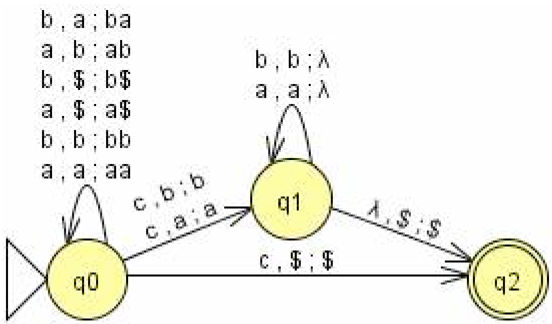
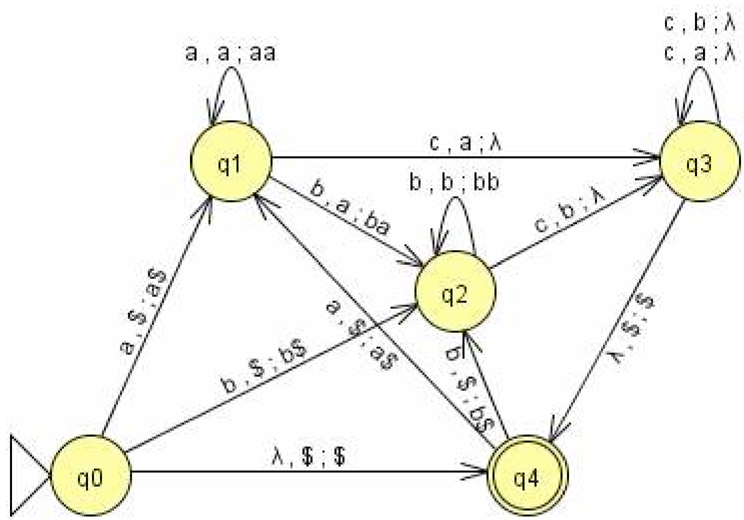
pg. 183

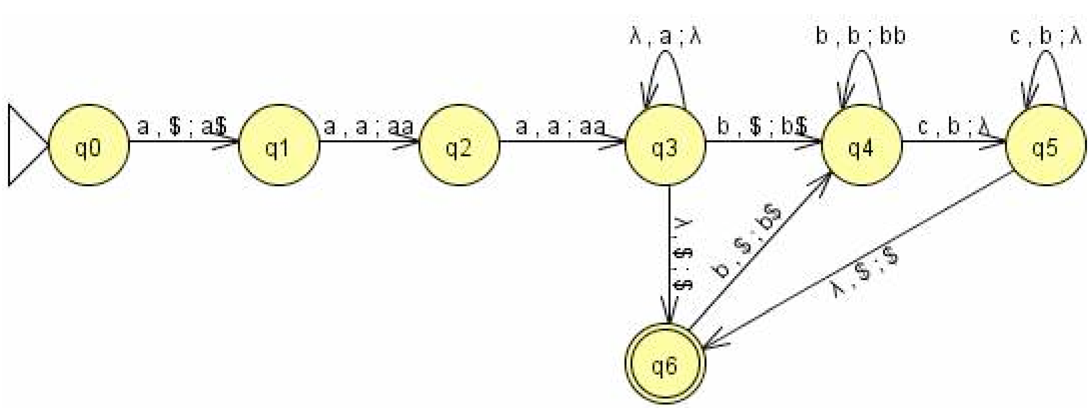
4(b) $L = \{wcw^R : w \in \{a, b\}^*\}$



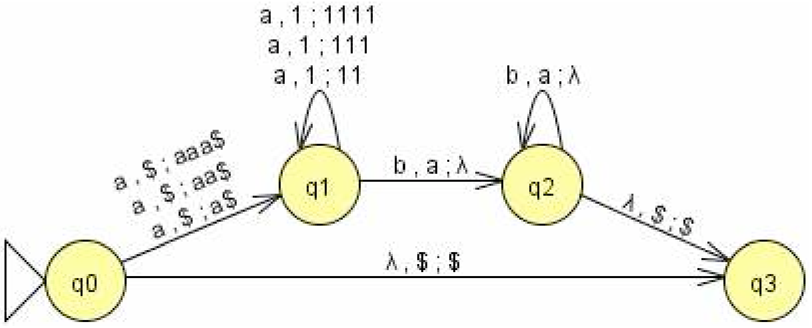
4(c) $L = \{a^n b^m c^{n+m} : n \geq 0, m \geq 0\}$



4(e) $L = \{a^3 b^n c^n : n \geq 0\}$



4(f) $L = \{a^n b^m : n \leq m \leq 3n\}$



Page #195, Prob# 5.

