Midterm 2

Midterm Review

General Info

Date and Time : Mar 21 , 12:00 pm – 1:00 pm
Topics: Chapter 2 and Chapter 3 (of the text)

The test will be Closed book and calculator is not allowed.

The students who registered for the **accommodation**, please send me an email and remind me (2 days before the test.)

DESIGNING CONTEXT-FREE GRAMMARS

Method 1: Based on this Fact: Many CFLs are the union of simpler CFLs.

- break a CFL into simpler pieces
- and then construct individual grammars for each piece
- merge the individual grammars into a grammar for the original language
  - By combining their rules and then adding the new rule

$$S \to S_1 | S_2 | \cdots | S_k,$$

Where the variables S_i are the start variables for the individual grammars.

DESIGNING CONTEXT-FREE GRAMMARS

- Example: Construct a grammar for the language

$$\{0^n1^n | n \geq 0\} \cup \{1^n0^n | n \geq 0\}.$$

- $L_1 = \{0^n1^n | n \geq 0\}$ Rules: $S_1 \rightarrow 0S_11 | \varepsilon$
- $L_2 = \{1^n0^n | n \geq 0\}$ Rules: $S_2 \rightarrow 1S_20 | \varepsilon$

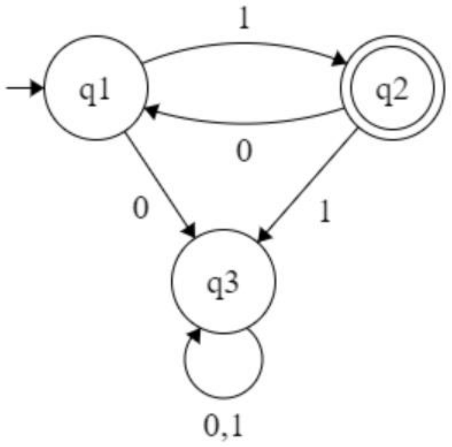
- $L_1 \cup L_2$ Rules:

$$S \rightarrow S1 | S2$$
$$S_1 \rightarrow 0S_11 | \varepsilon$$
$$S_2 \rightarrow 1S_20 | \varepsilon$$

DESIGNING CONTEXT-FREE GRAMMARS

Method 2: constructing a CFG for a regular language :

- construct a DFA for that language.
- convert the DFA into an equivalent CFG as follows.
  - Make a variable R_i for each state q_i of the DFA.
  - Add the rule $R_i \rightarrow aR_j$ to the CFG if $\delta(q_i,a)=q_j$.
  - Add the rule $R_i \rightarrow \varepsilon$ if q_i is an accept state of the DFA.
  - Make R_0 the start variable of the grammar, where q_0 is the start state of the machine.
  - the resulting CFG generates the same language that the DFA recognizes. (Verify on your own that)



- $V = \{R1, R2, R3\}$
- $\boldsymbol{R_1} \rightarrow 0R_3 \mid 1R_2 \quad R_2 \rightarrow 0R_1 \mid 1R_3 \quad R_3 \rightarrow 1R_3 \mid 1R_3$
- $R_2 \rightarrow \varepsilon$

DESIGNING CONTEXT-FREE GRAMMARS

Method 3: constructing a CFG for a Certain context-free languages :

- strings with two substrings that are “linked”
    - need to remember an unbounded amount of information about one of the substrings
        - to verify that it corresponds properly to the other substring.
    - **Example**: $\{0^n1^n | n \geq 0\}$

- **construct a CFG:** by using a rule of the form $R \rightarrow uRv$

DESIGNING CONTEXT-FREE GRAMMARS

Method 4: constructing a CFG for a language with a recursive structure:

- **Example**:

- Consider grammar $G_4 = (V, \Sigma, R, \langle EXPR \rangle)$
- $V = \{\langle EXPR \rangle, \langle TERM \rangle, \langle FACTOR \rangle\}$
- $\Sigma = \{a, +, \times, (,)\}$.
- The **rules** are
  - $\langle \boldsymbol{EXPR} \rangle \rightarrow \langle EXPR \rangle + \langle TERM \rangle | \langle TERM \rangle$
  - $\langle \boldsymbol{TERM} \rangle \rightarrow \langle TERM \rangle \times \langle FACTOR \rangle | \langle FACTOR \rangle$
  - $\langle \boldsymbol{FACTOR} \rangle \rightarrow (\langle EXPR \rangle) | a$

Some useful context-free grammars

- $S \rightarrow \varepsilon$ $\quad$ $\mathrm{L} = \{\}$
- $S \rightarrow a$ $\quad$ $\mathrm{L} = \{a\}$
- $S \rightarrow aS|\varepsilon$ $\quad$ $\mathrm{L} = \{a, aa, aaa, \dots\} = \{a^n | \, n > 0\}$
- $S \rightarrow Sa|\varepsilon$ $\quad$ $\mathrm{L} = \{a, aa, aaa, \dots\} = \{a^n | \, n > 0\}$
- $S \rightarrow aSa|\varepsilon$ $\quad$ $\mathrm{L} = \{aa, aaaa, aaaaaa, \dots\} = \{a^{2n} | \, n > 0\}$
- $S \rightarrow aSa|b$ $\quad$ $\mathrm{L} = \{aba, aabaa, aaabaaa, \dots\}$ $= \{a^n b a^n | \, n > 0\}$
- $S \rightarrow aaS|\varepsilon$ $\quad$ $\mathrm{L} = \{aa, aaaa, aaaaaa, \dots\} = \{a^{2n} | \, n > 0\}$
- $S \rightarrow aaS|b$ $\quad$ $\mathrm{L} = \{aa, aaaa, aaaaaa, \dots\} = \{a^{2n} b | \, n > 0\}$

Context-free grammar

Some useful context-free grammars

- $A \rightarrow aA|\varepsilon\,, B \rightarrow bB|\varepsilon\,,\ S|AB$ $\quad L = \{a^n b^m |\, n, m \geq 0\}$
- $A \rightarrow aAb|\varepsilon$ $\quad L = \{a^n b^n |\, n, m \geq 0\}$
- $A \rightarrow aaAb|\varepsilon$ $\quad L = \{a^{2n} b^n |\, n \geq 0\}$
- $A \rightarrow aaaAbb|\varepsilon$ $\quad L = \{a^{3n} b^{2n} |\, n \geq 0\}$

Give context-free grammars that generate the following languages.

$$\{\, w \in \{0,1\}^* \mid w \text{ contains at least three 1s} \,\}$$

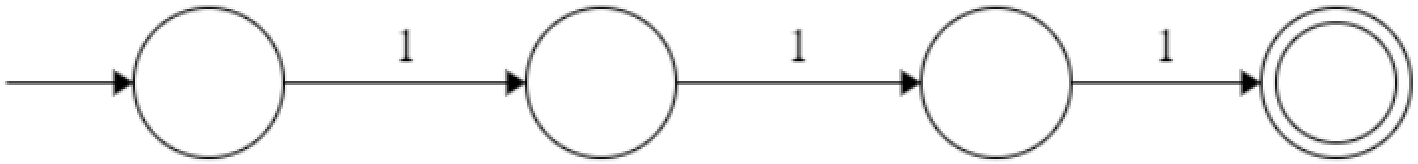
Give context-free grammars that generate the following languages.

$$\{ w \in \{0,1\}^* \mid w \text{ contains at least three 1s} \}$$

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S, X\}$, where S is the start variable; set of terminals $\Sigma = \{0, 1\}$; and rules

$$\begin{aligned} S &\rightarrow X1X1X1X \\ X &\rightarrow 0X \mid 1X \mid \varepsilon \end{aligned}$$

A rule for generating any string



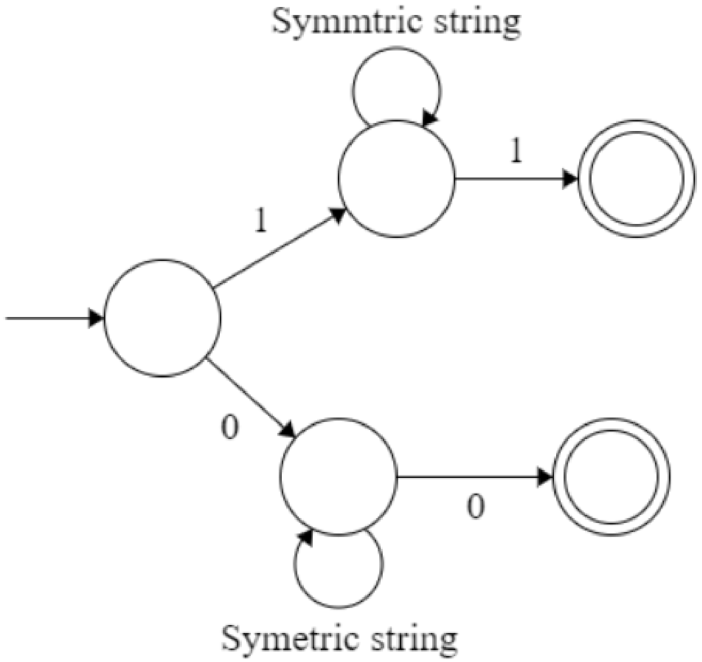
Give context-free grammars that generate the following languages.

$$\{\, w \in \{0,1\}^* \mid w = w^{\mathcal{R}} \text{ and } |w| \text{ is even} \,\}$$

Give context-free grammars that generate the following languages.

$$\{\, w \in \{0,1\}^* \mid w = w^{\mathcal{R}} \text{ and } |w| \text{ is even} \,\}$$

ε
00 , 11
0000,1001, 0110 , 1111



Context-free grammar

Give context-free grammars that generate the following languages.

$$\{ w \in \{0,1\}^* \mid w = w^{\mathcal{R}} \text{ and } |w| \text{ is even} \}$$

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = \{0, 1\}$; and rules

$$S \rightarrow 0S0 \mid 1S1 \mid \varepsilon$$

ε
00 , 11
0000,1001, 0110 , 1111

Context-free grammar

Give context-free grammars that generate the following languages.

$\{ w \in \{0,1\}^* \mid \text{the length of } w \text{ is odd and the middle symbol is } 0 \}$

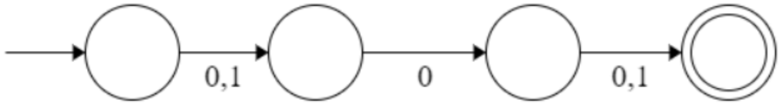
$L = \{0, 000, 001, 100, 101, \dots\}$

L is the generated language by G.

$S \rightarrow 0$

$S \rightarrow 000$
$S \rightarrow 001$
$S \rightarrow 100$
$S \rightarrow 101$

$S \rightarrow 0S0$
$S \rightarrow 0S1$
$S \rightarrow 1S0$
$S \rightarrow 1S1$



Context-free grammar

Give context-free grammars that generate the following languages.

$\{ w \in \{0,1\}^* \mid \text{the length of } w \text{ is odd and the middle symbol is } 0 \}$

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S\}$, where S is the start variable; set of terminals $\Sigma = \{0, 1\}$; and rules

$$S \rightarrow 0S0 \mid 0S1 \mid 1S0 \mid 1S1 \mid 0$$

$S \rightarrow 0$ ➡

$S \rightarrow 000$
$S \rightarrow 001$
$S \rightarrow 100$
$S \rightarrow 101$ ➡

$S \rightarrow 0S0$
$S \rightarrow 0S1$
$S \rightarrow 1S0$
$S \rightarrow 1S1$

Give context-free grammars that generate the following languages.

$$\{ a^i b^j c^k \mid i, j, k \geq 0, \text{ and } i = j \text{ or } i = k \}$$

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

$S_1 \rightarrow LR \mid \varepsilon$
$L \rightarrow aLb \mid \varepsilon$
$R \rightarrow cR \mid \varepsilon$

$S_2 \rightarrow aS_2c \mid M \mid \varepsilon$
$M \rightarrow bM \mid \varepsilon$

$$S \rightarrow S_1 \mid S_2$$

$S \rightarrow S_1 \mid S_2$
$S_1 \rightarrow LR \mid \varepsilon$
$L \rightarrow aLb \mid \varepsilon$
$R \rightarrow cR \mid \varepsilon$
$S_2 \rightarrow aS_2c \mid M \mid \varepsilon$
$M \rightarrow bM \mid \varepsilon$

$S \rightarrow LR \mid S_2$
$L \rightarrow aLb \mid \varepsilon$
$R \rightarrow cR \mid \varepsilon$
$S_2 \rightarrow aS_2c \mid M$
$M \rightarrow bM \mid \varepsilon$

Context-free grammar

Give context-free grammars that generate the following languages.

$\{ a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i + j = k \}$

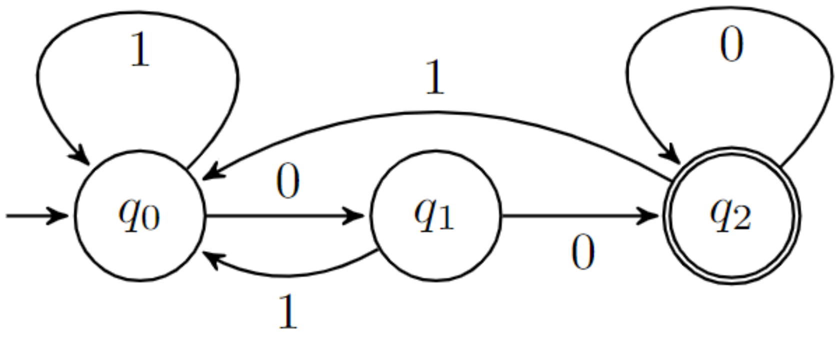
$$\{a^i b^j c^k \mid i, j, k \geq 0 \; and \; i + j = k\}$$
$$= \{a^i b^j c^j c^i \mid i, j, k \geq 0\}$$
$$= \{a^i b^j c^j c^i \mid i, j, k \geq 0\}$$

Answer: $G = (V, \Sigma, R, S)$ with set of variables $V = \{S, X\}$, where S is the start variable; set of terminals $\Sigma = \{a, b, c\}$; and rules

$$\begin{aligned} S &\rightarrow aSc \mid X \\ X &\rightarrow bXc \mid \varepsilon \end{aligned}$$

Context-free grammar

- Construct a context-free grammar for the following DFA:



The language of the DFA is defined by the grammar $G = (V, \Sigma, R, S_0)$ with $V = \{S_0, S_1, S_2\}$, $\Sigma = \{0, 1\}$, and R being the following set of rules:

$$
\begin{aligned}
S_0 &\to 0S_1 \mid 1S_0 \\
S_1 &\to 0S_2 \mid 1S_0 \\
S_2 &\to 0S_2 \mid 1S_0 \mid \epsilon
\end{aligned}
$$

Chomsky Normal Form

Convert the following CFG to CNF.

$S \rightarrow ASA \mid aB$
$A \rightarrow B \mid S$
$B \rightarrow b \mid \varepsilon$

1. add a new start variable

$S_0 \rightarrow S$
$S \rightarrow ASA \mid aB$
$A \rightarrow B \mid S$
$B \rightarrow b \mid \varepsilon$

2. remove the ε-rules
   $B \rightarrow \in$

$S_0 \rightarrow S$
$S \rightarrow ASA \mid aB \mid a$
$A \rightarrow B \mid S \mid \in$
$B \rightarrow b$

2. remove the ε-rules
   $A \rightarrow \in$

$S_0 \rightarrow S$
$S \rightarrow ASA \mid aB \mid a \mid AS \mid SA \mid S$
$A \rightarrow B \mid S$
$B \rightarrow b$

3. remove the unit rules
   $S \rightarrow S$

$S_0 \rightarrow S$
$S \rightarrow ASA \mid aB \mid a \mid AS \mid SA$
$A \rightarrow B \mid S$
$B \rightarrow b$

Convert the following CFG to CNF.

