Context Free Grammar (CFG)

Context-Free Grammar

- Grammar G1

$$A \rightarrow 0A$$
$$A \rightarrow 1B$$
$$B \rightarrow \#$$

Context-Free Grammar (Continuation…)

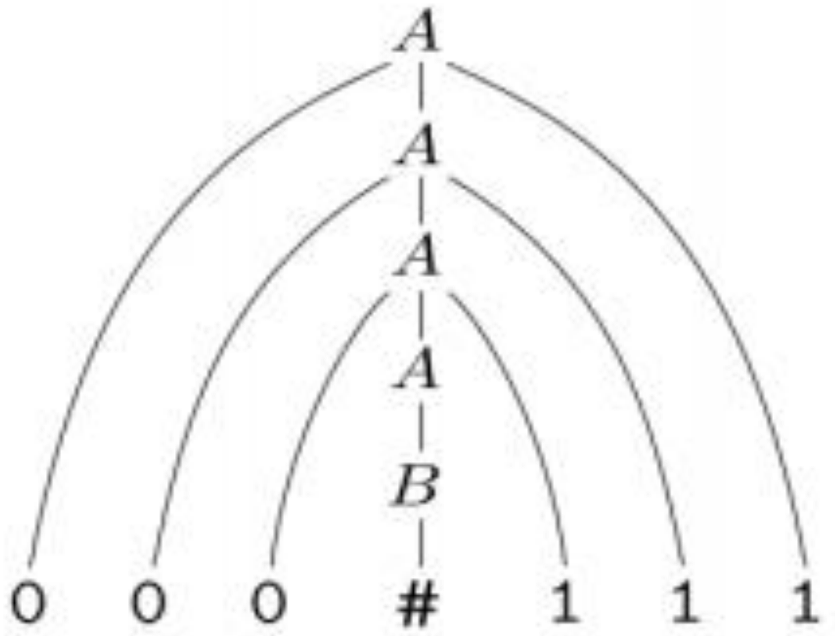


FIGURE 2.1
Parse tree for 000#111 in grammar G_1

Context-Free Grammar (Continuation…)

- Grammar G2

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle &\rightarrow \langle \text{CMPLX-NOUN} \rangle \mid \langle \text{CMPLX-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle &\rightarrow \langle \text{CMPLX-VERB} \rangle \mid \langle \text{CMPLX-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle &\rightarrow \langle \text{PREP} \rangle \langle \text{CMPLX-NOUN} \rangle \\
\langle \text{CMPLX-NOUN} \rangle &\rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CMPLX-VERB} \rangle &\rightarrow \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle &\rightarrow \texttt{a} \mid \texttt{the} \\
\langle \text{NOUN} \rangle &\rightarrow \texttt{boy} \mid \texttt{girl} \mid \texttt{flower} \\
\langle \text{VERB} \rangle &\rightarrow \texttt{touches} \mid \texttt{likes} \mid \texttt{sees} \\
\langle \text{PREP} \rangle &\rightarrow \texttt{with}
\end{aligned}
$$

Formal Definition of a Context-Free Grammar

DEFINITION 2.2

A ***context-free grammar*** is a 4-tuple (V, Σ, R, S), where

1. V is a finite set called the ***variables***,
2. Σ is a finite set, disjoint from V, called the ***terminals***,
3. R is a finite set of ***rules***, with each rule being a variable and a string of variables and terminals, and
4. $S \in V$ is the start variable.

Example 2.3

$G_3 = (\{S\}, \{a, b\}, R, S)$.

The set of rules (R), R, is $S \rightarrow aSb \mid SS \mid \epsilon$

Example 2.4

G_4 = (V, Σ, R, <EXPR>).

V is {<EXPR>, <TERM>, <FACTOR>},

and Σ is {a, +, ×, (,)}.

The rules are,

<EXPR> → <EXPR> + <TERM> | <TERM>

<TERM> → <TERM> × <FACTOR> | <FACTOR>

<FACTOR> → (<EXPR>) | a

Example 2.4 (Continuation…)