G:

$S \rightarrow aABB|aAA$,
$A \rightarrow aBB|a$,
$B \rightarrow bBB|A$.

The production $B \rightarrow A$ is not in Greibach form. So first we convert this to Greibach form we substitute A with its production, so that $B \rightarrow aBB|a$. So the G can be written in Greibach for as follows:

G:

$S \rightarrow aABB|aAA$,
$A \rightarrow aBB|a$,
$B \rightarrow bBB|aBB|a$.

We can now construct an npda M corresponding to G, where

$M = (\{q_0,q_1,q_f\},T,V \cup \{z\},\delta,q_0,z,\{q_f\})$, where $z \notin V$.

The input alphabet of M is identical with the set of terminals of G, i.e. $T = \{a,b\}$.
The stack alphabet contains the variables of the grammar, i.e. $V = \{S,A,B,z\}$

First we have the following rules relating to the initial and final states.

$\delta(q_0,\lambda,z) = \{(q_1,Sz)\}$,
$\delta(q_1,\lambda,z) = \{(q_f,z)\}$.

Now we write rules for each production. For example,

Rule for $S \rightarrow aABB$ is $\delta(q_1,a,S) = \{(q_1,ABB)\}$ and
Rule for $A \rightarrow a$ is $\delta(q_1,a,A) = \{(q_1, \lambda)\}$.

Following similar procedure, we can find rules for other productions. So we can write the npda for G as follows:

$\delta(q_0,\lambda,z) = \{(q_1,Sz)\}$,
$\delta(q_1,a,S) = \{(q_1,ABB), (q_1,AA)\}$,
$\delta(q_1,a,A) = \{(q_1,BB), (q_1, \lambda)\}$,
$\delta(q_1,b,B) = \{(q_1,BB)\}$,
$\delta(q_1,a,B) = \{(q_1,BB), (q_1, \lambda)\}$,
$\delta(q_1,\lambda,z) = \{(q_f,z)\}$.

Page# 196, Prob# 15.

Given the npda $M = (\{q_0,q_1\},\{a,b\},\{A,z\},\delta,q_0,z,\{q_1\})$, with transitions

$\delta(q_0,a,z) = \{(q_0,Az)\}$,
$\delta(q_0,b,A) = \{(q_0,AA)\}$,
$\delta(q_0,a,A) = \{(q_1, \lambda)\}$.

First we note that although the npda M has single accept state, it is not entered when the stack is empty. In order to satisfy the condition that the single accept state should be entered if and only if the stack is empty, we introduce a new state q_2 and an intermediate step in which we first remove the A from the stack to go the new state q_2 and then in next move we go from q_2 to the final state q_1 with the empty stack. So the new set of transition rules is

$\delta(q_0,a,z) = \{(q_0,Az)\}$,
$\delta(q_0,b,A) = \{(q_0,AA)\}$,
$\delta(q_0,a,A) = \{(q_2, \lambda)\}$.
$\delta(q_2,\lambda,z) = \{(q_1, \lambda)\}$.

Also we note that the condition that each move either increases or decreases the stack content by a single symbol is satisfied for both the given and the new transition rules.

Let us denote the variable (q_iAq_j) by A_{ij} and the variable (q_izq_j) by B_{ij}

For the last two transition rules yield the following productions:

$A_{02} \rightarrow a$,
$B_{21} \rightarrow \lambda$.

For the first transition rule, we can write the following productions:

$B_{00} \rightarrow aA_{00}B_{00} \mid aA_{01}B_{10} \mid aA_{02}B_{20}$,
$B_{01} \rightarrow aA_{00}B_{01} \mid aA_{01}B_{11} \mid aA_{02}B_{21}$,
$B_{02} \rightarrow aA_{00}B_{02} \mid aA_{01}B_{12} \mid aA_{02}B_{22}$.

Similarly for the 2^{nd} transition rule, we can write the following productions:

$A_{00} \rightarrow bA_{00}A_{00} \mid bA_{01}A_{10} \mid bA_{02}A_{20}$,
$A_{01} \rightarrow bA_{00}A_{01} \mid bA_{01}A_{11} \mid bA_{02}A_{21}$,
$A_{02} \rightarrow bA_{00}A_{02} \mid bA_{01}A_{12} \mid bA_{02}A_{22}$.

Now we note that the variables B_{10}, B_{11},B_{12}, B_{20}, B_{22}, A_{10},A_{11},A_{12},A_{20},A_{21} and A_{22} do not occur on the left side and we can eliminate the productions that contain these variables. Therefore we are left with the context-free grammar with following productions:

Page # 212, Prob# 8.

