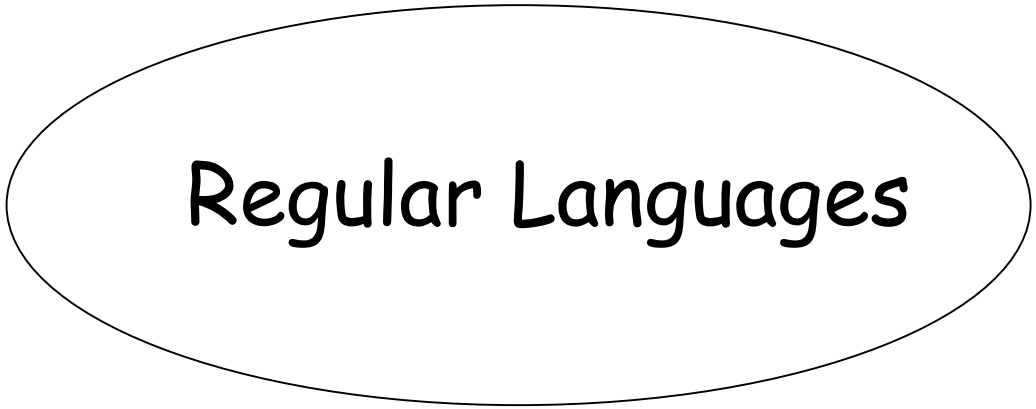
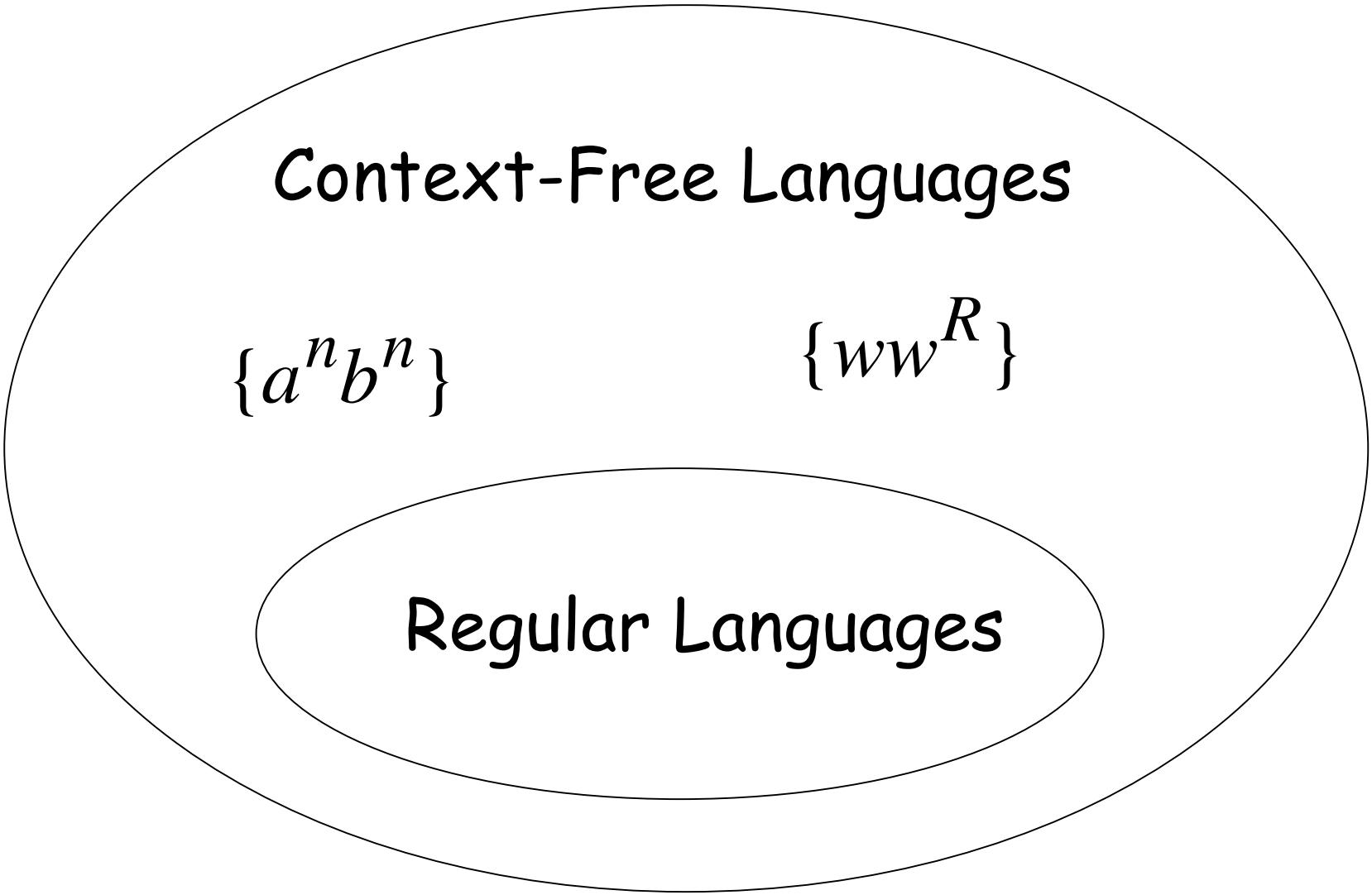
Context-Free Languages

