Introduction to the Theory of Computation 2023 — Midterm 2

Solutions

Problem 1 (35 pts). Consider the context-free grammar (V, Σ, R, S) with

$$V = \{S, B, D, X, Y\} \text{ and } \Sigma = \{a, b, c\},$$

where the rule set R contains the following rules:

$$\begin{aligned} S &\to XD \mid Y \\ X &\to aXb \mid ab \\ D &\to cD \mid \varepsilon \\ Y &\to aBc \mid aYc \\ B &\to bB \mid \varepsilon \end{aligned} \tag{1}$$

(a) (5 pts) We would like to find the language of this grammar. To begin, please respectively give the languages that the variables

$$B, D \text{ and } X$$

can generate.

(b) (5 pts) By the results of (a), please respectively give the languages that the variables

$$Y \text{ and } S$$

can generate. Note that you should write your answer in details.

(c) (5 pts) Is the string

$$a^2b^2c^2$$

ambiguous under the context-free grammar (1)?

(d) (10 pts) Please convert the grammar (1) to CNF by the procedure in Theorem 2.9 of the textbook (in our slides chap2_CNF2.pdf). In removing ? $\to \varepsilon$ rules, consider the order as

$$D, B.$$

In removing rules of a single variable on the right, consider the order as

$$S, X, Y.$$

(e) (10 pts) Let us consider a simplified context-free grammar $(\{S, B, Y\}, \Sigma, \tilde{R}, S)$, where $\tilde{R}$ contains the following rules.

$$\begin{aligned} S &\to Y \\ Y &\to aBc \mid aYc \\ B &\to bB \mid \varepsilon \end{aligned} \tag{2}$$

Please convert the grammar (2) to PDA by the procedure in Lemma 2.21 of the textbook (in our slides chap2_PDA3.pdf.) Hint: the converted PDA contains 9 nodes.

Solution.

(a) The variables B and D can respectively generate $L_B = \{b^j \mid j \geq 0\}$ and $L_D = \{c^k \mid k \geq 0\}$, and X generates $L_X = \{a^i b^j \mid i = j \geq 1\}$.

(b) Since $Y \to aBc \mid aYc$, we have the cases

(i) $Y \to aBc$. This rule generates $\{ab^jc \mid j \geq 0\}$.

(ii) $Y \to aYc$. This rule generates $\{a^i b^j c^k \mid i = k \geq 2, j \geq 0\}$.

Thus, after combining the cases (bi) and (bii), we imply that the rule $Y \to aBc \mid aYc$ generates

$$L_Y = \{a^i b^j c^k \mid i = k \geq 1, j \geq 0\}. \tag{3}$$

For the rule $S = XD \mid Y$, we have the cases

(1) $S \to XD$. Since $L_X = \{a^i b^j \mid i = j \geq 1\}$ and $L_D = \{c^k \mid k \geq 0\}$, the language of the concatenation XD is $\{a^i b^j c^k \mid i = j \geq 1, k \geq 0\}$.

(2) $S \to Y$. We have done the language L_Y in (3).

By combining the cases (b1) and (b2), we conclude that S can generate

$$\{a^i b^j c^k \mid (i = j \geq 1, k \geq 0) \text{ or } (i = k \geq 1, j \geq 0)\}.$$

Common mistake: details are missing.

(c) Here is the leftmost derivation.

$$\begin{aligned} S &\Rightarrow XD \\ &\Rightarrow aXbD \\ &\Rightarrow a^2b^2D \\ &\Rightarrow a^2b^2cD \\ &\Rightarrow a^2b^2c^2D \\ &\Rightarrow a^2b^2c^2 \end{aligned}$$

However, there exists another leftmost derivation.

$$\begin{aligned} S &\Rightarrow Y \\ &\Rightarrow aYc \\ &\Rightarrow a^2Bc^2 \\ &\Rightarrow a^2bBc^2 \\ &\Rightarrow a^2b^2Bc^2 \\ &\Rightarrow a^2b^2c^2 \end{aligned}$$

Therefore, the string

$$a^2b^2c^2$$

is ambiguous under the context-free grammar (1).

(d) We follow the procedure: i) add a new start variable, ii) eliminate all ε-rules, iii) eliminate all unit rules of the form $A \to B$, iv) convert the remaining rules into the CNF.