3. remove the unit rules

$S_0 \to S$

$S_0 \to ASA \mid aB \mid a \mid AS \mid SA$
$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to B \mid S$
$B \to b$

3. remove the unit rules

$A \to B$

$S_0 \to ASA \mid aB \mid a \mid AS \mid SA$
$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to S \mid b$
$B \to b$

3. remove the unit rules

$A \to S$

$S_0 \to ASA \mid aB \mid a \mid AS \mid SA$
$S \to ASA \mid aB \mid a \mid AS \mid SA$
$A \to b \mid ASA \mid aB \mid a \mid AS \mid SA$
$B \to b$

4. Not more than two variables

$S_0 \to ASA$, $S \to ASA$, $A \to ASA$

$S_0 \to AX \mid aB \mid a \mid AS \mid SA$
$S \to AX \mid aB \mid a \mid AS \mid SA$
$A \to b \mid AX \mid aB \mid a \mid AS \mid SA$
$B \to b$
$X \to SA$

Convert the following CFG to CNF. (Cont.)

5. change to a proper form

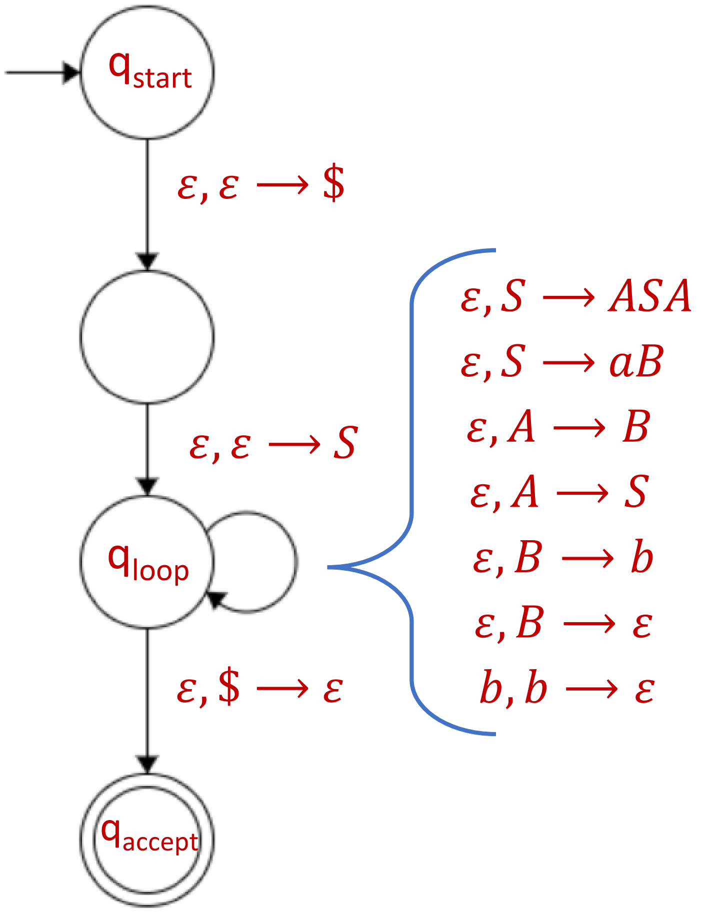
   $S_0 \rightarrow aB$, $S \rightarrow aB$, $A \rightarrow aB$

$S_0 \rightarrow AX \mid YB \mid a \mid AS \mid SA$
$S \rightarrow AX \mid YB \mid a \mid AS \mid SA$
$A \rightarrow b \mid AX \mid YB \mid a \mid AS \mid SA$
$B \rightarrow b$
$X \rightarrow SA$
$Y \rightarrow a$

Push-down automata

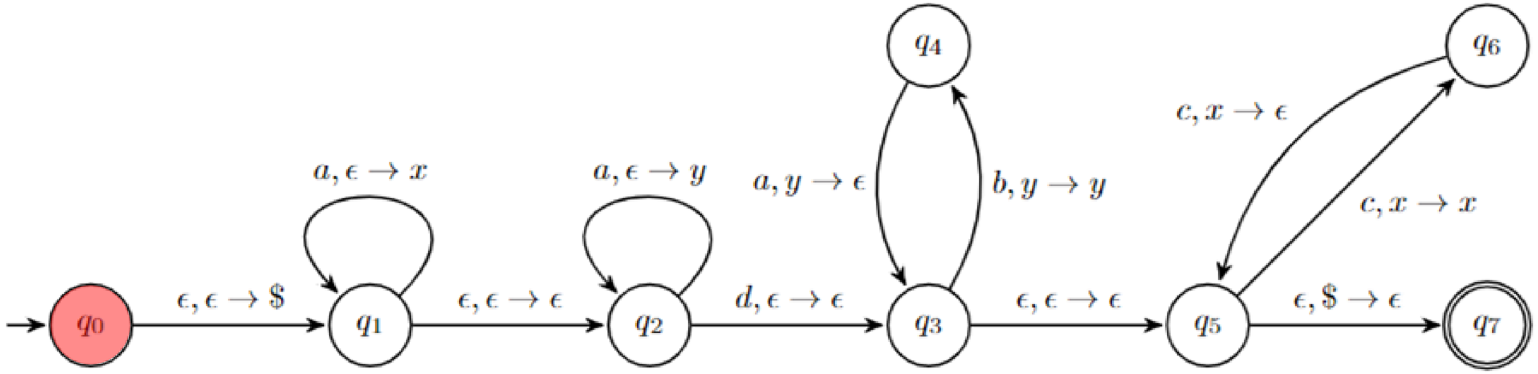
Convert the following CFG to PDA

S → ASA | aB
A → B | S
B → b | ε



Consider the following PDA:

- Show that the PDA accepts the word aaadbabacc.



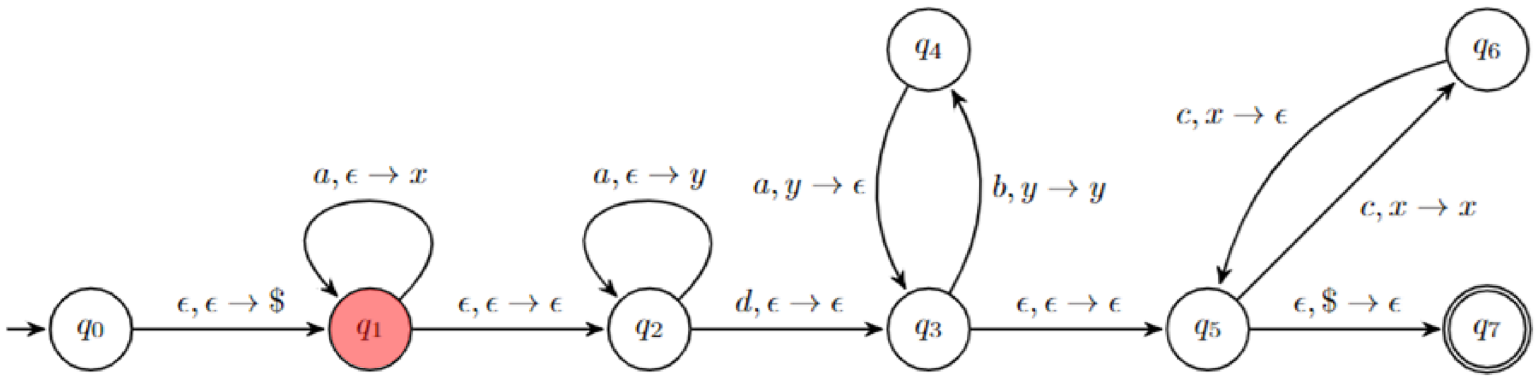
input state stack

$$(aaadbabacc, q_0, \epsilon) \to$$
$$\to (aaadbabacc, q_1, \$) \to$$
$$\to (aadbabacc, q_1, x\$) \to$$
$$\to (aadbabacc, q_2, x\$) \to$$
$$\to (adbabacc, q_2, yx\$) \to$$
$$\to (dbabacc, q_2, yyx\$) \to$$
$$\to (babacc, q_3, yyx\$) \to$$
$$\to (abacc, q_4, yyx\$) \to$$
$$\to (bacc, q_3, yx\$) \to$$
$$\to (acc, q_4, yx\$) \to$$
$$\to (cc, q_3, x\$) \to$$
$$\to (cc, q_5, x\$) \to$$
$$\to (c, q_6, x\$) \to$$
$$\to (\epsilon, q_5, \$) \to$$
$$\to (\epsilon, q_7, \epsilon).$$

Push-down automata

Consider the following PDA:

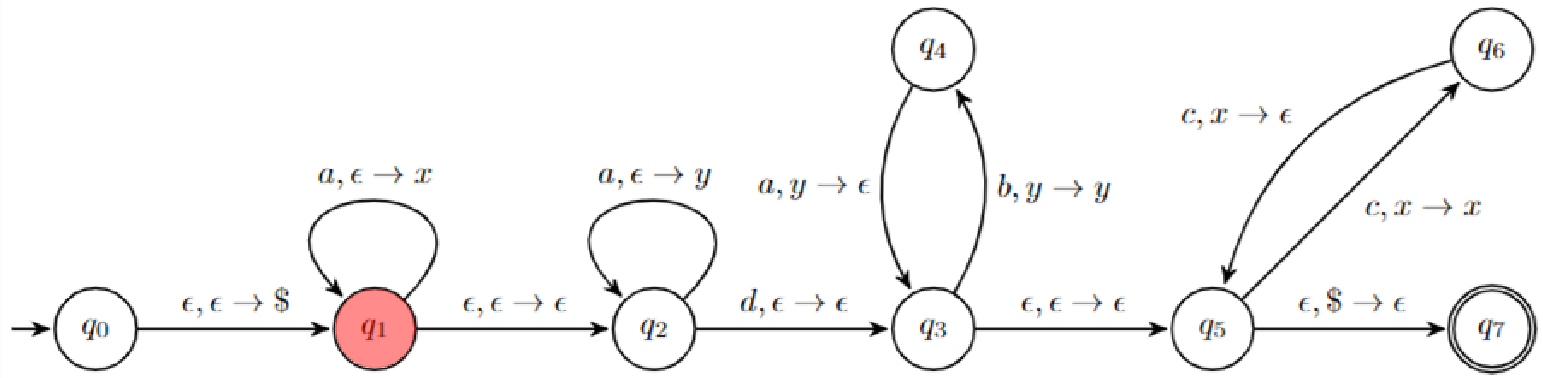
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to&(aaadbabacc, q_1, \$) \to \\
\to&(aadbabacc, q_1, x\$) \to \\
\to&(aadbabacc, q_2, x\$) \to \\
\to&(adbabacc, q_2, yx\$) \to \\
\to&(dbabacc, q_2, yyx\$) \to \\
\to&(babacc, q_3, yyx\$) \to \\
\to&(abacc, q_4, yyx\$) \to \\
\to&(bacc, q_3, yx\$) \to \\
\to&(acc, q_4, yx\$) \to \\
\to&(cc, q_3, x\$) \to \\
\to&(cc, q_5, x\$) \to \\
\to&(c, q_6, x\$) \to \\
\to&(\epsilon, q_5, \$) \to \\
\to&(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

- Show that the PDA accepts the word aaadbabacc.

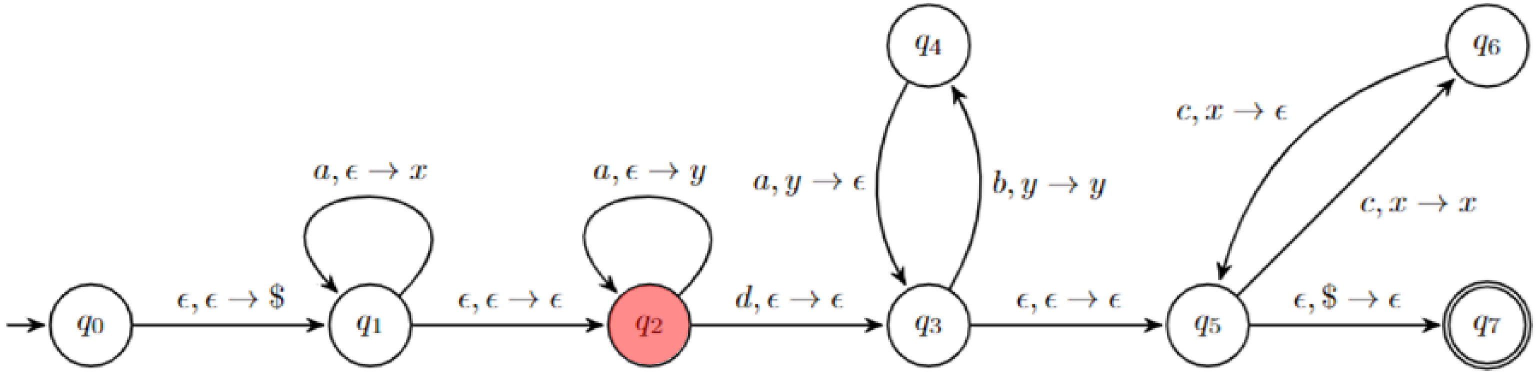


$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to&(aaadbabacc, q_1, \$) \to \\
\to&(aadbabacc, q_1, x\$) \to \\
\to&(aadbabacc, q_2, x\$) \to \\
\to&(adbabacc, q_2, yx\$) \to \\
\to&(dbabacc, q_2, yyx\$) \to \\
\to&(babacc, q_3, yyx\$) \to \\
\to&(abacc, q_4, yyx\$) \to \\
\to&(bacc, q_3, yx\$) \to \\
\to&(acc, q_4, yx\$) \to \\
\to&(cc, q_3, x\$) \to \\
\to&(cc, q_5, x\$) \to \\
\to&(c, q_6, x\$) \to \\
\to&(\epsilon, q_5, \$) \to \\
\to&(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Push-down automata

Consider the following PDA:

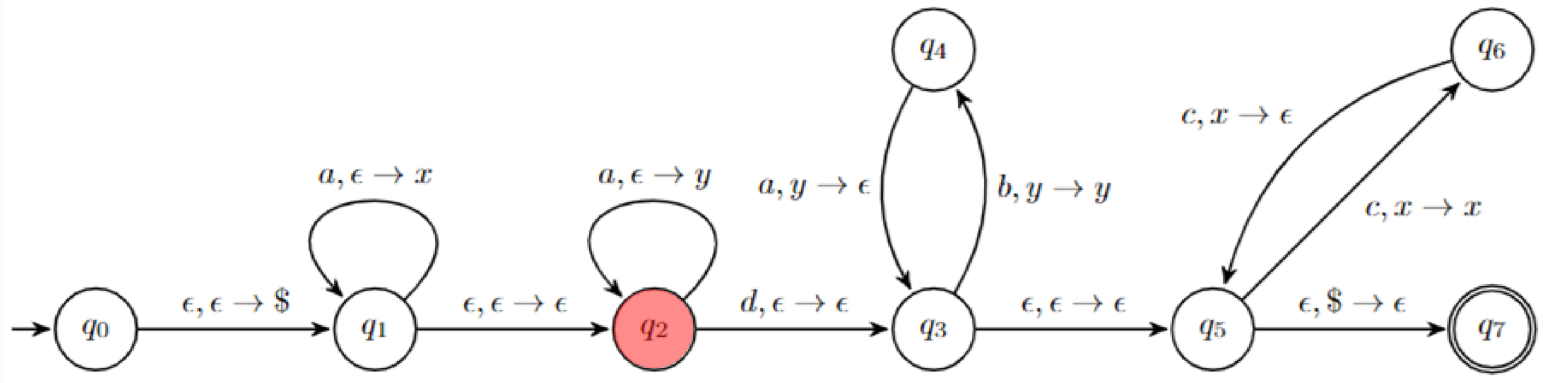
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to &(aaadbabacc, q_1, \$) \to \\
\to &(aadbabacc, q_1, x\$) \to \\
\to &(aadbabacc, q_2, x\$) \to \\
\to &(adbabacc, q_2, yx\$) \to \\
\to &(dbabacc, q_2, yyx\$) \to \\
\to &(babacc, q_3, yyx\$) \to \\
\to &(abacc, q_4, yyx\$) \to \\
\to &(bacc, q_3, yx\$) \to \\
\to &(acc, q_4, yx\$) \to \\
\to &(cc, q_3, x\$) \to \\
\to &(cc, q_5, x\$) \to \\
\to &(c, q_6, x\$) \to \\
\to &(\epsilon, q_5, \$) \to \\
\to &(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

