T065001: Introduction to Formal Languages

Lecture 6: Pushdown automata, context-free languages, grammars (1)

Chapter 2.1 in Sipser's textbook

2025-05-26

(Lecture slides by Yih-Kuen Tsay)

Introduction

- We have seen languages that cannot be described by any regular expression (or recognized by any finite automaton).
- *Context-free grammars* are a more powerful method for describing languages; they were first used in the study of natural languages.
- They play an important role in the specification and compilation of programming languages.
- The collection of languages associated with context-free grammars are called the *context-free languages* (CFLs).

Context-Free Grammars

- A *context-free grammar* (CFG) consists of a collection of *substitution rules* (or *productions*) such as:

$$\begin{array}{lcl} A & \to & 0A1 \\ A & \to & B \\ B & \to & \# \end{array} \quad \text{or alternatively} \quad \begin{array}{lcl} A & \to & 0A1 \mid B \\ B & \to & \# \end{array}$$

- Symbols A and B here are called *variables*; the other symbols 0, 1, and # are called *terminals*.

Context-Free Grammars

- A *context-free grammar* (CFG) consists of a collection of *substitution rules* (or *productions*) such as:

$$\begin{array}{lcl} A & \to & 0A1 \\ A & \to & B \\ B & \to & \# \end{array} \quad \text{or alternatively} \quad \begin{array}{lcl} A & \to & 0A1 \mid B \\ B & \to & \# \end{array}$$

- Symbols A and B here are called *variables*; the other symbols 0, 1, and # are called *terminals*.
- A grammar describes a language by *generating* each string of the language through a *derivation*.
  For example, the above grammar generates the string 000#111:
  $A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$.

Definition of a CFG

Definition (2.2)

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

1. V is a finite set of *variables*.
2. Σ ($\Sigma \cap V = \emptyset$) is a finite set of *terminals*.
3. R is a finite set of *rules*, each of the form $A \to w$, where $A \in V$ and $w \in (V \cup \Sigma)^*$.
4. $S \in V$ is the *start* symbol.

Definition of a CFG

Definition (2.2)

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

1. V is a finite set of *variables*.
2. Σ $(\Sigma \cap V = \emptyset)$ is a finite set of *terminals*.
3. R is a finite set of *rules*, each of the form $A \to w$, where $A \in V$ and $w \in (V \cup \Sigma)^*$.
4. $S \in V$ is the *start* symbol.

- If $A \to w$ is a rule, then uAv *yields* uwv, written as $uAv \Rightarrow uwv$.
- We write $u \overset{*}{\Rightarrow} v$ if $u = v$ or a sequence $u_1, u_2, \ldots, u_k$ $(k \geq 0)$ exists such that $u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \ldots \Rightarrow u_k \Rightarrow v$.

Definition of a CFG (cont.)

Notation: Let $G = (V, \Sigma, R, S)$ be a context-free grammar. Then $L(G) = \{w \in \Sigma^* \mid S \stackrel{*}{\Rightarrow} w\}$ = the set of all strings that G can generate.

Definition of a CFG (cont.)

Notation: Let $G = (V, \Sigma, R, S)$ be a context-free grammar. Then $L(G) = \{w \in \Sigma^* \mid S \stackrel{*}{\Rightarrow} w\}$ = the set of all strings that G can generate.

Remark: A grammar is often specified by writing down its rules only.

One can identify the variables as those symbols that appear on the left-hand side of the rules, and the terminals as the remaining symbols.

Furthermore, by convention, the start variable is the variable that appears on the left-hand side of the first rule.

Definition of a CFG (cont.)

Notation: Let $G = (V, \Sigma, R, S)$ be a context-free grammar. Then $L(G) = \{w \in \Sigma^* \mid S \stackrel{*}{\Rightarrow} w\}$ = the set of all strings that G can generate.

Remark: A grammar is often specified by writing down its rules only.

One can identify the variables as those symbols that appear on the left-hand side of the rules, and the terminals as the remaining symbols.

Furthermore, by convention, the start variable is the variable that appears on the left-hand side of the first rule.

