Solutions for Homework Five, CSE 355

1. (*7.1, 10 points*) Let M be the PDA defined by

$$Q = \{q_0, q_1, q_2\} \quad \Sigma = \{a, b\} \quad \Gamma = \{A\} \quad F = \{q_1, q_2\}$$

$$\begin{aligned} \delta(q_0, a, \lambda) &= \{[q_0, A]\} \\ \delta(q_0, \lambda, \lambda) &= \{[q_1, \lambda]\} \\ \delta(q_0, b, A) &= \{[q_2, \lambda]\} \\ \delta(q_1, \lambda, A) &= \{[q_1, \lambda]\} \\ \delta(q_2, b, A) &= \{[q_2, \lambda]\} \\ \delta(q_2, \lambda, A) &= \{[q_2, \lambda]\} \end{aligned}$$

a) Describe the language accepted by M.

b) Give the state diagram of M.

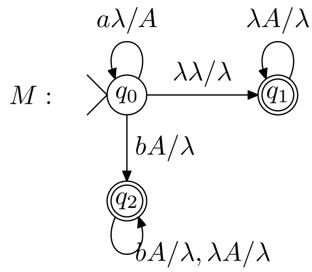
c) Trace all computations of the strings aab, abb, aba in M.

d) Show that $aabb, aaab \in L(M)$.

Solution:

a) The PDA M accepts the language $\{a^i b^j \mid 0 \le j \le i\}$. Processing an a pushes A onto the stack. Strings of the form a^i are accepted in state q_1. The transitions in q_1 empty the stack after the input has been read. A computation with input $a^i b^j$ enters state q_2 upon processing the first b. To read the entire input string, the stack must contain at least j A's. The transition $\delta(q_2, \lambda, A) = [q_2, \lambda]$ will pop any A's remaining on the stack.

b) The state diagram of M is



c) The computations of aab in M are as follows:

State	String	Stack
q_0	aab	λ
q_1	aab	λ

State	String	Stack
q_0	aab	λ
q_0	ab	A
q_1	ab	A
q_1	b	λ

State	String	Stack
q_0	aab	λ
q_0	ab	A
q_0	b	AA
q_1	b	AA
q_1	b	A
q_1	b	λ

State	String	Stack
q_0	aab	λ
q_0	ab	A
q_0	b	AA
q_2	λ	A
q_2	λ	λ

The computations of abb in M are as follows:

State	String	Stack
q_0	abb	λ
q_1	abb	λ

State	String	Stack
q_0	abb	λ
q_0	bb	A
q_1	bb	A
q_1	bb	λ

State	String	Stack
q_0	abb	λ
q_0	bb	A
q_2	b	λ

The computations of aba in M are as follows:

State	String	Stack
q_0	aba	λ
q_1	aba	λ

State	String	Stack
q_0	aba	λ
q_0	ba	A
q_1	ba	A
q_1	ba	λ

State	String	Stack
q_0	aba	λ
q_0	ba	A
q_2	a	λ

d) To show that the string $aabb$ and $aaab$ are in $L(M)$, we trace a computation of M that accepts these strings.

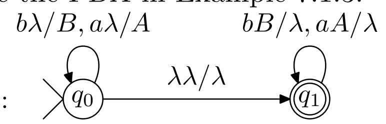
State	String	Stack
q_0	$aabb$	λ
q_0	abb	A
q_0	bb	AA
q_2	b	A
q_2	λ	λ

State	String	Stack
q_0	$aaab$	λ
q_0	aab	A
q_0	ab	AA
q_0	b	AAA
q_2	λ	AA
q_2	λ	A
q_2	λ	λ

■

2. (*7.2, 10 points*) Let M be the PDA in Example 7.1.3.

M :



a) Give the transition table of M.

b) Trace all computations of the strings ab, abb, $abbb$ in M.

c) Show that $aaaa, baab \in L(M)$.

d) Show that $aaa, ab \notin L(M)$.

