Introduction to the Theory of Computation 2024 — Midterm 2

Solutions

Problem 1 (30 pts). Consider the CFG (V, Σ, R, S) with

$$V = \{S, B, J\} \text{ and } \Sigma = \{b, j\},$$

where the rule set R contains the following rules:

$$\begin{aligned} S &\to B \mid BJ \\ B &\to BB \mid b \mid \varepsilon \\ J &\to jJb \mid j \mid \varepsilon \end{aligned} \tag{1}$$

(a) (5 pts) Please provide leftmost derivations for the input strings

$$jjb \text{ and } bbbb$$

by using CFG (1). What you need to give is a sequence of derivations. No need to draw a tree.

(b) (5 pts) Is the string

$$j$$

derived ambiguously in CFG (1)? Please provide your reasons for determining ambiguity in CFG (1).

(c) (10 pts) Convert CFG (1) to CNF by the following steps:

(i) Add a new start state.

(ii) Remove $X \to \varepsilon$ with the order

$$B \to \varepsilon,\ J \to \varepsilon,\ S \to \varepsilon,$$

for any variable X that is not the start state.

(iii) Handle $X \to Y$, for all variables X and Y. Please follow the order

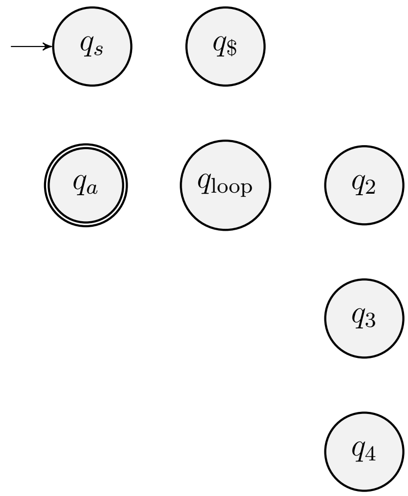
$$J \to J,\ J \to B,\ J \to S,\ B \to J,\ B \to B,\ B \to S,\ S \to J,\ S \to B,\ S \to S,$$
$$S_0 \to B,\ S_0 \to J,\ S_0 \to S.$$

(iv) Convert $X \to u_1 u_2 u_3$, where $k \geq 3$ and each u_i is a variable or terminal symbol.

(v) Replace any terminal u_i in the preceding rules with $U_1 \to j$ and $U_2 \to b$.

Please ensure that all intermediate steps are clearly documented, showing how CFG (1) is transformed into CNF.

(d) (10 pts) Please convert CFG (1) to a PDA with the following draft



by the procedure outlined in Lemma 2.21 of the textbook (in our slides chap2_PDA3.pdf). Please note that we do not allow adding states.

Solution.

(a) We show the leftmost derivations of those strings on the following.

(i) jjb.

$$S \to BJ \to \varepsilon J \to \varepsilon jJb \to \varepsilon jjb.$$

(ii) $bbbb$.

$$S \to B \to BB \to BBB \to BBBB \to bBBB \to bbBB \to bbbB \to bbbb.$$

(b) Since j can be derived by the following leftmost derivations

(i)

$$S \to BJ \to \varepsilon J \to \varepsilon j, \text{and}$$

(ii)

$$S \to BJ \to BBJ \to \varepsilon BJ \to \varepsilon\varepsilon J \to \varepsilon\varepsilon j,$$

j is derived ambiguously in CFG (1)

(c)
- Add $S_0 \to S$.

$$\begin{aligned} S_0 &\to S \\ S &\to B \mid BJ \\ B &\to BB \mid b \mid \varepsilon \\ J &\to jJb \mid j \mid \varepsilon \end{aligned}$$

- Remove $B \to \varepsilon$

$$\begin{aligned} S_0 &\to S \\ S &\to B \mid BJ \mid J \mid \varepsilon \\ B &\to BB \mid b \mid B \\ J &\to jJb \mid j \mid \varepsilon \end{aligned}$$

- Remove $J \to \varepsilon$

$$\begin{aligned} S_0 &\to S \\ S &\to B \mid BJ \mid J \mid \varepsilon \\ B &\to BB \mid b \mid B \\ J &\to jJb \mid j \mid jb \end{aligned}$$

- Remove $S \to \varepsilon$

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to B \mid BJ \mid J \\ B &\to BB \mid b \mid B \\ J &\to jJb \mid j \mid jb \end{aligned}$$

- Remove $B \to B$

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to B \mid BJ \mid J \\ B &\to BB \mid b \\ J &\to jJb \mid j \mid jb \end{aligned}$$

- Remove $S \to J$

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to B \mid BJ \mid jJb \mid j \mid jb \\ B &\to BB \mid b \\ J &\to jJb \mid j \mid jb \end{aligned}$$

- Remove $S \to B$

$$\begin{aligned} S_0 &\to S \mid \varepsilon \\ S &\to BB \mid b \mid BJ \mid jJb \mid j \mid jb \\ B &\to BB \mid b \\ J &\to jJb \mid j \mid jb \end{aligned}$$