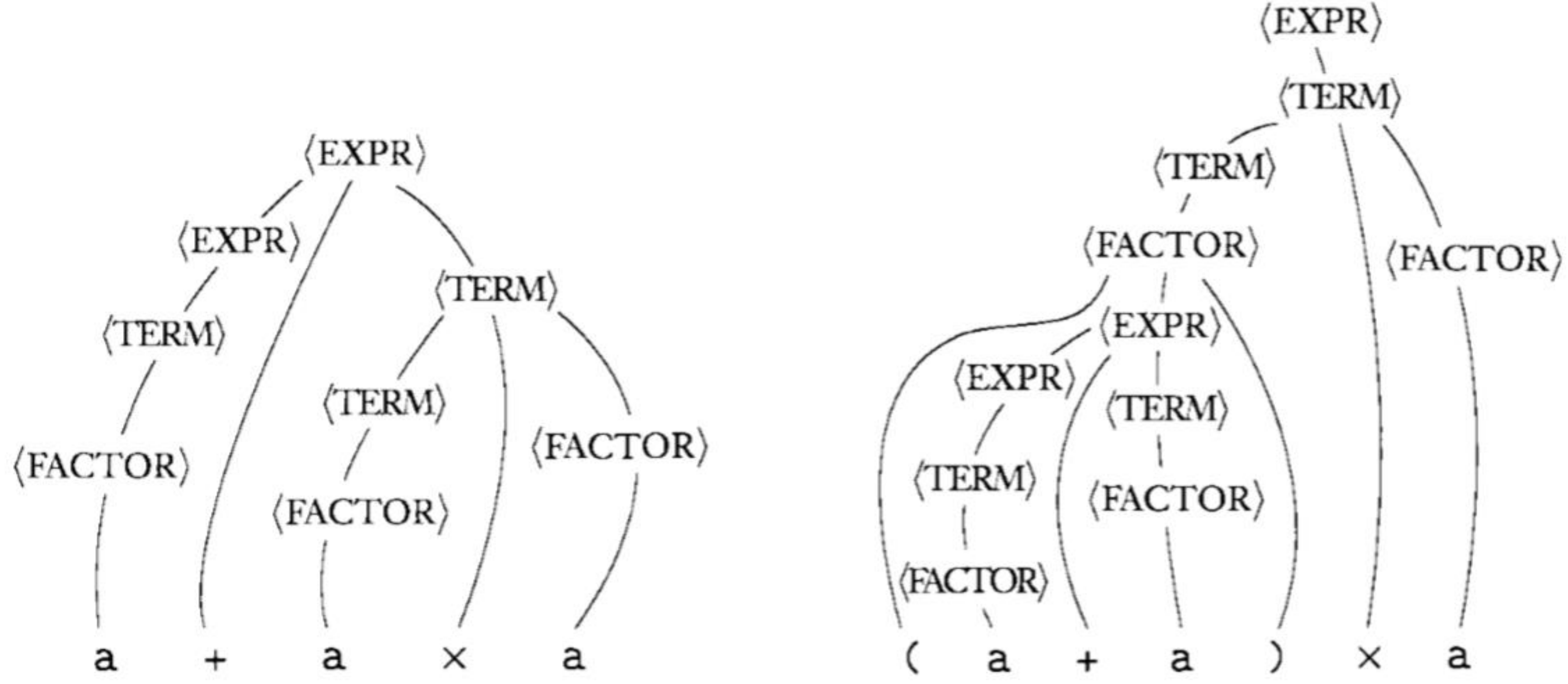


FIGURE 2.5
Parse trees for the strings a+a×a and (a+a)×a

Definitions

- Leftmost Derivation
- Rightmost Derivation
- Leftmost Derivation Parse Tree
- Rightmost Derivation Parse Tree
- Top-Down Parse Tree
- Bottom-Up Parse Tree

Designing Context-Free Grammars

L = $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

L = $\{0^n 1^{2n} \mid n \geq 0\} \cup \{1^{2n} 0^n \mid n \geq 0\}$

Designing Context-Free Grammars (Continuation…)

L(M) = {w| w contains 001 as substring}

L(M) = {w| w contains 1001 as substring}

Designing Context-Free Grammars (Continuation…)