Example: Let G_1 be the CFG from before:

$$A \to 0A1 \mid B$$
$$B \to \#$$

Here, $L(G_1) =$?

Definition of a CFG (cont.)

Notation: Let $G = (V, \Sigma, R, S)$ be a context-free grammar. Then $L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}$ = the set of all strings that G can generate.

Remark: A grammar is often specified by writing down its rules only.

One can identify the variables as those symbols that appear on the left-hand side of the rules, and the terminals as the remaining symbols.

Furthermore, by convention, the start variable is the variable that appears on the left-hand side of the first rule.

Example: Let G_1 be the CFG from before: $A \to 0A1 \mid B$
$B \to \#$

Here, $L(G_1) = \{0^n \# 1^n \mid n \geq 0\}$.

Definition of a CFG (cont.)

Notation: Let $G = (V, \Sigma, R, S)$ be a context-free grammar. Then $L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}$ = the set of all strings that G can generate.

Remark: A grammar is often specified by writing down its rules only.

One can identify the variables as those symbols that appear on the left-hand side of the rules, and the terminals as the remaining symbols.

Furthermore, by convention, the start variable is the variable that appears on the left-hand side of the first rule.

Example: Let G_1 be the CFG from before:

$$A \to 0A1 \mid B$$
$$B \to \#$$

Here, $L(G_1) = \{0^n \# 1^n \mid n \geq 0\}$.

Definition: A language X is called a context-free language (CFL) if there exists some context-free grammar G such that $L(G) = X$.

Definition of a CFG (cont.)

Example: Let G_1 be the CFG from before: $A \to 0A1 \mid B$
$B \to \#$

The derivation $A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$ of the string 000#111 can be represented pictorially as a parse tree:

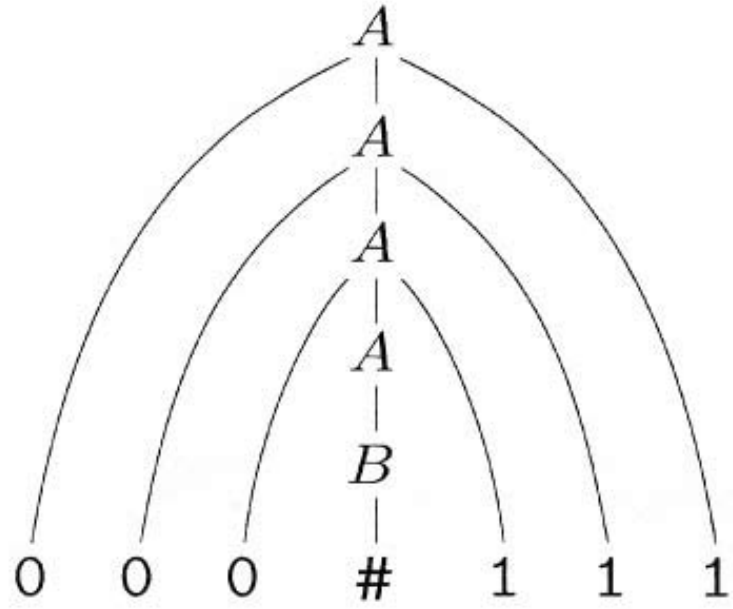

FIGURE **2.1**
Parse tree for 000#111 in grammar G_1

Example CFGs

Define G_2 as the following simplified model of the English language:

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

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$\langle \text{SENTENCE} \rangle \Rightarrow$$

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$\langle \text{SENTENCE} \rangle \Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle$$

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$\begin{aligned} \langle \text{SENTENCE} \rangle &\Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\ &\Rightarrow \langle \text{CMPLX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \end{aligned}$$

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$\begin{aligned}
\langle \text{SENTENCE} \rangle &\Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{CMPLX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle
\end{aligned}$$

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{CMPLX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{the}\ \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle
\end{aligned}
$$

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{CMPLX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{the}\ \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{the boy}\ \langle \text{VERB-PHRASE} \rangle
\end{aligned}
$$

Example CFGs (cont.)