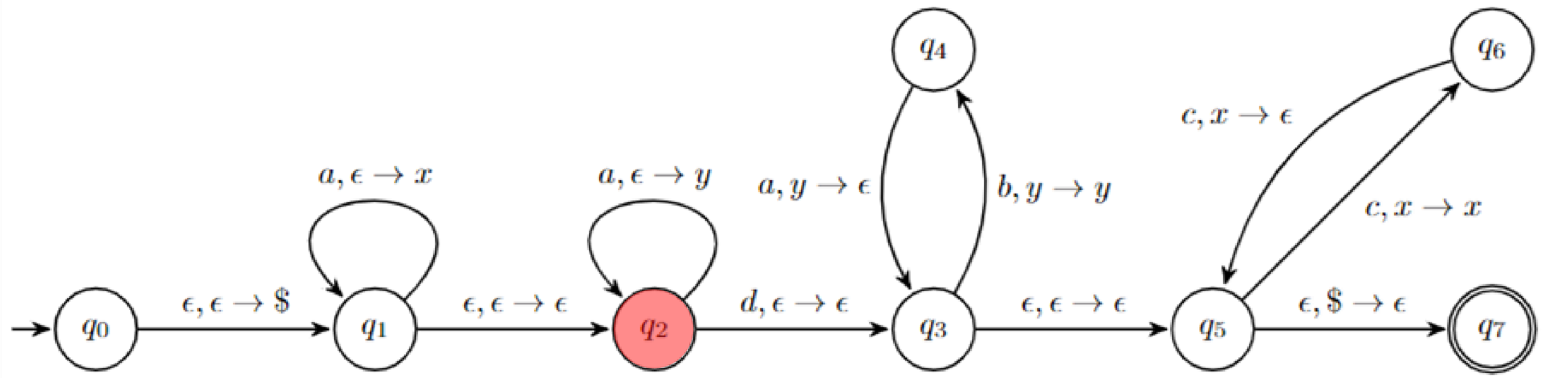
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to&(aaadbabacc, q_1, \$) \to \\
\to&(aadbabacc, q_1, x\$) \to \\
\to&(aadbabacc, q_2, x\$) \to \\
\to&(adbabacc, q_2, yx\$) \to \\
\to&(dbabacc, q_2, yyx\$) \to \\
\to&(babacc, q_3, yyx\$) \to \\
\to&(abacc, q_4, yyx\$) \to \\
\to&(bacc, q_3, yx\$) \to \\
\to&(acc, q_4, yx\$) \to \\
\to&(cc, q_3, x\$) \to \\
\to&(cc, q_5, x\$) \to \\
\to&(c, q_6, x\$) \to \\
\to&(\epsilon, q_5, \$) \to \\
\to&(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

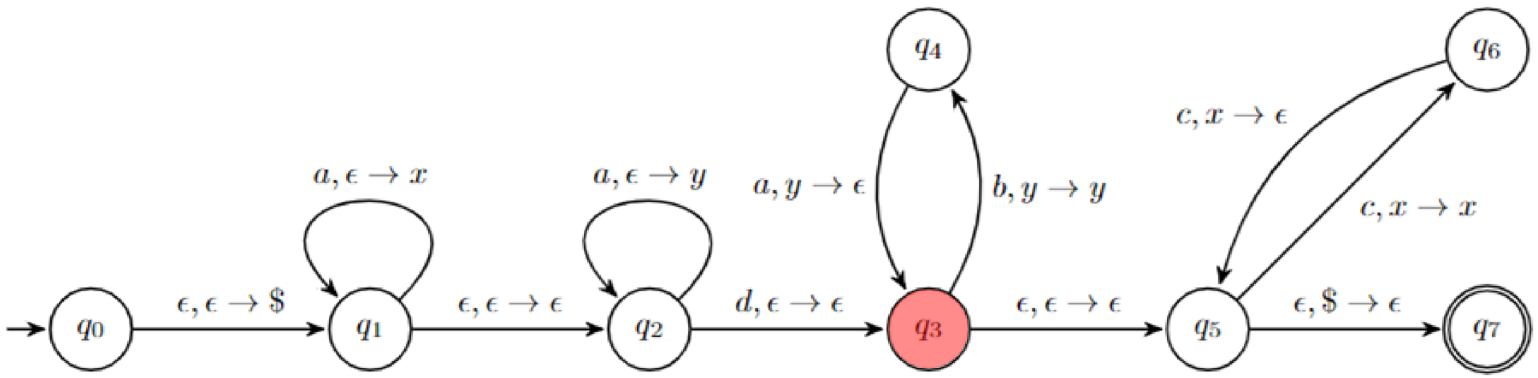
- Show that the PDA accepts the word aaadbabacc.



$$(aaadbabacc, q_0, \epsilon) \to$$
$$\to (aaadbabacc, q_1, \$) \to$$
$$\to (aadbabacc, q_1, x\$) \to$$
$$\to (aadbabacc, q_2, x\$) \to$$
$$\to (adbabacc, q_2, yx\$) \to$$
$$\to (dbabacc, q_2, yyx\$) \to$$
$$\to (babacc, q_3, yyx\$) \to$$
$$\to (abacc, q_4, yyx\$) \to$$
$$\to (bacc, q_3, yx\$) \to$$
$$\to (acc, q_4, yx\$) \to$$
$$\to (cc, q_3, x\$) \to$$
$$\to (cc, q_5, x\$) \to$$
$$\to (c, q_6, x\$) \to$$
$$\to (\epsilon, q_5, \$) \to$$
$$\to (\epsilon, q_7, \epsilon).$$

Consider the following PDA:

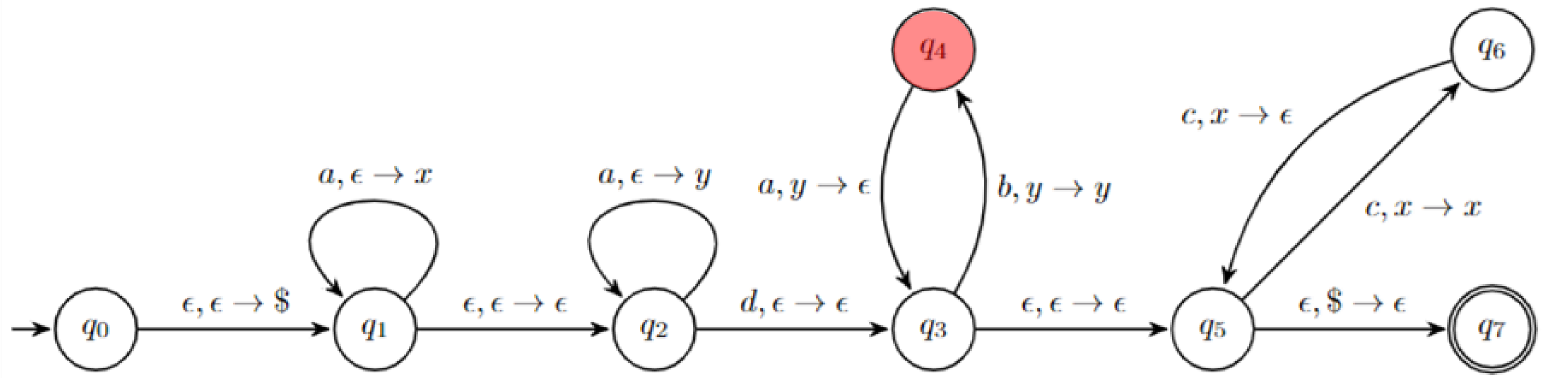
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to &(aaadbabacc, q_1, \$) \to \\
\to &(aadbabacc, q_1, x\$) \to \\
\to &(aadbabacc, q_2, x\$) \to \\
\to &(adbabacc, q_2, yx\$) \to \\
\to &(dbabacc, q_2, yyx\$) \to \\
\to &(babacc, q_3, yyx\$) \to \\
\to &(abacc, q_4, yyx\$) \to \\
\to &(bacc, q_3, yx\$) \to \\
\to &(acc, q_4, yx\$) \to \\
\to &(cc, q_3, x\$) \to \\
\to &(cc, q_5, x\$) \to \\
\to &(c, q_6, x\$) \to \\
\to &(\epsilon, q_5, \$) \to \\
\to &(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

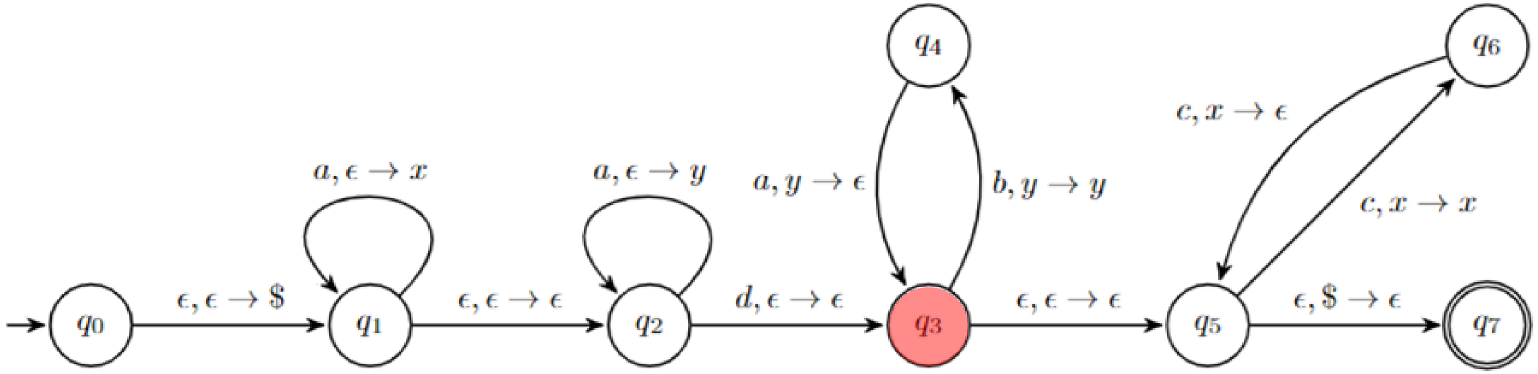
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to&(aaadbabacc, q_1, \$) \to \\
\to&(aadbabacc, q_1, x\$) \to \\
\to&(aadbabacc, q_2, x\$) \to \\
\to&(adbabacc, q_2, yx\$) \to \\
\to&(dbabacc, q_2, yyx\$) \to \\
\to&(babacc, q_3, yyx\$) \to \\
\to&(abacc, q_4, yyx\$) \to \\
\to&(bacc, q_3, yx\$) \to \\
\to&(acc, q_4, yx\$) \to \\
\to&(cc, q_3, x\$) \to \\
\to&(cc, q_5, x\$) \to \\
\to&(c, q_6, x\$) \to \\
\to&(\epsilon, q_5, \$) \to \\
\to&(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

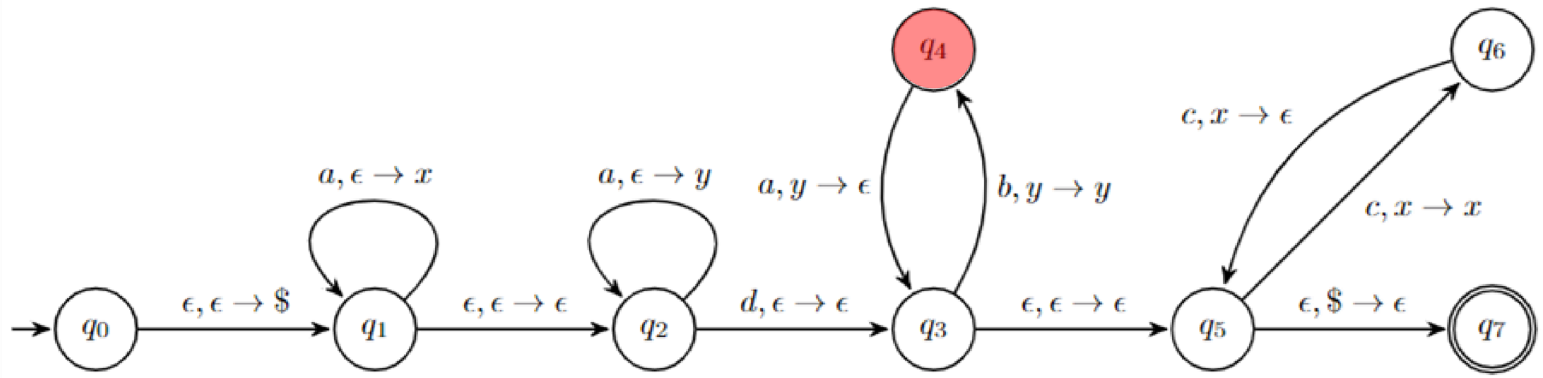
- Show that the PDA accepts the word aaadbabacc.



$$(aaadbabacc, q_0, \epsilon) \to$$
$$\to (aaadbabacc, q_1, \$) \to$$
$$\to (aadbabacc, q_1, x\$) \to$$
$$\to (aadbabacc, q_2, x\$) \to$$
$$\to (adbabacc, q_2, yx\$) \to$$
$$\to (dbabacc, q_2, yyx\$) \to$$
$$\to (babacc, q_3, yyx\$) \to$$
$$\to (abacc, q_4, yyx\$) \to$$
$$\to (bacc, q_3, yx\$) \to$$
$$\to (acc, q_4, yx\$) \to$$
$$\to (cc, q_3, x\$) \to$$
$$\to (cc, q_5, x\$) \to$$
$$\to (c, q_6, x\$) \to$$
$$\to (\epsilon, q_5, \$) \to$$
$$\to (\epsilon, q_7, \epsilon).$$

Consider the following PDA:

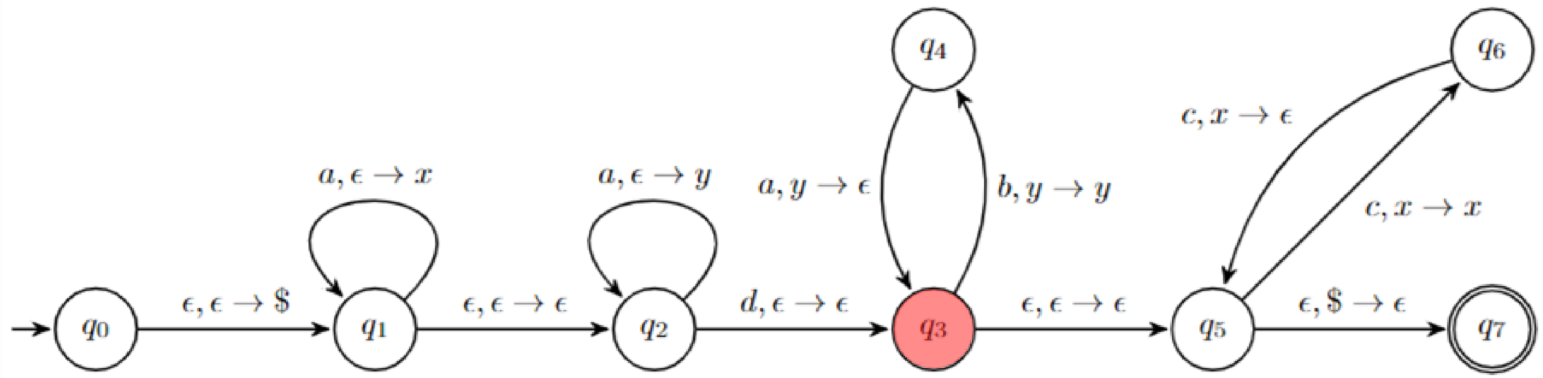
- Show that the PDA accepts the word aaadbabacc.



$$(aaadbabacc, q_0, \epsilon) \to$$
$$\to (aaadbabacc, q_1, \$) \to$$
$$\to (aadbabacc, q_1, x\$) \to$$
$$\to (aadbabacc, q_2, x\$) \to$$
$$\to (adbabacc, q_2, yx\$) \to$$
$$\to (dbabacc, q_2, yyx\$) \to$$
$$\to (babacc, q_3, yyx\$) \to$$
$$\to (abacc, q_4, yyx\$) \to$$
$$\to (bacc, q_3, yx\$) \to$$
$$\to (acc, q_4, yx\$) \to$$
$$\to (cc, q_3, x\$) \to$$
$$\to (cc, q_5, x\$) \to$$
$$\to (c, q_6, x\$) \to$$
$$\to (\epsilon, q_5, \$) \to$$
$$\to (\epsilon, q_7, \epsilon).$$

Consider the following PDA:

- Show that the PDA accepts the word aaadbabacc.

