Leftmost Derivation & Rightmost Derivation

Definition 4.16 Leftmost and Rightmost Derivations

A derivation in a context-free grammar is a *leftmost* derivation (LMD) if, at each step, a production is applied to the leftmost variable-occurrence in the current string. A rightmost derivation (RMD) is defined similarly.

Example (Leftmost Derivation)

$E \rightarrow T + E \mid T$

$T \rightarrow F * T \mid F$

$F \rightarrow a$

String: $a + a * a$

E

$T + E$

$F + E$

$a + E$

$a + T$

$a + F * T$

$a + a * T$

$a + a * F$

$a + a * a$

Rightmost Derivation

E

T + E

T + T

T + F * T

T + F * F

T + F * a

T + a * a

F + a * a

a + a * a

Leftmost Derivation & Rightmost Derivation

Theorem 4.17

If G is a context-free grammar, then for every $x \in L(G)$, these three statements are equivalent:

1. x has more than one derivation tree.
2. x has more than one leftmost derivation.
3. x has more than one rightmost derivation.

$E \to E+E \mid E*E \mid a$

$a+a*a$ (2 Leftmost Derivations)

12

E

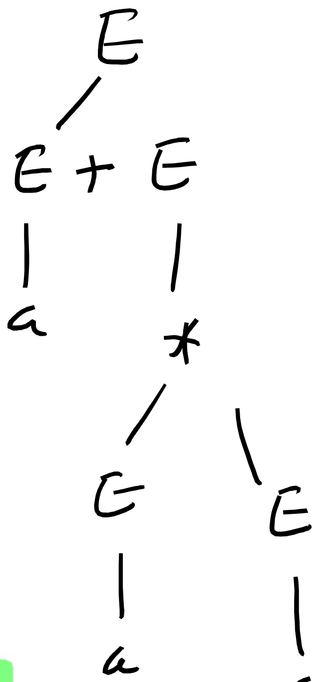
$E+E$

$a+E$

$a+E*E$

$a+a*E$

$a+a*a$



E

$E*E$

$E+E*E$

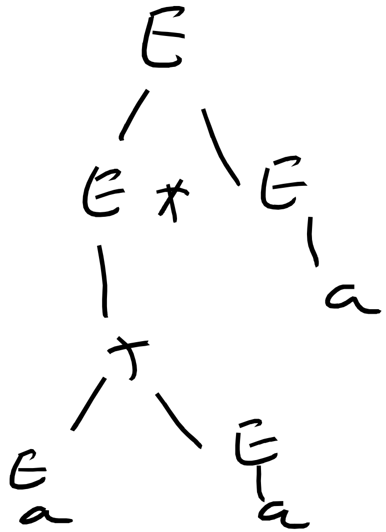
$a+E*E$

$a+a*E$

$a+a*a$

18

Let's take $a=3$



Ambiguous Grammar

A context-free grammar G is *ambiguous* if for at least one $x \in L(G)$, x has more than one derivation tree (or, equivalently, more than one leftmost derivation).

Ambiguous Grammar

EXAMPLE 4.20 Ambiguity in the CFG for *Expr* in Example 4.2

In the grammar G in Example 4.2, with productions

$$S \rightarrow a \mid S+S \mid S*S \mid (S)$$

the string $a+a*a$ has the two leftmost derivations

$$S \Rightarrow S+S \Rightarrow a+S \Rightarrow a+S*S \Rightarrow a+a*S \Rightarrow a+a*a$$

$$S \Rightarrow S*S \Rightarrow S+S*S \Rightarrow a+S*S \Rightarrow a+a*S \Rightarrow a+a*a$$

which correspond to the derivation trees in Figure 4.22.