Solution:

a)

$$Q = \{q_0, q_1\}$$
$$\Sigma = \{a, b\}$$
$$\Gamma = \{A, B\}$$
$$F = \{q_1\}$$

$$\delta(q_0, b, \lambda) = \{[q_0, B]\}$$
$$\delta(q_0, a, \lambda) = \{[q_0, A]\}$$
$$\delta(q_0, \lambda, \lambda) = \{[q_1, \lambda]\}$$
$$\delta(q_1, b, B) = \{[q_1, \lambda]\}$$
$$\delta(q_1, a, A) = \{[q_1, \lambda]\}$$

b) The computations of ab in M are as follows:

State	String	Stack
q_0	ab	λ
q_1	ab	λ

State	String	Stack
q_0	ab	λ
q_0	b	A
q_1	b	A

State	String	Stack
q_0	ab	λ
q_0	b	A
q_0	λ	BA
q_1	λ	BA

The computations of abb in M are as follows:

State	String	Stack
q_0	abb	λ
q_1	abb	λ

State	String	Stack
q_0	abb	λ
q_0	bb	A
q_1	bb	A

State	String	Stack
q_0	abb	λ
q_0	bb	A
q_0	b	BA
q_1	b	BA
q_1	λ	A

State	String	Stack
q_0	abb	λ
q_0	bb	A
q_0	b	BA
q_0	λ	BBA
q_1	λ	BBA

The computations of $abbb$ in M are as follows:

State	String	Stack
q_0	$abbb$	λ
q_1	$abbb$	λ

State	String	Stack
q_0	$abbb$	λ
q_0	bbb	A
q_1	bbb	A

State	String	Stack
q_0	$abbb$	λ
q_0	bbb	A
q_0	bb	BA
q_1	bb	BA
q_1	b	A

State	String	Stack
q_0	$abbb$	λ
q_0	bbb	A
q_0	bb	BA
q_0	b	BBA
q_1	b	BBA
q_1	λ	BA

State	String	Stack
q_0	$abbb$	λ
q_0	bbb	A
q_0	bb	BA
q_0	b	BBA
q_0	λ	$BBBA$
q_1	λ	$BBBA$

c) To show that the string $aaaa$ and $baab$ are in $L(M)$, we trace a computation of M that accepts these strings.

State	String	Stack
q_0	$aaaa$	λ
q_0	aaa	A
q_0	aa	AA
q_1	aa	AA
q_1	a	A
q_1	λ	λ

State	String	Stack
q_0	$baab$	λ
q_0	aab	B
q_0	ab	AB
q_1	ab	AB
q_1	b	B
q_1	λ	λ

d) To show that the string aaa and ab are not in $L(M)$, we trace all computations of these strings in M, and check whether none of them accepts these strings. We have listed all the computations of ab in (b), and none of them accepts it. Now we trace all computations of aaa in M

State	String	Stack
q_0	aaa	λ
q_1	aaa	λ

State	String	Stack
q_0	aaa	λ
q_0	aa	A
q_1	aa	A
q_1	a	λ

State	String	Stack
q_0	aaa	λ
q_0	aa	A
q_0	a	AA
q_1	a	AA
q_1	λ	A

State	String	Stack
q_0	aaa	λ
q_0	aa	A
q_0	a	AA
q_0	λ	AAA
q_1	λ	AAA

Since none of the computations above is accepted, we have aaa is not in M.

■

3. (*7.3, 10 points*) Construct PDAs that accept each of the following languages.

a) $\{a^i b^j \mid 0 \leq i \leq j\}$

b) $\{a^i c^j b^i \mid i, j \geq 0\}$

c) $\{a^i b^j c^k \mid i + k = j\}$

d) $\{w \mid w \in \{a, b\}^*$ and w has twice as many a's as b's$\}$

e) $\{a^i b^i \mid i \geq 0\} \cup a^* \cup b^*$

f) $\{a^i b^j c^k \mid i = j \text{ or } j = k\}$

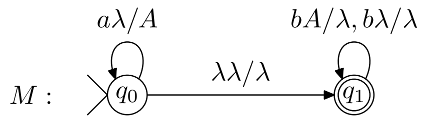
g) $\{a^i b^j \mid i \neq j\}$

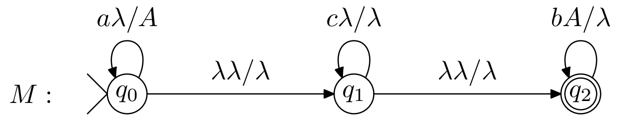
h) $\{a^i b^j \mid 0 \leq i \leq j \leq 2i\}$