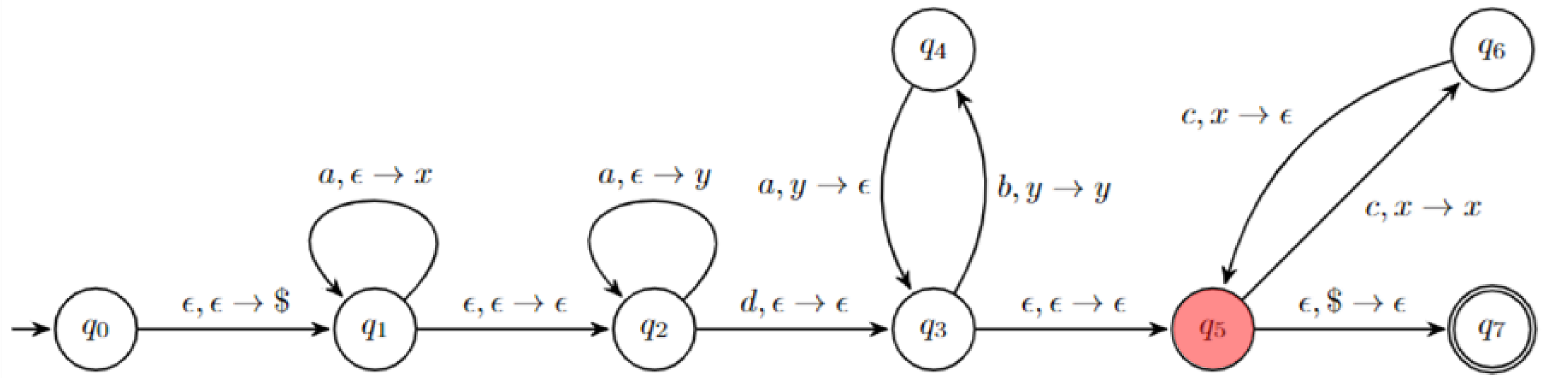


$$(aaadbabacc, q_0, \epsilon) \to$$
$$\to (aaadbabacc, q_1, \$) \to$$
$$\to (aadbabacc, q_1, x\$) \to$$
$$\to (aadbabacc, q_2, x\$) \to$$
$$\to (adbabacc, q_2, yx\$) \to$$
$$\to (dbabacc, q_2, yyx\$) \to$$
$$\to (babacc, q_3, yyx\$) \to$$
$$\to (abacc, q_4, yyx\$) \to$$
$$\to (bacc, q_3, yx\$) \to$$
$$\to (acc, q_4, yx\$) \to$$
$$\to (cc, q_3, x\$) \to$$
$$\to (cc, q_5, x\$) \to$$
$$\to (c, q_6, x\$) \to$$
$$\to (\epsilon, q_5, \$) \to$$
$$\to (\epsilon, q_7, \epsilon).$$

Push-down automata

Consider the following PDA:

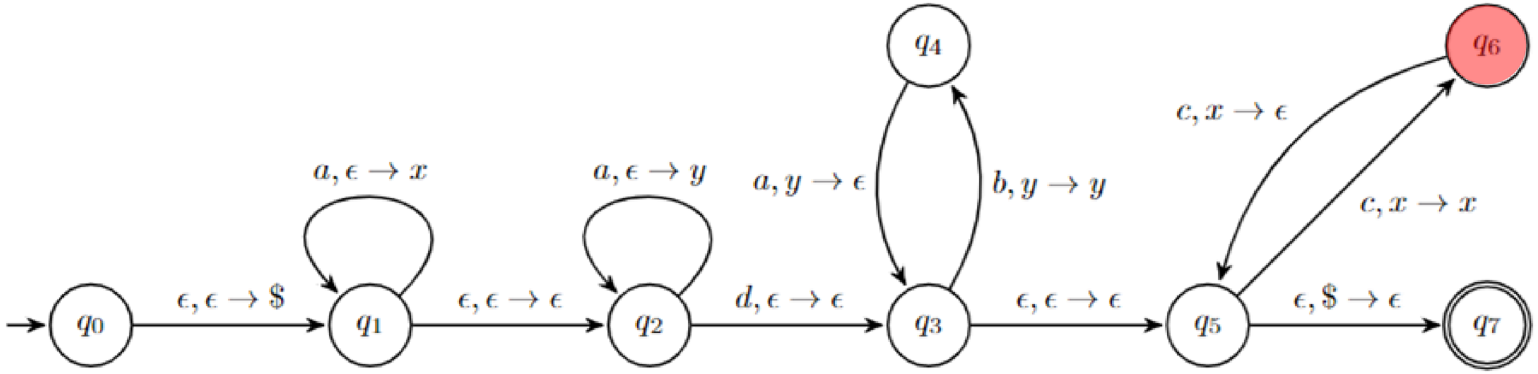
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to &(aaadbabacc, q_1, \$) \to \\
\to &(aadbabacc, q_1, x\$) \to \\
\to &(aadbabacc, q_2, x\$) \to \\
\to &(adbabacc, q_2, yx\$) \to \\
\to &(dbabacc, q_2, yyx\$) \to \\
\to &(babacc, q_3, yyx\$) \to \\
\to &(abacc, q_4, yyx\$) \to \\
\to &(bacc, q_3, yx\$) \to \\
\to &(acc, q_4, yx\$) \to \\
\to &(cc, q_3, x\$) \to \\
\to &(cc, q_5, x\$) \to \\
\to &(c, q_6, x\$) \to \\
\to &(\epsilon, q_5, \$) \to \\
\to &(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

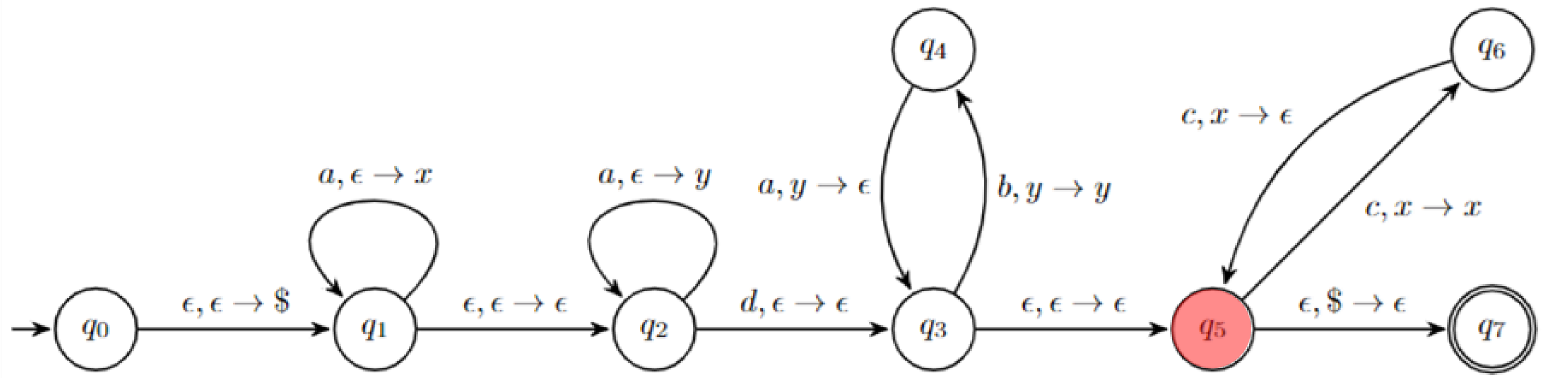
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to &(aaadbabacc, q_1, \$) \to \\
\to &(aadbabacc, q_1, x\$) \to \\
\to &(aadbabacc, q_2, x\$) \to \\
\to &(adbabacc, q_2, yx\$) \to \\
\to &(dbabacc, q_2, yyx\$) \to \\
\to &(babacc, q_3, yyx\$) \to \\
\to &(abacc, q_4, yyx\$) \to \\
\to &(bacc, q_3, yx\$) \to \\
\to &(acc, q_4, yx\$) \to \\
\to &(cc, q_3, x\$) \to \\
\to &(cc, q_5, x\$) \to \\
\to &(c, q_6, x\$) \to \\
\to &(\epsilon, q_5, \$) \to \\
\to &(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

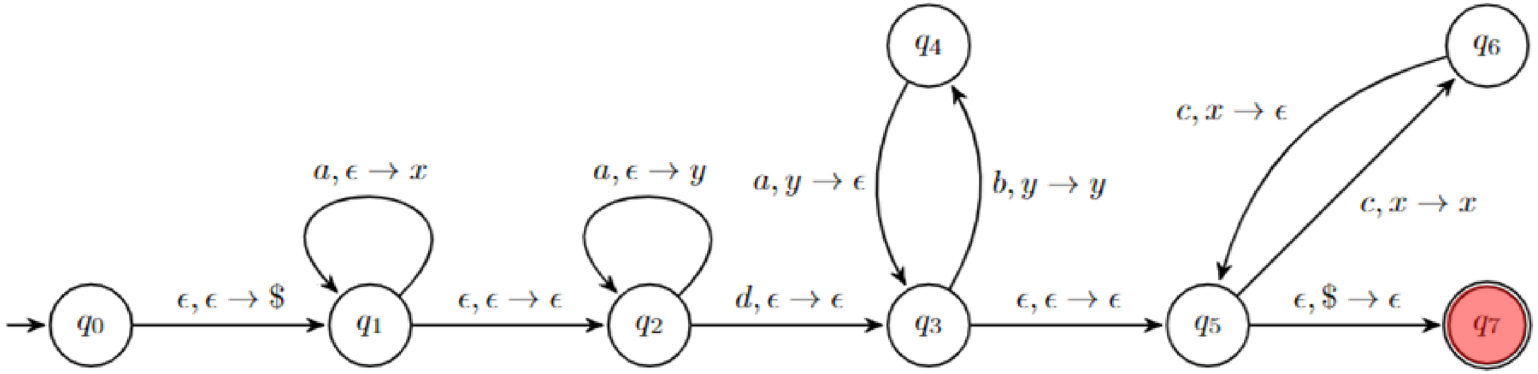
- Show that the PDA accepts the word aaadbabacc.



$$(aaadbabacc, q_0, \epsilon) \to$$
$$\to (aaadbabacc, q_1, \$) \to$$
$$\to (aadbabacc, q_1, x\$) \to$$
$$\to (aadbabacc, q_2, x\$) \to$$
$$\to (adbabacc, q_2, yx\$) \to$$
$$\to (dbabacc, q_2, yyx\$) \to$$
$$\to (babacc, q_3, yyx\$) \to$$
$$\to (abacc, q_4, yyx\$) \to$$
$$\to (bacc, q_3, yx\$) \to$$
$$\to (acc, q_4, yx\$) \to$$
$$\to (cc, q_3, x\$) \to$$
$$\to (cc, q_5, x\$) \to$$
$$\to (c, q_6, x\$) \to$$
$$\to (\epsilon, q_5, \$) \to$$
$$\to (\epsilon, q_7, \epsilon).$$

Consider the following PDA:

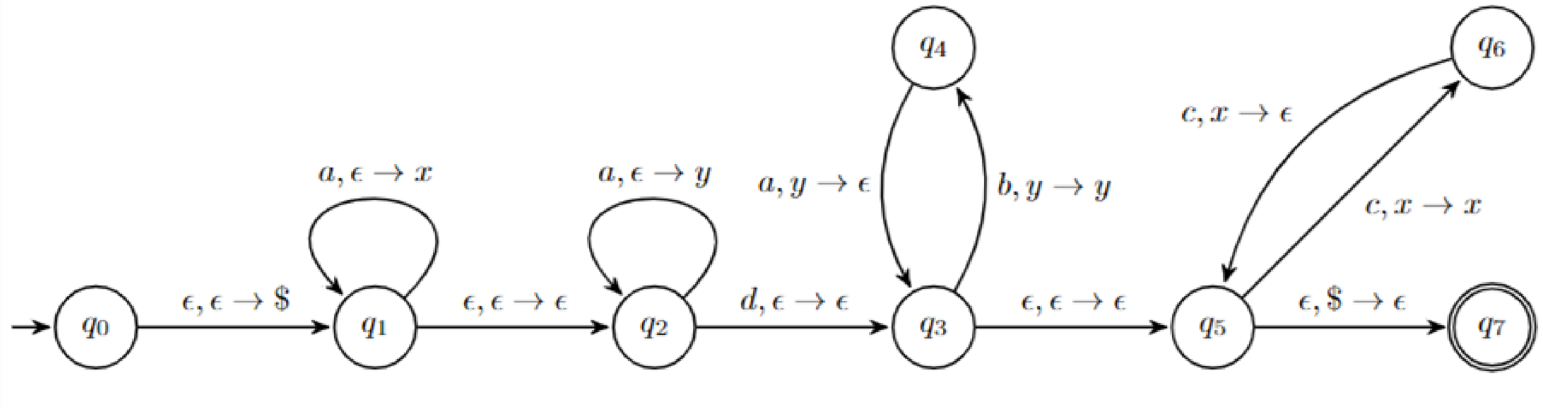
- Show that the PDA accepts the word aaadbabacc.



$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
&\to (aaadbabacc, q_1, \$) \to \\
&\to (aadbabacc, q_1, x\$) \to \\
&\to (aadbabacc, q_2, x\$) \to \\
&\to (adbabacc, q_2, yx\$) \to \\
&\to (dbabacc, q_2, yyx\$) \to \\
&\to (babacc, q_3, yyx\$) \to \\
&\to (abacc, q_4, yyx\$) \to \\
&\to (bacc, q_3, yx\$) \to \\
&\to (acc, q_4, yx\$) \to \\
&\to (cc, q_3, x\$) \to \\
&\to (cc, q_5, x\$) \to \\
&\to (c, q_6, x\$) \to \\
&\to (\epsilon, q_5, \$) \to \\
&\to (\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

- Show that the PDA accepts the word aaadbabacc.

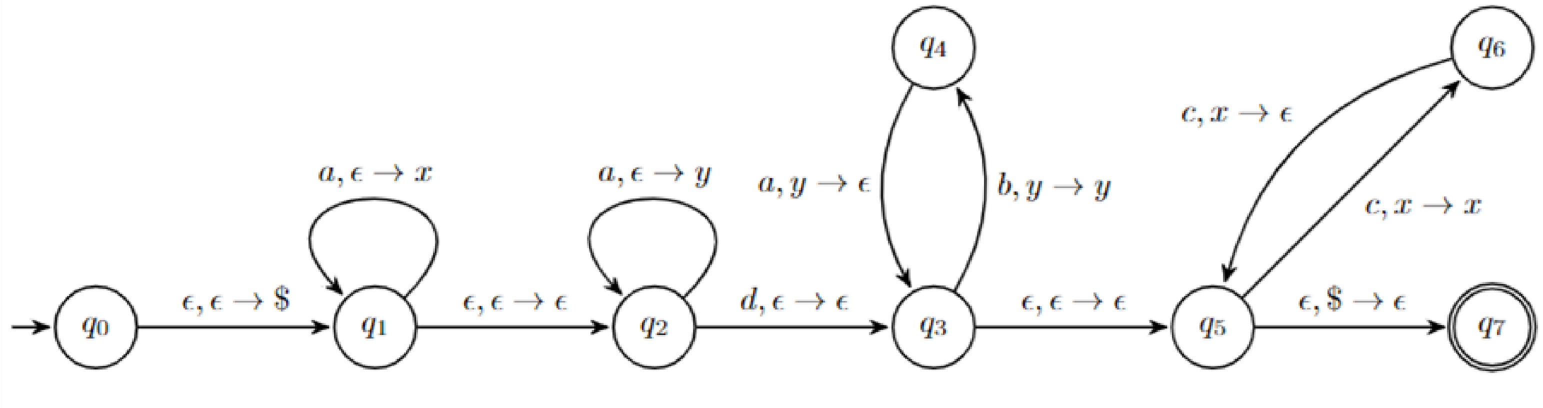


- Which language L does the given PDA accept?

$$
\begin{aligned}
& (aaadbabacc, q_0, \epsilon) \to \\
\to & (aaadbabacc, q_1, \$) \to \\
\to & (aadbabacc, q_1, x\$) \to \\
\to & (aadbabacc, q_2, x\$) \to \\
\to & (adbabacc, q_2, yx\$) \to \\
\to & (dbabacc, q_2, yyx\$) \to \\
\to & (babacc, q_3, yyx\$) \to \\
\to & (abacc, q_4, yyx\$) \to \\
\to & (bacc, q_3, yx\$) \to \\
\to & (acc, q_4, yx\$) \to \\
\to & (cc, q_3, x\$) \to \\
\to & (cc, q_5, x\$) \to \\
\to & (c, q_6, x\$) \to \\
\to & (\epsilon, q_5, \$) \to \\
\to & (\epsilon, q_7, \epsilon).
\end{aligned}
$$

Consider the following PDA:

- Show that the PDA accepts the word aaadbabacc.