Certain English sentences can be generated by G_2:

$$
\begin{aligned}
\langle \text{SENTENCE} \rangle &\Rightarrow \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{CMPLX-NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{the}\ \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{the boy}\ \langle \text{VERB-PHRASE} \rangle \\
&\Rightarrow \texttt{the boy}\ \langle \text{CMPLX-VERB} \rangle \\
&\Rightarrow \texttt{the boy}\ \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
&\Rightarrow \texttt{the boy sees}\ \langle \text{NOUN-PHRASE} \rangle \\
&\Rightarrow \texttt{the boy sees}\ \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
&\Rightarrow \texttt{the boy sees a}\ \langle \text{NOUN} \rangle \\
&\Rightarrow \texttt{the boy sees a flower}
\end{aligned}
$$

Example CFGs (cont.)

- $G_3 = (\{S\}, \{(,)\}, R, S)$, where R is defined by $S \to (S) \,|\, SS \,|\, \varepsilon$.
  $L(G_3) = ?$

Example CFGs (cont.)

- $G_3 = (\{S\}, \{(,)\}, R, S)$, where R is defined by $S \to (S) \mid SS \mid \varepsilon$.

  $L(G_3)$ is the language of all strings of properly nested parentheses such as ( ) (( )).

Example CFGs (cont.)

- $G_3 = (\{S\}, \{(,)\}, R, S)$, where R is defined by $S \to (S) \mid SS \mid \varepsilon$.

  $L(G_3)$ is the language of all strings of properly nested parentheses such as ( ) (( )).

Proof: Let C be the set of all strings of properly nested parentheses.

Example CFGs (cont.)

- $G_3 = (\{S\}, \{(,)\}, R, S)$, where R is defined by $S \to (S) \mid SS \mid \varepsilon$.

  $L(G_3)$ is the language of all strings of properly nested parentheses such as ( ) (( )).

Proof: Let C be the set of all strings of properly nested parentheses.

$C \subseteq L(G_3)$: Use induction on the string length.

Base case: OK because ε is generated by $S \to \varepsilon$.

General case: Let s be any string in C of length $i > 0$.
[Induction hypothesis: $S \stackrel{*}{\Rightarrow} u$ for every $u \in C$ with $|u| < i$]
Write $s = (u)v$ for possibly empty $u, v \in C$ with $|u| < i$, $|v| < i$. By the induction hypothesis, $S \stackrel{*}{\Rightarrow} u$ and $S \stackrel{*}{\Rightarrow} v$.
We have $S \Rightarrow SS \Rightarrow (S)S \stackrel{*}{\Rightarrow} (u)v = s$. Thus, $s \in L(G_3)$.

Example CFGs (cont.)

- $G_3 = (\{S\}, \{(,)\}, R, S)$, where R is defined by $S \to (S) \mid SS \mid \varepsilon$.

  $L(G_3)$ is the language of all strings of properly nested parentheses such as ( ) (( )).

Proof: Let C be the set of all strings of properly nested parentheses.

$C \subseteq L(G_3)$: Use induction on the string length.

Base case: OK because ε is generated by $S \to \varepsilon$.

General case: Let s be any string in C of length $i > 0$.
[Induction hypothesis: $S \stackrel{*}{\Rightarrow} u$ for every $u \in C$ with $|u| < i$]
Write $s = (u)v$ for possibly empty $u, v \in C$ with $|u| < i$, $|v| < i$. By the induction hypothesis, $S \stackrel{*}{\Rightarrow} u$ and $S \stackrel{*}{\Rightarrow} v$.
We have $S \Rightarrow SS \Rightarrow (S)S \stackrel{*}{\Rightarrow} (u)v = s$. Thus, $s \in L(G_3)$.

$C \supseteq L(G_3)$: Any string generated by G_3 belongs to C due to $S \to (S)$ being the only substitution rule that introduces terminals.

Example CFGs (cont.)