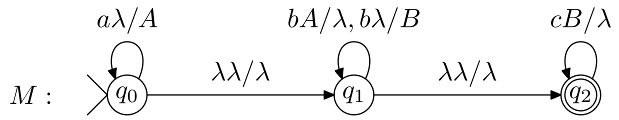
i) $\{a^{i+j} b^i c^j \mid i, j > 0\}$

j) The set of palindromes over $\{a, b\}$

Solution:

a) 

b) 

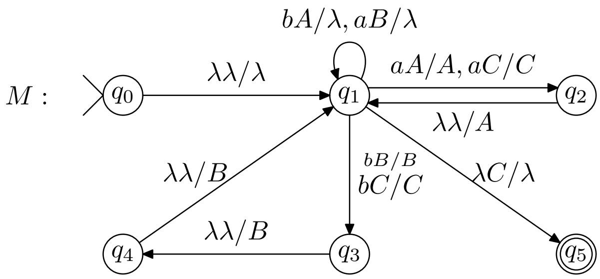
c) 

d) The pushdown automaton defined by the transitions

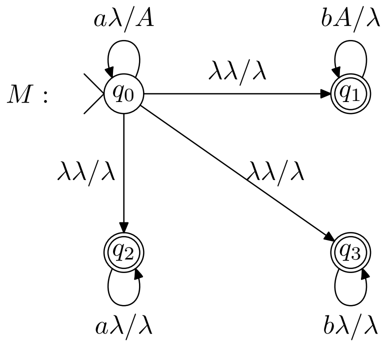
$$\begin{aligned}
\delta(q_0, \lambda, \lambda) &= \{[q_1, C]\} \\
\delta(q_1, a, A) &= \{[q_2, A]\} \\
\delta(q_1, a, C) &= \{[q_2, C]\} \\
\delta(q_1, b, B) &= \{[q_3, B]\} \\
\delta(q_1, b, C) &= \{[q_3, C]\} \\
\delta(q_1, a, B) &= \{[q_1, \lambda]\} \\
\delta(q_1, b, A) &= \{[q_1, \lambda]\} \\
\delta(q_1, \lambda, C) &= \{[q_5, \lambda]\} \\
\delta(q_2, \lambda, \lambda) &= \{[q_1, A]\} \\
\delta(q_3, \lambda, \lambda) &= \{[q_4, B]\} \\
\delta(q_4, \lambda, \lambda) &= \{[q_1, B]\}
\end{aligned}$$

accepts strings that have twice as many a's as b's. A computation begins by pushing a C onto the stack, which serves as a bottom-maker throughout the computation. The stack is used to record the relationship between the number of a's and b's scanned during the computation. The stacktop will be a C when the number of a's processed is exactly twice the number of b's processed. The stack will contain n A's if the automaton has read n more a's than b's. If n more b's than a's have been read, the stack will hold $2n$ B's. When an a is read with an A or C on the top of the stack, an A is pushed onto the stack. This is accomplished by the transition to q_2. If a B is on the top of the stack, the stack is popped removing one b. If a b is read with a C or B on the stack, two B's are pushed onto the stack. Processing a b with an A on the stack pops the A.

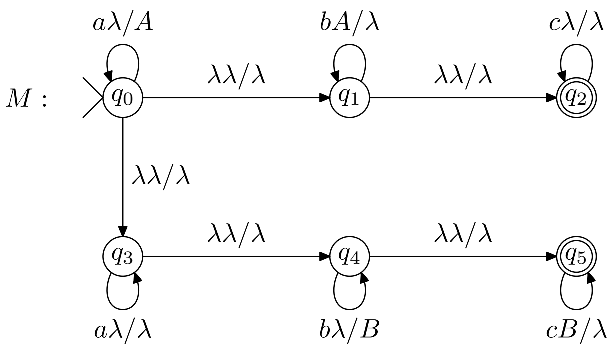
The lone accepting state of the automation is q_5. If the input string has twice as many a's as b's, the transition to q_5 pops the C, terminates the computation, and accepts the string.