L = $\{0^n 1^n \mid n \geq 0\}$

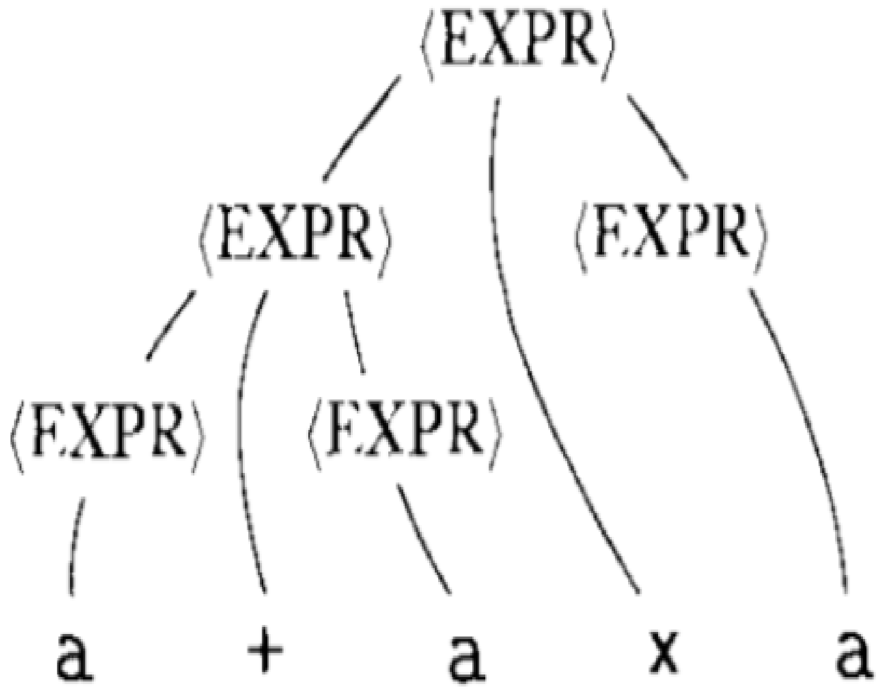
L = $\{0^n 1^{2n} \mid n \geq 0\}$

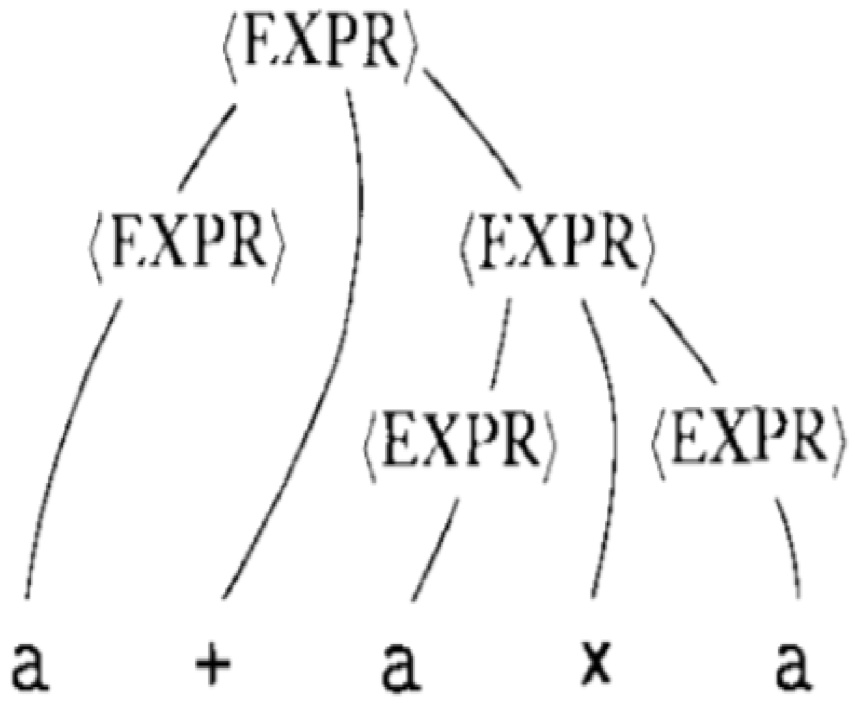
Ambiguity

Grammar, G_5.

<EXPR> → <EXPR> + <EXPR> |
<EXPR> × <EXPR> |
(<EXPR>) |
a

Ambiguity (Continuation…)





Ambiguity (Continuation…)

DEFINITION **2.7**

A string w is derived ***ambiguously*** in context-free grammar G if it has two or more different leftmost derivations. Grammar G is ***ambiguous*** if it generates some string ambiguously.

Ambiguity (Continuation…)

- Sometimes when we have an ambiguous grammar we can find an unambiguous grammar that generates the same language.
- Some context-free languages, however, can be generated only by ambiguous grammars.
- Such languages are called inherently ambiguous.

•The language **$\{a^i b^j c^k \mid i=j \text{ or } j=k \text{ where } i,j,k>=0\}$** is inherently ambiguous.

Chomsky Normal Form

DEFINITION 2.8

A context-free grammar is in ***Chomsky normal form*** if every rule is of the form

$$A \to BC$$
$$A \to a$$

where a is any terminal and A, B, and C are any variables—except that B and C may not be the start variable. In addition we permit the rule $S \to \varepsilon$, where S is the start variable.

Chomsky Normal Form (Continuation…)

THEOREM 2.9

Any context-free language is generated by a context-free grammar in Chomsky normal form.