- $G_4 = \big(\{\langle \text{EXPR} \rangle\}, \{\text{a}, +, \times, (,)\}, R, \langle \text{EXPR} \rangle\big)$, where R is:

$$\begin{array}{rcl} \langle \text{EXPR} \rangle & \rightarrow & \langle \text{EXPR} \rangle + \langle \text{EXPR} \rangle \mid \\ & & \langle \text{EXPR} \rangle \times \langle \text{EXPR} \rangle \mid \\ & & (\langle \text{EXPR} \rangle) \mid \text{a} \end{array}$$

$L(G_4) = ?$

Example CFGs (cont.)

- $G_4 = \big(\{\langle \text{EXPR} \rangle\}, \{\text{a}, +, \times, (,)\}, R, \langle \text{EXPR} \rangle\big)$, where R is:

$$\begin{array}{rcl} \langle \text{EXPR} \rangle & \rightarrow & \langle \text{EXPR} \rangle + \langle \text{EXPR} \rangle \mid \\ & & \langle \text{EXPR} \rangle \times \langle \text{EXPR} \rangle \mid \\ & & (\langle \text{EXPR} \rangle) \mid \text{a} \end{array}$$

$L(G_4)$ is the language of algebraic expressions with the operations $+$ and $\times$ and a constant a such as $(a + a) \times a$.

Example CFGs (cont.)

- $G_4 = (\{\langle \text{EXPR} \rangle\}, \{\text{a}, +, \times, (,)\}, R, \langle \text{EXPR} \rangle)$, where R is:

$$\begin{aligned} \langle \text{EXPR} \rangle \quad \rightarrow \quad & \langle \text{EXPR} \rangle + \langle \text{EXPR} \rangle \mid \\ & \langle \text{EXPR} \rangle \times \langle \text{EXPR} \rangle \mid \\ & (\langle \text{EXPR} \rangle) \mid \text{a} \end{aligned}$$

  $L(G_4)$ is the language of algebraic expressions with the operations $+$ and $\times$ and a constant a such as $(a + a) \times a$.

- G_4 generates the string $\text{a} + \text{a} \times \text{a}$ in two different ways. (Bad.)

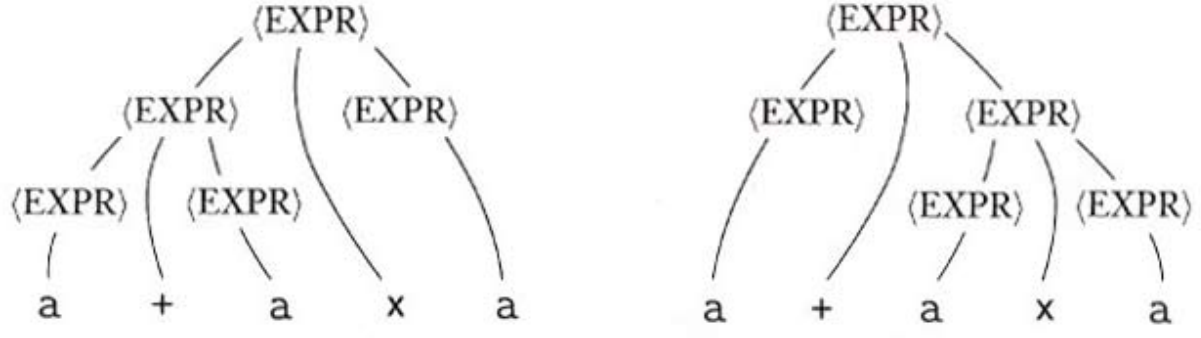

FIGURE **2.6**

Ambiguity

- A derivation of a string in a grammar is a *leftmost derivation* if at every step the leftmost remaining variable is the one replaced.
- A parse tree represents one unique leftmost derivation.

Definition (2.7)

A string is derived *ambiguously* in a grammar if it has two or more different leftmost derivations (or parse trees). A grammar is *ambiguous* if it generates some string ambiguously.

Ambiguity

- A derivation of a string in a grammar is a *leftmost derivation* if at every step the leftmost remaining variable is the one replaced.
- A parse tree represents one unique leftmost derivation.