- Which language L does the given PDA accept?

$$L = \{a^n a^s d(ba)^s c^{2n} \in \{a, b, c, d\}^* \mid n, s \geq 0\}$$

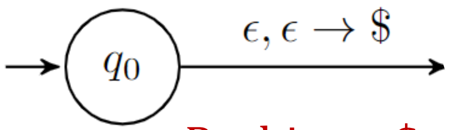
$$
\begin{aligned}
&(aaadbabacc, q_0, \epsilon) \to \\
\to &(aaadbabacc, q_1, \$) \to \\
\to &(aadbabacc, q_1, x\$) \to \\
\to &(aadbabacc, q_2, x\$) \to \\
\to &(adbabacc, q_2, yx\$) \to \\
\to &(dbabacc, q_2, yyx\$) \to \\
\to &(babacc, q_3, yyx\$) \to \\
\to &(abacc, q_4, yyx\$) \to \\
\to &(bacc, q_3, yx\$) \to \\
\to &(acc, q_4, yx\$) \to \\
\to &(cc, q_3, x\$) \to \\
\to &(cc, q_5, x\$) \to \\
\to &(c, q_6, x\$) \to \\
\to &(\epsilon, q_5, \$) \to \\
\to &(\epsilon, q_7, \epsilon).
\end{aligned}
$$

Push-down automata

Consider the following PDA:

- Create a PDA that recognizes the following context free language:

$$L = \{a^* w c^k \mid w \in \{a,b\}^* \text{ and } k = |w|_a \ (k = \text{the number of } as \text{ in } w)\}$$

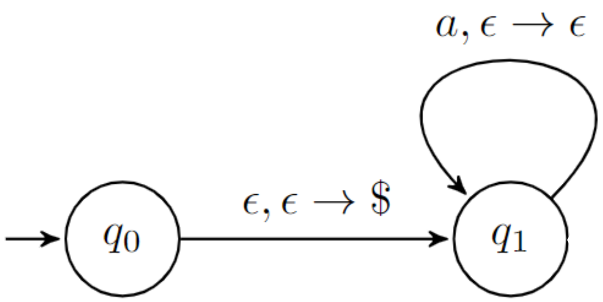


Pushing a $
onto the stack

Consider the following PDA:

- Create a PDA that recognizes the following context free language:

$$L = \{a^* w c^k \mid w \in \{a,b\}^* \text{ and } k = |w|_a \ (k = \text{the number of } a\text{s in } w)\}$$

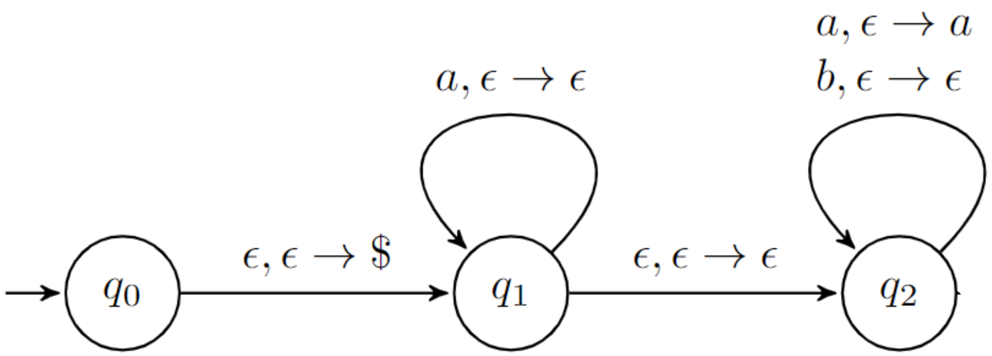


a^*

Consider the following PDA:

- Create a PDA that recognizes the following context free language:

$$L = \{a^* w c^k \mid w \in \{a,b\}^* \text{ and } k = |w|_a \ (k = \text{the number of } as \text{ in } w)\}$$



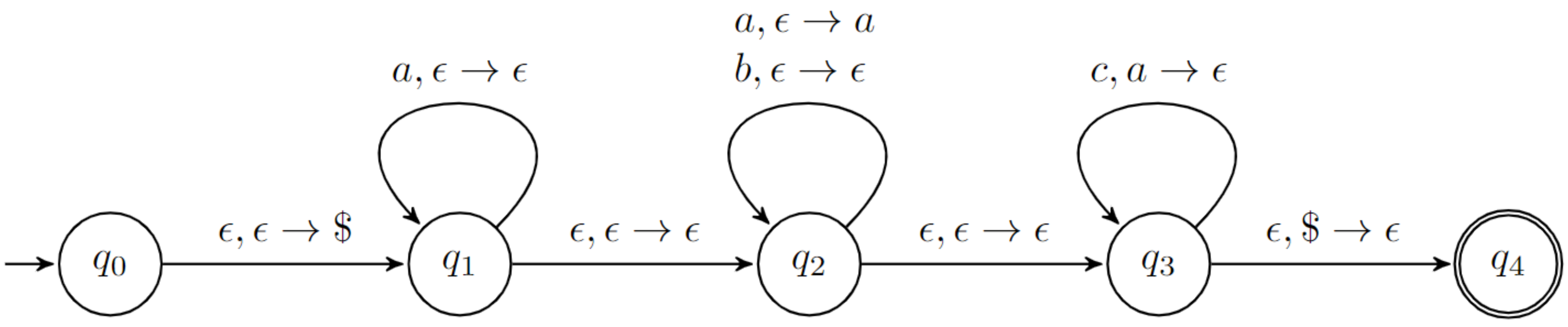
w

$stack: \underbrace{a \dots a}_{|w|_a}$

Consider the following PDA:

- Create a PDA that recognizes the following context free language:

$$L = \{a^* w c^k \mid w \in \{a, b\}^* \text{ and } k = |w|_a \ (k = \text{the number of } a\text{s in } w)\}$$

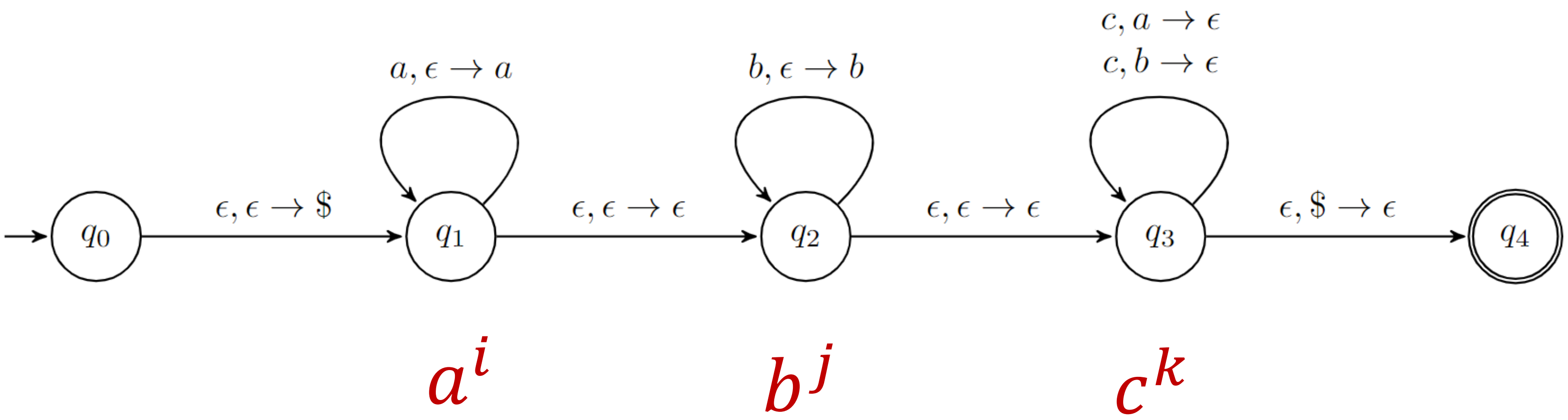


c^k

Consider the following PDA:

- Create a PDA that recognizes the following language.

$$L = \{a^i b^j c^k \mid i, j \geq 0,\ k = i + j\}$$

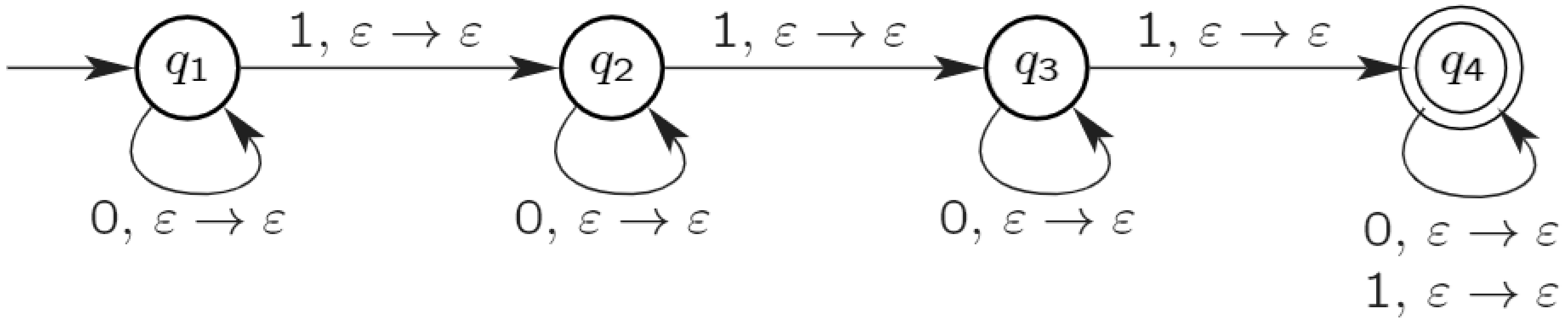


Push-down automata

Consider the following PDA:

- Create a PDA that recognizes the following language.

$$A = \{ w \in \{0,1\}^* \mid w \text{ contains at least three 1s} \}$$

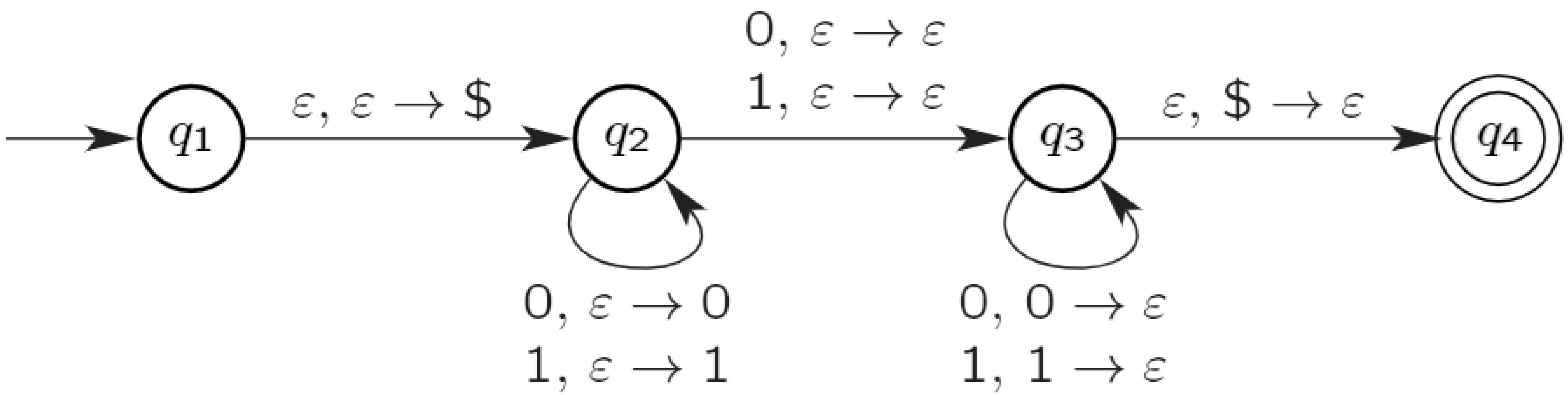


Push-down automata

Consider the following PDA:

- Create a PDA that recognizes the following language.

$$B = \{ w \in \{0,1\}^* \mid w = w^{\mathcal{R}} \text{ and the length of } w \text{ is odd} \}$$

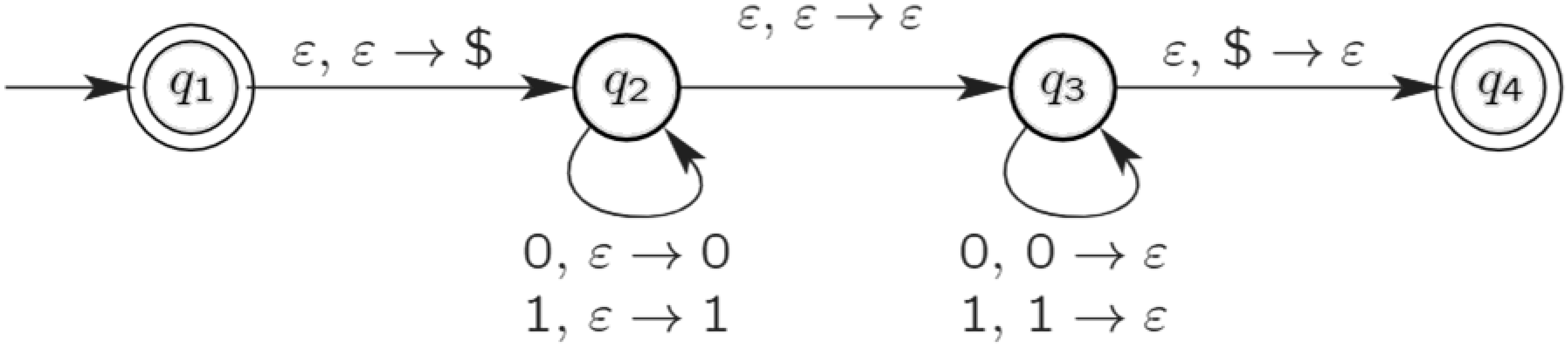


Push-down automata

Consider the following PDA:

- Create a PDA that recognizes the following language.