Proof Idea:

- Converting any CFG G into Chomsky Normal Form (CNF) by following some steps.
- Rules that violate the conditions are replaced with equivalent ones that are satisfactory.
- In all steps we patch up the grammar to be sure that it still generates the same language

Chomsky Normal Form (Continuation…)

- A context free grammar (CGF) is in Chomsky Normal Form (CNF) if all production rules satisfy one of the following conditions:
  - A non-terminal generating a terminal (e.g.; A->a)
  - A non-terminal generating two non-terminals (e.g.; A->BC)
  - Start variable generating ε. (e.g.; S-> ε)

Chomsky Normal Form (Continuation…)

- **Step-01:** Eliminate start variable from RHS of a rule.
  - First we add a new start variable $\mathbf{S_0}$
  - Then we add the Rule $\mathbf{S_0 \rightarrow S}$ where S is the original start variable

Chomsky Normal Form (Continuation…)

- **Step-02:** Eliminate null production rule from all variables except start variables.
  - We remove all **A -> ε** rules where **A** is not the start variable
  - Then for each occurrence of an A on the RHS of a rule, add new rule with that occurrence deleted.
    - For example: if **R -> uAv**, add new rule **R-> uv**
    - For example: if **R -> uAvAw**, add new rule **R-> uvAw, R-> uAvw** and **R -> uvw**

Chomsky Normal Form (Continuation…)

- **Step-03:** Eliminate all unit production rules.
  - We remove a unit rule **A -> B**
  - Then whenever a rule **B -> u** appears where **u** is a string of variables and terminals, we add the rule **A -> u** unless this was a unit rule previously removed
  - Repeat this steps until we removed all unit rules

Chomsky Normal Form (Continuation…)

- **Step-04:** Finally we convert all remaining rules into the proper form
  - Replace each rule $\mathbf{A \rightarrow u_1u_2u_3\ldots u_k}$ where $\mathbf{k \geq 3}$ and each $\mathbf{u_i}$ is a variable or terminal symbol with the rules $\mathbf{A \rightarrow u_1A_1}$, $\mathbf{A_1 \rightarrow u_2A_2}$, $\mathbf{A_2 \rightarrow u_3A_3}$ **and** $\mathbf{A_{k-2} \rightarrow u_{k-1}u_k}$ where $\mathbf{A_i}$ are new variables
  - If $\mathbf{k=2}$ we replace any terminal $\mathbf{u_i}$ in the preceding rule with the new variable $\mathbf{U_i}$ and add the rule $\mathbf{U_i} \rightarrow \mathbf{u_i}$

Chomsky Normal Form (Continuation…)

- **Example 2.10:** Convert **CFG G_6** into equivalent **CNF**

$$
\begin{aligned}
S &\rightarrow ASA \mid \mathtt{a}B \\
A &\rightarrow B \mid S \\
B &\rightarrow \mathtt{b} \mid \varepsilon
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- After applying **Step-01** we get,

$$
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid aB \\
A &\rightarrow B \mid S \\
B &\rightarrow b \mid \varepsilon
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- Removing rules **B -> ε** and adding rules **A -> ε** by following **Step-02** we get,

$$
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid aB \mid \mathbf{a} \\
A &\rightarrow B \mid S \mid \boldsymbol{\varepsilon} \\
B &\rightarrow b
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- Removing rules **A -> ε** by repeating **Step-02** we get,

$$
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid \mathrm{a}B \mid \mathrm{a} \mid \boldsymbol{SA} \mid \boldsymbol{AS} \mid \boldsymbol{S} \\
A &\rightarrow B \mid S \\
B &\rightarrow \mathrm{b}
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- Removing unit rule **S -> S** by following **Step-03** we get,

$$
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
A &\rightarrow B \mid S \\
B &\rightarrow \mathtt{b}
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- Removing unit rule $\mathbf{S_0 \text{ -> } S}$ and adding **RHS rules of S** as **RHS rules of S_0** by following **Step-03** we get,

$$
\begin{aligned}
S_0 &\rightarrow \boldsymbol{ASA} \mid \mathbf{a}\boldsymbol{B} \mid \mathbf{a} \mid \boldsymbol{SA} \mid \boldsymbol{AS} \\
S &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
A &\rightarrow B \mid S \\
B &\rightarrow \mathtt{b}
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- Removing unit rule **A -> B** and adding **RHS rules of B** as **RHS rules of A** by following **Step-03** we get,

$$
\begin{aligned}
S_0 &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
S &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
A &\rightarrow S \mid \mathbf{b} \\
B &\rightarrow \mathtt{b}
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- Removing unit rule **A -> S** and adding **RHS rules of S** as **RHS rules of A** by following **Step-03** we get,

$$
\begin{aligned}
S_0 &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
S &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
A &\rightarrow \mathtt{b} \mid ASA \mid \mathtt{a}B \mid \mathtt{a} \mid SA \mid AS \\
B &\rightarrow \mathtt{b}
\end{aligned}
$$