- Remove $S_0 \to S$

$$\begin{aligned} S_0 &\to BB \mid b \mid BJ \mid jJb \mid j \mid jb \mid \varepsilon \\ S &\to BB \mid b \mid BJ \mid jJb \mid j \mid jb \\ B &\to BB \mid b \\ J &\to jJb \mid j \mid jb \end{aligned}$$

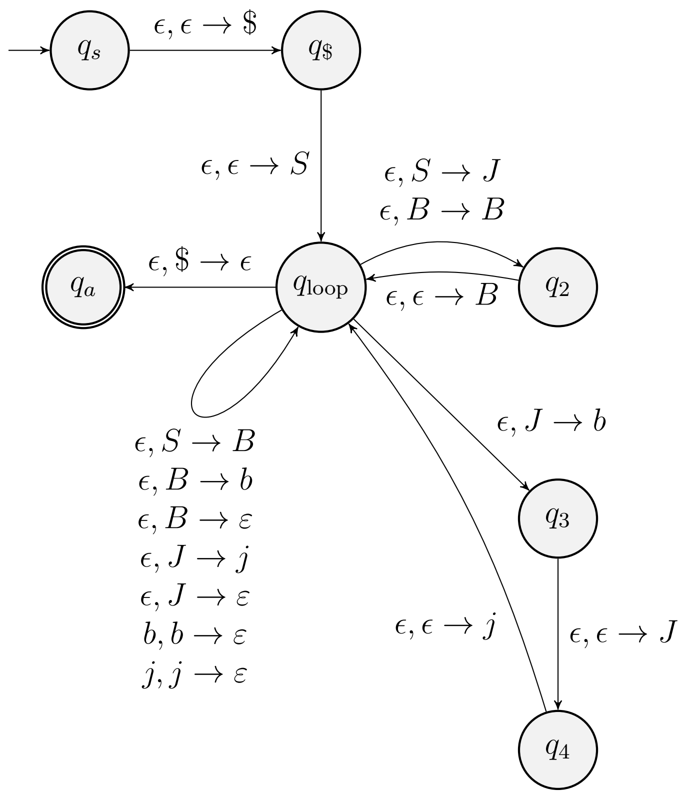
- Convert $A \to u_1 u_2 \cdots u_3$, where $k \geq 3$.

$$\begin{aligned} S_0 &\to BB \mid b \mid BJ \mid jA_1 \mid j \mid jb \mid \varepsilon \\ S &\to BB \mid b \mid BJ \mid jA_1 \mid j \mid jb \\ B &\to BB \mid b \\ J &\to jA_1 \mid j \mid jb \\ A_1 &\to Jb \end{aligned}$$

- Convert remaining rules.

$$\begin{aligned}
S_0 &\to BB \mid b \mid BJ \mid U_1A_1 \mid j \mid U_1U_2 \mid \varepsilon \\
S &\to BB \mid b \mid BJ \mid U_1A_1 \mid j \mid U_1U_2 \\
B &\to BB \mid b \\
J &\to U_1A_1 \mid j \mid U_1U_2 \\
A_1 &\to JU_2 \\
U_1 &\to j \\
U_2 &\to b
\end{aligned}$$

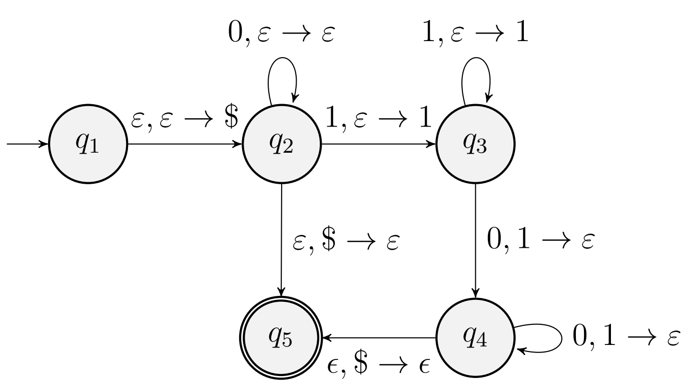
(d) Please see the following diagram.



Problem 2 (10 pts). Consider the following language

$$\{0^m 1^n 0^n \mid m \geq 0, n \geq 0\} \tag{2}$$

We have a 5-state PDA



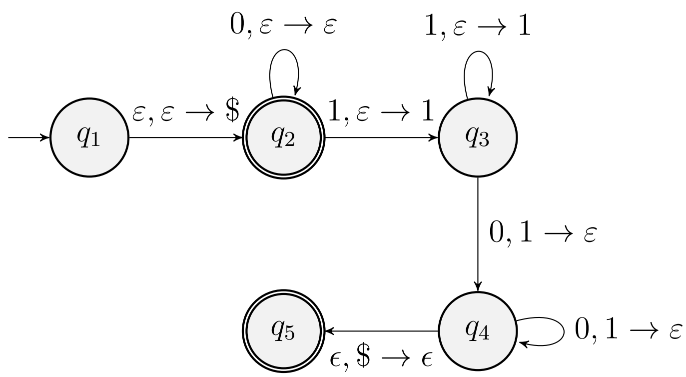
for the language (2) with $\Sigma = \{0, 1\}$ and $\Gamma = \{1, \$\}$.

