1. (**60 points**)

(a) (**5 points**) Let Σ be the alphabet $\{\mathsf{a}, \mathsf{b}\}$. Give a context free grammar for the language, A_1, where

$$A_1 \quad = \quad \{w \in \Sigma^* \mid \exists n \in \mathbb{Z}^{\geq 0}.\ w = \mathsf{a}^n \mathsf{b}^{2n}\}$$

Note: in all problems, you don not need to write your grammar in CNF or any other "special" form. In fact, you should write your rules so that it is clear why they generate the specified language, and if it is not obvious, add a short explanation of the intuition behind your solution.

Solution:

$$S_0 \quad \rightarrow \quad \epsilon \quad \mid \quad \mathsf{a}\, S_0\, \mathsf{bb}$$

This is solution is "obvious" enough that it doesn't require further explanation.

(b) (**10 points**) Describe a PDA that recognizes language A_1. You can just draw a transition diagram where edges are labeled as in *Sipser*.
Solution:
See the PDA of Figure 1.

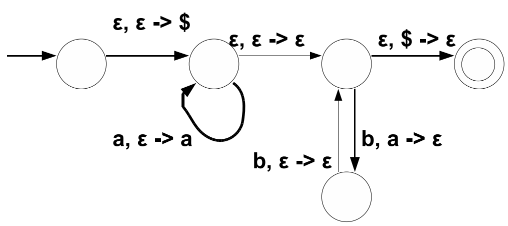

Figure 1: PDA for problem 1b.

(c) **(15 points)** Let Σ be the alphabet $\{\mathsf{a}, \mathsf{b}\}$. Give a context free grammar for the language, A_2, where

$$A_2 \quad = \quad \{w \in \Sigma^* \mid \#a(w) = 2\#b(w)\}$$

where $\#a(w)$ denotes the number of a's in w and likewise for $\#b(w)$. My grammar is fairly short, but it requires a bit of explanation to see that it is correct. Make sure that you include enough of an explanation of why your grammar is correct that your solution is convincing.

Solution:

$$\begin{array}{rcl} S_0 \rightarrow \epsilon & \mid & S_0\, S_0 \\ & \mid & \mathsf{a}\, S_0\, \mathsf{a}\, S_0\, \mathsf{b} \\ & \mid & \mathsf{a}\, S_0\, \mathsf{b}\, S_0\, \mathsf{a} \\ & \mid & \mathsf{b}\, S_0\, \mathsf{a}\, S_0\, \mathsf{a} \end{array}$$

I'll claim that it is obvious that every string generated by this grammar has twice as many a's as b's. Now, I'll show that every string that has twice as many a's as b's is generated by this grammar. Let $f(s) = \#a(s) - 2\#b(s)$.

Let w be an arbitrary string in A_2. I'll sketch the induction proof that $w \in A_2$.

If $w = \epsilon$, then w is generated by the derivation $S_0 \Rightarrow \epsilon$.

Otherwise, if we can find non-empty strings $x, y \in A_2$ such that $w = xy$, then $S_0 \Rightarrow S_0\, S_0 \Rightarrow xy = w$.

Otherwise, For any non-empty strings x and y with $xy = w$, $f(x) \neq 0$.

If $f(x) > 0$ for all x as described above, Then, w must be of the form $\mathsf{aa}u\mathsf{b}$ for some string $u \in \Sigma^*$, and $f(u) = 0$. Thus $u \in A_2$ and we get

$$\begin{array}{lcll}
S_0 & \Rightarrow & \mathsf{a}\,\S_0\mathsf{a}\,\S_0\,\mathsf{b} & \\
 & S_0 \;\rightarrow\; \mathsf{a}S_0\mathsf{a}S_0\mathsf{b} & & \\
 & \Rightarrow & \mathsf{aa}\,\S_0\,\mathsf{b}, & S_0 \;\rightarrow\; \epsilon \\
 & \overset{*}{\Rightarrow} & \mathsf{aa}\,u\,\mathsf{b}, & S_0 \;\overset{*}{\Rightarrow}\; u, \text{ by ind. hyp.} \\
 & = & w &
\end{array}$$

Thus, w is generated by the grammar.

