Problem 1  (5 pts)

Consider the following language

\[ \{ww^R \mid w \in \{0,1\}^*\} \]

and its 4-state PDA mentioned in the lecture. Draw a tree to simulate the following input string.

0000

Note that the tree must be a complete one. You cannot just use one path to accept/reject the string.

Answer

The tree is as following.

\[
\text{start} \rightarrow q_1, \{\} \xrightarrow{\epsilon} q_2, \{\$\} \xrightarrow{\epsilon} q_3, \{\$\} \xrightarrow{\epsilon} q_4, \{\$\} \times
\]

\[
\downarrow 0
\]

\[
q_2, \{0,\$\} \xrightarrow{\epsilon} q_3, \{0,\$\}
\]

\[
\downarrow 0 \quad \downarrow 0
\]

\[
q_3, \{0,0,\$\} \leftarrow q_2, \{0,0,\$\} \quad q_3, \{0,\$\} \xrightarrow{\epsilon} q_4, \{\$\} \times
\]

\[
\downarrow 0 \quad \downarrow 0
\]

\[
q_3, \{0,\$\} \quad q_2, \{0,0,0,\$\} \xrightarrow{\epsilon} q_3, \{0,0,0,\$\} \downarrow 0
\]

\[
q_4, \{\} \leftarrow q_3, \{\$\} \quad q_2, \{0,0,0,0,\$\} \xrightarrow{\epsilon} q_3, \{0,0,0,0,\$\} \quad q_3, \{0,0,\$\}
\]

\[
\checkmark \quad \times \quad \times
\]
Problem 2  (25 pts)

Consider the following language

\[ \{ w \mid w \in \{0, 1\}^* \text{ has an even number of zeros} \} \]

and its DFA

(a) (10 pts) In our lecture, we describe a way to transform each link to a rule of CFG. Use that way to generate a CFG for this language.

(b) (15 pts) Generate a CFG for this language with the smallest number of rules. Explain why yours has the smallest number.

Answer

(a) We can get the following CFG.

\[ V = \{ R_0, R_1, R_2 \} \quad \Sigma = \{ 0, 1 \} \quad S = R_1 \]

The rules are

\[ R_1 \rightarrow 0R_2 \mid 1R_1 \mid \epsilon \]
\[ R_2 \rightarrow 0R_1 \mid 1R_2 \]

(b) The language with smallest rules are

\[ V = \{ R_0 \} \quad \Sigma = \{ 0, 1 \} \quad S = R_0 \]

The rules are

\[ R_0 \rightarrow R_00R_00R_0 \mid 1R_0 \mid \epsilon \]

We explain that

language of CFG = language of DFA
First, by the first rule, any string generated by the CFG must have an even number of zeros. Thus,
\[
\text{language of CFG } \subseteq \text{ language of DFA}
\]
Second, for any string that has an even number of zeros, we can represent it as
\[
1^*01^*0...01^*01^*
\]
By our rules, \(1^*\) can be generated by the second and third rules. Thus,
\[
\text{language of CFG } \supseteq \text{ language of DFA}
\]
Therefore, the proof is complete.
One rule is impossible: It is because the CFG with only one rule can generate at most one string. However, there are a lot of strings in the language.
Two rules are impossible: It is because we need to generate 1 and 11. By the above argument, we need two rules without 0, but we also need to generate 0. There must be another rule for 0.
Thus, it is impossible to have a CFG for this language with less than 3 rules.

**Another solution**

\[
R_0 \rightarrow 0R_00R_0 \mid 1R_0 \mid \epsilon
\]

**Problem 3  (45 pts)**

Consider the following NFA,

![NFA Diagram]

We assume \(\Sigma = \{1\}\)

(a) (5 pts) Modify it to be a PDA and give the formal definition. Note that you can use only two states.

(b) (15 pts) We would like to use the procedure in Lemma 2.27 to transform the PDA obtained in (a) to a CFG. To begin, we modify the PDA to another PDA with no more than 3 states satisfying

(i) It has a single accept state.
(ii) It empties its stack before accepting.

(iii) Each transition either pushes a symbol onto the stack or pops one off the stack, but it
does not do both at the same time.

Then please generate the CFG.

(c) (20 pts) Convert the CFG obtained in (b) to CNF. You must show the details.

(d) (5 pts) Simplify the CNF in (c) to the smallest number of rules. You must explain
why some rules in your CNF are not needed and therefore the rule set can be
simplified to your answer.