(a) (5 pts) Please complete the following table of δ function.

	0			1			ε		
	1	\$	ε	1	\$	ε	1	\$	ε
q_1									
q_2									
q_3									
q_4									
q_5									

You do not need to write done $\emptyset$ and can leave these place emtpy.

(b) (5 pts) If we modify the PDA to a 6-state DPDA



where the rejected state q_r is not shown in the diagram for the readability. Please show the table of δ function. You do not need to write done $\emptyset$ and can leave these place emtpy.

Solution.

(a) Please see the following table.

	0			1			ε		
	1	\$	ε	1	\$	ε	1	\$	ε
q_1									$\{(q_2, \$)\}$
q_2			$\{(q_2, \varepsilon)\}$			$\{(q_3, 1)\}$		$\{(q_5, \varepsilon)\}$	
q_3	$\{(q_4, \varepsilon)\}$					$\{(q_3, 1)\}$			
q_4	$\{(q_4, \varepsilon)\}$							$\{(q_5, \varepsilon)\}$	
q_5									

Empty slots have the value $\emptyset$.

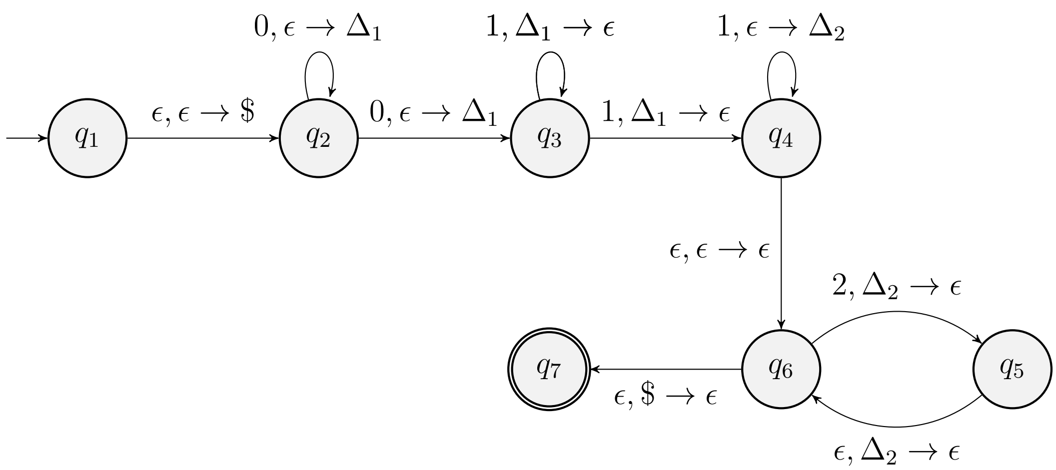
(b) The δ function of this diagram is

	0			1			ε		
	1	\$	ε	1	\$	ε	1	\$	ε
q_1									$(q_2, \$)$
q_2			(q_2, ε)			$(q_3, 1)$			
q_3	(q_4, ε)	(q_r, ε)				$(q_3, 1)$			
q_4	(q_4, ε)			(q_r, ε)				(q_5, ε)	
q_5	(q_r, ε)	(q_r, ε)		(q_r, ε)	(q_r, ε)				
q_r	(q_r, ε)	(q_r, ε)		(q_r, ε)	(q_r, ε)				

Empty slots have the value $\emptyset$.

Problem 3 (40 pts). Please consider the following PDA P with

- $Q = \{q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$ is the set of states.
- $\Sigma = \{0, 1, 2\}$ is the input alphabet.
- $\Gamma = \{\$, \Delta_1, \Delta_2\}$ is the stack alphabet.



(a) (5 pts) Please follow page 5 of our slides in chap2_PDA2.pdf to simulate the given PDA P on the input string **01112** by drawing the corresponding simulation trees. Then, determine whether the PDA P **accepts** this input string based on your simulation.

(b) (10 pts) What is the language recognized by P? Please provide the details to explain your answer.

(c) (15 pts) Please identify and explain which of the following conditions mentioned in page 2 of "chap_PDA4.pdf" is not satisfied by PDA P.

- A single accept state.
- Stack should be empty before accepting.
- Each transition should either pushes or pops something from the stack, but not both at the same time.

Next, please convert PDA P to P' so that it satisfies all three conditions mentioned above. Please note that your Σ, Γ and Q must be the same. You are allowed only to change δ (i.e., the links between nodes). You need to provide reasons for your changes.