Chomsky Normal Form (Continuation…)

- By following **Step-04** we replace
  - **S_0 -> ASA by S_0 -> AA_1** and **A_1 -> SA**
  - **S-> ASA by S -> AA_1** and **A_1 -> SA**
  - **A-> ASA by A -> AA_1** and **A_1 -> SA**
  - **S_0 -> aB by S_0 -> UB** and **U -> a**
  - **S-> aB by S -> UB** and **U -> a**
  - **A-> aB by A -> UB** and **U -> a**

$$
\begin{aligned}
S_0 &\rightarrow AA_1 \mid UB \mid \mathtt{a} \mid SA \mid AS \\
S &\rightarrow AA_1 \mid UB \mid \mathtt{a} \mid SA \mid AS \\
A &\rightarrow \mathtt{b} \mid AA_1 \mid UB \mid \mathtt{a} \mid SA \mid AS \\
A_1 &\rightarrow SA \\
U &\rightarrow \mathtt{a} \\
B &\rightarrow \mathtt{b}
\end{aligned}
$$

Pushdown Automata

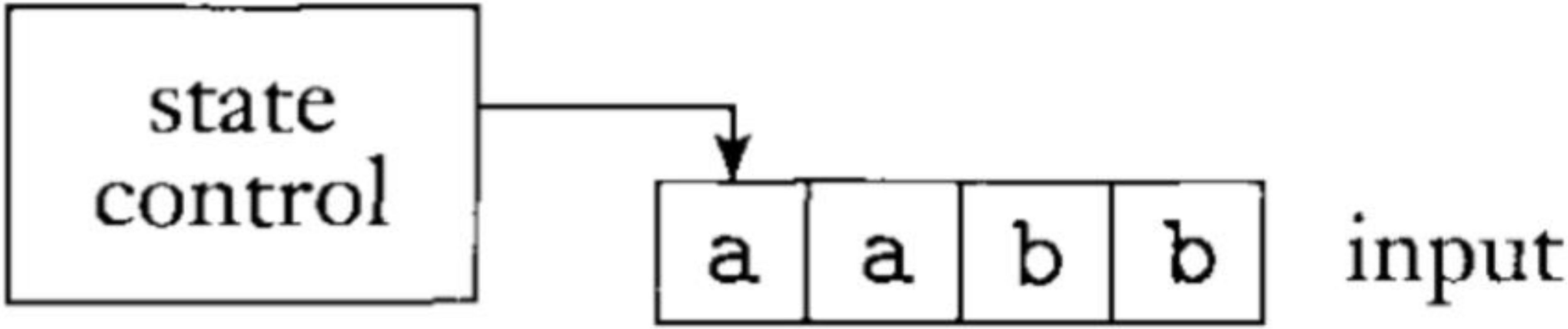


FIGURE 2.11
Schematic of a finite automaton

Pushdown Automata (Continuation…)