If $f(x) < 0$ for all x as described above, Then an argument analagous to the one above shows that w is of the form $\mathsf{b}u\mathsf{aa}$ and is generated by the grammar.

Otherwise, $f(x)$ must change sign as we consider longer prefixes of w, but $f(x)$ is never 0. Note that if $f(x \cdot c) > f(x)$ for some $c \in \Sigma$, then $c = \mathsf{a}$ and $f(x \cdot c) = f(x) + 1$. Thus, the sign change in f must be from positive to negative. We conclude that w has the form $\mathsf{a}u\mathsf{b}v\mathsf{a}$ and

$$\begin{array}{lcll}
S_0 & \Rightarrow & \mathsf{a}\,\S_0\mathsf{b}\,\S_0\,\mathsf{a} & \\
 & S_0 \;\rightarrow\; \mathsf{a}S_0\mathsf{b}S_0\mathsf{a} & & \\
 & \overset{*}{\Rightarrow} & \mathsf{a}u\mathsf{b}\,v\,\mathsf{a}, & S_0 \;\overset{*}{\Rightarrow}\; u, v, \text{ by ind. hyp.} \\
 & = & w &
\end{array}$$

Thus, w is generated by the grammar.

This completes the proof (sketch). The langauge generated by the grammar given above is A_2.

(d) (**10 points**) Describe a PDA that recognizes language A_2. You can just draw a transition diagram where edges are labeled as in *Sipser*.

Solution:

See the PDA of Figure 2.

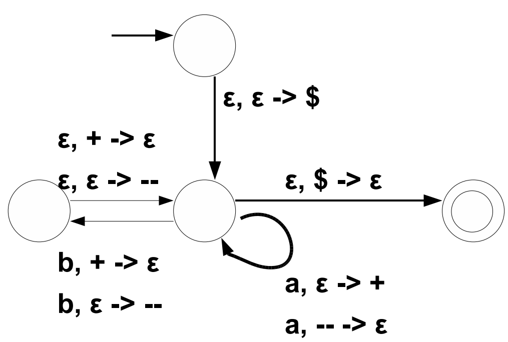

Figure 2: PDA for problem 1d.

(e) (**10 points**) Let Σ be the alphabet $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$. Give a context free grammar for the language, A_3, where

$$A_3 \quad = \quad \{w \in \Sigma^* \mid \exists i, j \in \mathbb{Z}^{\geq 0}.\ w = \mathsf{a}^i\mathsf{b}^j\mathsf{c}^{i+j}\}$$

Solution:

$$\begin{array}{lclclcl}
S_0 & \rightarrow & \epsilon & \mid & S_1 & \mid & \mathsf{a}\,S_0\,\mathsf{c} \\
S_1 & \rightarrow & \epsilon & \mid & \mathsf{b}\,S_1\,\mathsf{c} & &
\end{array}$$

S_0 is the start variable.

This one merits a bit of explanation. The rules for S_0 can derive ϵ (i.e. $\mathsf{a}^0\mathsf{b}^0\mathsf{c}^0$) or strings of the form $\mathsf{a}^iS_1\mathsf{c}^i$ (equivalently, $\mathsf{a}^i\mathsf{b}^0S_1\mathsf{c}^i$. Likewise, S_1 derives strings of the form $\mathsf{b}^j\mathsf{c}^j$. Thus, the grammar produces all strings of the form $\mathsf{a}^i\mathsf{b}^j\mathsf{c}^j\mathsf{c}^i = \mathsf{a}^i\mathsf{b}^j\mathsf{c}^{i+j}$, and no others. This is the language A_2.

(f) (**10 points**) Describe a PDA that recognizes language A_3. You can just draw a transition diagram where edges are labeled as in *Sipser*.
Solution:
See the PDA of Figure 3.