- Add $S_0 \to S$.

$$\begin{aligned} S_0 &\to S \\ S &\to XD \mid Y \\ X &\to aXb \mid ab \\ D &\to cD \mid \varepsilon \\ Y &\to aBc \mid aYc \\ B &\to bB \mid \varepsilon \end{aligned}$$

- Remove $D \to \varepsilon$.

$$\begin{aligned} S_0 &\to S \\ S &\to XD \mid X \mid Y \\ X &\to aXb \mid ab \\ D &\to cD \mid c \\ Y &\to aBc \mid aYc \\ B &\to bB \mid \varepsilon \end{aligned}$$

- Remove $B \to \varepsilon$.

$$\begin{aligned} S_0 &\to S \\ S &\to XD \mid X \mid Y \\ X &\to aXb \mid ab \\ D &\to cD \mid c \\ Y &\to aBc \mid ac \mid aYc \\ B &\to bB \mid b \end{aligned}$$

- Remove $S_0 \to S$.

$$\begin{aligned} S_0 &\to XD \mid X \mid Y \\ X &\to aXb \mid ab \\ D &\to cD \mid c \\ Y &\to aBc \mid ac \mid aYc \\ B &\to bB \mid b \end{aligned}$$

- Remove $S_0 \to X$.

$$\begin{aligned} S_0 &\to XD \mid aXb \mid ab \mid Y \\ X &\to aXb \mid ab \\ D &\to cD \mid c \\ Y &\to aBc \mid ac \mid aYc \\ B &\to bB \mid b \end{aligned}$$

- Remove $S_0 \to Y$.

$$\begin{aligned}
S_0 &\to XD \mid aXb \mid ab \mid aBc \mid ac \mid aYc \\
X &\to aXb \mid ab \\
D &\to cD \mid c \\
Y &\to aBc \mid ac \mid aYc \\
B &\to bB \mid b
\end{aligned}$$

- Add $V_1 \to Xb$.

$$\begin{aligned}
S_0 &\to XD \mid aV_1 \mid ab \mid aBc \mid ac \mid aYc \\
X &\to aV_1 \mid ab \\
D &\to cD \mid c \\
Y &\to aBc \mid ac \mid aYc \\
B &\to bB \mid b \\
V_1 &\to Xb
\end{aligned}$$

- Add $V_2 \to Bc$.

$$\begin{aligned}
S_0 &\to XD \mid aV_1 \mid ab \mid aV_2 \mid ac \mid aYc \\
X &\to aV_1 \mid ab \\
D &\to cD \mid c \\
Y &\to aV_2 \mid ac \mid aYc \\
B &\to bB \mid b \\
V_1 &\to Xb \\
V_2 &\to Bc
\end{aligned}$$

- Add $V_3 \to Yc$.

$$\begin{aligned}
S_0 &\to XD \mid aV_1 \mid ab \mid aV_2 \mid ac \mid aV_3 \\
X &\to aV_1 \mid ab \\
D &\to cD \mid c \\
Y &\to aV_2 \mid ac \mid aV_3 \\
B &\to bB \mid b \\
V_1 &\to Xb \\
V_2 &\to Bc \\
V_3 &\to Yc
\end{aligned}$$

- Replace any terminals in the preceding rules with the new variables.

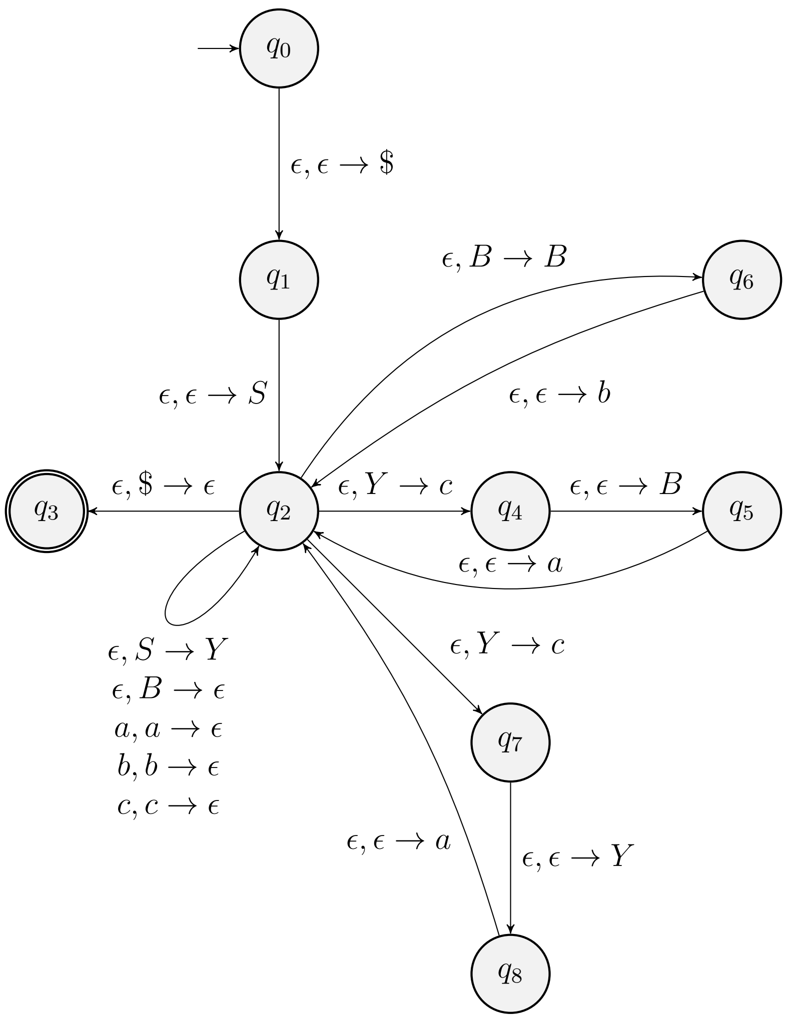
$$
\begin{aligned}
S_0 &\to XD \mid U_1V_1 \mid U_1U_2 \mid U_1V_2 \mid U_1U_3 \mid U_1V_3 \\
X &\to U_1V_1 \mid U_1U_2 \\
D &\to U_3D \mid c \\
Y &\to U_1V_2 \mid U_1U_3 \mid U_1V_3 \\
B &\to U_2B \mid b \\
V_1 &\to XU_2 \\
V_2 &\to BU_3 \\
V_3 &\to YU_3 \\
U_1 &\to a \\
U_2 &\to b \\
U_3 &\to c
\end{aligned}
$$