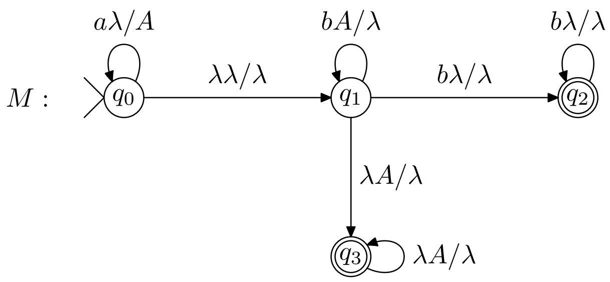
e)



f)



g)



h) The language $L = \{a^i b^j \mid 0 \le i \le j \le 2 \cdot i\}$ is generated by the context-free grammar

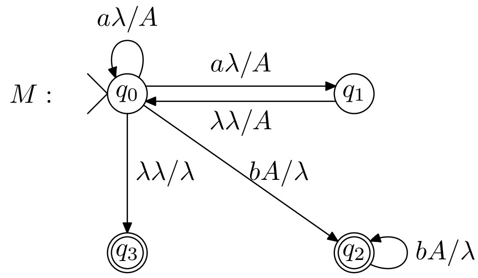
$$S \to aSB \mid \lambda$$
$$B \to bb \mid b$$

The B rule generates one or two b's for each a. A pushdown automaton M that accepts L uses the a's to record an acceptable number of matching b's on the stack. Upon processing an a, the computation nondeterministically pushes one or two A's onto the stack. The transitions of M are

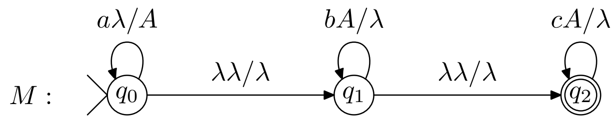
$$\delta(q_0, a, \lambda) = \{[q_1, A]\}$$
$$\delta(q_0, \lambda, \lambda) = \{[q_3, \lambda]\}$$
$$\delta(q_0, a, \lambda) = \{[q_0, A]\}$$
$$\delta(q_0, b, A) = \{[q_2, \lambda]\}$$
$$\delta(q_1, \lambda, \lambda) = \{[q_0, A]\}$$
$$\delta(q_2, b, A) = \{[q_2, \lambda]\}$$

The states q_2 and q_3 are the accepting states of M. The null string is accepted in q_3. For a nonnull string $a^i b^j \in L$, one of the computations will push exactly j A's onto the stack. The stack is emptied by processing the b's in q_2.

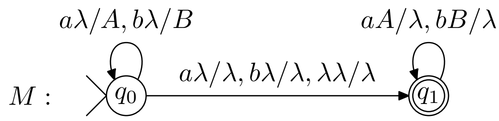
The state diagram of the PDA is



i)



j)



■

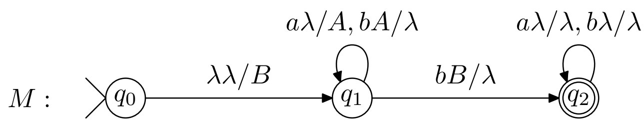
4. (*7.7, 10 points*) Let L be the language $\{w \in \{a, b\}^* \mid w$ has a prefix containing more b's than a's.$\}$. For example, $baa, abba, abbaaa \in L$, but $aab, aabbab \notin L$.

a) Construct a PDA that accepts L by final state.

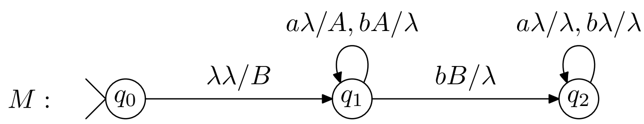
b) Construct a PDA that accepts L by empty stack.