$$C = \{ w \in \{0,1\}^* \mid w = w^{\mathcal{R}} \}$$

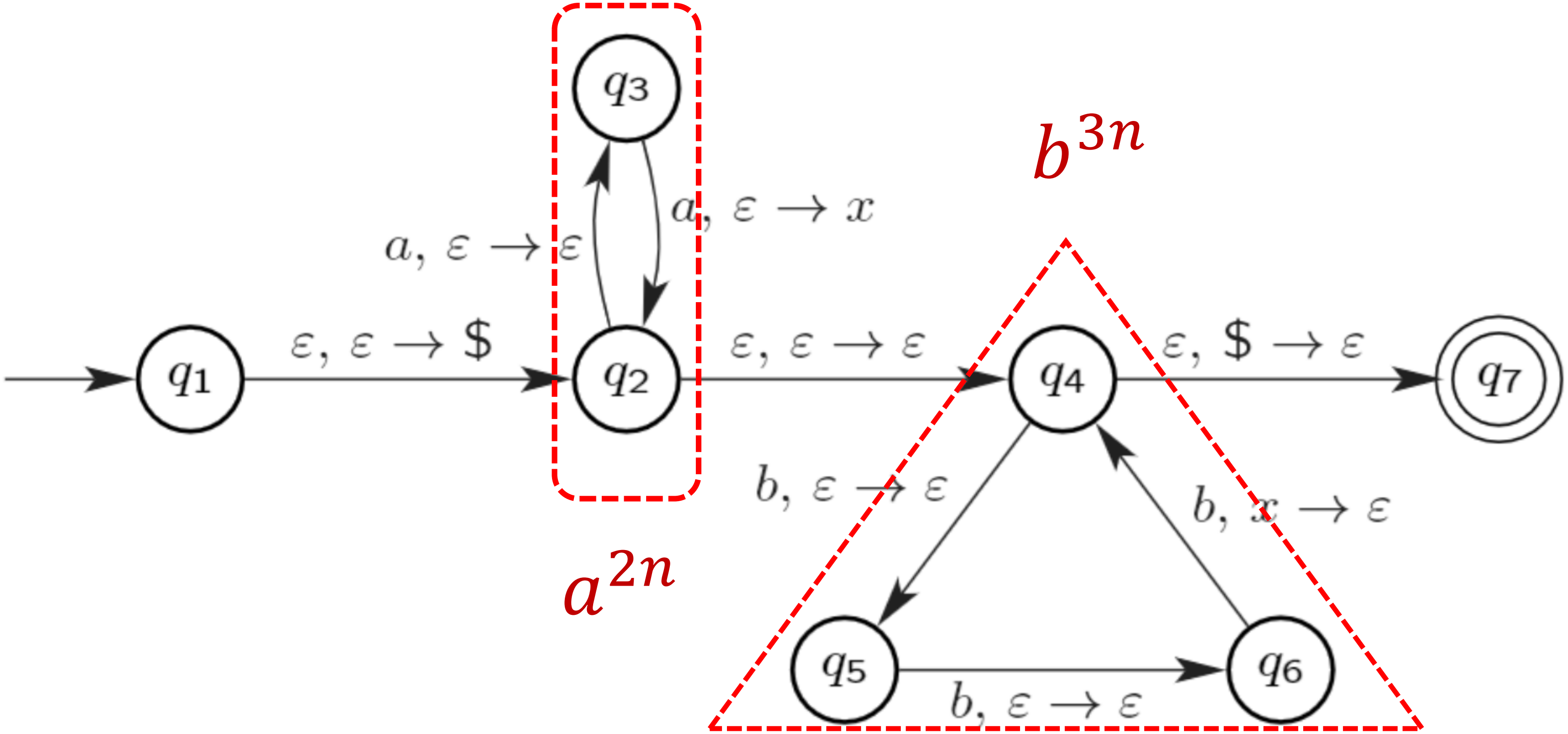


Push-down automata

Consider the following PDA:

- Create a PDA that recognizes the following language.

$$F = \{ a^{2n}b^{3n} \mid n \geq 0 \}$$



Consider the following PDA:

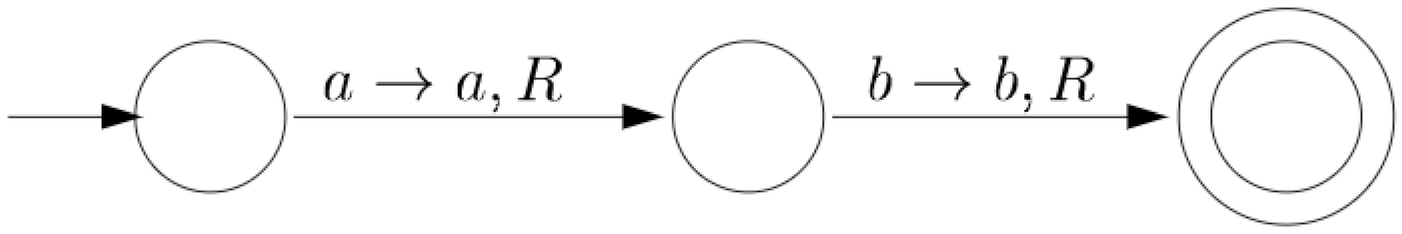
- Design a Turing Machine that accepts the following language.

$$L = \{ab(a+b)^*\}$$

Consider the following PDA:

- Design a Turing Machine that accepts the following language.

$$L = \{ab(a+b)^*\}$$



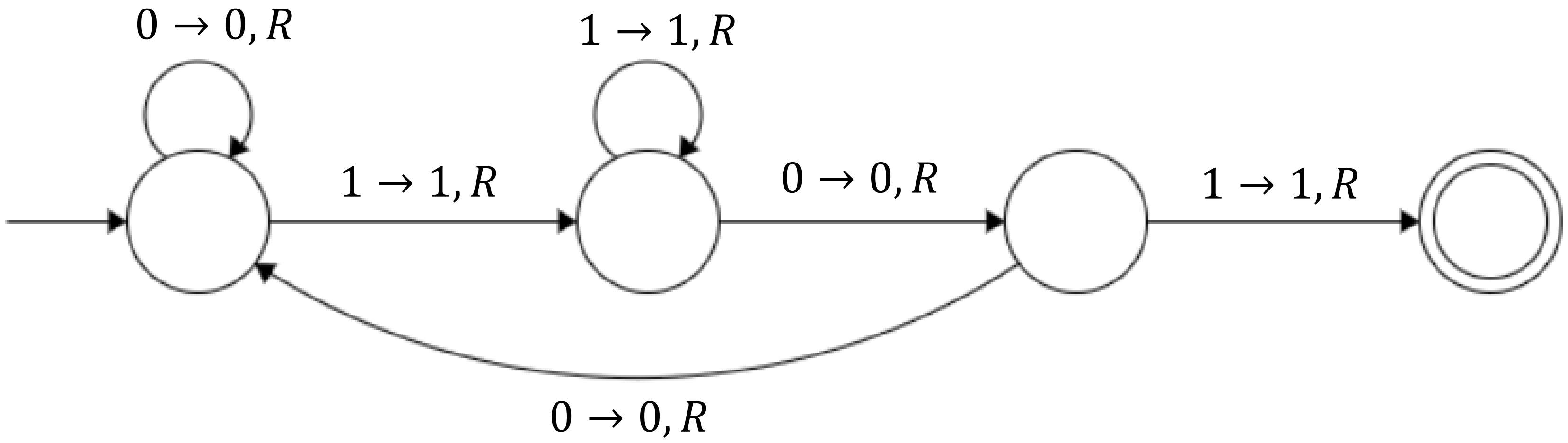
Answer.

- The Turing machine for language L is shown in Figure above .
- Notice that the machine doesn't have to read all the input string in order to accept the string.
- This is because we know the input string alphabet which is {a, b}, and after string ab any string made from the input alphabet can follow.

Consider the following PDA:

- Design a Turing Machine that accepts strings containing 101.

- The language is regular.
- The TM, will be similar to the FA that recognizes the language.

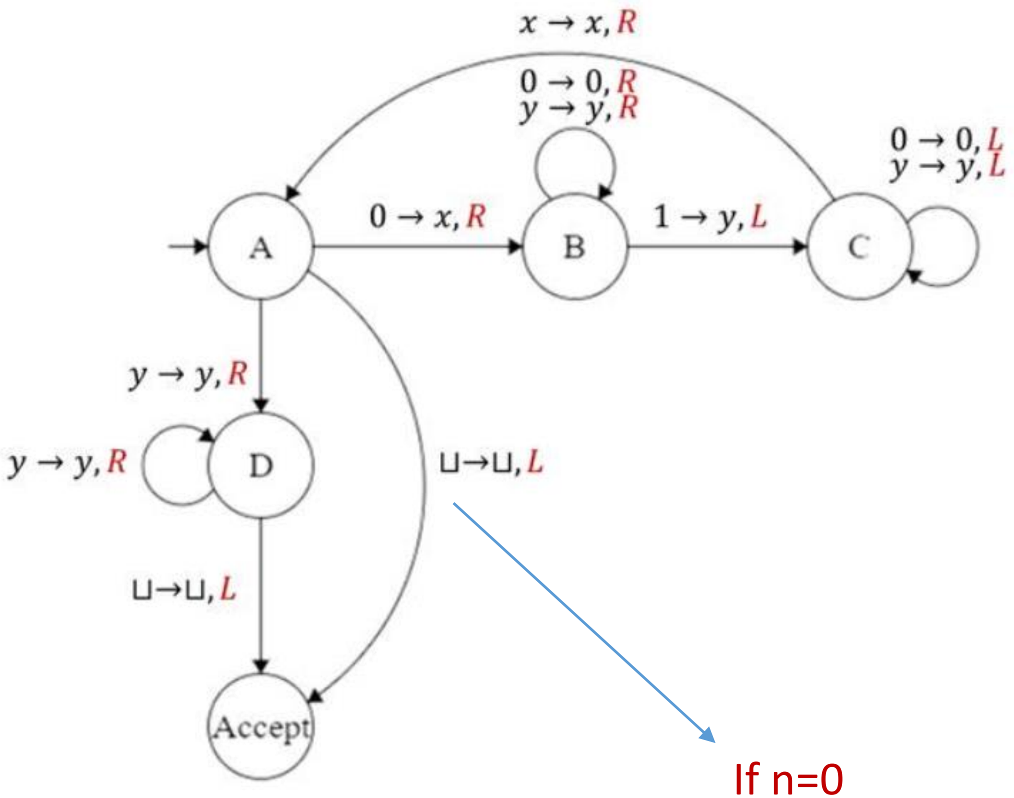


Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n 1^n \mid n \geq 0\}$.

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



If n=0

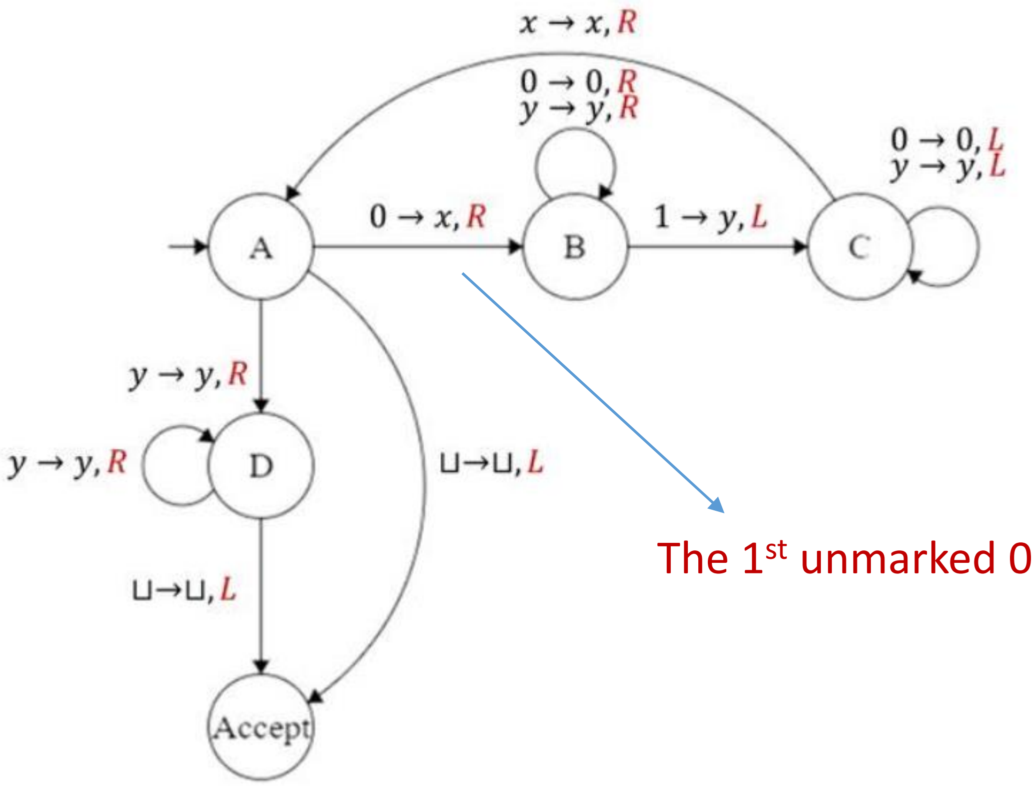
Turing Machines

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n1^n \mid n \geq 0\}$.

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



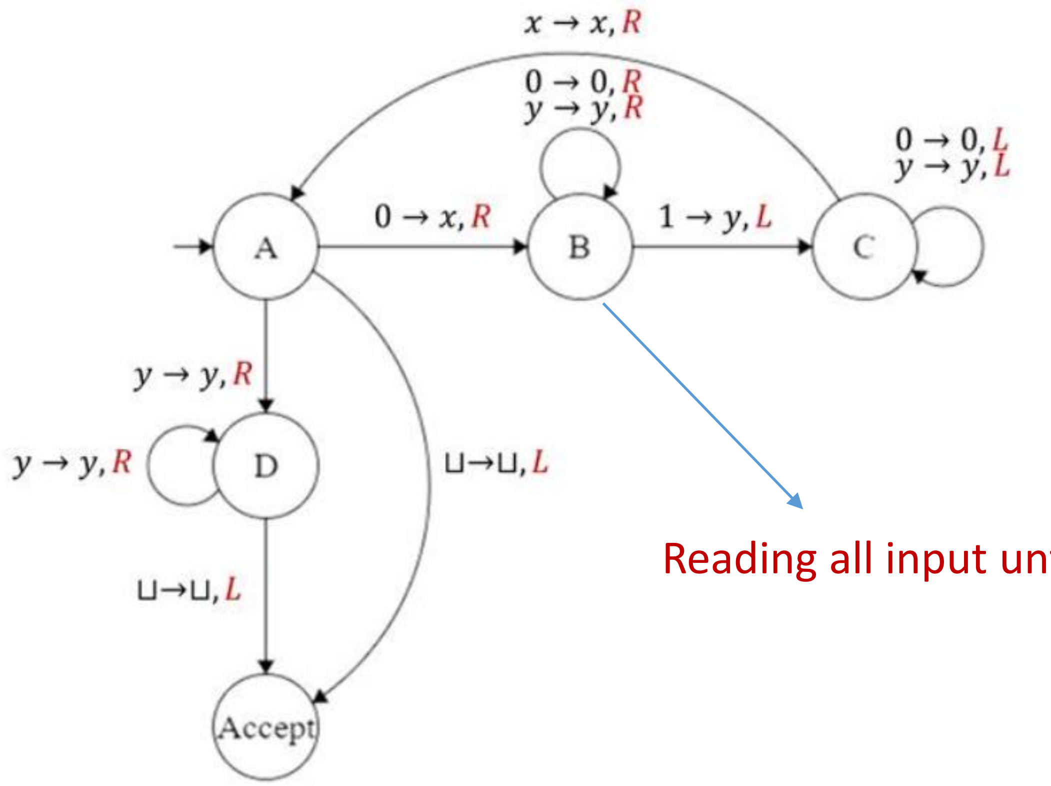
The 1^{st} unmarked 0 will be marked by x

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n 1^n\}$. .