$\{a^n b^n\}$ $\{ww^R\}$

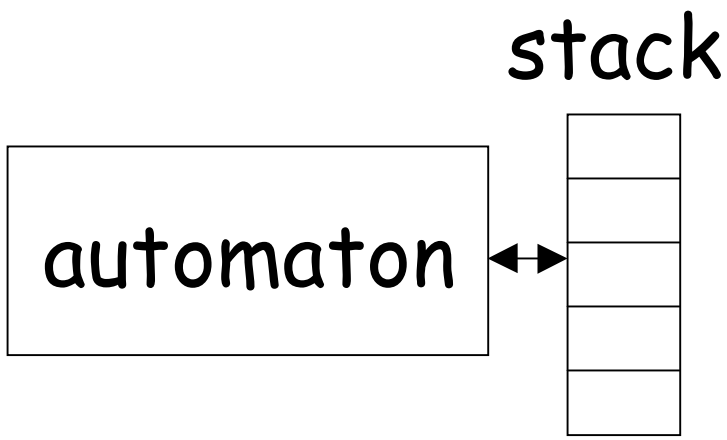



Context-Free Languages
$\{a^n b^n\}$
$\{ww^R\}$
Regular Languages

Context-Free Languages

Context-Free Grammars

Pushdown Automata



Context-Free Grammars

A context-free grammar G:

$$S \to aSb$$

$$S \to \lambda$$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$$

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

$$L(G) = \{a^n b^n : n \geq 0\}$$

((( )))

A context-free grammar G:

$$S \to aSa$$

$$S \to bSb$$

$$S \to \lambda$$

$$S \Rightarrow aSa \Rightarrow abSba \Rightarrow abaSaba \Rightarrow abaaba$$

$$S \rightarrow aSa$$

$$S \rightarrow bSb$$

$$S \rightarrow \lambda$$

$$L(G) = \{ww^R\}$$

A context-free grammar G:

$$S \to aSb$$

$$S \to SS$$

$$S \to \lambda$$

$$S \Rightarrow SS \Rightarrow aSbS \Rightarrow aSbS \Rightarrow abS$$

$$\Rightarrow abaSb \Rightarrow abab$$

$$S \to aSb$$

$$S \to SS$$

$$S \to \lambda$$

$$L(G) = \{w \quad : n_a(w) = n_b(w),$$
$$n_a(v) \geq n_b(v),$$
$$\text{any prefix } v\}$$

() ((( ))) (( ))

Definition of Context-Free Grammars

Any grammar G=(V,T,S,P)