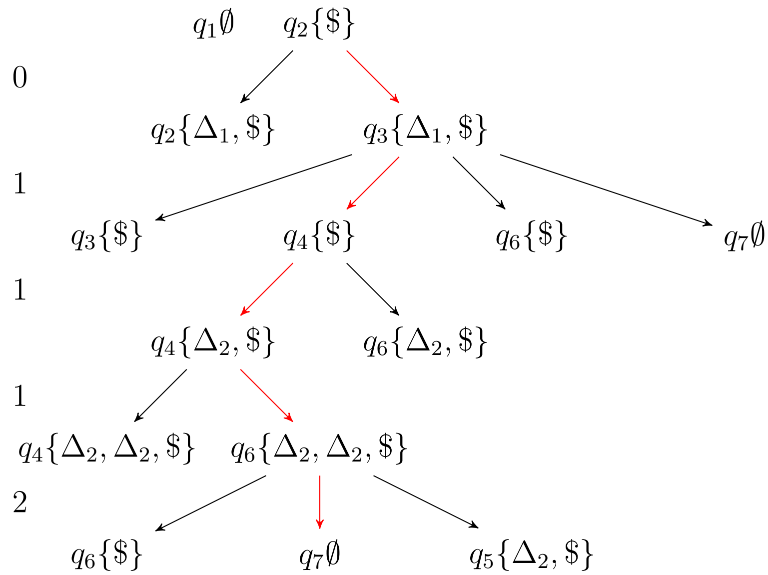
Hint: See if you can remove one link and add another one so that the language is the same.

Additionally, to confirm that your P' recognizes same language as in (b), you need to do (a) again with your P'. Please draw the corresponding simulation trees and determine whether the P' **accepts** the input string **01112**.

(d) (10 pts) Based on (c), convert P' to a CFG by using the procedure in Lemma 2.27 of the textbook. For simplicity, you only need to write each $A_{pq} \to aA_{rs}b$ rule. The rules $A_{pq} \to A_{pr}A_{rq}$ and $A_{pp} \to \epsilon$ are not needed. In order to prepare for $A_{pq} \to aA_{rs}b$ rules, please give table(s) for each stack alphabet t pushed/popped, similar to what we had in slides.

Solution.

(a) For input string **01112**



This PDA **accepts** input string **01112**.

(b) We observe that states q_1, q_2, and q_3 push m Δ_1 into the stack while reading 0s and then pop m Δ_1 when reading 1's. Moreover, since states q_0 and q_1 cannot reach q_7 without reading 0's, the language cannot include ϵ. As a result, we can summarize that states q_1, q_2, and q_3 handle the following language:

$$\{0^m 1^m \mid m > 0\}.$$

For the other states, we observe that the loop between q_6 and q_5 pops $2n$ Δ_2 from the stack while reading 2's. Therefore, q_4 must push $2n$ Δ_2 into the stack while reading 1's. Moreover, because q_4 can reach q_7 by ϵ, the language includes ϵ. As a result, we can summarize that the other states handle the following language

$$\{1^{2n} 2^n \mid n \geq 0\}.$$

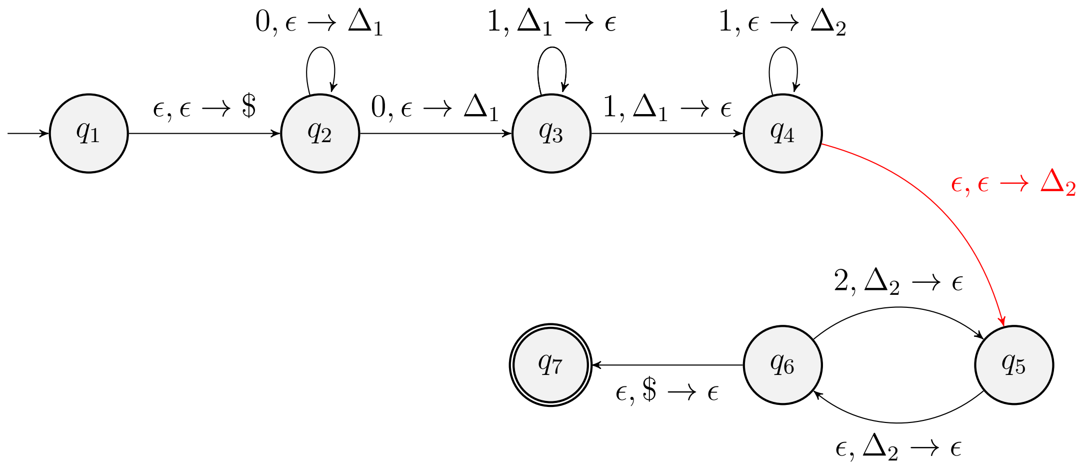
Since the first language will complete before handling the second language. Therefore, the PDA P recognizes the following language