Finally, we have the CNF of the grammar (1).

Common mistakes:

- Maintain epsilon rules incorrectly.
- Miss on adding a new start variable.
- Incorrectly follow the procedure of textbook when convert the remaining rules into the proper form.

(e) Please see the following diagram.

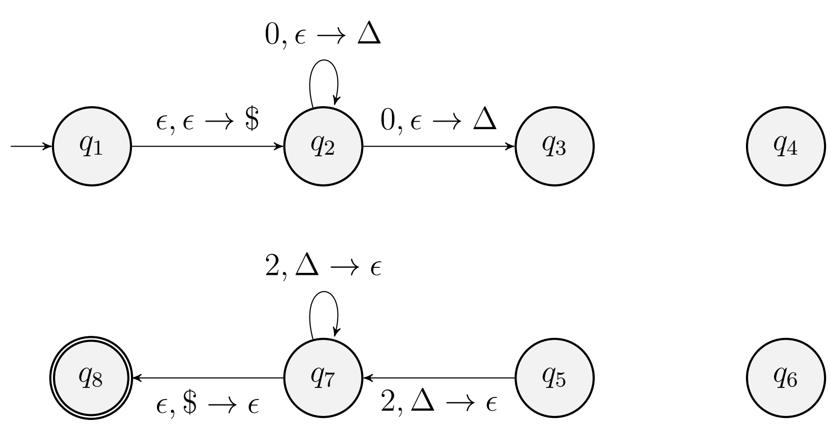


Problem 2 (15 pts). Let $\Sigma = \{0, 1, 2\}$. Consider the language

$$L_2 = \{0^a 1^b 2^c \mid a, b, c \text{ are positive integers and form an arithmetic sequence. }\}$$

The sequence $\{a, b, c\}$ is an arithmetic sequence if $b - a = c - b$. For example, both "011222" and "000112" are in L_2, but "00122" is not in L_2.

(a) (10 pts) Finish the following PDA to recognize L_2 without adding new states. The stack alphabet Γ is restricted to be $\{\$, \Delta\}$.



From q_1 to q_3 we push a Δ upon receiving an input character "0". From q_5 to q_8 we pop a Δ upon receiving an input character "2" then accept the string if the stack is empty in the long run. **You must explain your idea of the added links.**

Hint: because

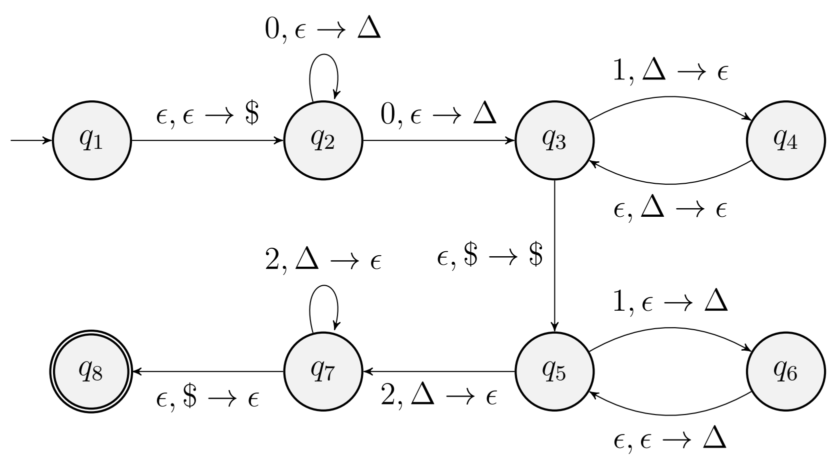
$$a + c = 2b,$$

try to think how we cancel a 0's with certain 1's. Note that if a is odd, then c must also be odd. On the other hand, if a is even, then c must also be even.