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



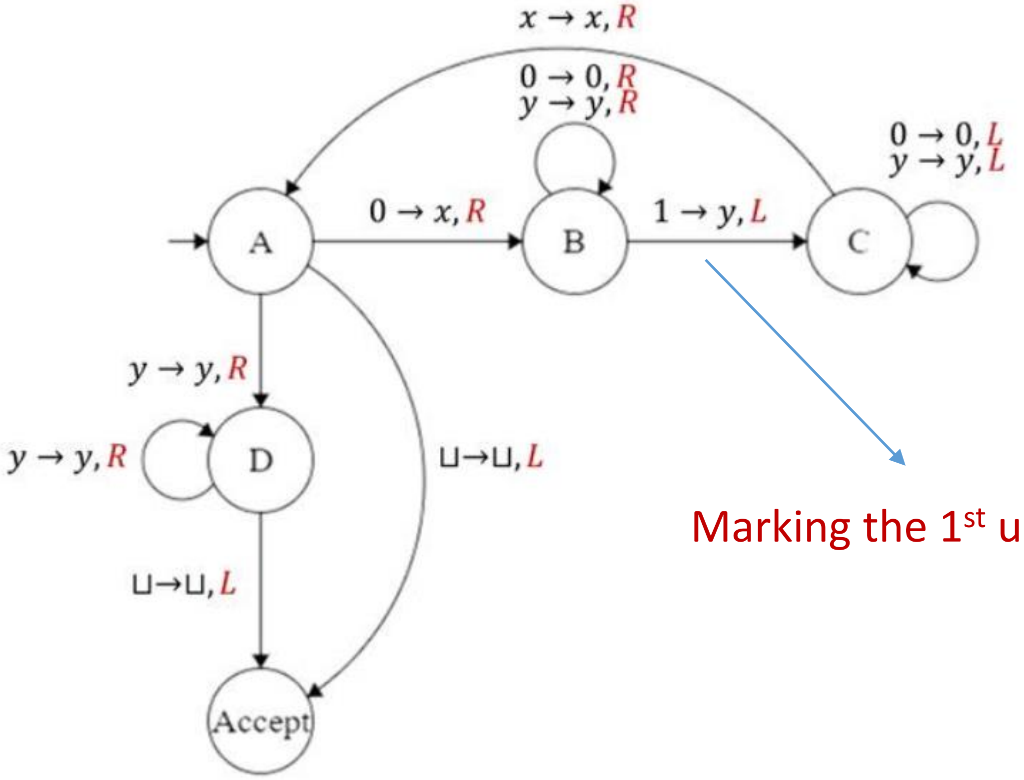
Reading all input until the 1st unmarked 1.

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n 1^n\}$. .

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



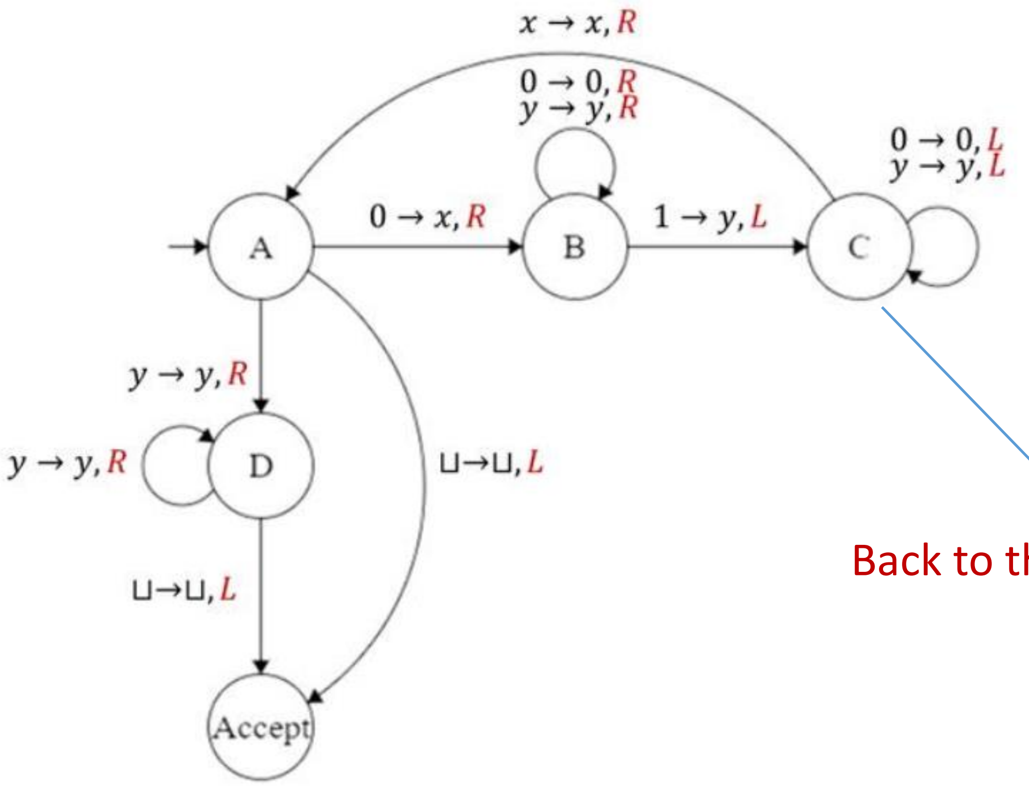
Marking the 1st unmarked 1.

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n 1^n\}$. .

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



Back to the last marked 0 (x).

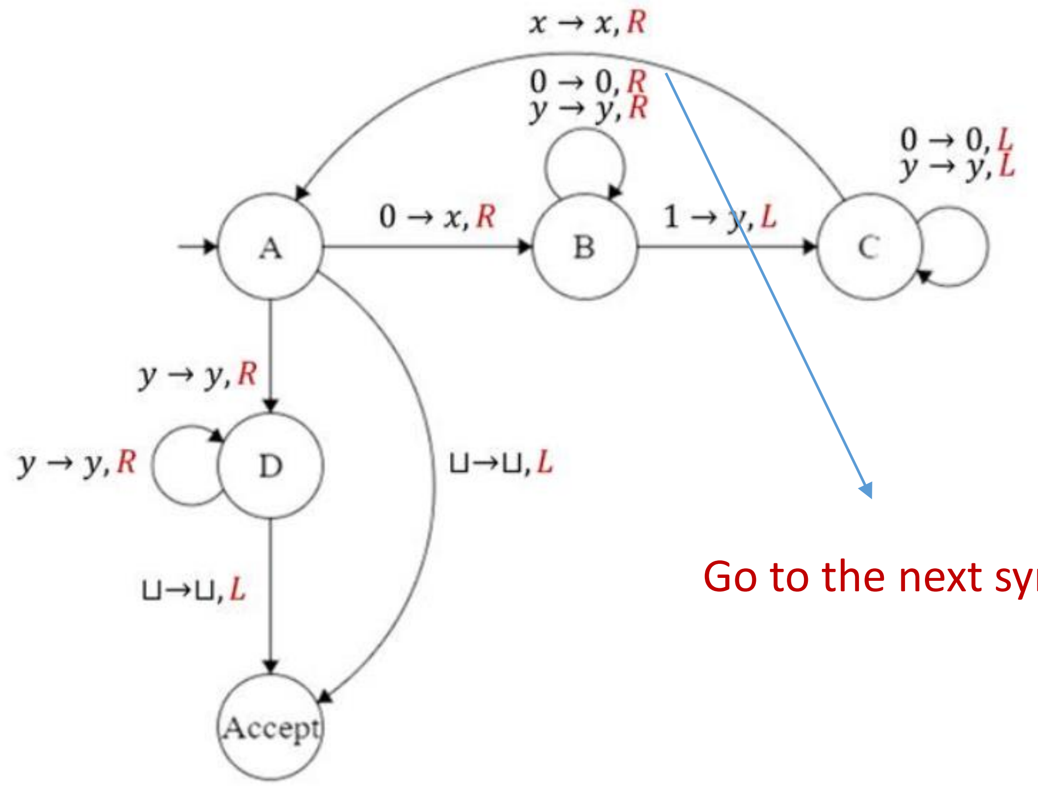
Turing Machines

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n 1^n\}$. .

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



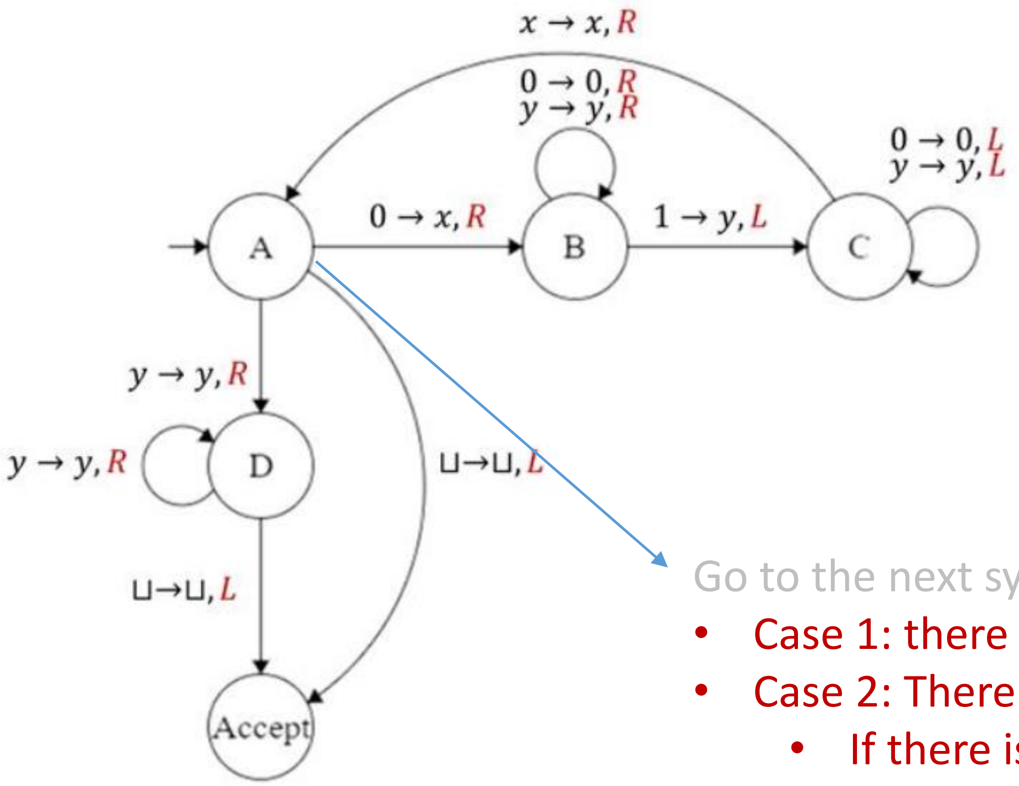
Go to the next symbol

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n1^n\}$. .

0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



Go to the next symbol

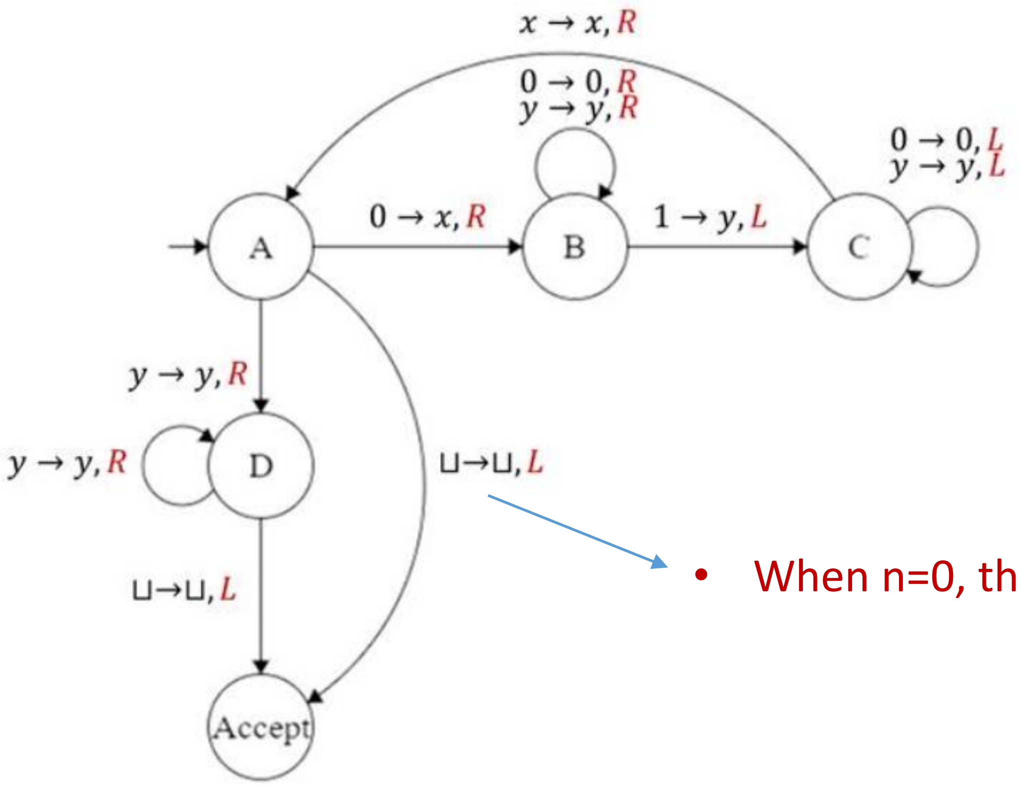
- Case 1: there is another 0 , then the process will be repeated
- Case 2: There is no more 0 (just y), then read all y's.
    - If there is no 1 more 1's (just blank symbol), accept.

Consider the following PDA:

- Design a Turing Machine that accepts the language $\{0^n1^n\}$. .

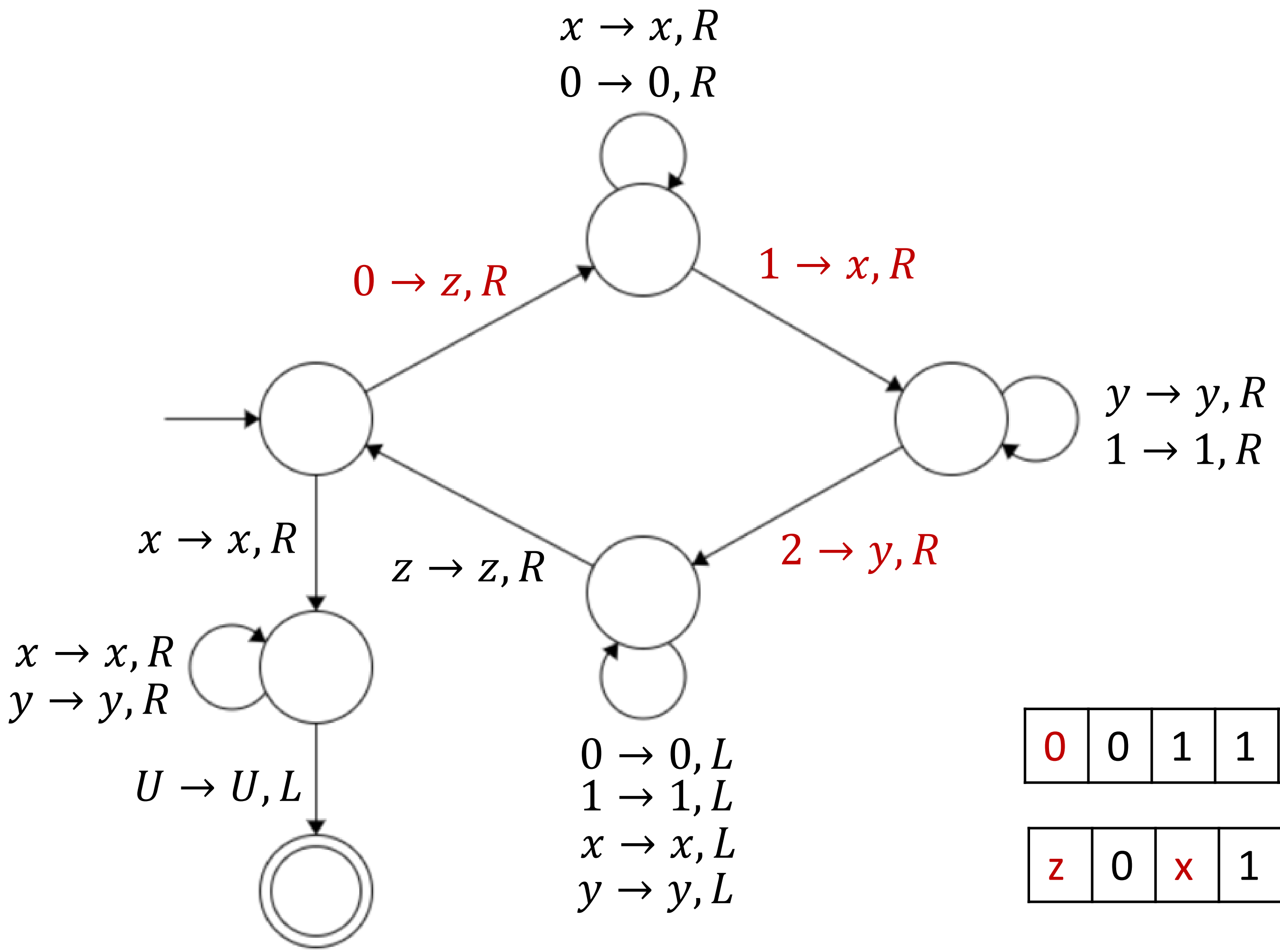
0	0	0	1	1	1	U			

x	0	0	y	1	1	U			



- When n=0, the string is accepted.