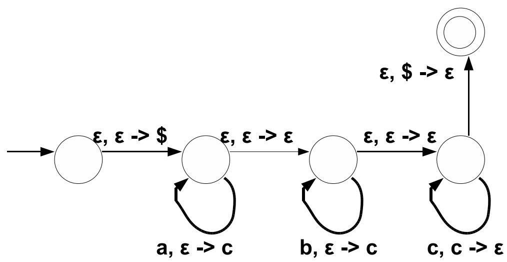

Figure 3: PDA for problem 1f.

2. (**10 points**) Prove that language B described below is not context free.

$$B \quad = \quad \{w \in \{\mathsf{a}, \mathsf{b}\}^* \mid (w = w^{\mathcal{R}}) \wedge (\#a(w) = \#b(w))\}$$

where $w^{\mathcal{R}}$ is the reverse of w. In English, B is the language of all palindromes that contain an equal number of a's and b's.
Solution:
Let p be a proposed pumping lemma constant, and let $w = a^p b^{2p} a^p \in B$. If $w = uvxyz$ with $|vxy| \leq p$ and $|vy| > 0$, it must be the case that $\#a(vy) = \#b(vy)$; otherwise uv^2xy^2z clearly has an unequal number of a's and b's. Thus we assume $\#a(vy) = \#b(vy)$. Without loss of generality, we assume that vxy is a substring of $a^p b^p$. Since v necessarily begins with a, then uv^2xy^2z has a prefix of a^{p+1}. This implies that $uv^2xy^2z \neq uv^2xy^2z^{\mathcal{R}}$, because this string has a postfix of ba^p.

3. (**20 points**) One of the languages described below is context free and the other is not. Determine which is which. Give a CFG or describe a PDA for the context-free language, and use the pumping lemma to prove that the other language is not context free. For both languages the alphabet is $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$.

$$C_1 = \{w \mid \exists i, j \in \mathbb{Z}^{\geq 0}.\ w = \mathsf{a}^i \mathsf{b}^j \mathsf{c}^i \mathsf{d}^j\}$$
$$C_2 = \{w \mid \exists i, j \in \mathbb{Z}^{\geq 0}.\ w = \mathsf{a}^i \mathsf{b}^j \mathsf{c}^j \mathsf{d}^i\}$$

Solution:
C_1 is not context-free: Let p be a proposed pumping lemma constant, and let $w = a^p b^p c^p d^p \in C_1$. If $w = uvxyz$ with $|vxy| \leq p$ and $|vy| > 0$, then assume without loss of generality that v only contains an a or a b (or both) (as the cases of v containing c or d is analogous, as is the case of y containing a particular symbol). If v contains an a, then y does not contain a c. On the other hand, if v contains a b, then y does not contain a d. In either case, $uv^2xy^2z \notin C_1$, as either the number of a's increase while the number of c's do not, or the number of b's increase while the number of d's do not.
C_2 is context-free, because it is given by the following grammer:
$S \rightarrow aSd|T$
$T \rightarrow bTc|\epsilon$

4. (**20 points**) One of the languages described below is context free and the other is not. Determine which is which. Give a CFG or describe a PDA for the context-free language, and use the pumping lemma to prove that the other language is not context free. For both languages the alphabet is $\{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}$.

$D_1 \quad = \quad \{x_1 \mathsf{c} x_2 \mathsf{c} \cdots x_k \mid$ each $x_i \in \{\mathsf{a}, \mathsf{b}\}^*$, and for every $i, j \in 1 \ldots k$, if $i \neq j$, then $x_i \neq x_j.\}$
$D_2 \quad = \quad \{x_1 \mathsf{c} x_2 \mathsf{c} \cdots x_k \mid$ each $x_i \in \{\mathsf{a}, \mathsf{b}\}^*$, there is some pair $i, j \in 1 \ldots k$ with $i \neq j$ and $x_i \neq x_j.\}$

Solution:

D_1 is not context-free: Let p be a proposed pumping lemma constant, and let $w = a^0ca^1ca^2c...ca^{p-1}ca^p \in D_1$. If $w = uvxyz$ with $|vxy| \leq p$ and $|vy| > 0$, we consider two cases. (1) If vy contains a c, then (wlog, assume v contains the c) $v = qcr$ for some strings q and r, and $v^3 = qcrqcrqcr$ has two common substrings delimited by c, which is the longest prefix of rq that does not contain a c. Therefore, $uv^3xy^3z \notin D_1$. If vy does not contain a c, then v is contained within ca^ic for some $1 \leq 1 \leq p$. Then, $uv^0xy^0z \notin D_1$ because no string of length i appears but there are still a total of p c-symbols, so there are two equal strings by the pigeon-hole principle.

D_2 is context free. The key observation is that if there are two x_i's that differ, then there is an i such that $x_i \neq x_{i+1}$. Figure 4 shows a PDA that recognizes language D_2.

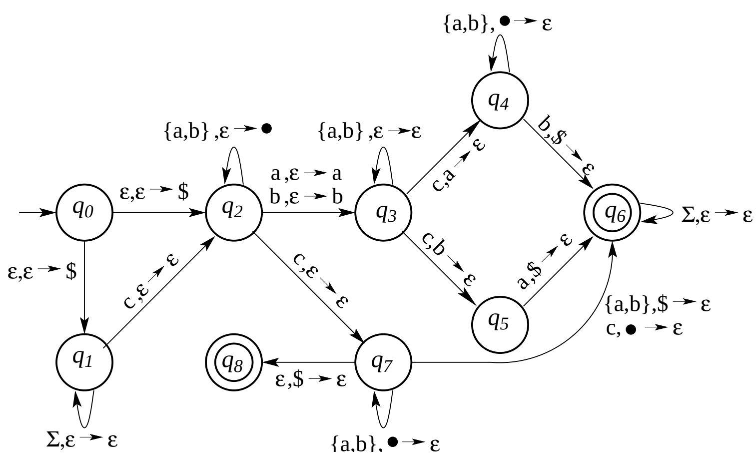

Figure 4: PDA for language D_2 (problem 4).

The PDA initially pushes an endmarker, $\$$ onto the stack. It moves directly to state q_2 if $x_1 \neq x_2$. Otherwise, it moves to state q_1 to skip over $x_1, x_2, \ldots$ to get to a pair that differ.

Now, note that if x_i and x_{i+1} differ then either they have the same lengths but have different symbols in some position OR they have different lengths. If they have the same lengths, then in state q_2 the PDA pushes markers, $\bullet$'s, onto the stack until it reaches a symbol that differs for the two strings. It pushes this symbol for the x_i string onto the stack and transitions to state q_3. In state q_3, the PDA skips over the rest of x_i. When it reaches the c that separates x_i from x_{i+1} it transitions to state q_4 if the symbol that it has guessed will be different was an a in string x_i and to state q_5 if it was a b. In state q_4, the PDA pops markers until it reaches the symbol in the same position as the a in string x_i. If the corresponding symbol in x_{i+1} is a b, the PDA transitions to state q_6 and accepts. The operation in state q_5 is similar.

If x_i and x_{i+1} have different lengths, the PDA stays in state q_2 the entire time that it reads x_i and transitions to state q_7 when it reads the c that separates x_i from x_{i+1}. At this point, the number of markers on the stack is equal to the length of x_i. In state q_7, the PDA pops off one marker for each symbol of x_{i+1}. If $|x_{i+1}| > |x_i|$ then the PDA will read an a or b when the $\$$ marker is on the top of the stack and it will transition to state q_6 and accept once it finishes reading the string. If $|x_{i+1}| < |x_i|$ and x_{i+1} is not the last substring, then the PDA will read a c while there are still one or more $\bullet$ markers on the stack. Again, the PDA will transition to state q_6 and eventually accept. Finally, if $x_{i+1}| < |x_i|$ and x_{i+1} is the last substring, then the PDA will reach the end of the input while there are still one or more $\bullet$ markers on the stack. In this case, the PDA will transtion to state q_8 and accept.