8(b).

$L = \{a^n b^j a^k b^l : n+j \leq k+l\}$ is context free, as we can easily find an npda M for this language as follows:

$M = (\{q_0,q_1,q_2,q_3,q_4,q_5\},\{a,b\},\{0,z\},\delta,q_0,z,\{q_1\})$, and the transitions are

$\delta(q_0,\lambda,\lambda) = (q_1,z)$,
$\delta(q_1,a, \lambda) = (q_1,0)$,
$\delta(q_1,b, \lambda) = (q_2, 0)$,
$\delta(q_2,b, \lambda) = (q_2, 0)$,
$\delta(q_2,a, 0) = (q_3, \lambda)$,
$\delta(q_3,a, 0) = (q_3, \lambda)$,
$\delta(q_3,b, 0) = (q_4, \lambda)$,
$\delta(q_4,b, 0) = (q_4, \lambda)$,
$\delta(q_4,\lambda, 0) = (q_4, \lambda)$,
$\delta(q_4,\lambda,z) = (q_5, \lambda)$.

8(e).

$L = \{a^n b^j a^k b^l : n \leq k, j \leq l\}$ is NOT context free, as we can find contradiction to the pumping lemma as follows:

Consider pumping length m and the string $w = a^m b^m a^m b^m$

Assume L is context free, then by pumping lemma, w = uvxyz, where $|vxy| \leq m$, $|vy| \geq 1$. Now vxy cannot contain characters a,b from more than two adjacent groups, as alternate groups are separated at least by m characters. It can be shown that for any choice of vxy, we pump vy to violate at least one of the conditions $n \leq k$ and $j \leq l$. For example, if we choose vxy from 2nd group of b and 3rd group of a, then we can pump up so that $j \geq l$. So the pumped string is not in L, which is a contradiction.

8(f).

$L = \{a^n b^n c^j : n \leq j \}$ is NOT context free as can be shown by pumping lemma.

Consider pumping length m and the string $w = a^m b^m c^m$.
Now, any choice of vxy cannot involve all a,b and c, since a and c are separated by m characters. Therefore, by pumping appropriately we can always violate the condition $n \leq j$ or the condition $|a| = |b|$.

$A_{02} \rightarrow a$,
$B_{21} \rightarrow \lambda$,
$B_{00} \rightarrow aA_{00}B_{00}$,
$B_{01} \rightarrow aA_{00}B_{01} \mid aA_{02}B_{21}$,
$B_{02} \rightarrow aA_{00}B_{02}$,
$A_{00} \rightarrow bA_{00}A_{00}$,
$A_{01} \rightarrow bA_{00}A_{01}$,
$A_{02} \rightarrow bA_{00}A_{02}$.

8(g).

$L = \{ w \in \{a,b,c\}^* : n_a(w) = n_b(w) = 2n_c(w)\}$ is NOT context-free as we can argue that we cannot construct an npda for L.

If we try to construct an npda such that we push twice when reading c, and pop once when reading a or b then we get $2n_c(w) = n_a(w)+n_b(w)$, but there is no way to tell whether $n_a(w) = n_b(w)$. On the other hand, if we construct the npda such that $n_a(w) = n_b(w)$, then there is no way to tell if $n_a(w) = 2n_c(w)\}$.

$n_c(\omega)$

220/12 $L = \{w \in \{a,b,c\}^* : n_a(w) \neq n_b(w) \cup n_a(w) \neq n_c(w)\}$

+ L is context-free, b/c:

1) $L = L_1 \cup L_2$, where $L_1 = \{w \in \{a,b,c\}^* : n_a(w) \neq n_b(w)\}$
$L_2 = \{w \in \{a,b,c\}^* : n_a(w) \neq n_c(w)\}$

(1.1) It is easy to show L_1, L_2 are ~~regular~~ CF by construct. npda
2) Context-free L are closed under union

+ $\bar{L}$ is not context-free b/c:

[$n_a \neq n_b \cup n_a \neq n_c \Leftrightarrow n_a = n_b \cap n_a = n_c \Leftrightarrow n_a = n_b = n_c$]

1) $\bar{L} = \{w \in \{a,b,c\}^* : n_a(w) = n_b(w) = n_c(w)\}$

2) $\bar{L} \cap L(a^* b^* c^*) = \{a^n b^n c^n : n \geq 0\}$

($L(a^* b^* c^*)$: regular; $\{a^n b^n c^n : n \geq 0\}$: not context-free, easy to show by pumping lemma)

3) CF Ls are closed under regular intersection