Definition (2.7)

A string is derived *ambiguously* in a grammar if it has two or more different leftmost derivations (or parse trees). A grammar is *ambiguous* if it generates some string ambiguously.

E.g., the grammar G_4 in the example above is ambiguous. However, G_4 has an equivalent unambiguous grammar...See the next slide.

Ambiguity

- A derivation of a string in a grammar is a *leftmost derivation* if at every step the leftmost remaining variable is the one replaced.
- A parse tree represents one unique leftmost derivation.

Definition (2.7)

A string is derived *ambiguously* in a grammar if it has two or more different leftmost derivations (or parse trees). A grammar is *ambiguous* if it generates some string ambiguously.

E.g., the grammar G_4 in the example above is ambiguous. However, G_4 has an equivalent unambiguous grammar...See the next slide.

Remark: Certain languages such as $\{a^i b^j c^k \mid i = j \text{ or } j = k\}$ are inherently ambiguous and only admit ambiguous grammars.

For practical applications like programming languages where a program must have a unique interpretation, ambiguous grammars are undesirable.

Example CFGs (cont.)

An unambiguous grammar G_5 with $L(G_5) = L(G_4)$ for G_4 defined before:

$$
\begin{array}{rrcl}
G_5: & \langle \text{EXPR} \rangle & \rightarrow & \langle \text{EXPR} \rangle + \langle \text{TERM} \rangle \mid \langle \text{TERM} \rangle \\
 & \langle \text{TERM} \rangle & \rightarrow & \langle \text{TERM} \rangle \times \langle \text{FACTOR} \rangle \mid \langle \text{FACTOR} \rangle \\
 & \langle \text{FACTOR} \rangle & \rightarrow & (\langle \text{EXPR} \rangle) \mid \mathtt{a}
\end{array}
$$