$$\{0^m 1^{m+2n} 2^n \mid m > 0, n \geq 0\}.$$

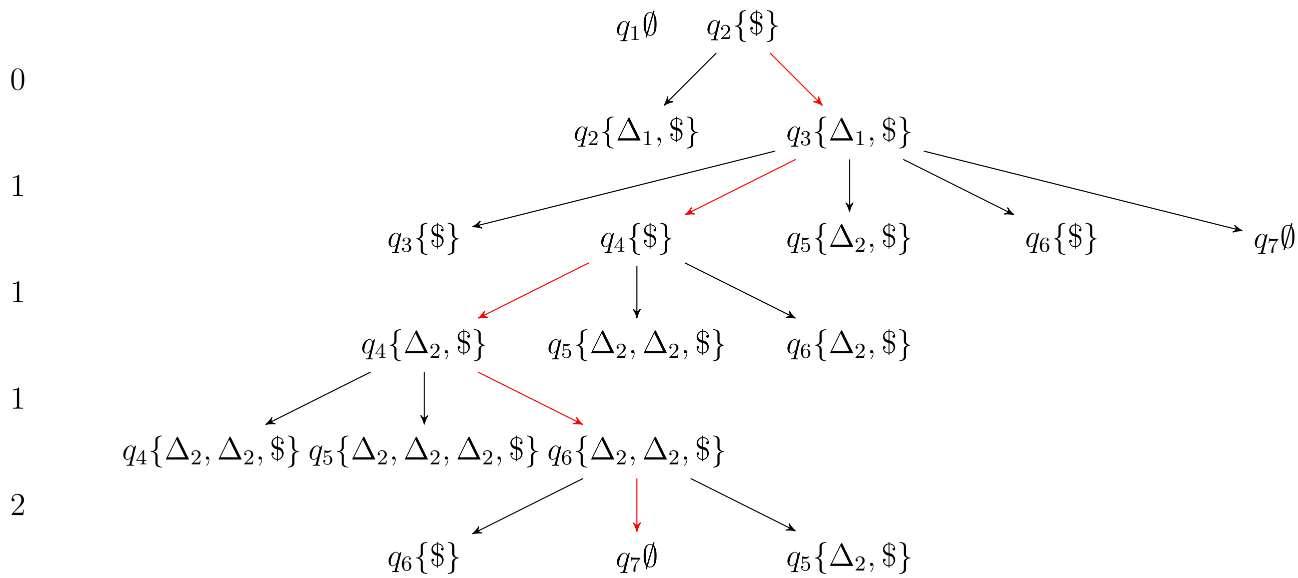
(c) The PDA P has an $\epsilon \to \epsilon$ transition from q_4 to q_6, which violates the third needed condition. Thus, we need to handle the transition $q_4 \to q_6$. We know that in the original figure, q_4, q_5, q_6 handle the following language

$$\{1^{2n} 2^n \mid n \geq 0\}.$$

Our strategy is to push $2n$ Δ_2 into stack by 1's, and then pop up $2n$ Δ_2 by the loop between q_6 and q_5 when reading 2's. To construct PDA P', our new strategy is to push $2n+1$ Δ_2 into stack and immediately pop up one Δ_2 via the path from q_5 to q_6. This ensures that the stack contains exactly $2n$ Δ_2 before reading 2's, allowing P' to pop up $2n$ Δ_2 through same loop between q_6 and q_5.