Solution:

a)



b)



■

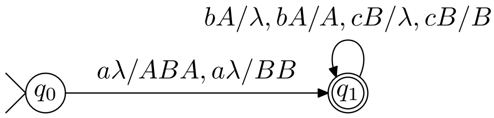
5. (*7.12, 20 points*) Use the technique of Theorem 7.3.1 to construct a PDA that accepts the languages of the Greibach normal form grammar.

$$S \to aABA \mid aBB$$
$$A \to bA \mid b$$
$$B \to cB \mid c$$

Solution: The state diagram for the extended PDA obtained from the grammar is

$Q = \{q_0, q_1\}$
$\Sigma = \{a, b, c\}$
$\Gamma = \{A, B\}$
$F = \{q_1\}$

$\delta(q_0, a, \lambda) = \{[q_1, ABA], [q_1, BB]\}$
$\delta(q_1, b, A) = \{[q_1, A], [q_1, \lambda]\}$
$\delta(q_1, c, B) = \{[q_1, B], [q_1, \lambda]\}$

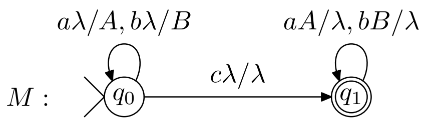


■

6. (*7.15, 20 points*) Let M be the PDA in Example 7.1.1.

$Q = \{q_0, q_1\}$
$\Sigma = \{a, b, c\}$
$\Gamma = \{A, B\}$
$F = \{q_1\}$

$\delta(q_0, a, \lambda) = \{[q_0, A]\}$
$\delta(q_0, b, \lambda) = \{[q_0, B]\}$
$\delta(q_0, c, \lambda) = \{[q_1, \lambda]\}$
$\delta(q_1, a, A) = \{[q_1, \lambda]\}$
$\delta(q_1, b, B) = \{[q_1, \lambda]\}$



a) Trace the computation in M that accepts *bbcbb*.

b) Use the technique from Theorem 7.3.2 to construct a grammar G that accepts $L(M)$.

c) Give the derivation of *bbcbb* in G.

Solution:

a)

State	String	Stack
q_0	$bbcbb$	λ
q_0	$bcbb$	B
q_0	cbb	BB
q_1	bb	BB
q_1	b	B
q_1	λ	λ

b) First we add transitions to M as follows.

$$
\begin{aligned}
\delta(q_0, a, \lambda) &= \{[q_0, A]\} \\
\delta(q_0, a, A) &= \{[q_0, AA]\} \\
\delta(q_0, a, B) &= \{[q_0, AB]\} \\
\delta(q_0, b, \lambda) &= \{[q_0, B]\} \\
\delta(q_0, b, A) &= \{[q_0, BA]\} \\
\delta(q_0, b, B) &= \{[q_0, BB]\} \\
\delta(q_0, c, \lambda) &= \{[q_1, \lambda]\} \\
\delta(q_0, c, A) &= \{[q_1, A]\} \\
\delta(q_0, c, B) &= \{[q_1, B]\} \\
\delta(q_1, a, A) &= \{[q_1, \lambda]\} \\
\delta(q_1, b, B) &= \{[q_1, \lambda]\}
\end{aligned}
$$

Second the rules of the equivalent grammar G and the transition from which they were constructed are presented in Table 1.

c)

$$
\begin{array}{ll}
[q_0, bbcbb, \lambda] & S \Rightarrow \langle q_0, \lambda, q_1\rangle \\
\vdash [q_0, bcbb, B] & \Rightarrow b\langle q_0, B, q_1\rangle \\
\vdash [q_0, cbb, BB] & \Rightarrow bb\langle q_0, B, q_1\rangle\langle q_1, B, q_1\rangle \\
\vdash [q_1, bb, BB] & \Rightarrow bbc\langle q_1, B, q_1\rangle\langle q_1, B, q_1\rangle \\
\vdash [q_1, b, B] & \Rightarrow bbcb\langle q_1, \lambda, q_1\rangle\langle q_1, B, q_1\rangle \\
 & \Rightarrow bbcb\langle q_1, B, q_1\rangle \\
\vdash [q_1, \lambda, \lambda] & \Rightarrow bbcbb\langle q_1, \lambda, q_1\rangle \\
 & \Rightarrow bbcbb
\end{array}
$$

■

7. (*7.17, 20 points*) Use the pumping lemma to prove that each of the following languages is not context-free.

a) $\{a^k \mid k \text{ is a perfect square}\}$

b) $\{a^i b^j c^i d^j \mid i, j \geq 0\}$

c) $\{a^i b^{2i} a^i \mid i \geq 0\}$

d) $\{a^i b^j c^k \mid 0 < i < j < k < 2i\}$

e) $\{ww^R w \mid w \in \{a, b\}^*\}$

f) The set of finite-length prefixes of the infinite string

$$abaabaaabaaaab \cdots ba^n ba^{n+1} b \cdots$$

Solution:

a) Assume that language L consisting of strings over $\{a\}$ whose lengths are a perfect square is context-free. By the pumping lemma, there is a number k such that every string in L with length k or more can be written $uvwxy$ where