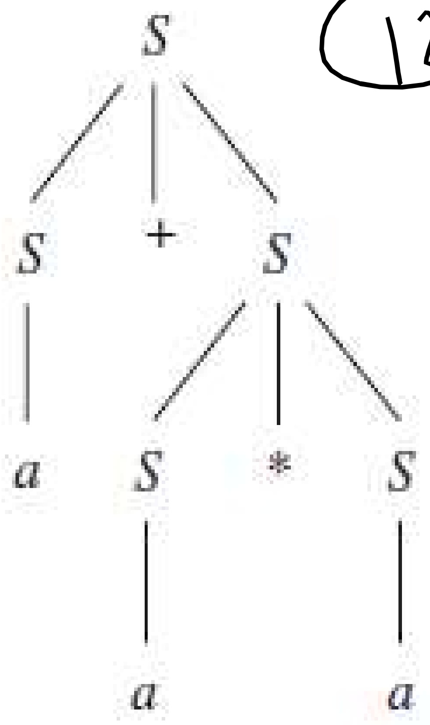
a=3

Two Left-most Derivation Trees

Top

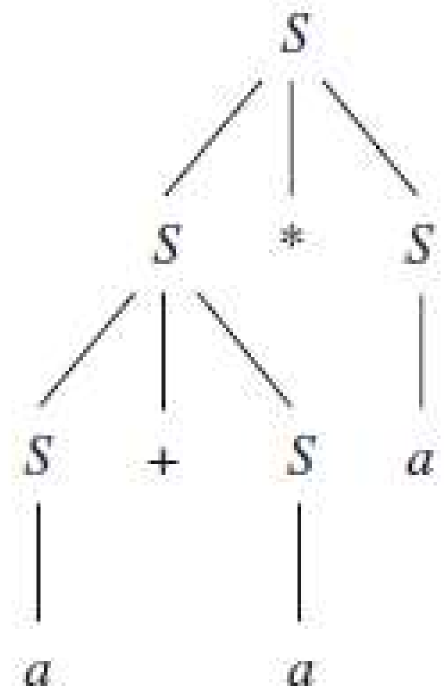
Bottom

12



(a)

18



(b)

Top

Bottom

Show that following grammars are Ambiguous

a. $S \to SS \mid a \mid b$ (aaa)

b. $S \to ABA$ $A \to aA \mid \Lambda$ $B \to bB \mid \Lambda$ (a)

c. $S \to aSb \mid aaSb \mid \Lambda$ (aaabb)

d. $S \to aSb \mid abS \mid \Lambda$ (ab)

S
a S b
a b

S
a b S
a b

Ambiguous Grammar

Show that the CFG with productions

$$S \rightarrow a \mid Sa \mid bSS \mid SSb \mid SbS$$

is ambiguous. baaa

S
bSS
baS
baSa
baaa

S
bSS
bSaS
baaS
baaa

Ambiguous Grammar

EXAMPLE 4.19 The Dangling *else*

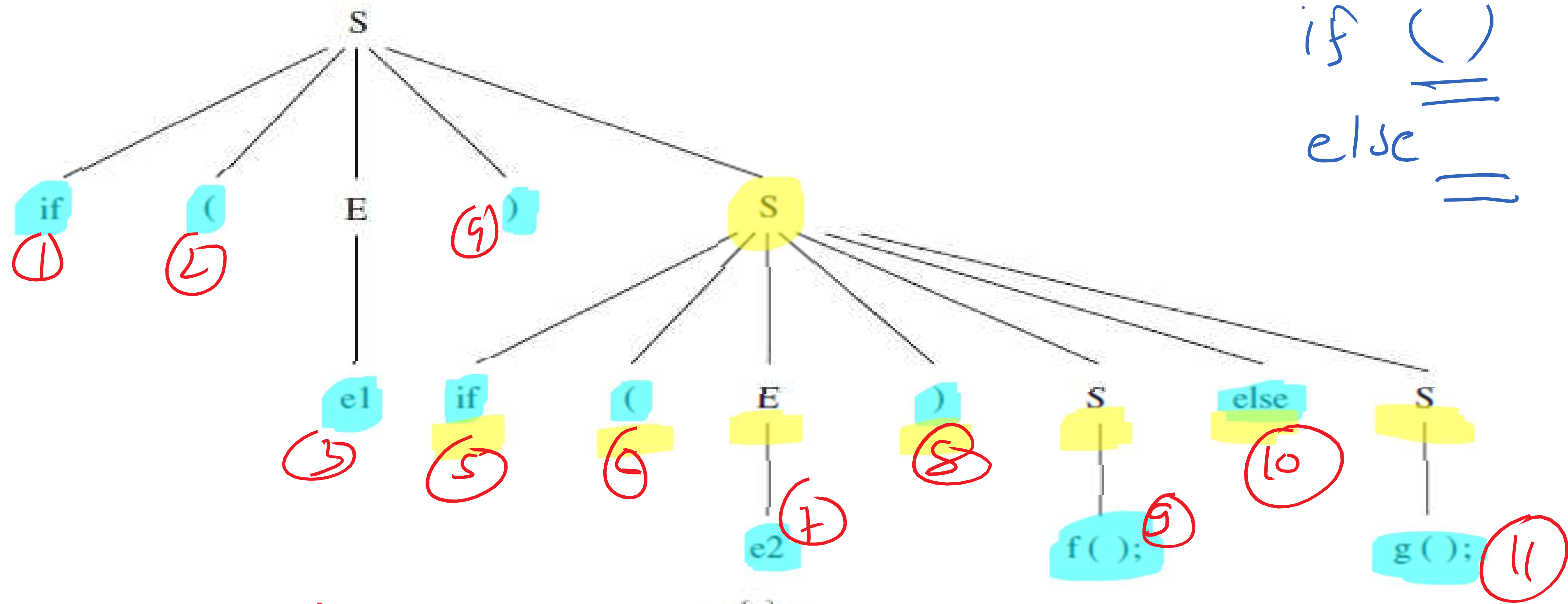
In the C programming language, an if-statement can be defined by these grammar rules:

$$
\begin{aligned}
S \rightarrow\ & \text{if } (E)\ S \mid \\
& \text{if } (E)\ S \text{ else } S \mid \\
& OS
\end{aligned}
$$

OS = Other Statement

Dangling "Else" Problem

if ( )
if ( )
else

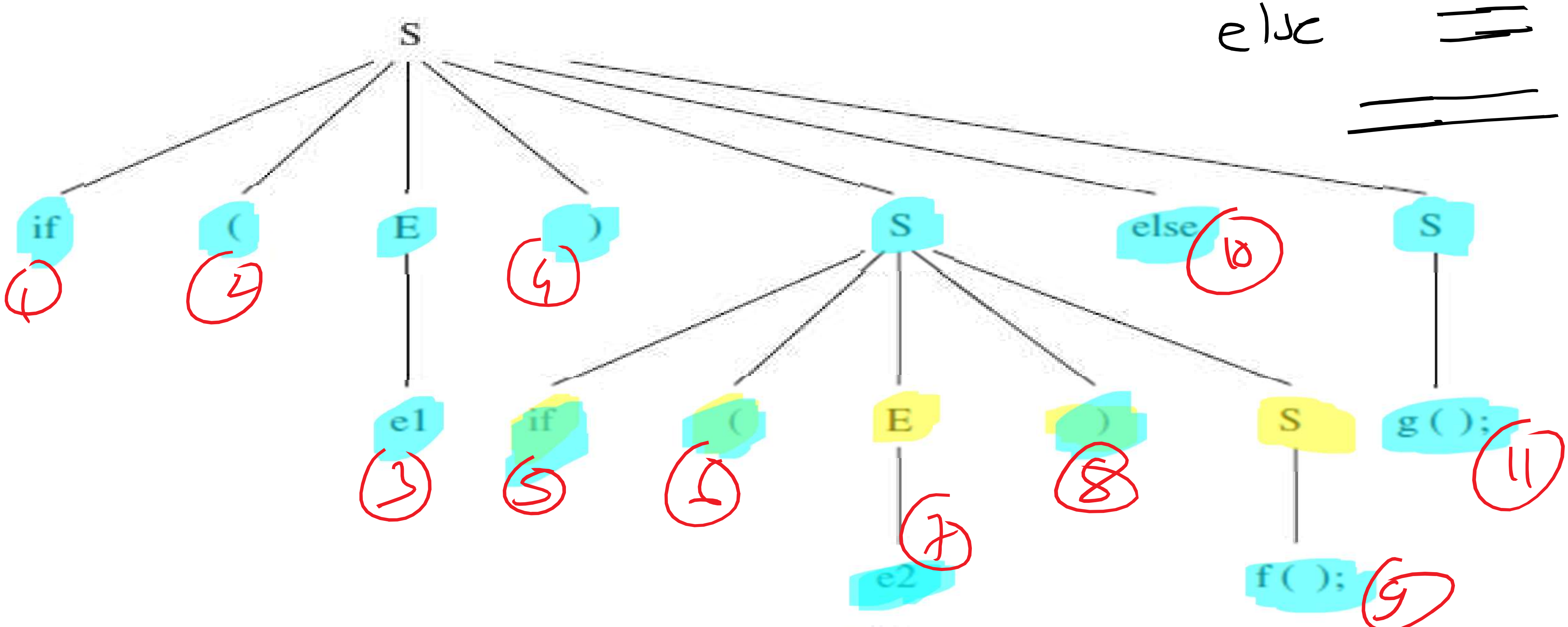


(a)

if (e1) if (e2) f(); else g();

Dangling "Else" Problem

if (   )
    if ( )
else



(b)

if (e1) if (e2) f(); else g();

Top-Down Parsing

- We can eliminate non-Determinism **upto some extent** by re-writing the grammar rules