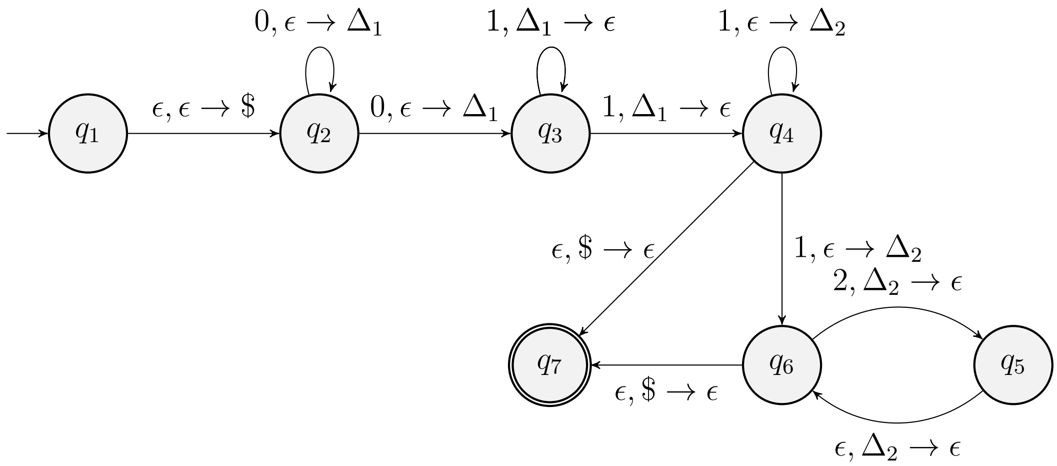
For input string **01112**



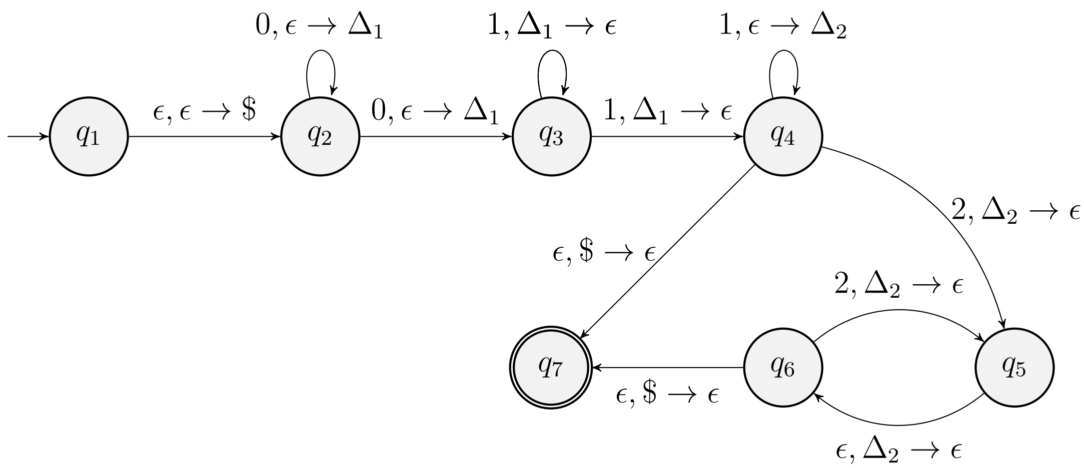
This PDA P' **accepts** input string **01112**.

We also accept the following PDAs; however, we did not provide the reason and the corresponding simulation trees.

Other solution I: Please consider the following PDA.



Other solution II: Please consider the following PDA.



(d) The corresponding CFG is the following:

- $t = \Delta_1$:

p	r	s	q	a	b	rules
2	2	3	3	0	1	$A_{23} \to 0A_{23}1$
2	2	3	4	0	1	$A_{24} \to 0A_{23}1$
2	3	3	3	0	1	$A_{23} \to 0A_{33}1$
2	3	3	4	0	1	$A_{24} \to 0A_{33}1$

- $t = \Delta_2$:

p	r	s	q	a	b	rules
4	4	5	6	1	ϵ	$A_{46} \to 1A_{45}$
4	4	6	5	1	2	$A_{45} \to 1A_{46}2$
4	5	5	6	ϵ	ϵ	$A_{46} \to A_{55}$
4	5	6	5	ϵ	2	$A_{45} \to A_{56}2$

- $t = \$$:

p	r	s	q	a	b	rules
1	2	6	7	ϵ	ϵ	$A_{17} \to A_{26}$

For other solutions, the corresponding CFGs are on the following.

Other solution I: The CFG which converting PDA (cI) is

- $t = \Delta_1$:

p	r	s	q	a	b	rules
2	2	3	3	0	1	$A_{23} \to 0A_{23}1$
2	2	3	4	0	1	$A_{24} \to 0A_{23}1$
2	3	3	3	0	1	$A_{23} \to 0A_{33}1$
2	3	3	4	0	1	$A_{24} \to 0A_{33}1$

- $t = \Delta_2$:

p	r	s	q	a	b	rules
4	4	5	6	1	ϵ	$A_{46} \to 1A_{45}$
4	4	6	5	1	2	$A_{45} \to 1A_{46}2$
4	6	5	6	1	ϵ	$A_{46} \to 1A_{65}$
4	6	6	5	1	2	$A_{45} \to 1A_{66}2$

- $t = \$$:

p	r	s	q	a	b	rules
1	2	6	7	ϵ	ϵ	$A_{17} \to A_{26}$
1	2	4	7	ϵ	ϵ	$A_{17} \to A_{24}$

Other solution II: The CFG which converting PDA (cII) is

- $t = \Delta_1$:

p	r	s	q	a	b	rules
2	2	3	3	0	1	$A_{23} \to 0A_{23}1$
2	2	3	4	0	1	$A_{24} \to 0A_{23}1$
2	3	3	3	0	1	$A_{23} \to 0A_{33}1$
2	3	3	4	0	1	$A_{24} \to 0A_{33}1$

- $t = \Delta_2$:

p	r	s	q	a	b	rules
4	4	4	5	1	2	$A_{45} \to 1A_{44}2$
4	4	5	6	1	ϵ	$A_{46} \to 1A_{45}$
4	4	6	5	1	2	$A_{45} \to 1A_{46}2$

- $t = \$$:

p	r	s	q	a	b	rules
1	2	4	7	ϵ	ϵ	$A_{17} \to A_{24}$
1	2	6	7	ϵ	ϵ	$A_{17} \to A_{26}$

Problem 4 (20 pts). In this problem, you will design a TM for

mapping a **non-negative** binary number into an equivalent quaternary number.

Let $\boldsymbol{x}$ be a string representing a binary number

$$\boldsymbol{x} = x_1x_2\cdots, \text{where } x_i \in \{0,1\}.$$

We have the following table to describe the relationship between different numerical systems.