Transition	Rule
	$S \to \langle q_0, \lambda, q_1 \rangle$
$\delta(q_0, a, \lambda) = \{[q_0, A]\}$	$\langle q_0, \lambda, q_0 \rangle \to a \langle q_0, A, q_0 \rangle$
	$\langle q_0, \lambda, q_1 \rangle \to a \langle q_0, A, q_1 \rangle$
$\delta(q_0, a, A) = \{[q_0, AA]\}$	$\langle q_0, A, q_0 \rangle \to a \langle q_0, A, q_0 \rangle \langle q_0, A, q_0 \rangle$
	$\langle q_0, A, q_0 \rangle \to a \langle q_0, A, q_1 \rangle \langle q_1, A, q_0 \rangle$
	$\langle q_0, A, q_1 \rangle \to a \langle q_0, A, q_0 \rangle \langle q_0, A, q_1 \rangle$
	$\langle q_0, A, q_1 \rangle \to a \langle q_0, A, q_1 \rangle \langle q_1, A, q_1 \rangle$
$\delta(q_0, a, B) = \{[q_0, AB]\}$	$\langle q_0, B, q_0 \rangle \to a \langle q_0, A, q_0 \rangle \langle q_0, B, q_0 \rangle$
	$\langle q_0, B, q_0 \rangle \to a \langle q_0, A, q_1 \rangle \langle q_1, B, q_0 \rangle$
	$\langle q_0, B, q_1 \rangle \to a \langle q_0, A, q_0 \rangle \langle q_0, B, q_1 \rangle$
	$\langle q_0, B, q_1 \rangle \to a \langle q_0, A, q_1 \rangle \langle q_1, B, q_1 \rangle$
$\delta(q_0, b, \lambda) = \{[q_0, B]\}$	$\langle q_0, \lambda, q_0 \rangle \to b \langle q_0, B, q_0 \rangle$
	$\langle q_0, \lambda, q_1 \rangle \to b \langle q_0, B, q_1 \rangle$
$\delta(q_0, b, A) = \{[q_0, BA]\}$	$\langle q_0, A, q_0 \rangle \to b \langle q_0, B, q_0 \rangle \langle q_0, A, q_0 \rangle$
	$\langle q_0, A, q_0 \rangle \to b \langle q_0, B, q_1 \rangle \langle q_1, A, q_0 \rangle$
	$\langle q_0, A, q_1 \rangle \to b \langle q_0, B, q_0 \rangle \langle q_0, A, q_1 \rangle$
	$\langle q_0, A, q_1 \rangle \to b \langle q_0, B, q_1 \rangle \langle q_1, A, q_1 \rangle$
$\delta(q_0, b, B) = \{[q_0, BB]\}$	$\langle q_0, B, q_0 \rangle \to b \langle q_0, B, q_0 \rangle \langle q_0, B, q_0 \rangle$
	$\langle q_0, B, q_0 \rangle \to b \langle q_0, B, q_1 \rangle \langle q_1, B, q_0 \rangle$
	$\langle q_0, B, q_1 \rangle \to b \langle q_0, B, q_0 \rangle \langle q_0, B, q_1 \rangle$
	$\langle q_0, B, q_1 \rangle \to b \langle q_0, B, q_1 \rangle \langle q_1, B, q_1 \rangle$
$\delta(q_0, c, \lambda) = \{[q_1, \lambda]\}$	$\langle q_0, \lambda, q_0 \rangle \to c \langle q_1, \lambda, q_0 \rangle$
	$\langle q_0, \lambda, q_1 \rangle \to c \langle q_1, \lambda, q_1 \rangle$
$\delta(q_0, c, A) = \{[q_1, A]\}$	$\langle q_0, A, q_0 \rangle \to c \langle q_1, A, q_0 \rangle$
	$\langle q_0, A, q_1 \rangle \to c \langle q_1, A, q_1 \rangle$
$\delta(q_0, c, B) = \{[q_1, B]\}$	$\langle q_0, B, q_0 \rangle \to c \langle q_1, B, q_0 \rangle$
	$\langle q_0, B, q_1 \rangle \to c \langle q_1, B, q_1 \rangle$
$\delta(q_1, a, A) = \{[q_1, \lambda]\}$	$\langle q_1, A, q_0 \rangle \to a \langle q_1, \lambda, q_0 \rangle$
	$\langle q_1, A, q_1 \rangle \to a \langle q_1, \lambda, q_1 \rangle$
$\delta(q_1, b, B) = \{[q_1, \lambda]\}$	$\langle q_1, B, q_0 \rangle \to b \langle q_1, \lambda, q_0 \rangle$
	$\langle q_1, B, q_1 \rangle \to b \langle q_1, \lambda, q_1 \rangle$
	$\langle q_0, \lambda, q_0 \rangle \to \lambda$
	$\langle q_1, \lambda, q_1 \rangle \to \lambda$