Answer

(a) The PDA diagram

\[
\text{start} \rightarrow q_0 \xrightarrow{1, \epsilon \rightarrow \epsilon} q_1
\]

\[Q = \{q_0, q_1\}\]
\[\Sigma = \{1\}\]
\[\Gamma = \emptyset\]

\[\text{start state} = q_0\]
\[F = \{q_1\}\]

\[\delta(q, \sigma, \gamma) = \begin{cases} 
\{(q_1, \epsilon)\}, & q = q_0, \sigma = 1, \gamma = \epsilon, \ q \in Q, \sigma \in \Sigma, \gamma \in \Gamma_
\emptyset, & \text{otherwise}
\end{cases}\]

(b) To satisfy those condition, consider

\[
\text{start} \rightarrow q_1 \xrightarrow{1, \epsilon \rightarrow 1} q_2 \xrightarrow{\epsilon, 1 \rightarrow \epsilon} q_3
\]

By Lemma 2.27, we can generate a CFG from above PDA. For simplicity, we denote \(A_{q_i, q_j}\)
as $A_{ij}$

\[ S \rightarrow A_{13} \\
A_{11} \rightarrow A_{11}A_{11}|A_{12}A_{21}|A_{13}A_{31}|\epsilon \\
A_{12} \rightarrow A_{11}A_{12}|A_{12}A_{22}|A_{13}A_{32} \\
A_{13} \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|1A_{22} \\
A_{21} \rightarrow A_{21}A_{11}|A_{22}A_{21}|A_{23}A_{31} \\
A_{22} \rightarrow A_{21}A_{12}|A_{22}A_{22}|A_{23}A_{32}|\epsilon \\
A_{23} \rightarrow A_{21}A_{13}|A_{22}A_{23}|A_{23}A_{33} \\
A_{31} \rightarrow A_{31}A_{11}|A_{32}A_{21}|A_{33}A_{31} \\
A_{32} \rightarrow A_{31}A_{12}|A_{32}A_{22}|A_{33}A_{32} \\
A_{33} \rightarrow A_{31}A_{13}|A_{32}A_{23}|A_{33}A_{33}|\epsilon \\

(c) First we remove $\epsilon$,

\[ S \rightarrow A_{13} \\
A_{11} \rightarrow A_{11}A_{11}|A_{12}A_{21}|A_{13}A_{31}|A_{11} \\
A_{12} \rightarrow A_{11}A_{12}|A_{12}A_{22}|A_{13}A_{32}|A_{12} \\
A_{13} \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|A_{13}|1A_{22}|1 \\
A_{21} \rightarrow A_{21}A_{11}|A_{22}A_{21}|A_{23}A_{31}|A_{21} \\
A_{22} \rightarrow A_{21}A_{12}|A_{22}A_{22}|A_{23}A_{32}|A_{22} \\
A_{23} \rightarrow A_{21}A_{13}|A_{22}A_{23}|A_{23}A_{33}|A_{23} \\
A_{31} \rightarrow A_{31}A_{11}|A_{32}A_{21}|A_{33}A_{31}|A_{31} \\
A_{32} \rightarrow A_{31}A_{12}|A_{32}A_{22}|A_{33}A_{32}|A_{32} \\
A_{33} \rightarrow A_{31}A_{13}|A_{32}A_{23}|A_{33}A_{33}|A_{33} \]
Then we remove unit rules,

\[
S \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|1A_{22}|1 \\
A_{11} \rightarrow A_{11}A_{11}|A_{12}A_{21}|A_{13}A_{31} \\
A_{12} \rightarrow A_{11}A_{12}|A_{12}A_{22}|A_{13}A_{32} \\
A_{13} \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|1A_{22}|1 \\
A_{21} \rightarrow A_{21}A_{11}|A_{22}A_{21}|A_{23}A_{31} \\
A_{22} \rightarrow A_{21}A_{12}|A_{22}A_{22}|A_{23}A_{32} \\
A_{23} \rightarrow A_{21}A_{13}|A_{22}A_{23}|A_{23}A_{33} \\
A_{31} \rightarrow A_{31}A_{11}|A_{32}A_{21}|A_{33}A_{31} \\
A_{32} \rightarrow A_{31}A_{12}|A_{32}A_{22}|A_{33}A_{32} \\
A_{33} \rightarrow A_{31}A_{13}|A_{32}A_{23}|A_{33}A_{33}
\]