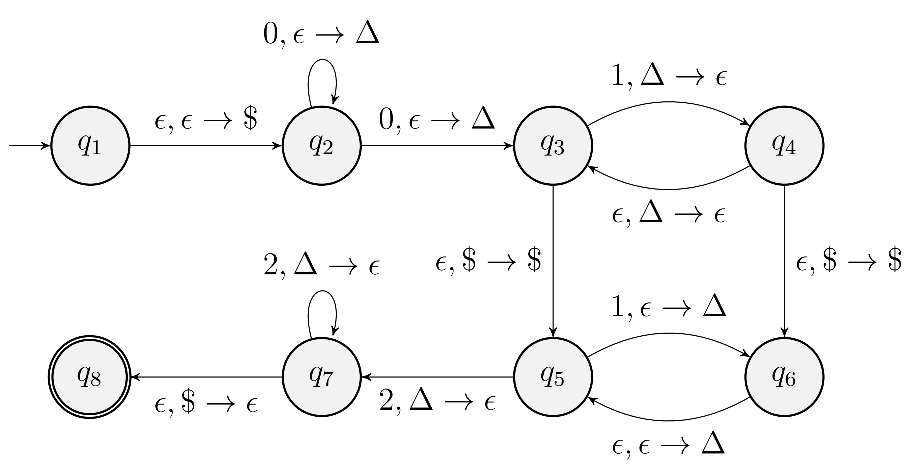
(b) (5 pts) Following page 5 of the slide "chap2_PDA2.pdf" or just like what we did in the previous exam, please simulate your PDA in (a) on the two strings "012" and "00112" by drawing trees. Then determine whether the PDA accepts the two strings according to your simulation.

Solution.

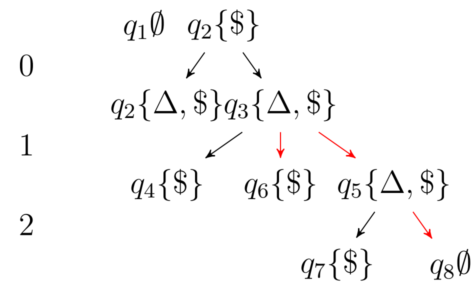
(a) Let's begin with the situation with an even a. We use a loop between q_3 and q_4 so that $a/2$ 1's are processed and a Δ's are popped. Now the stack is empty, so we use a loop between q_5 and q_6 to process the rest of 1^b, which has length $c/2$ and puch c Δ's to the stack. Thus the diagram is now as follows



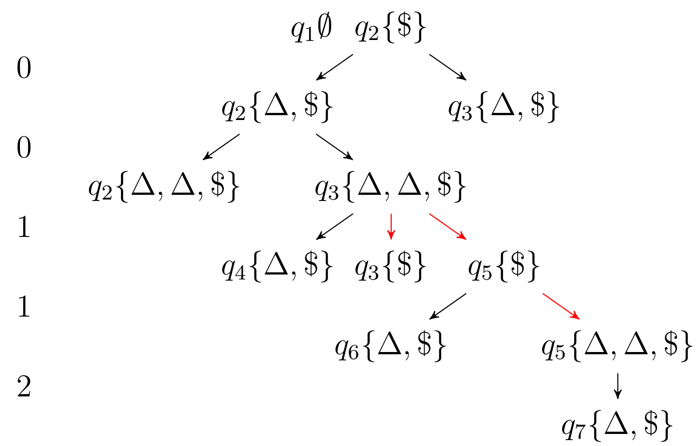
We then must handle the situation if a is odd. We find that when all a Δ's have been popped up, we are at q_4 and $(a + 1)/2$ 1's have been processed. Then we go to q_6 and the loop between q_5 and q_6 processes $(c - 1)/2$ 1's and brings c Δ's to stack.