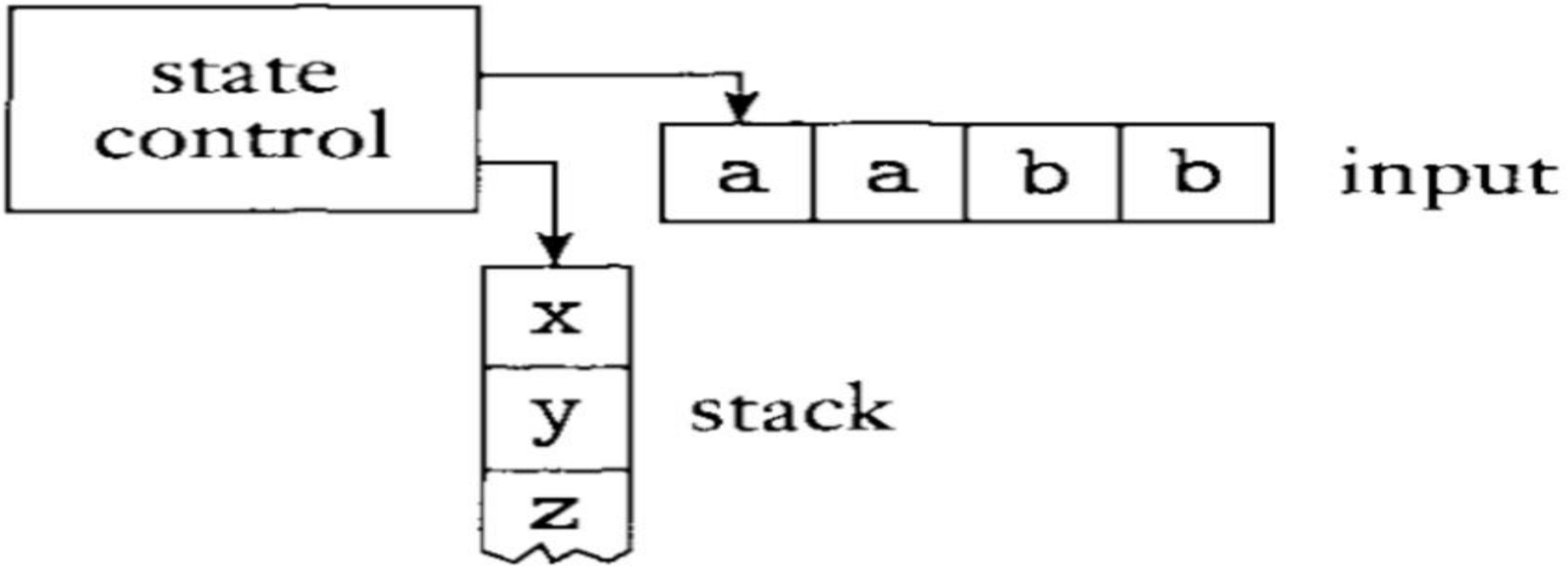


FIGURE 2.12
Schematic of a pushdown automaton

Formal Definition of a Pushdown Automaton

DEFINITION 2.13

A *pushdown automaton* is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, Γ, and F are all finite sets, and

1. Q is the set of states,
2. Σ is the input alphabet,
3. Γ is the stack alphabet,
4. $\delta: Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

Examples of Pushdown Automaton

EXAMPLE 2.14

The following is the formal description of the PDA (page 110) that recognizes the language $\{0^n 1^n | \, n \geq 0\}$. Let M_1 be $(Q, \Sigma, \Gamma, \delta, q_1, F)$, where

$Q = \{q_1, q_2, q_3, q_4\}$,

$\Sigma = \{0,1\}$,

$\Gamma = \{0, \$\}$,

$F = \{q_1, q_4\}$, and

δ is given by the following table, wherein blank entries signify $\emptyset$.

Input:	0			1			ε		
Stack:	0	\$	ε	0	\$	ε	0	\$	ε
q_1									$\{(q_2, \$)\}$
q_2			$\{(q_2, 0)\}$	$\{(q_3, \varepsilon)\}$					
q_3				$\{(q_3, \varepsilon)\}$				$\{(q_4, \varepsilon)\}$	
q_4									

Examples of Pushdown Automaton (Continuation…)