V: variables

T: terminal symbols

S: start variable

P: production rules of the form

$$A \to x$$

x : string of variables and terminals

Definition of Context-Free Languages

A language L is context-free

if and only if

there is a grammar G with $L = L(G)$

Derivation Order

1. $S \to AB$
2. $A \to aaA$
3. $A \to \lambda$
4. $B \to Bb$
5. $B \to \lambda$

Leftmost derivation:

$$S \overset{1}{\Rightarrow} AB \overset{2}{\Rightarrow} aaAB \overset{3}{\Rightarrow} aaB \overset{4}{\Rightarrow} aaBb \overset{5}{\Rightarrow} aab$$

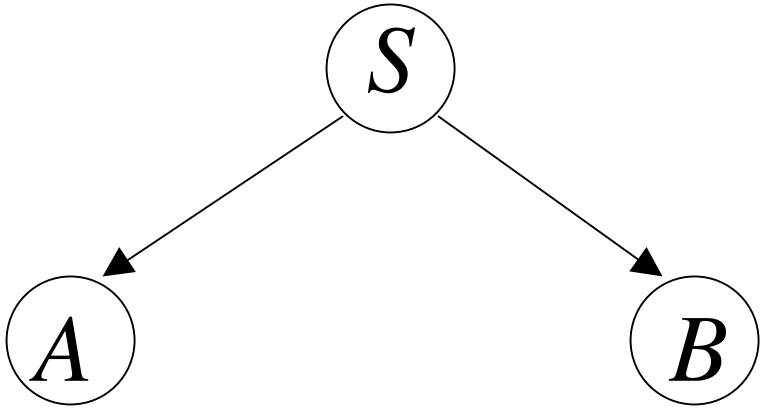
Rightmost derivation:

$$S \overset{1}{\Rightarrow} AB \overset{4}{\Rightarrow} ABb \overset{5}{\Rightarrow} Ab \overset{2}{\Rightarrow} aaAb \overset{3}{\Rightarrow} aab$$

Derivation Trees

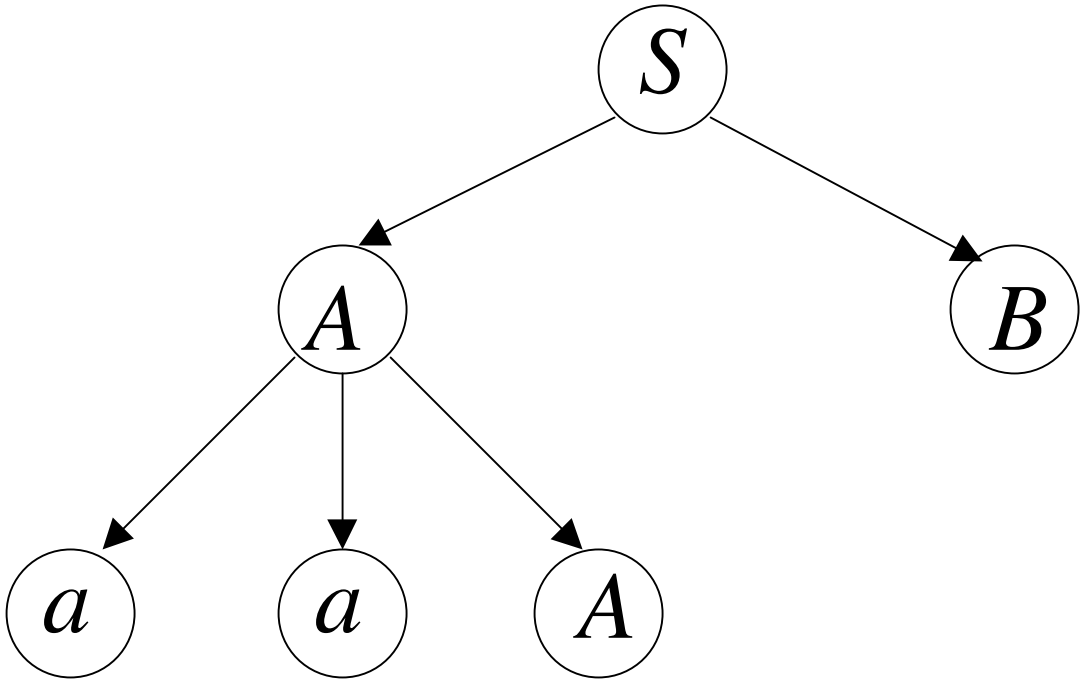
$S \rightarrow AB$ $A \rightarrow aaA \mid \lambda$ $B \rightarrow Bb \mid \lambda$