Decimal	0	1	2	3	4	5	6	7
Binary	0000	0001	0010	0011	0100	0101	0110	0111
Quaternary	00	01	02	03	10	11	12	13

When doing the mapping, you can group every 2-bits in the binary number to make the mapping easier. For example, assume we have a number 213 in the decimal system, and the mapping can be completed by

$$\boldsymbol{x} = 11011000 = \underline{11}\ \underline{01}\ \underline{10}\ \underline{00} \to \underline{3}\ \underline{1}\ \underline{2}\ \underline{0} = 3120.$$

(a) (15 pts) Now, let us constrain the input $\boldsymbol{x}$ to have even length (i.e., $|\boldsymbol{x}| \bmod 2 = 0$,) and define the output $\boldsymbol{y}$ as

$$\boldsymbol{y} = y_1 y_2 \cdots, \text{where } y_i \in \{0, 1, 2, 3\}.$$

The TM has the following initial configuration

$$\boldsymbol{x}\# \sqcup \cdots,$$

and the final configuration

$$\underbrace{\sqcup \cdots \sqcup}_{|\boldsymbol{x}|} \# \boldsymbol{y} \sqcup \cdots.$$

For example, if 11011000 is the input, we have

$$11011000\# \sqcup \cdots$$

in the beginning. After we run the TM, the tape content should become

$$\underbrace{\sqcup \cdots \sqcup}_{8} \#3120 \sqcup \cdots.$$

To achieve this conversion, we sequentially process every two bits. For every $x_i x_{i+1}$, we calculate

$$2x_i + x_{i+1}$$

to get the corresponding $\boldsymbol{y}$ components.

Please follow the steps to design your TM:

Step 1: Process the bit x_i and replace it with $\sqcup$.

Step 2: Move to the corresponding position of $\boldsymbol{y}$ in the right side of #, and store the value $2 \times x_i$.

Step 3: Move back to x_{i+1}. Process the bit x_{i+1} and replace it with $\sqcup$.

Step 4: Move to the corresponding position of $\boldsymbol{y}$ in the right side of #, and change the value from $2x_i$ to $2x_i + x_{i+1}$.

Step 5: Move back to the current leftmost bit and repeat Step 1 until all bits in $\boldsymbol{x}$ have been read.

Note that your TM should satisfy

- $\Sigma = \{\#, 0, 1\}$,
- $\Gamma = \{\#, 0, 1, 2, 3, \sqcup\}$,
- No more than 9 states (the rejected states are excluded), and
- We only consider moving the head right or left in the Turing machine.

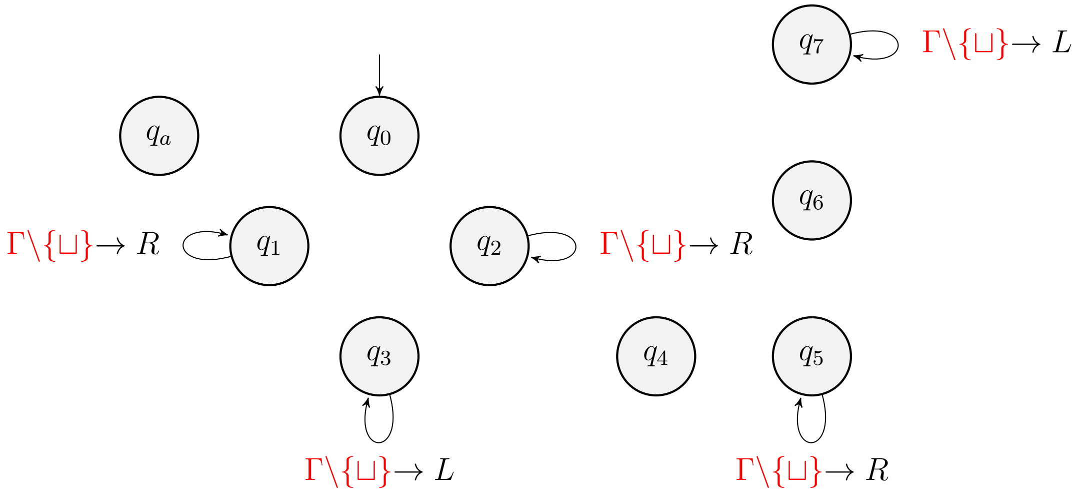
For the following links between two nodes

$$0 \to R, 1 \to R, 2 \to R, 3 \to R, \# \to R,$$

you are allowed to use

$$\Gamma \backslash \{\sqcup\} \to R$$

instead. We require you to complete the following diagram



All you need to do is add links to the states above. We allow $\boldsymbol{x} = \varepsilon$, so

$$\#$$

is accepted. Please draw the resultant diagram on the answer sheet and do not submit this page as your answer.