Finally we add a new variable and rules to be a CNF,

\[
S \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|1A_{22}|1 \\
A_{11} \rightarrow A_{11}A_{11}|A_{12}A_{21}|A_{13}A_{31} \\
A_{12} \rightarrow A_{11}A_{12}|A_{12}A_{22}|A_{13}A_{32} \\
A_{13} \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|1A_{22}|1 \\
A_{21} \rightarrow A_{21}A_{11}|A_{22}A_{21}|A_{23}A_{31} \\
A_{22} \rightarrow A_{21}A_{12}|A_{22}A_{22}|A_{23}A_{32} \\
A_{23} \rightarrow A_{21}A_{13}|A_{22}A_{23}|A_{23}A_{33} \\
A_{31} \rightarrow A_{31}A_{11}|A_{32}A_{21}|A_{33}A_{31} \\
A_{32} \rightarrow A_{31}A_{12}|A_{32}A_{22}|A_{33}A_{32} \\
A_{33} \rightarrow A_{31}A_{13}|A_{32}A_{23}|A_{33}A_{33} \\
B \rightarrow 1
\]

(d) Because \( S \) is the start variable, the first rule applied is

\[
S \rightarrow A_{11}A_{13}|A_{12}A_{23}|A_{13}A_{33}|1A_{22}|1.
\]

If we take

\[
S \rightarrow A_{11}A_{13},
\]
we argue that no string can be generated. The reason is that we must have

\[ S \rightarrow A_1 \cdots A_1 A_1, \]

where \( A_1 \) becomes a terminal. However there is no such a rule. By a similar argument, we have that none of

\[
S \rightarrow A_{12} A_{23} \\
S \rightarrow A_{13} A_{33} \\
S \rightarrow B A_{22}
\]

can generate a string. Therefore, because from \( S \) we can only generate 1, all other rules are useless. Thus, \( S \rightarrow 1 \) becomes the CFG with the smallest number of rules.

Common Mistakes

(a)

- \( F \) is a set.

(c)

- You cannot remove \( A_{11} \rightarrow A_{11} A_{11} \) while removing \( A_{11} \rightarrow \epsilon \)

Problem 4 (25 pts)

(a) (15 pts) Design a Turing machine with no more than 10 states (including \( q_{\text{accept}} \) and \( q_{\text{reject}} \)) that accepts the language

\[ \{ \#w\#a^n | w \in \{0, 1\}^* \text{ and is a binary representation for } n \}, \]

where \( \Sigma = \{\#, 0, 1, a\}, \Gamma = \{\#, 0, 1, a, \times, \omega\} \). For example, the following strings are accepted

- ##
- #0#
- #01#a
- #110#aaaaaa

(b) (10 pts) Use the Turing machine obtained in (a) to simulate the following strings

(i) #010#aa
(ii) #10#a
Answer

(a) The concept is in each round, minus $w$ by one as well as cross off one $a$. Here we describe a Turing machine $M$ that recognize this language.

$M_1 =$ "On input string $s$

1. Find the right most element of $w$
2. (Minus $w$ by one) Find the right most 1 and replace it by 0, and replace every 0 right of it by 1.
3. If in stage 2 the tape contains no 1 but still contains $a$, reject.
4. If in stage 2 the tape contains neither 1 nor $a$, accept.
5. (Cross off one $a$) Find the left most $a$ and cross it off (i.e., replace it with $\times$).
6. If in stage 5 the tape contains no $a$, reject.
7. Go to stage 1."

The state diagram of $M_1$. Note that links not shown go to $q_r$

(b) (i) For #010#aa
For this problem, (a) and (b) are graded together, so you only see one number as your total points of this problem.

Alternative solution of 4(a)

1. This answer also crosses off one $a$ while it minuses $w$ by 1. But when it minuses $w$, it doesn’t need to find the right most 1 of $w$ first.

$$M_2 = \text{"On input string } s\text{"

1. Find the right most element of $w$

2. (Minus $w$ by one) Go left and replace every scanned 0 with 1, until touching a $\#$, then go to stage 3. (This indicate before this stage, $w$ is 0). Or until a 1, then replace the 1 with 0 and go to stage 4.

3. If the tape contains no $a$ accept, else, reject.

4. (Cross off one $a$) Find the left most $a$ and cross it off (i.e., replace it with $\times$).

5. If in stage 4 the tape contains no $a$, reject.

6. Go to stage 1."
2. This answer crosses off nearly half of the $a$ string instead of crossing it one by one.

$$M_3 = \text{"On input string } s\text{\"}$$

1. If the right most digit of $w$(not including $\times$) is 1, go to stage 2; it’s
   0, go to stage 3; it has no character left, go to stage 4.
2. (Minus $w$ by one) Replace this 1 with 0 and cross off one $a$. If no $a$
   left, reject.
3. (Divide $w$ by two) Cross off the 0. If the length of $a$ is odd, reject,
   else, cross off half of the $a$ string.
4. If the tape contains no $a$ accept, else, reject.
5. Go to stage 1."
The state diagram of $M_3$. Note that links not shown go to $q_r$. 