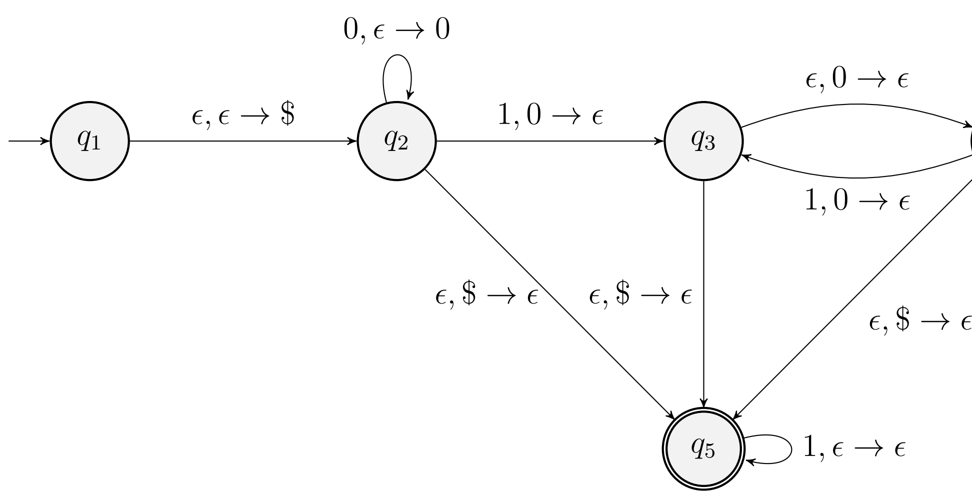
(b) For input “012”:



For input “00112”:



Problem 3 (35 pts). Consider the following PDA P with $\Sigma = \{0, 1\}$ and $\Gamma = \{0, \$\}$.



(a) (5 pts) What is the language recognized by P?

(b) (5 pts) Can you add a link from q_5 to q_4 (and may remove the link of $q_5 \to q_5$) for satisfying the following conditions?

- Single accept state
- Stack empty before accepting

- Each transition pushes or pops something from the stack, but not both

Note: adding a link is sufficient. You are not allowed to change neither Σ nor Γ.

(c) (10 pts) Convert your new diagram in (b) 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.

(d) (5 pts) Explain why our given PDA P is not a DPDA. If you don't know where to start, try to see if you can complete the δ table in (e). Note that we consider the PDA P but not the modification in (b).

(e) (10 pts) Now we aim to construct a DPDA for the same language. From issues you pointed out in (d), can you make some minor adjustment of the PDA P (without changing Σ, Γ and Q) to have a DPDA? To show that your resulting diagram is a DPDA, please complete the following formal definition. In particular, you need to finish the δ table and the F set.

- $Q = \{q_1, q_2, q_3, q_4, q_5, q_r\}$ is the set of states.
- $\Sigma = \{0, 1\}$ is the input alphabet.
- $\Gamma = \{0, \$\}$ is the stack alphabet.
- δ is the transition function:

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

- q_1 is the start state.
- $F = \{ \qquad\qquad\qquad \}$ is the set of accept states.

Solution.

(a) The PDA accepts strings of the form 0^m1^n for certain condition of m and n. To figure out the condition, we consider m for the following cases.

Case 1: For $m = 0$, the string is accepted for any n via

$$q_1 \to q_2 \to q_5 \underbrace{\to \cdots \to q_5}_{\text{all other input 1's}} .$$

Case 2: For an even m, $m > 0$, the string would be accepted if $n \geq m/2$ via

$$q_1 \to q_2 \to \underbrace{\cdots \to q_2}_{m \text{ input 0's}} \underbrace{\to q_3 \to q_4 \cdots \to q_3 \to q_4}_{(m/2) \text{ input 1's}} \to q_5 \underbrace{\to \cdots \to q_5}_{\text{all other input 1's}} .$$

Case 3: For an odd m, the string would be accepted if $n \geq (m+1)/2$ via

$$q_1 \to q_2 \to \underbrace{\cdots \to q_2}_{m \text{ input 0's}} \underbrace{\to q_3 \to q_4 \cdots \to q_3 \to q_4}_{(m-1)/2 \text{ input 1's}} \underbrace{\to q_3}_{1 \text{ input 1's}} \to q_5 \underbrace{\to \cdots \to q_5}_{\text{all other input 1's}}.$$

Up to this point, we know that $L(P)$ has

$$0^m 1^{m/2} 1^k, \forall m \text{ is even}, m \geq 0, k \geq 0$$

and

$$0^m 1^{(m+1)/2} 1^k, \forall m \text{ is odd}, k \geq 0.$$

This can be simplified to

$$0^m 1^n, \forall m \text{ is even}, m \geq 0, n \geq m/2$$

and

$$0^m 1^n, \forall m \text{ is odd}, n \geq (m+1)/2.$$

When m is odd, the set of n satisfying $n \geq (m+1)/2$ is

$$\left\{ \frac{m+1}{2}, \frac{m+1}{2} + 1, \frac{m+1}{2} + 2, \ldots \right\},$$