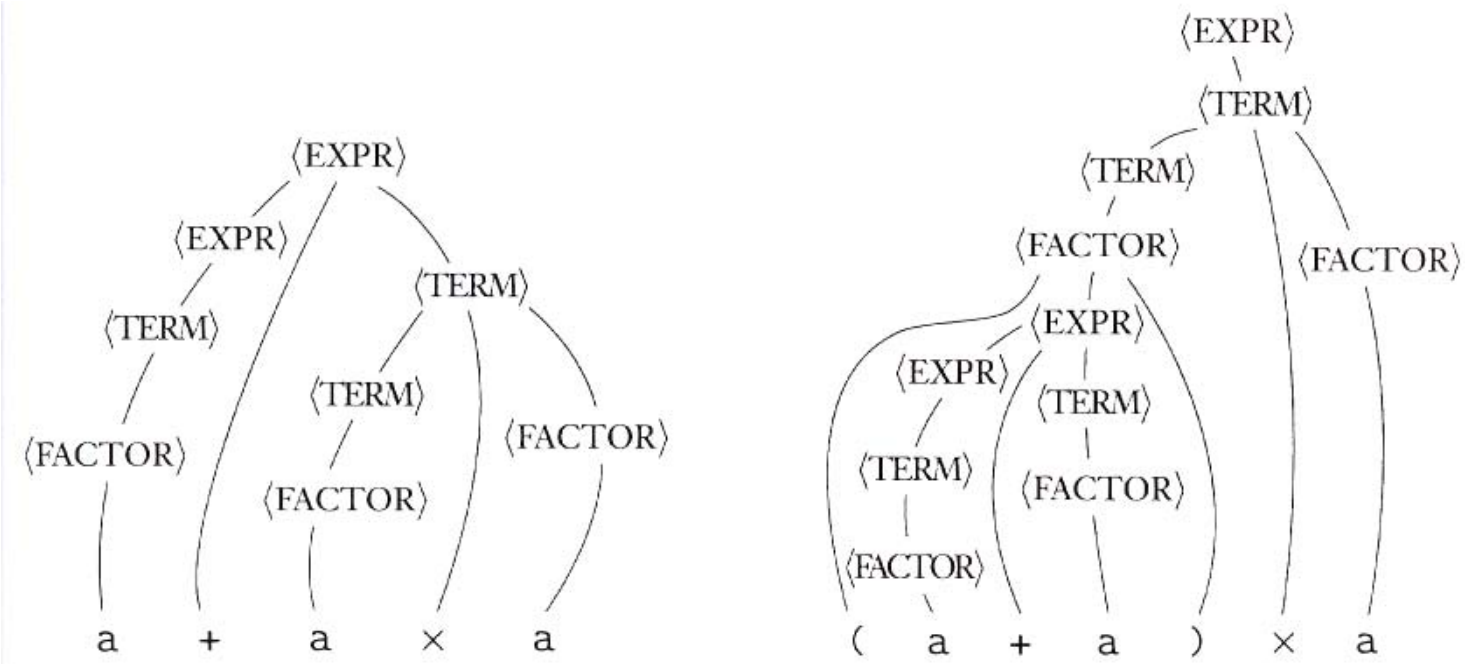

FIGURE **2.5**
Parse trees for the strings a+a×a and (a+a)×a

Designing CFGs

Example: To design a grammar for $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$, break it into smaller pieces as follows.

Designing CFGs

Example: To design a grammar for $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$, break it into smaller pieces as follows.

1. Construct a grammar for $\{0^n 1^n \mid n \geq 0\}$: $\quad S_1 \to 0\, S_1\, 1 \mid \varepsilon$

Designing CFGs

Example: To design a grammar for $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$, break it into smaller pieces as follows.

1. Construct a grammar for $\{0^n 1^n \mid n \geq 0\}$: $S_1 \to 0\, S_1\, 1 \mid \varepsilon$
2. Construct a grammar for $\{1^n 0^n \mid n \geq 0\}$: $S_2 \to 1\, S_2\, 0 \mid \varepsilon$

Designing CFGs

Example: To design a grammar for $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$, break it into smaller pieces as follows.

1. Construct a grammar for $\{0^n 1^n \mid n \geq 0\}$: $S_1 \to 0\, S_1\, 1 \mid \varepsilon$
2. Construct a grammar for $\{1^n 0^n \mid n \geq 0\}$: $S_2 \to 1\, S_2\, 0 \mid \varepsilon$
3. Add the rule $S \to S_1 \mid S_2$ and let S be the start variable.

This yields:

$$S \to S_1 \mid S_2$$
$$S_1 \to 0\, S_1\, 1 \mid \varepsilon$$
$$S_2 \to 1\, S_2\, 0 \mid \varepsilon$$

Chomsky Normal Form

- When working with context-free grammars, it is often convenient to have them in simplified form.

Definition (2.8)

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

$$A \rightarrow BC \quad \text{or}$$
$$A \rightarrow a$$

where a is any terminal and A, B, and C are any variables, except that B and C cannot be the start variable.

In addition,

$$S \rightarrow \varepsilon$$

is permitted if S is the start variable.

Chomsky Normal Form (cont.)

Theorem (2.9)

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

Chomsky Normal Form (cont.)

Theorem (2.9)

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

1. Add $S_0 \to S$, where S_0 is a new start symbol and S is the old one.

Chomsky Normal Form (cont.)

Theorem (2.9)

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

1. Add $S_0 \to S$, where S_0 is a new start symbol and S is the old one.
2. While an ε-rule $A \to \varepsilon$ with $A \neq S_0$ exists: Remove it, and for each occurrence of A on the right-hard side of a rule, add a rule with that occurrence deleted unless it is a previously removed ε-rule. E.g., $R \to uAvAw$ causes us to add $R \to uvAw$, $R \to uAvw$, $R \to uvw$.

Chomsky Normal Form (cont.)

Theorem (2.9)

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

1. Add $S_0 \to S$, where S_0 is a new start symbol and S is the old one.
2. While an ε-rule $A \to \varepsilon$ with $A \neq S_0$ exists: Remove it, and for each occurrence of A on the right-hard side of a rule, add a rule with that occurrence deleted unless it is a previously removed ε-rule. E.g., $R \to uAvAw$ causes us to add $R \to uvAw$, $R \to uAvw$, $R \to uvw$.
3. While a unit rule $A \to B$ exists: Remove it, and for each $B \to u$, add $A \to u$ unless it is a previously removed unit rule.