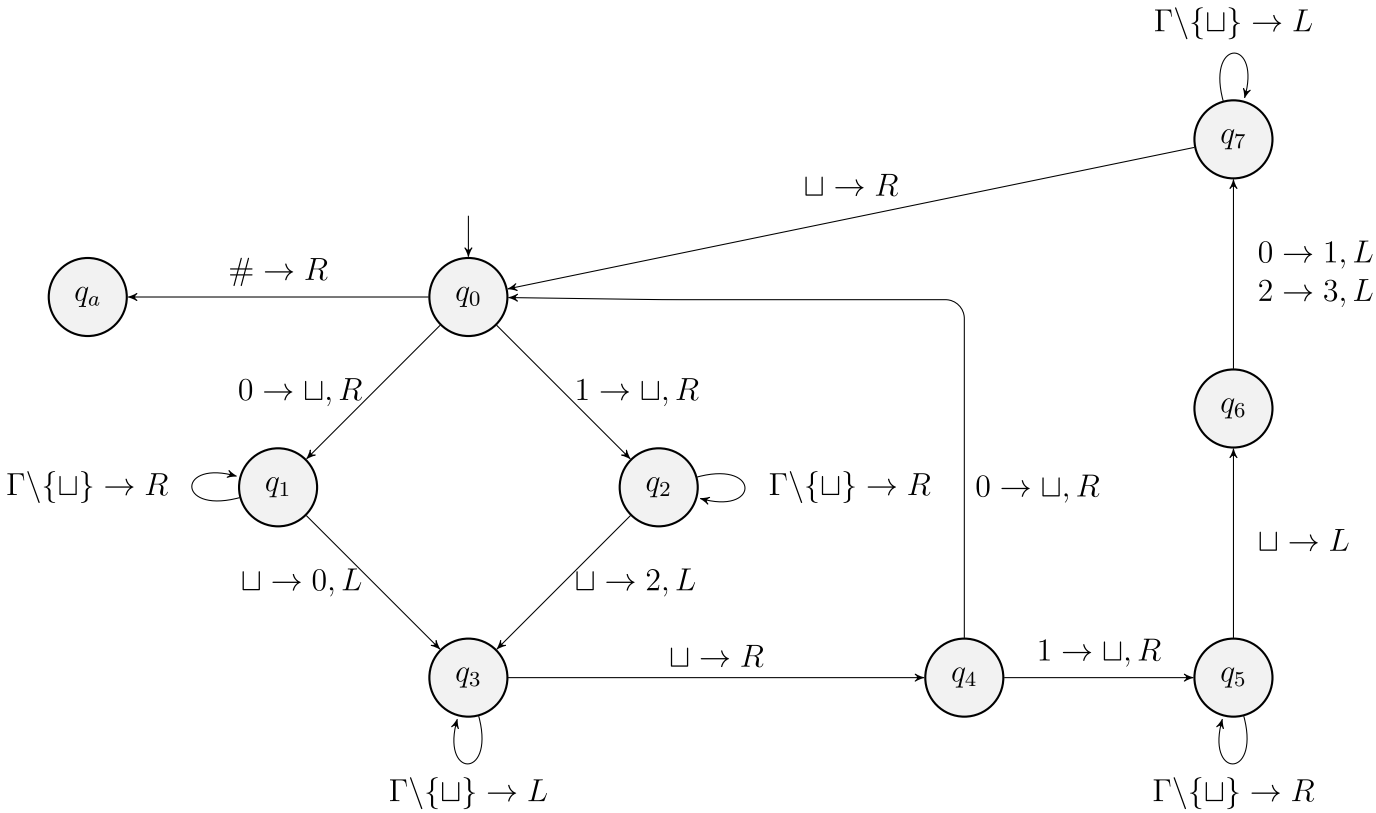
(b) (5 pts) Please simulate your TM in (a) on the string

$$11\#$$

Note that you need to show the entire simulation until your Turing Machine stops.

Solution.

(a) Please see the following diagram:



The meaning of each states:

$\{q_0\}$: Step 1.

$\{q_1, q_2\}$: Step 2.

$\{q_3\}$: Step 3. Move back to x_{i+1}.

$\{q_4\}$: Step 3. Process the bit x_{i+1} and replace it with $\sqcup$.

$\{q_5, q_6\}$: Step 4.

$\{q_7\}$: Step 5.

(b) For the string "11#", the simulation of the Turing machine is on the following.

$$
\begin{array}{lllllllllll}
 & q_0 11\# \sqcup \sqcup & \Rightarrow & \sqcup q_2 1\# \sqcup \sqcup & \Rightarrow & \sqcup 1 q_2 \# \sqcup \sqcup & \Rightarrow & \sqcup 1\# q_2 \sqcup \sqcup & \Rightarrow & \sqcup 1 q_3 \# 2 \sqcup \\
\Rightarrow & \sqcup q_3 1\# 2 \sqcup & \Rightarrow & q_3 \sqcup 1\# 2 \sqcup & \Rightarrow & \sqcup q_4 1\# 2 \sqcup & \Rightarrow & \sqcup \sqcup q_5 \# 2 \sqcup & \Rightarrow & \sqcup \sqcup \# q_5 2 \sqcup \\
\Rightarrow & \sqcup \sqcup \# 2 q_5 \sqcup & \Rightarrow & \sqcup \sqcup \# q_6 2 \sqcup & \Rightarrow & \sqcup \sqcup q_7 \# 3 \sqcup & \Rightarrow & \sqcup q_7 \sqcup \# 3 \sqcup & \Rightarrow & \sqcup \sqcup q_0 \# 3 \sqcup \\
\Rightarrow & \sqcup \sqcup \# q_a 3 \sqcup & & & & & & & &
\end{array}
$$