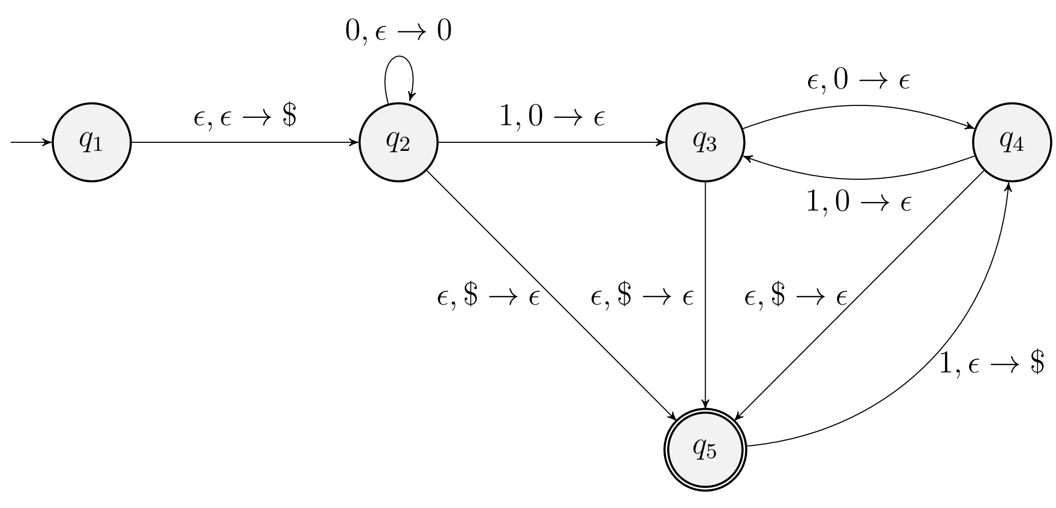
and the set of n satisfying $n \geq m/2$ is also

$$\left\{ \frac{m+1}{2}, \frac{m+1}{2} + 1, \frac{m+1}{2} + 2, \ldots \right\}.$$

Therefore, the representation of strings in $L(P)$ can be simplified to

$$0^m 1^n, n \geq m/2, n \geq 0, m \geq 0.$$

(b) The diagram is like the following.



The original PDA has satisfied the first two conditions. Thus, we only have to handle the $q_5 \to q_5$ link. This can be done by pushing and popping a "\$" upon reading a "1" at q_5.

(c) The corresponding CFG is the following:

- $u = 0$:

p	r	s	q	a	b	rules
2	2	2	3	0	1	$A_{23} \to 0A_{22}1$
2	2	3	4	0	ϵ	$A_{24} \to 0A_{23}$
2	2	4	3	0	1	$A_{23} \to 0A_{24}1$

- $u = \$$:

p	r	s	q	a	b	rules
1	2	2	5	ϵ	ϵ	$A_{15} \to A_{22}$
1	2	3	5	ϵ	ϵ	$A_{15} \to A_{23}$
1	2	4	5	ϵ	ϵ	$A_{15} \to A_{24}$
5	4	2	5	1	ϵ	$A_{55} \to 1A_{42}$
5	4	3	5	1	ϵ	$A_{55} \to 1A_{43}$
5	4	4	5	1	ϵ	$A_{55} \to 1A_{44}$

Common mistake: the rules $A_{55} \to 1A_{42}$ and $A_{55} \to 1A_{43}$ are missed.

(d) At q_2 we see that $\delta(q_2, 0, \epsilon) = \{q_2\}$ and $\delta(q_2, \epsilon, \$) = \{q_5\}$, violating the rule

$$\text{only one of } \delta(q_2, 0, \$),\ \delta(q_2, 0, \epsilon),\ \delta(q_2, \epsilon, \$) \text{ and } \delta(q_2, \epsilon, \epsilon) \text{ is not } \emptyset.$$