- There are two main techniques :
    1. Left-Factoring of Grammar
    2. Elimination of Left Recursion

Left –factoring of Grammar

A non-terminal with two or more productions, whose right-hand sides start with the same grammar symbols, adds non-determinism to the Parser

Replace productions

$$A \rightarrow \alpha\ \beta_1 \mid \alpha\ \beta_2 \mid \ldots \mid \alpha\ \beta_n \mid \gamma$$

with

$$A \rightarrow \alpha\ A_R \mid \gamma$$
$$A_R \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

Left –factoring of Grammar

Replace productions

$$A \rightarrow \alpha\ \beta_1 \mid \alpha\ \beta_2 \mid \ldots \mid \alpha\ \beta_n \mid \gamma$$

with

$$A \rightarrow \alpha\ A_R \mid \gamma$$
$$A_R \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

Consider the grammar:

$E \rightarrow T + E \mid T - E \mid T$

$T \rightarrow F * T \mid F$

$F \rightarrow a$

Left-factored Grammar:

$E \rightarrow T E_R$
$E_R \rightarrow +E \mid -E \mid$ ^
$T \rightarrow F T_R$
$T_R \rightarrow *T \mid$ ^
$F \rightarrow a$

Left –factoring of Grammar

Suppose you want to derive 'a' in the previous grammar

Grammar:

$E \rightarrow TE_R$
$E_R \rightarrow +E \mid -E \mid \wedge$
$T \rightarrow FT_R$
$T_R \rightarrow *T \mid \wedge$
$F \rightarrow a$

Original Grammar (Non-Deterministic)	Left- Factored Grammar (Deterministic)
E	E
T+ E or T-E or T (Trial and Error)	TE_R
T	FT_RE_R
F*T or F (Trial and Error)	aT_RE_R
F	$a\ E_R$
a	a

Left –factoring of Grammar (Example-2)

Replace productions

$$A \rightarrow \alpha\,\beta_1 \mid \alpha\,\beta_2 \mid \ldots \mid \alpha\,\beta_n \mid \gamma$$

with

$$A \rightarrow \alpha\,A_R \mid \gamma$$
$$A_R \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

Consider the grammar:

S→ iEtSeS | iEtS | a

E→ b

Left-factored Grammar:

S→ iEtSS_R | a
S_R→ eS | ^
E→ b

Left –factoring of Grammar

Suppose you want to derive **'ibtibtaea'** in the previous grammar

Here, Grammar is Ambiguous

Original Grammar (Non-Deterministic)	Left- Factored Grammar (Deterministic)
S	S
iEtSeS or iEtS (Trial and Error)	$iEtSS_R$
	$ibtSS_R$
	$ibtiEtSS_RS_R$
	$ibtibtSS_RS_R$
	$ibtibtaS_RS_R$
	$ibtibtaeSS_R$ or $ibtibtaS_R$
	$ibtibtaeaS_R$ or ibtibtaeS
	Ibtibtaea or ibtibtaea

Grammar:

$S \rightarrow iEtSS_R \mid a$
$S_R \rightarrow eS \mid \wedge$
$E \rightarrow b$

Left –factoring of Grammar (Example-3)

Replace productions

$$A \rightarrow \alpha\, \beta_1 \mid \alpha\, \beta_2 \mid \ldots \mid \alpha\, \beta_n \mid \gamma$$

with

$$A \rightarrow \alpha\, A_R \mid \gamma$$
$$A_R \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

S→ T$

T→ [T] | [] T | [T]T | []

S→ T$

T→ [$\mathbf{T_R}$

$\mathbf{T_R}$→ T] |] T | T]T |]

Left –factoring of Grammar (Example-3)

$S \rightarrow T\$$

$T \rightarrow [\ \mathbf{T_R}$

$\mathbf{T_R} \rightarrow T\ ]\ |\ ]\ T\ |\ T\ ]\ T\ |\ ]$

$S \rightarrow T\$$

$T \rightarrow [\ \mathbf{T_R}$

$\mathbf{T_R} \rightarrow T\ ]\ \mathbf{M}\ |\ ]\ \mathbf{N}$

$\mathbf{M} \rightarrow T\ |\ \wedge$

$\mathbf{N} \rightarrow T\ |\ \wedge$

$S \rightarrow T\$$

$T \rightarrow [\ \mathbf{T_R}$

$\mathbf{T_R} \rightarrow T\ ]\ \mathbf{M}\ |\ ]\ \mathbf{M}$

$\mathbf{M} \rightarrow T\ |\ \wedge$