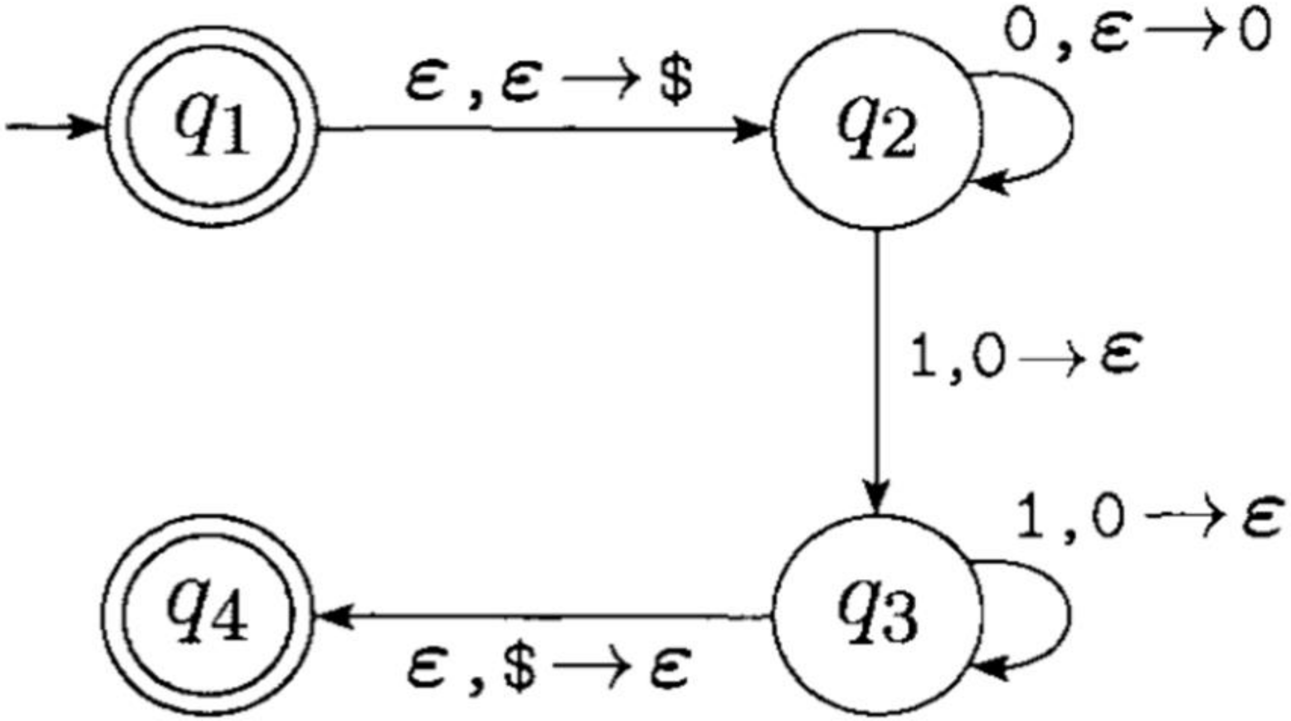


FIGURE 2.15
State diagram for the PDA M_1 that recognizes $\{0^n 1^n \mid n \geq 0\}$

Examples of Pushdown Automaton (Continuation…)

EXAMPLE 2.16

This example illustrates a pushdown automaton that recognizes the language

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

Examples of Pushdown Automaton (Continuation)

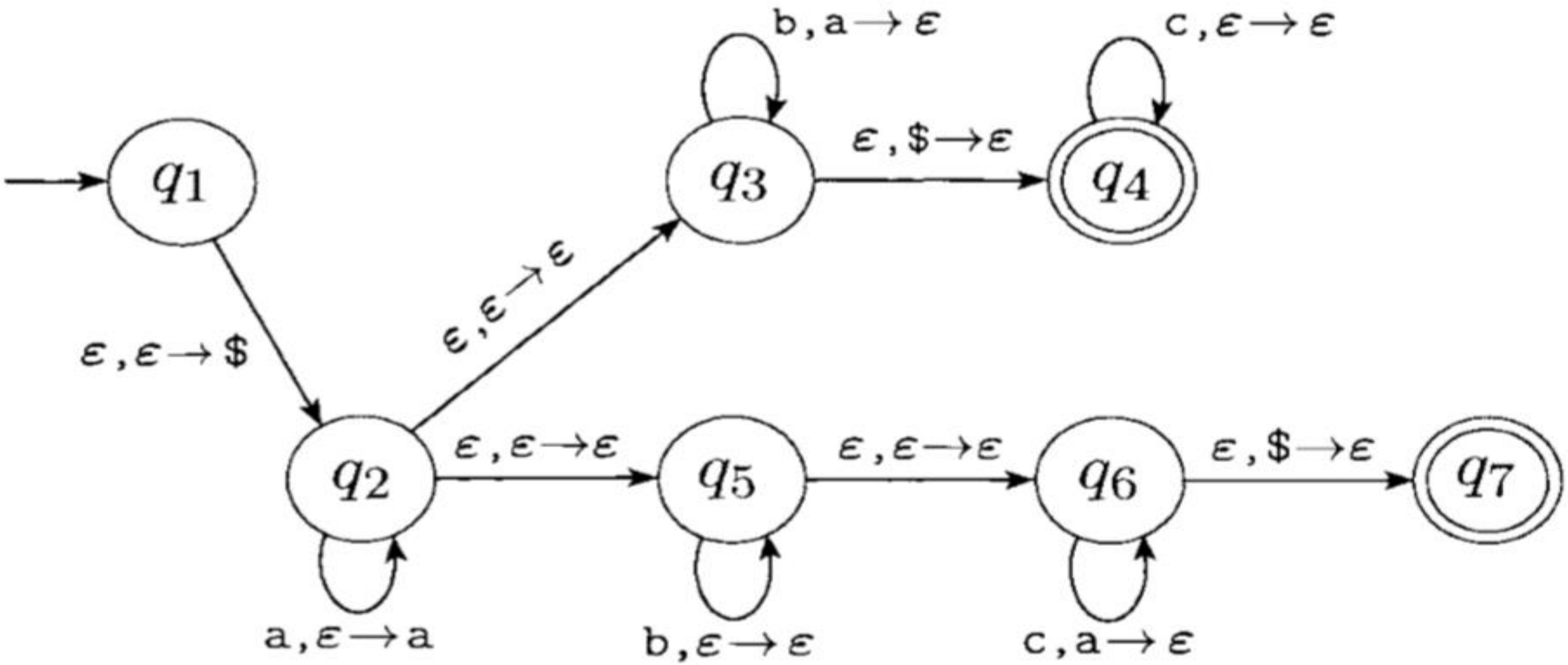


FIGURE 2.17
State diagram for PDA M_2 that recognizes
$\{a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i = j \text{ or } i = k\}$

Examples of Pushdown Automaton (Continuation…)

EXAMPLE 2.18

In this example we give a PDA M_3 recognizing the language $\{ww^{\mathcal{R}} | w \in \{0,1\}^*\}$.