$S \Rightarrow AB$



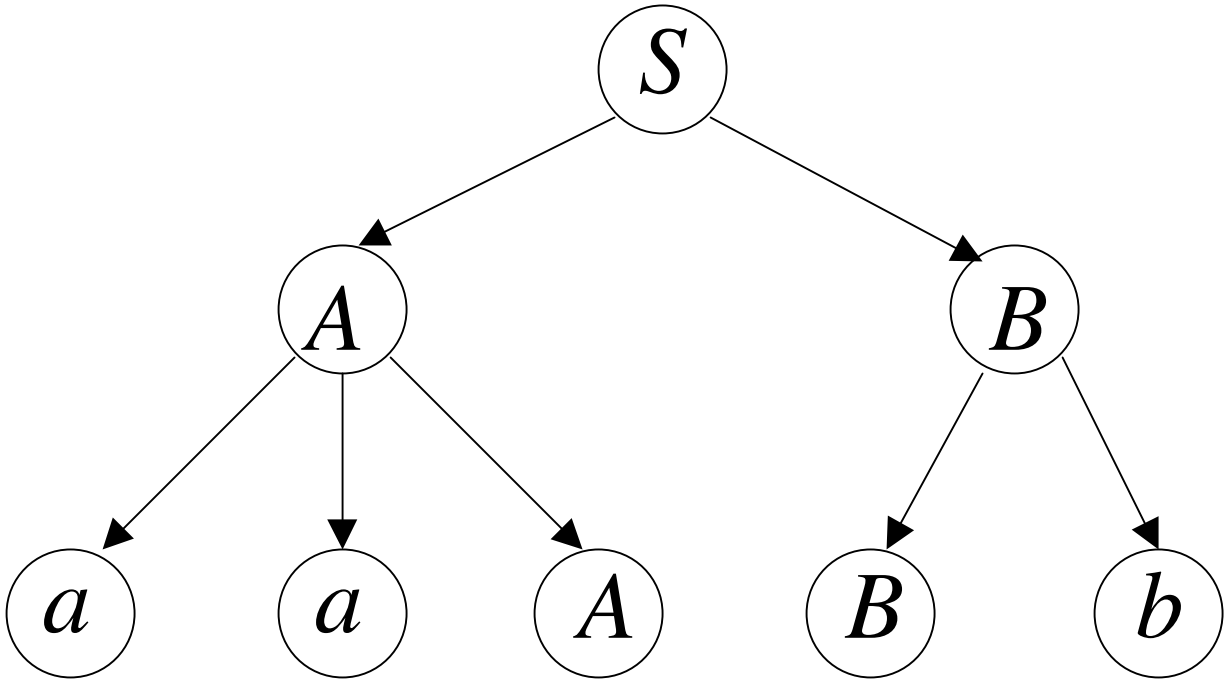
$S \rightarrow AB$ $A \rightarrow aaA \mid \lambda$ $B \rightarrow Bb \mid \lambda$