Table 1: The rules of the equivalent grammar G and the transition from which they were constructed (Problem 7.15 (b))

(i) $length(vwx) \leq k$

(ii) v and x are not both null

(iii) $uv^iwx^iy \in L$ for all $i \geq 0$.

The string $z = a^{k^2}$ must have a decomposition $uvwxy$ that satisfies the preceding conditions. Consider the length of the string uv^2wx^2y obtained by pumping $uvwxy$.

$$\begin{aligned} length(z) =& length(uv^2wx^2y) \\ =& lenght(uvwxy) + length(v) + length(x) \\ =& k^2 + length(v) + length(x) \\ \leq& k^2 + k \\ <& (k+1)^2 \end{aligned}$$

Since the length of z is greater than k^2 but less than $(k+1)^2$, we conclude that $z \notin L$ and that L is not context-free.

b) Assume that language $L = \{a^ib^jc^id^j \mid i, j \geq 0\}$ is context-free. By the pumping lemma, there is a number k such that every string in L with length k or more can be written $uvwxy$ where

(i) $length(vwx) \leq k$

(ii) v and x are not both null

(iii) $uv^iwx^iy \in L$ for all $i \geq 0$.

The string $z = a^kb^kc^kd^k$ must have a decomposition $uvwxy$ that satisfies the preceding conditions. Consider the string uv^2wx^2y obtained by pumping $uvwxy$. Since v and x are not both null by condition (ii), we have that vwx contains at least one terminal. Without loss of generality, assume vwx contains a terminal which is either a or c (similar argument for the case that the terminal is either b or d). Condition (i) requires the length of vwx to be at most k. This implies that vwx is a string that cannot contain both a and c types of terminal. Thus uv^2wx^2y increases the number of either a's or c's, but not the both, compared with $uvwxy$. Hence $uv^2wx^2y \notin L$, a contradiction. We conclude that L is not context-free.

c) Assume that language $L = \{a^ib^{2i}a^i \mid i \geq 0\}$ is context-free. By the pumping lemma, there is a number k such that every string in L with length k or more can be written $uvwxy$ where

(i) $length(vwx) \leq k$

(ii) v and x are not both null

(iii) $uv^iwx^iy \in L$ for all $i \geq 0$.

The string $z = a^kb^{2k}a^k$ must have a decomposition $uvwxy$ that satisfies the preceding conditions. Consider the string uv^2wx^2y obtained by pumping $uvwxy$. Since by assumption $uv^2wx^2y \in L$, we must have that the union of v and x contains both a type and b type of terminals. Otherwise it only increases one type of terminal while keeping the other the same, thereby no longer in L. Further more, condition (i) requires the length of vwx to be at most k. This implies that the substring vwx of z cannot contain a's from both sides of the b's substring. Therefore uv^2wx^2y only increases the number of a's either preceding or after b's, but not both. Hence $uv^2wx^2y \notin L$, and consequently, L is not context-free.

d) Assume that language $L = \{a^i b^j c^k \mid 0 < i < j < k < 2i\}$ is context-free. By the pumping lemma, there is a number k such that every string in L with length k or more can be written $uvwxy$ where

(i) $length(vwx) \leq k$

(ii) v and x are not both null

(iii) $uv^iwx^iy \in L$ for all $i \geq 0$.