Examples of Pushdown Automaton (Continuation…)

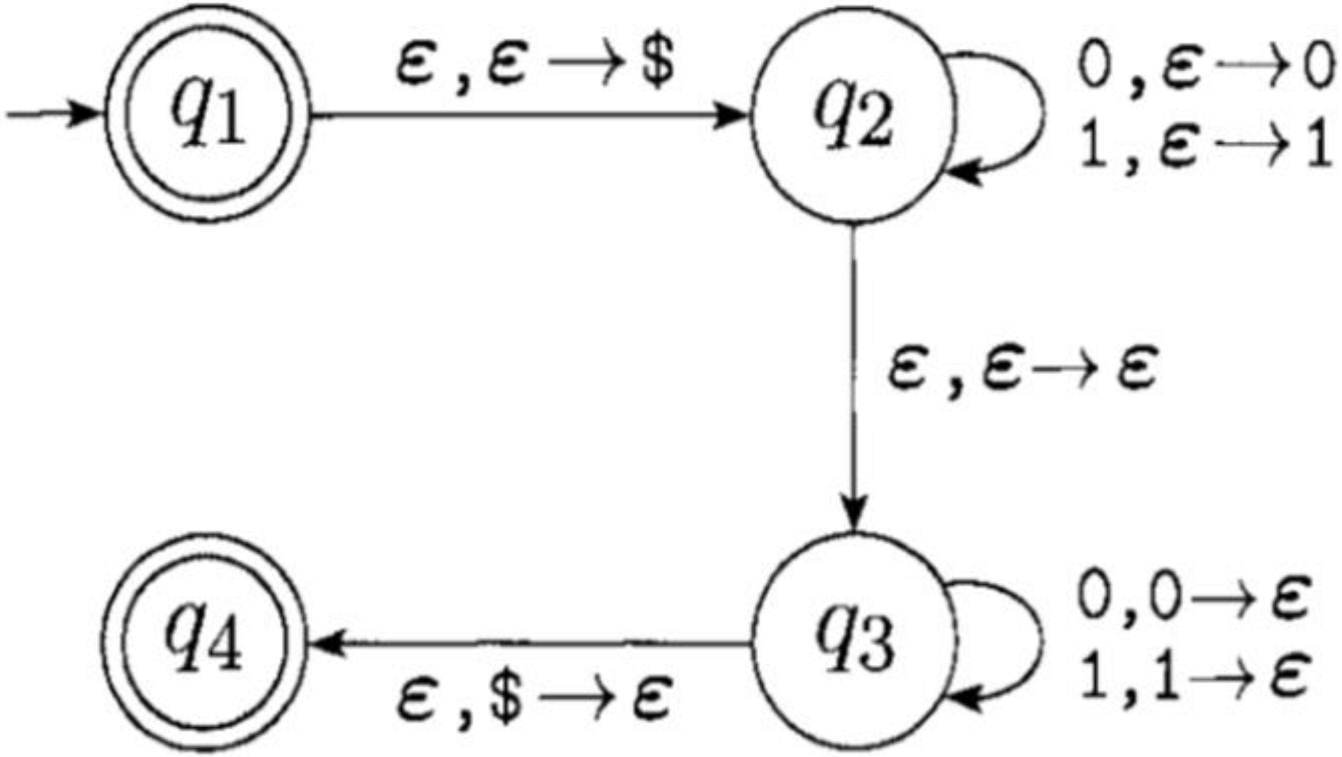


FIGURE 2.19
State diagram for the PDA M_3 that recognizes $\{ww^{\mathcal{R}} | \, w \in \{0,1\}^*\}$

Practice Problems of Pushdown Automaton

- Design a NPDA for the following languages:
  - $\{0^n1^{2n}$ | where n>=1}
  - $\{0^{2n}1^n$ | where n>=1}
  - $\{a^nb^mc^r$ | where m,n,r>=0 and r=n+m}
  - $\{a^nb^mc^r$ | where m,n,r>=0 and r=n-m}


Thank you