Draw the diagram for a TM that accepts the language $\{0^n 1^n 2^n\}$.



0	0	1	1	2	2	U			

z	0	x	1	y	2	U			

Q: Draw the diagram for a TM that that shifts the entire input string one cell to the right.

Example 1:

- Use the pumping lemma to show that the language $B = \{a^n b^n c^n | n \geq 0\}$ is not context free.

Proof by Contradiction:

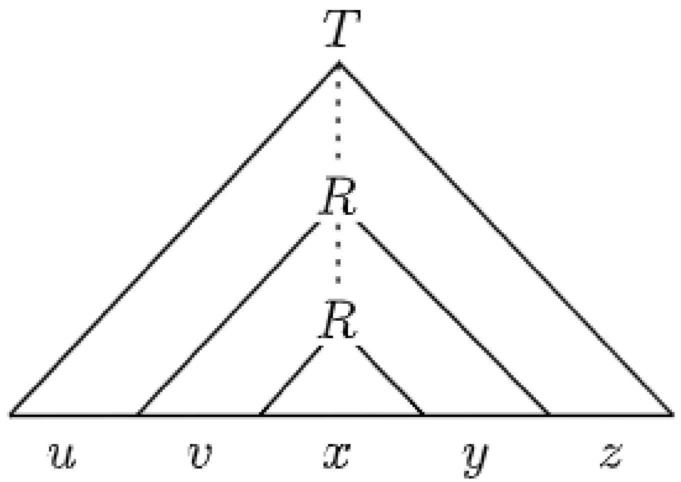
- We assume that B is a CFL and obtain a contradiction.
- Let p be the pumping length for B that is guaranteed to exist by the pumping lemma.
- Select the string $s = a^p b^p c^p$.
- Clearly s is a member of B and of length at least p.
- The **pumping lemma** states that s can be **pumped**,
  - but we show that it **cannot**.

- In other words, we show that **no matter how we divide s** into $uvxyz$,
  - **one of the three conditions** of the lemma is violated

Proof by Contradiction: (Cont.)

- First, **condition 2** stipulates that either v or y is nonempty. $(|vy| > 0)$
- Then we consider one of two cases:
  1. **When both v and y contain only one type of alphabet symbol**
  2. **When either v or y contains more than one type of symbol**

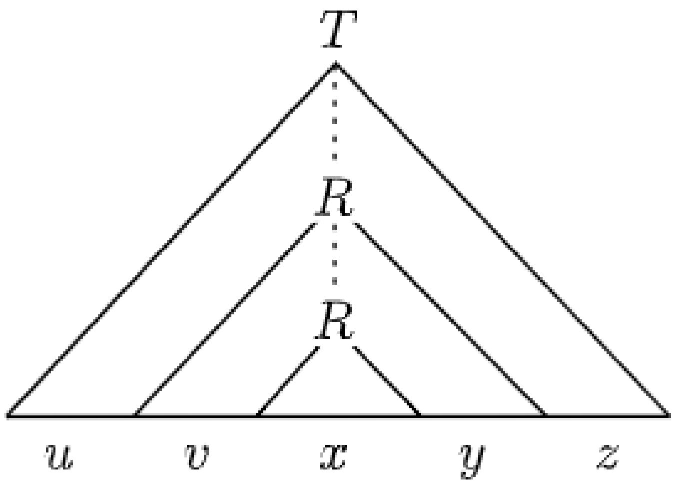
$$B = \{a^n b^n c^n | n \geq 0\}$$



Proof by Contradiction: (Cont.)

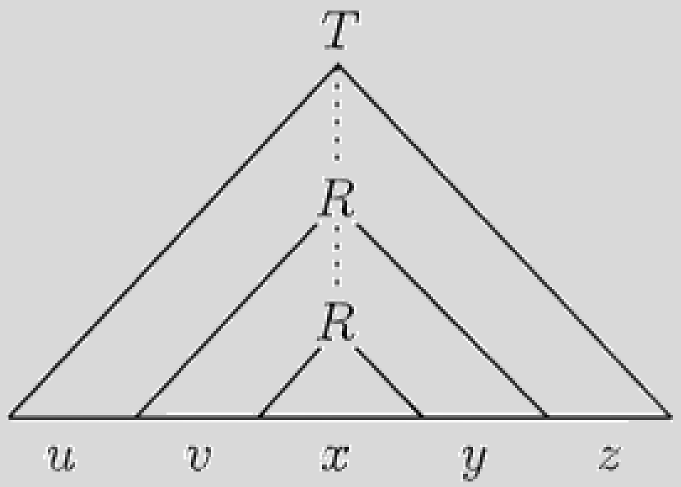
- First, **condition 2** stipulates that either v or y is nonempty. $(|vy| > 0)$
- Then we consider one of two cases:
  1. **When both v and y contain only one type of alphabet symbol,**
     - v does not contain both **a's** and **b's** or both **b's** and **c's**, and the same holds for y.

$$B = \{a^n b^n c^n | n \geq 0\}$$



Proof by Contradiction: (Cont.)

- First, **condition 2** stipulates that either v or y is nonempty. $(|vy| > 0)$
- Then we consider one of two cases:
  1. **When both v and y contain only one type of alphabet symbol,**
     - v does not contain both **a's** and **b's** or both **b's** and **c's**, and the same holds for y.
     - In this case, the string uv^2xy^2z cannot contain equal numbers of a's, b's, and c's.
     - Therefore, it cannot be a member of B.
     - That violates condition 1 of the lemma and is thus a contradiction.



$$B = \{a^n b^n c^n | n \geq 0\}$$

Proof by Contradiction: (Cont.)

- First, **condition 2** stipulates that either v or y is nonempty. $(|vy| > 0)$
- Then we consider one of two cases:

1. **When both v and y contain only one type of alphabet symbol,**
    - v does not contain both **a's** and **b's** or both **b's** and **c's**, and the same holds for y.
    - In this case, the string uv^2xy^2z cannot contain equal numbers of a's, b's, and c's.
    - Therefore, it cannot be a member of B.
    - That violates condition 1 of the lemma and is thus a contradiction.

    - Example: aaaaa bbbbb ccccc or aaaaa bbbbb ccccc

      v y v y=ε

Proof by Contradiction: (Cont.)

2. **When either v or y contains more than one type of symbol,**
   - uv^2xy^2z may contain equal numbers of the three alphabet symbols
     - but not in the correct order.
   - Hence it cannot be a member of B and a contradiction occurs.

   - Example: aaaa**ab**bbbb**b**ccccc → aaaa**abab**bbbb**bb**ccccc

     v y uv^2xy^2z

Example 2:

- Use the pumping lemma to show that the language $C = \{a^i b^j c^k | 0 \leq i \leq j \leq k\}$ is not context free.

Example 2:

- Use the pumping lemma to show that the language $C = \{a^i b^j c^k | 0 \leq i \leq j \leq k\}$ is not context free.

Proof by Contradiction:

- We assume that C is a CFL and obtain a contradiction.
- Let p be the pumping length for C that is guaranteed to exist by the pumping lemma.
- Select the string $s = a^p b^p c^p$.

Example 2:

- Use the pumping lemma to show that the language $C = \{a^i b^j c^k | 0 \leq i \leq j \leq k\}$ is not context free.

Proof by Contradiction:

- We assume that C is a CFL and obtain a contradiction.
- Let p be the pumping length for C that is guaranteed to exist by the pumping lemma.
- Select the string $s = a^p b^p c^p$.
- Clearly s is a member of C and of length at least p.
- The **pumping lemma** states that s can be **pumped**,
  - but we show that it **cannot**.

- In other words, we show that **no matter how we divide s** into $uvxyz$,
  - **one of the three conditions** of the lemma is violated

Proof by Contradiction: (Cont.)

- Let s = $uvxyz$.
- Then we consider two cases:
  1. **When both v and y contain only one type of alphabet symbol,**
  2. **When either v or y contains more than one type of symbol**

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

1. **When both v and y contain only one type of alphabet symbol,**
   - v **does not contain** both a's and b's or both b's and c's, and the same holds for y.
   - Because v and y contain only one type of alphabet symbol, one of the symbols a, b, or c **doesn't appear** in v or y.
   - We further subdivide this case into three subcases according to **which symbol does not appear**.

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

1. **When both v and y contain only one type of alphabet symbol,**
   - v **does not contain** both a's and b's or both b's and c's, and the same holds for y.
     - **The a's do not appear.** Then we try **pumping down** to obtain the string $uv^0xy^0z = uxz$.

     - **Example:** aaaaa bbbbb ccccc → v=bb , y=cc

$$C = \{a^i b^j c^k \mid 0 \le i \le j \le k\}$$

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

1. **When both v and y contain only one type of alphabet symbol,**
   - **v does not contain** both a's and b's or both b's and c's, and the same holds for y.
     - **The a's do not appear.** Then we try pumping down to obtain the string $uv^0xy^0z = uxz$.
     - **Example:** aaaaa bbbbb ccccc → v=bb , y=cc
     - That contain s the same number of a's as s does,
       - but it contains fewer b's or fewer c's.
     - Therefore, it is not a member of C,
       - and a **contradiction** occurs.

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

1. **When both v and y contain only one type of alphabet symbol,**
    - **The b's do not appear.** Then either **a's** or **c's** must appear in v or y because both can't be the empty string.

    - **Example:** aaaaa bbbbb ccccc → v=aa , y=cc

$$C = \{a^i b^j c^k \mid 0 \le i \le j \le k\}$$

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

1. **When both v and y contain only one type of alphabet symbol,**
    - **The b's do not appear.** Then either **a's** or **c's** must appear in v or y because both can't be the empty string.

    - **Example:** aaaaa bbbbb ccccc → v=aa , y=cc

    - If **a's appear**,
        - the string uv^2xy^2z contains more a's than b's,
        - so it is **not in C**.
    - If **c's appear**,
        - the string uv^0xy^0z contains more b's than c's,
        - so it is **not in C**.
    - Either way, a **contradiction** occurs.

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

1. **When both v and y contain only one type of alphabet symbol,**
   - **The b's do not appear.** Then either **a's** or **c's** must appear in v or y because both can't be the empty string.

   - **Example:** aaaaa bbbbb ccccc → v=aa , y=cc

   - If **a's appear**,
     - the string uv^2xy^2z contains more a's than b's,
     - so it is **not in C**.

$$C = \{a^i b^j c^k \mid 0 \le i \le j \le k\}$$

Pumping Lemma for CFLs

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:
  1. **When both v and y contain only one type of alphabet symbol,**
     - **The c's do not appear.** Then the string uv^2xy^2z contains more a's or more b's than c's,
       - so it is **not in C**, and a **contradiction** occurs.

Proof by Contradiction: (Cont.)

- Let s = uvxyz.
- Then we consider two cases:

2. **When either v or y contains more than one type of symbol**
    - Then the string uv^2xy^2z will not contain the symbols in the correct order.
        - so it is **not in C**, and a **contradiction** occurs.

Example 3:

- Use the pumping lemma to show that the language $D = \{ww | w \in \{0,1\}*\}$ is not context free.

Example 3:

- Use the pumping lemma to show that the language $D = \{ww | w \in \{0,1\}*\}$ is not context free.

Proof by Contradiction:

- We assume that D is a CFL and obtain a contradiction.
- Let p be the pumping length for D that is guaranteed to exist by the pumping lemma.
- Select the string $s = 0^p 1^p 0^p 1^p$.

Example 3:

- Use the pumping lemma to show that the language $D = \{ww | w \in \{0,1\} *\}$ is not context free.

Proof by Contradiction:

- We assume that D is a CFL and obtain a contradiction.
- Let p be the pumping length for D that is guaranteed to exist by the pumping lemma.
- Select the string $s = 0^p 1^p 0^p 1^p$.

Example 3:

- Use the pumping lemma to show that the language $D = \{ww|w \in \{0,1\} *\}$ is not context free.

Proof by Contradiction:

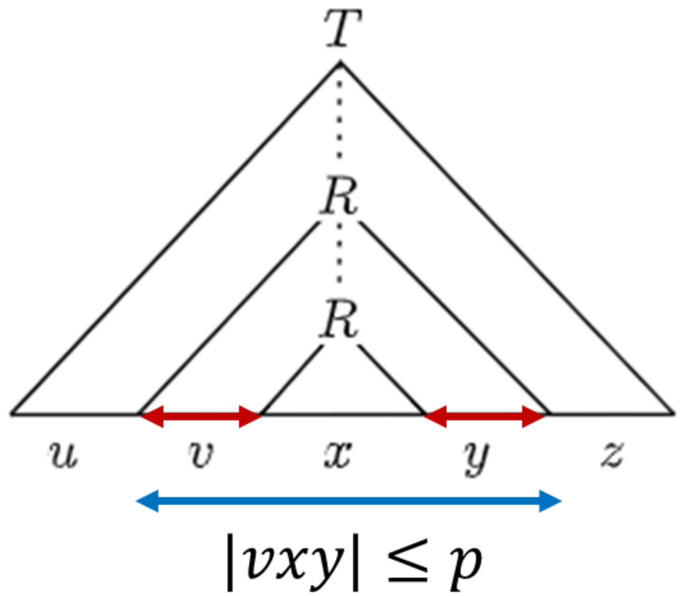
- We assume that D is a CFL and obtain a contradiction.
- Let p be the pumping length for D that is guaranteed to exist by the pumping lemma.
- Select the string $s = 0^p 1^p 0^p 1^p$.
- Clearly s is a member of D and of length at least p.
- The **pumping lemma** states that s can be **pumped**,
  - but we show that it **cannot**.

- In other words, we show that **no matter how we divide s** into $uvxyz$,
  - **one of the three conditions** of the lemma is violated

Proof by Contradiction (Cont.):

- We use **condition 3** of the pumping lemma
  - It says that we can pump s by dividing $s = uvxyz$, where $|vxy| \leq p$.

- We show that if **condition 3** holds, then the **other conditions** must fail.



Proof by Contradiction (Cont.):

- We use **condition 3** of the pumping lemma
  - It says that we can pump s by dividing $s = uvxyz$, where $|vxy| \leq p$.
- We show that if condition 3 holds, then the other conditions must fail.
- **Case 1:** the **substring occurs only in the first half** of s, pumping s up to uv^2xy^2z moves a **1** into the first position of the second half,
  - and so it cannot be of the form ww.

$$s = 0^p 1^p \,\vdots\, 0^p 1^p$$

midpoint

$$\underbrace{00000\ 11111}_{vxy}\ 00000\ 11111$$

Proof by Contradiction (Cont.):

- **Case 2:** similarly, if vxy occurs in the second half of s, pumping s up to uv^2xy^2z moves a **0** into the last position of the first half, and so it cannot be of the form ww.

$$s = 0^p 1^p \mid 0^p 1^p$$

midpoint

$$00000\ 11111\ 0\underbrace{0000\ 111}_{vxy}11$$

Proof by Contradiction (Cont.):

- **Case3:** the substring vxy straddles the midpoint of s,
  - when we try to pump s down to uxz it has the form $0^p1^i0^j1^p$, where i and j cannot both be p.
  - This string is not of the form ww.

$$s = 0^p1^p \,\vdots\, 0^p1^p$$

midpoint

$$\text{00000 111}\underbrace{\text{11 000}}_{vxy}\text{00 11111}$$

Then this string fails the conditions of pumping lemma.

Consider the following PDA:

- Prove that the following language is not context-free.

$$L = \{a^n : \ n \text{ is prime}\}$$