$$S \Rightarrow AB \Rightarrow aaAB$$



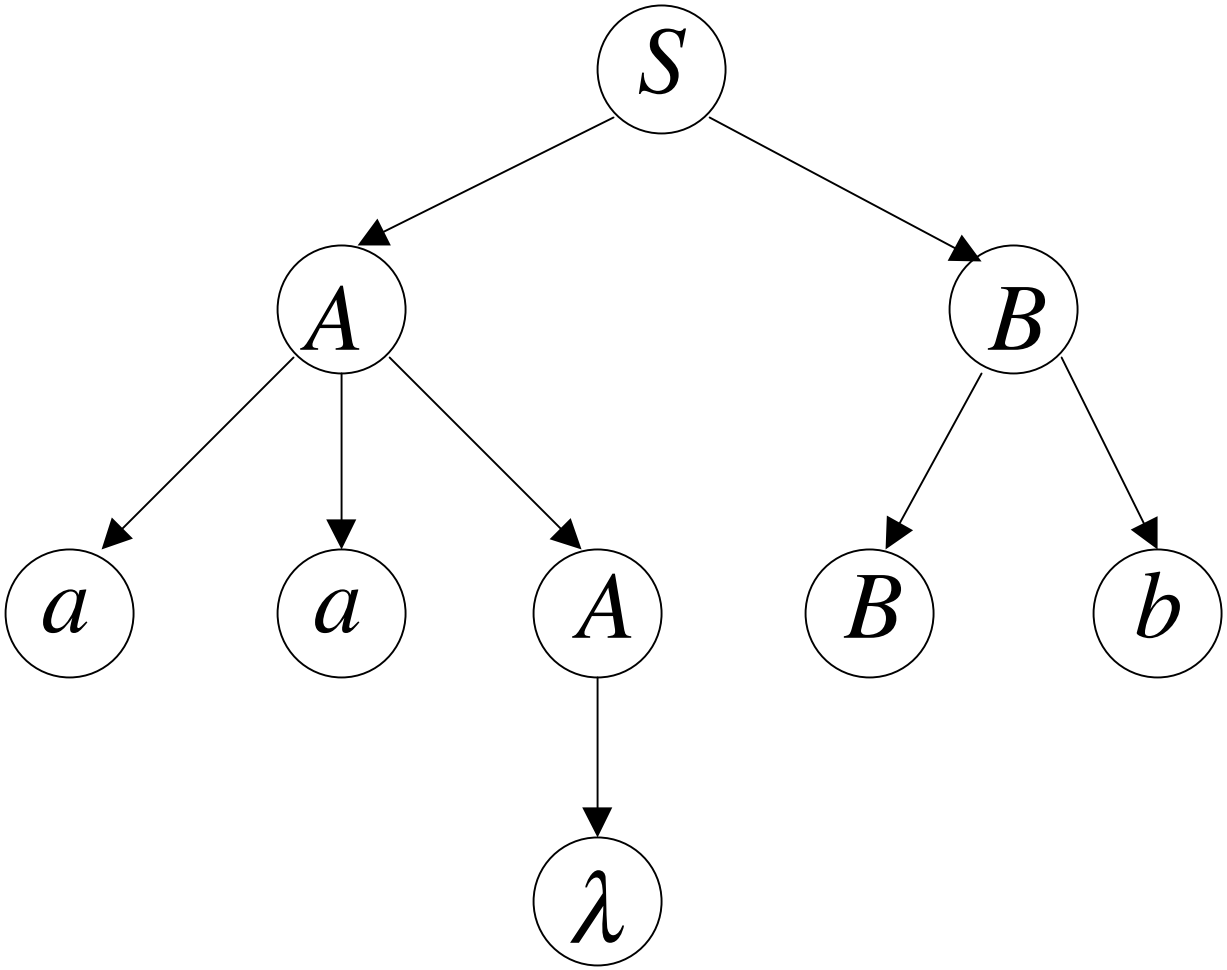
$S \rightarrow AB$ $A \rightarrow aaA \mid \lambda$ $B \rightarrow Bb \mid \lambda$