Chomsky Normal Form (cont.)

Theorem (2.9)

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

1. Add $S_0 \to S$, where S_0 is a new start symbol and S is the old one.
2. While an ε-rule $A \to \varepsilon$ with $A \neq S_0$ exists: Remove it, and for each occurrence of A on the right-hard side of a rule, add a rule with that occurrence deleted unless it is a previously removed ε-rule. E.g., $R \to uAvAw$ causes us to add $R \to uvAw$, $R \to uAvw$, $R \to uvw$.
3. While a unit rule $A \to B$ exists: Remove it, and for each $B \to u$, add $A \to u$ unless it is a previously removed unit rule.
4. Replace each rule $A \to u_1 u_2 \dots u_k$ such that $k \geq 3$ by $A \to u_1 A_1$, $A_1 \to u_2 A_2$, $\dots$, $A_{k-2} \to u_{k-1} u_k$, where the A_i's are new variables.

Chomsky Normal Form (cont.)

Theorem (2.9)

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

1. Add $S_0 \to S$, where S_0 is a new start symbol and S is the old one.
2. While an ε-rule $A \to \varepsilon$ with $A \neq S_0$ exists: Remove it, and for each occurrence of A on the right-hard side of a rule, add a rule with that occurrence deleted unless it is a previously removed ε-rule. E.g., $R \to uAvAw$ causes us to add $R \to uvAw$, $R \to uAvw$, $R \to uvw$.
3. While a unit rule $A \to B$ exists: Remove it, and for each $B \to u$, add $A \to u$ unless it is a previously removed unit rule.
4. Replace each rule $A \to u_1 u_2 \dots u_k$ such that $k \geq 3$ by $A \to u_1 A_1$, $A_1 \to u_2 A_2$, $\dots$, $A_{k-2} \to u_{k-1} u_k$, where the A_i's are new variables.
5. For each rule $A \to u_1 u_2$, if any u_i is a terminal then replace its occurrence in A by a new variable U_i and add $U_i \to u_i$.

An Example Conversion

Let us apply the described procedure to convert the following CFG to Chomsky normal form.

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

An Example Conversion

Let us apply the described procedure to convert the following CFG to Chomsky normal form.

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

- Add a new start symbol.

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

An Example Conversion (cont.)

- Remove ε rule $B \longrightarrow \varepsilon$.

$$
\begin{array}{rcl}
S_0 & \longrightarrow & S \\
S & \longrightarrow & ASA \mid aB \mid a \\
A & \longrightarrow & B \mid S \mid \varepsilon \\
B & \longrightarrow & b \mid \cancel{\varepsilon}
\end{array}
$$

An Example Conversion (cont.)

- Remove ε rule $B \longrightarrow \varepsilon$.

$$\begin{array}{rcl} S_0 & \longrightarrow & S \\ S & \longrightarrow & ASA \mid aB \mid a \\ A & \longrightarrow & B \mid S \mid \varepsilon \\ B & \longrightarrow & b \mid \cancel{\varepsilon} \end{array}$$

- Remove $A \longrightarrow \varepsilon$.

$$\begin{array}{rcl} S_0 & \longrightarrow & S \\ S & \longrightarrow & ASA \mid aB \mid a \mid SA \mid AS \mid S \\ A & \longrightarrow & B \mid S \mid \cancel{\varepsilon} \\ B & \longrightarrow & b \end{array}$$

An Example Conversion (cont.)

- Remove unit rule $S \longrightarrow S$.

$$
\begin{array}{rcl}
S_0 & \longrightarrow & S \\
S & \longrightarrow & ASA \mid aB \mid a \mid SA \mid AS \mid \cancel{S} \\
A & \longrightarrow & B \mid S \\
B & \longrightarrow & b
\end{array}
$$

An Example Conversion (cont.)