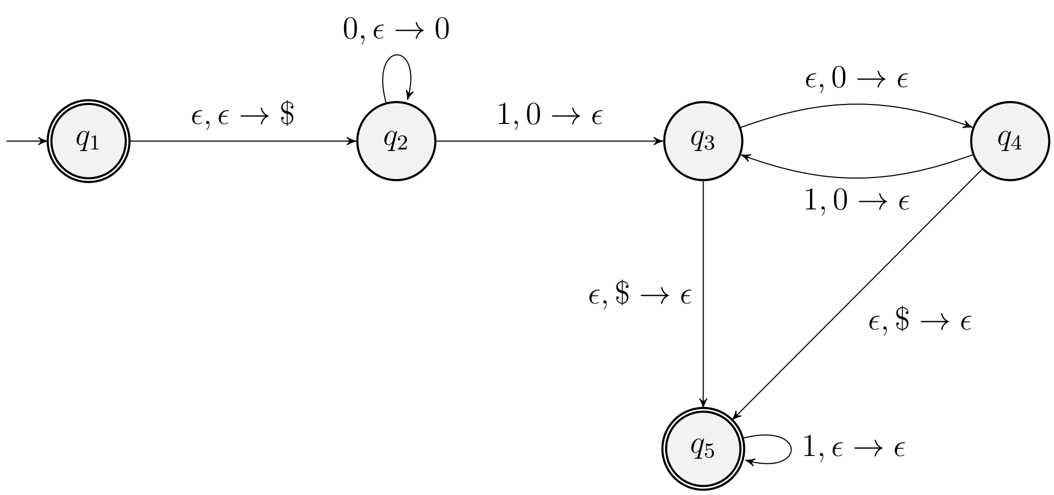
Thus, the PDA is not a DPDA.

Common mistake: some have that

$$\delta(q_4, 0, 0) = \delta(q_4, \epsilon, 0) = \delta(q_4, 0, \epsilon) = \delta(q_4, \epsilon, \epsilon) = \emptyset.$$

But if you check examples in our slides, we can do $\delta(q_4, 0, 0) = q_r$ and we didn't say the example is not a DPDA. Here we have a situation that if the diagram is not changed, then we do not have a DPDA.

(e) The $q_2 \to q_5$ link is only used to accept the empty string. We can modify q_1 to be an accept state to remove this link and result in the following diagram.



Thus, the transition function δ is the following:

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

And the set of accept set $F = \{q_1, q_5\}$.

Problem 4 (15 pts). Xiao-Ming is a train conductor. After the train arrived at the terminal station and was not in service, he needs to check whether there is any passenger in the train. Let us use the knowledge of this course to help Xiao-Ming.

When the train is in service, there are passengers getting on and off the train. Let us consider these types of passengers: adults and children, and further denote an adult and a child getting on(off) the train as $A_+(A_-)$ and $C_+(C_-)$, respectively. While the train is not in service, we use the strings with $\{A_+, A_-, C_+, C_-\}$ to represent for the passing situations, which should satisfy the following rule.

- A_+ and C_+ must be happened before A_- and C_-, respectively.

(a) (10 pts) We would like to design a Turing machine that accepts all of the strings with the situation

"there is no person in the train."

For example,

$$A_+C_+C_-A_-$$

is accepted, but

$$A_+A_+C_+C_-A_- \text{ and } A_-A_+$$

are rejected. We also accept $\sqcup$. Our idea is that

$$\begin{aligned}&\text{eliminating the first observed } A_+(C_+) \text{ with the first observed } A_-(C_-) \text{ in the tape,}\\ &\text{and we use } \times \text{ to denote the eliminated } A_+(C_+) \text{ and } A_-(C_-).\end{aligned} \tag{4}$$

Then, we move the head to the first location of the tape. To achieve this, we use a special symbol F to denote the eliminated A_+ or C_+ in the first location of the tape. After that, recursively execute the procedure (4) until none of alphabet in the string can be modified to $\times$. In the final, we judge the string should be accepted or rejected.

Therefore, we concludes the aforementioned idea to the following steps.

Step 1: When reading $A_+(C_+)$, modify it to

$$\begin{cases} F, & \text{as } A_+(C_+) \text{ is the first string in the tape,} \\ \times, & \text{otherwise.} \end{cases}$$

Step 2: Moving right, until the machine reads $A_-(C_-)$.

Step 3: Modify $A_-(C_-)$ to $\times$ and move left.

Step 4: Move left as reading A_+, C_+, A_-, C_-, $\times$, until the machine reads F then move right.

Step 5: Move right when reading $\times$. If the machine reads $A_+ (C_+)$, go to Step 1. If the machine reads $\sqcup$, go right and accept the string.

Please follow these steps to draw a Turing machine with

$$\Sigma = \{A_+, A_-, C_+, C_-\} \text{ and } \Gamma = \{A_+, A_-, C_+, C_-, \times, F\},$$

