$, and $mi \neq mj$ for some i, j$\}$
(a) Give a nondeterministic PDA that accepts L.
(b) Write a context-free grammar that generates L.
(c) Prove that L is not regular.

Solutions

1. This is a very simple mechanical process that you should have no difficulty carrying out, and getting the following PDA, M = ({p, q}, {[,]}, {S, [,]}, Δ, p, {q}), where

Δ = {((p, ε, ε), (q, S)),
((q, ε, S), (q, ε)), ((q, ε, S), (q, SS)), ((q, ε, S), (q, [S])),
((q, [, [), (q, ε)), ((q,],]), (q, ε))}

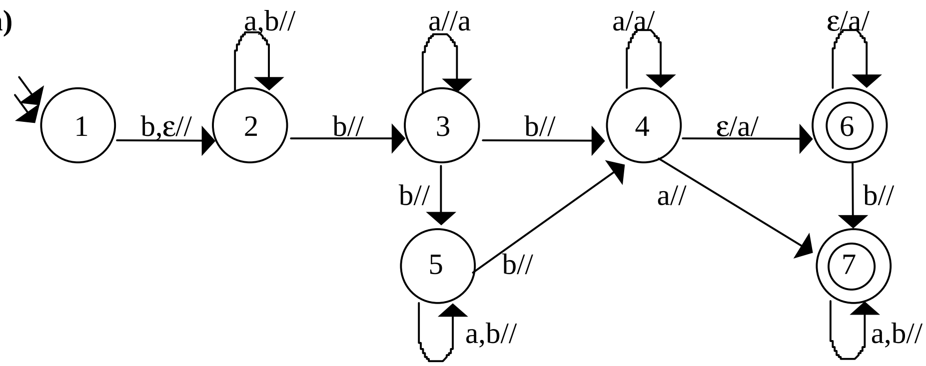
2. (a) L(M) = $\{a^n b^n a : n \geq 0\}$
(b)



3. Don't even try to use the grammar construction algorithm. Just observe that $L = \{a^n b^n b^m c^p : m \geq p \text{ and } n \text{ and } p \geq 0\}$, or, alternatively $\{a^n b^m c^p : m \geq n + p \text{ and } n \text{ and } p \geq 0\}$. It can be generated by the following rules:

$S \rightarrow S_1S_2$
$S_1 \rightarrow aS_1b$ /* S_1 generates the $a^n b^n$ part. */
$S_1 \rightarrow \varepsilon$
$S_2 \rightarrow bS_2$ /* S_2 generates the $b^m c^p$ part. */
$S_2 \rightarrow bS_2c$
$S_2 \rightarrow \varepsilon$

4. (a)



We use state 2 to skip over an arbitrary number of ba^i groups that aren't involved in the required mismatch.
We use state 3 to count the first group of a's we care about.
We use state 4 to count the second group and make sure it's not equal to the first.
We use state 5 to skip over an arbitrary number of ba^i groups in between the two we care about.
We use state 6 to clear the stack in the case that the second group had fewer a's than the first group did.
We use state 7 to skip over any remaining ba^i groups that aren't involved in the required mismatch.

(b) $S \rightarrow A'bLA'$ /* L will take care of two groups where the first group has more a's */
$S \rightarrow A'bRA'$ /* R will take care of two groups where the second group has more a's */
$L \rightarrow ab \mid aL \mid aLa$
$R \rightarrow ba \mid Ra \mid aRa$
$A' \rightarrow bAA' \mid \varepsilon$
$A \rightarrow aA \mid \varepsilon$

(c) Let $L_1 = ba^*ba^*$, which is obviously regular.
If L is regular then
$L_2 = L \cap L_1$ is regular.
$L_2 = ba^n ba^m, n \neq m$
$\neg L_2 \cap L_1$ must also be regular.
But $\neg L_2 \cap L_1 = ba^n ba^m, n = m$, which can easily be shown, using the pumping theorem, not to be regular.