Without loss of generality, we assume $k > 2$, since we can always increase k while maintaining the three conditions above. Then the string $z = a^k b^{k+1} c^{k+2}$ is in L and must have a decomposition $uvwxy$ that satisfies the preceding conditions. Consider the string uv^kwx^ky obtained by pumping $uvwxy$. Condition (i) requires the length of vwx to be at most k. This implies that vwx is a string containing only one type of terminal or the concatenation of either a and b types, or b and c types. If c is not contained in vwx, pumping v and x only increases the number of a's or b's. Thus the new string cannot keep the number of a's less than the number of b's which is less than the number of c's, i.e. $k + 2$. If c is contained in vwx, then a is not contained in vwx. Thus uv^kwx^ky would have at least $(k + 2) + (k - 1) = 2k + 1$ number of c's while keeping the number of a's the same, i.e. k. Hence $uv^kwx^ky \notin L$, and consequently, L is not context-free.

e) Assume that language $L = \{ww^Rw \mid w \in \{a, b\}^*\}$ is context-free. By the pumping lemma, there is a number k such that every string in L with length k or more can be written $uvwxy$ where

(i) $length(vwx) \leq k$

(ii) v and x are not both null

(iii) $uv^iwx^iy \in L$ for all $i \geq 0$.

The string $z = (a^kb^k)(a^kb^k)^R(a^kb^k) = a^kb^{2k}a^{2k}b^k$ must have a decomposition $uvwxy$ that satisfies the preceding conditions. By condition (ii), we have v and x have at least one terminal. Without loss of generality, assume that at least one a is in v or x (similar argument for the case of at least one b in v or x). Condition (i) requires the length of vwx to be at most k. This implies that the substring vwx of z cannot contain a's from both sides of b^{2k}. If the a's in the substring vwx of z are before b^{2k}, then uv^2wx^2y increases the number of a's before b^{2k} while keeping the number of a's after b^{2k} the same as $2k$. Hence uv^2wx^2y is no long in $L = \{ww^Rw \mid w \in \{a, b\}^*\}$. If the a's in the substring vwx of z are after b^{2k}, we have $uv^2wx^2y \notin L$ by similar argument. Therefore L is not context-free.

f) Assume that the language L consisting of prefixes of string

$$abaabaaabaaaab \cdots ba^nba^{n+1}b \cdot$$

is context-free and let k be the number specified by the pumping lemma. Consider the string $z = abaab \cdots ba^kb$, which is in the language and has length greater than k. Thus z can be written $uvwxy$ where

(i) $length(vwx) \neq k$

(ii) v and x are not both null

(iii) $uv^iwx^iy \in L$ for all $i \geq 0$.

To show that the assumption that L is context-free produces a contradiction, we examine all possible decomposition of z that satisfy the conditions of the pumping lemma. By (ii), one or both of v and x must be nonnull. In the following argument we assume that $v \neq \lambda$.

Case 1: v has no b's. In this case, v consists solely of a's and lies between two consecutive b's. That is, v occurs in z in a position of the form

$$\cdots ba^nba^iva^jba^{n+2}b\cdots$$

where $i + length(v) + j = n + 1$. Pumping v produces an incorrect number of a's following ba^nb and, consequently, the resulting string is not in the language.

Case 2: v has two or more b's. In this case, v contains a substring ba^nb. Pumping v produces a string with two substrings of the form ba^nb. No string with this property is in L.

Case 3: v has one b. Then v can be written a^iba^j and occurs in z as

$$\cdots ba^{n-1}ba^{n-i}va^{n+1-j}b\cdots$$

Pumping v produces the substring

$$\cdots ba^{n-1}ba^{n-i}a^iba^ja^iba^ja^{n+1-j}b\cdots = \cdots ba^{n-1}ba^nba^{j+i}ba^{n+1}b\cdots,$$

which cannot occur in a string in L.

Regardless of its makeup, pumping any nonnull substring v of z produces a string that is not in the language L. A similar argument shows that pumping x produces a string not in L whenever x is nonnull. Since one of v or x is nonnull, there is no decomposition of z that satisfies the requirements of the pumping lemma and we conclude that the language is not context-free.

■