- Remove unit rule $S \longrightarrow S$.

$$\begin{array}{rcl} S_0 & \longrightarrow & S \\ S & \longrightarrow & ASA \mid aB \mid a \mid SA \mid AS \mid \cancel{S} \\ A & \longrightarrow & B \mid S \\ B & \longrightarrow & b \end{array}$$

- Remove $S_0 \longrightarrow S$.

$$\begin{array}{rcl} S_0 & \longrightarrow & \cancel{S} \mid ASA \mid aB \mid a \mid SA \mid AS \\ S & \longrightarrow & ASA \mid aB \mid a \mid SA \mid AS \\ A & \longrightarrow & B \mid S \\ B & \longrightarrow & b \end{array}$$

An Example Conversion (cont.)

- Remove $A \longrightarrow B$.

$$
\begin{array}{rcl}
S_0 & \longrightarrow & ASA \mid aB \mid a \mid SA \mid AS \\
S & \longrightarrow & ASA \mid aB \mid a \mid SA \mid AS \\
A & \longrightarrow & \cancel{B} \mid S \mid b \\
B & \longrightarrow & b
\end{array}
$$

An Example Conversion (cont.)

- Remove $A \to B$.

$$\begin{array}{rcl} S_0 & \to & ASA \mid aB \mid a \mid SA \mid AS \\ S & \to & ASA \mid aB \mid a \mid SA \mid AS \\ A & \to & \cancel{B} \mid S \mid b \\ B & \to & b \end{array}$$

- Remove $A \to S$.

$$\begin{array}{rcl} S_0 & \to & ASA \mid aB \mid a \mid SA \mid AS \\ S & \to & ASA \mid aB \mid a \mid SA \mid AS \\ A & \to & \cancel{S} \mid b \mid ASA \mid aB \mid a \mid SA \mid AS \\ B & \to & b \end{array}$$

An Example Conversion (cont.)

- Convert $S_0 \longrightarrow ASA$, $S \longrightarrow ASA$, and $A \longrightarrow ASA$.

$$
\begin{aligned}
S_0 &\longrightarrow AA_{1,1} \mid aB \mid a \mid SA \mid AS \\
S &\longrightarrow AA_{2,1} \mid aB \mid a \mid SA \mid AS \\
A &\longrightarrow b \mid AA_{3,1} \mid aB \mid a \mid SA \mid AS \\
A_{1,1} &\longrightarrow SA \\
A_{2,1} &\longrightarrow SA \\
A_{3,1} &\longrightarrow SA \\
B &\longrightarrow b
\end{aligned}
$$

An Example Conversion (cont.)

- Convert $S_0 \longrightarrow aB$, $S \longrightarrow aB$, and $A \longrightarrow aB$.

$$
\begin{aligned}
S_0 &\longrightarrow AA_{1,1} \mid U_1B \mid a \mid SA \mid AS \\
S &\longrightarrow AA_{2,1} \mid U_2B \mid a \mid SA \mid AS \\
A &\longrightarrow b \mid AA_{3,1} \mid U_3B \mid a \mid SA \mid AS \\
A_{1,1} &\longrightarrow SA \\
A_{2,1} &\longrightarrow SA \\
A_{3,1} &\longrightarrow SA \\
U_1 &\longrightarrow a \\
U_2 &\longrightarrow a \\
U_3 &\longrightarrow a \\
B &\longrightarrow b
\end{aligned}
$$

An Example Conversion (cont.)

- Convert $S_0 \to aB$, $S \to aB$, and $A \to aB$.

$$
\begin{aligned}
S_0 &\to AA_{1,1} \mid U_1B \mid a \mid SA \mid AS \\
S &\to AA_{2,1} \mid U_2B \mid a \mid SA \mid AS \\
A &\to b \mid AA_{3,1} \mid U_3B \mid a \mid SA \mid AS \\
A_{1,1} &\to SA \\
A_{2,1} &\to SA \\
A_{3,1} &\to SA \\
U_1 &\to a \\
U_2 &\to a \\
U_3 &\to a \\
B &\to b
\end{aligned}
$$

This grammar is in Chomsky normal form and generates the same language as the original one.