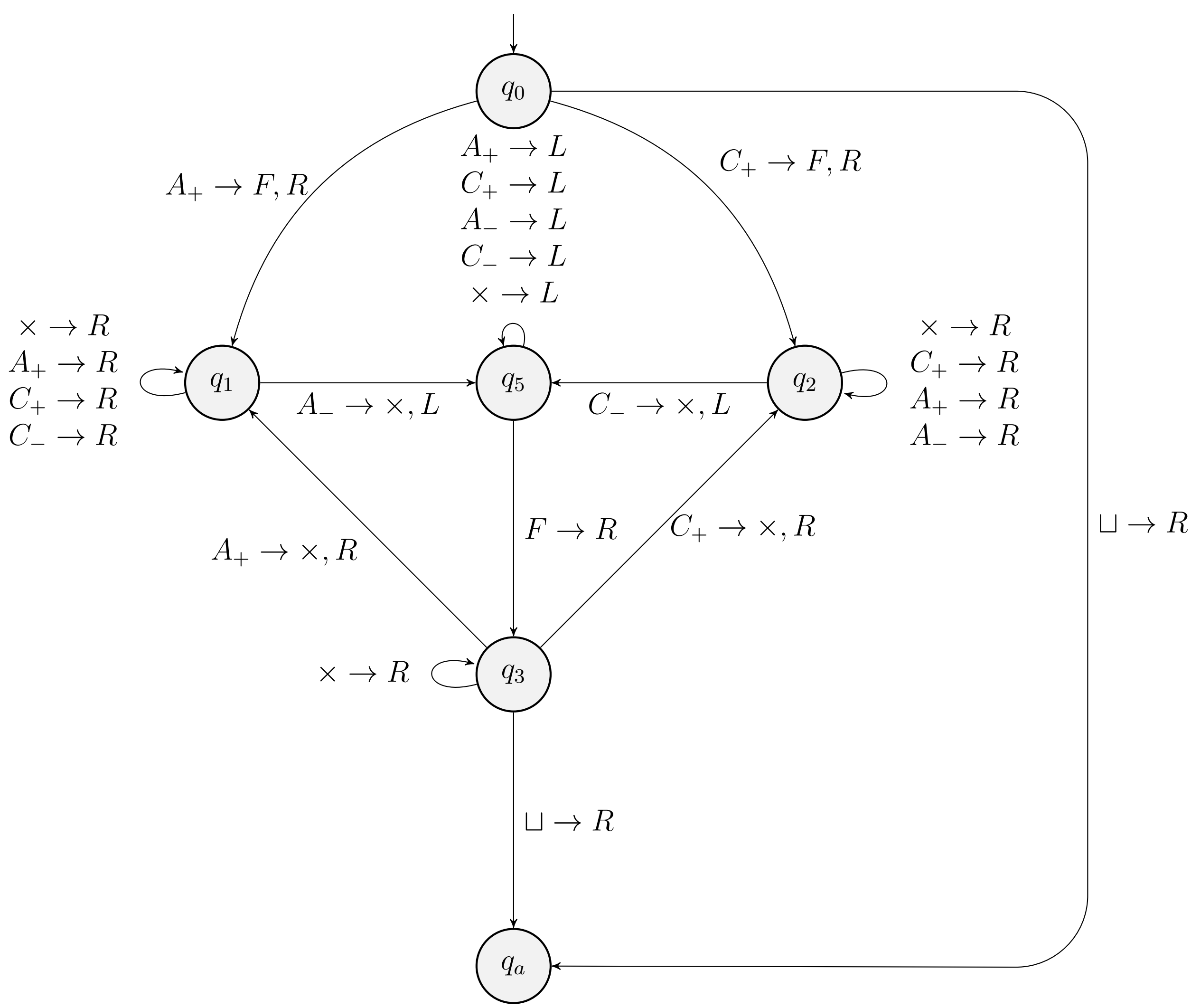
which must be no more than 6 states (the reject state excluded). Note that we only consider moving the head right or left in the Turing machine, and the links to the reject state q_r is not needed.

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

$$A_+A_+A_-A_-.$$

Solution.

(a) Please see the following diagram.



Common mistakes:

- Draw a double circle state.

- Reject empty string.
- Accept strings such as "A_+C_-".
- The solution does not follow our precedure.
- Draw some non-deterministic links.
- Use over 6 states.

(b) For the string $A_+A_+A_-A_-$, the simulation of the Turing machine is on the following.

$$\begin{array}{lllllll} & q_0A_+A_+A_-A_- & \Rightarrow & Fq_1A_+A_-A_- & \Rightarrow & FA_+q_1A_-A_- & \Rightarrow & Fq_5A_+\times A_- \\ \Rightarrow & q_5FA_+\times A_- & \Rightarrow & Fq_3A_+\times A_- & \Rightarrow & F\times q_1\times A_- & \Rightarrow & F\times\times q_1A_- \\ \Rightarrow & F\times q_5\times\times & \Rightarrow & Fq_5\times\times\times & \Rightarrow & q_5F\times\times\times & \Rightarrow & Fq_3\times\times\times \\ \Rightarrow & F\times q_3\times\times & \Rightarrow & F\times\times q_3\times & \Rightarrow & F\times\times\times q_3 & \Rightarrow & F\times\times\times\sqcup q_a \end{array}$$