$$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaABb$$



$S \to AB$ $A \to aaA \mid \lambda$ $B \to Bb \mid \lambda$

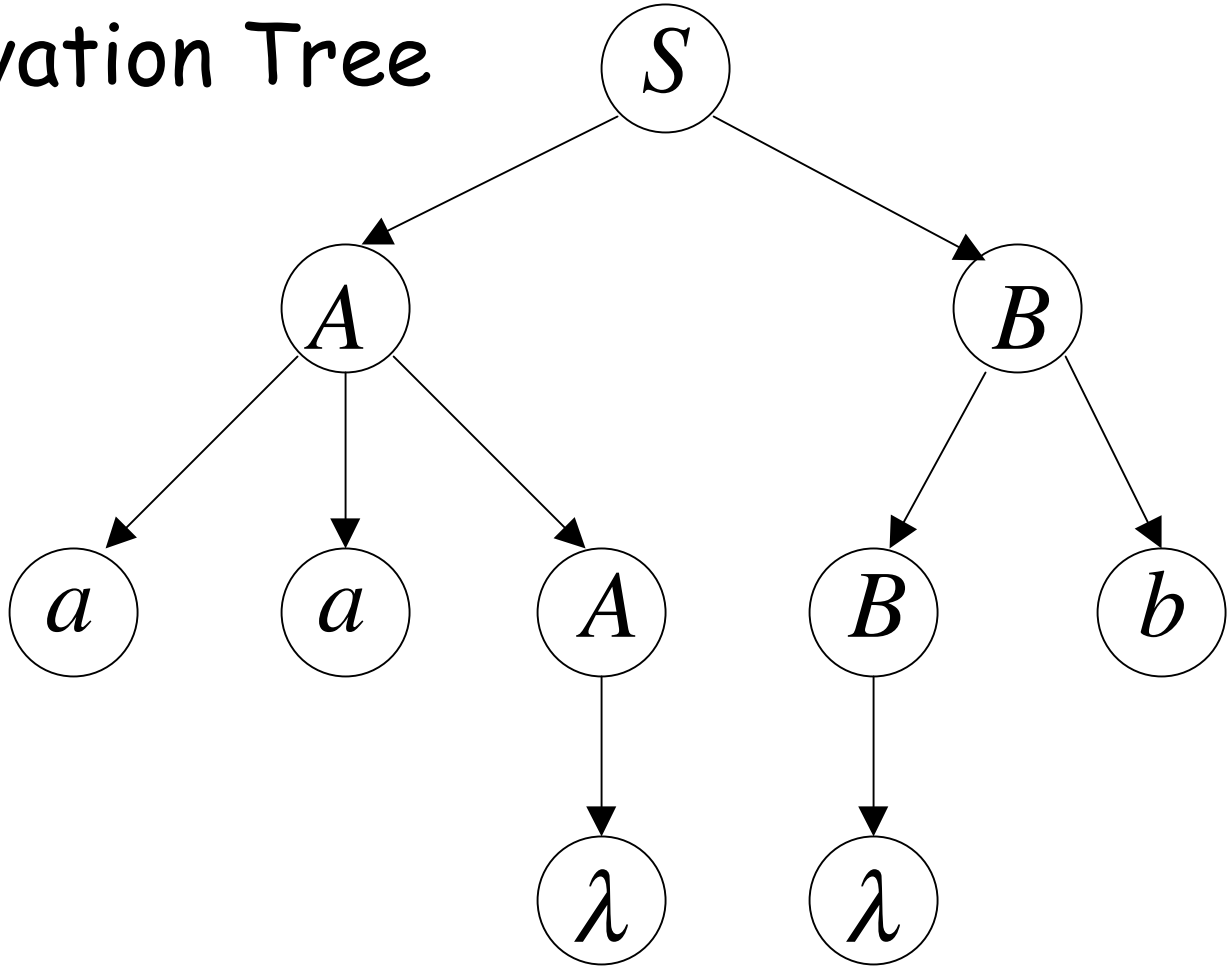
$$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaABb \Rightarrow aaBb$$



$S \to AB$ $A \to aaA \mid \lambda$ $B \to Bb \mid \lambda$

$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaABb \Rightarrow aaBb \Rightarrow aab$

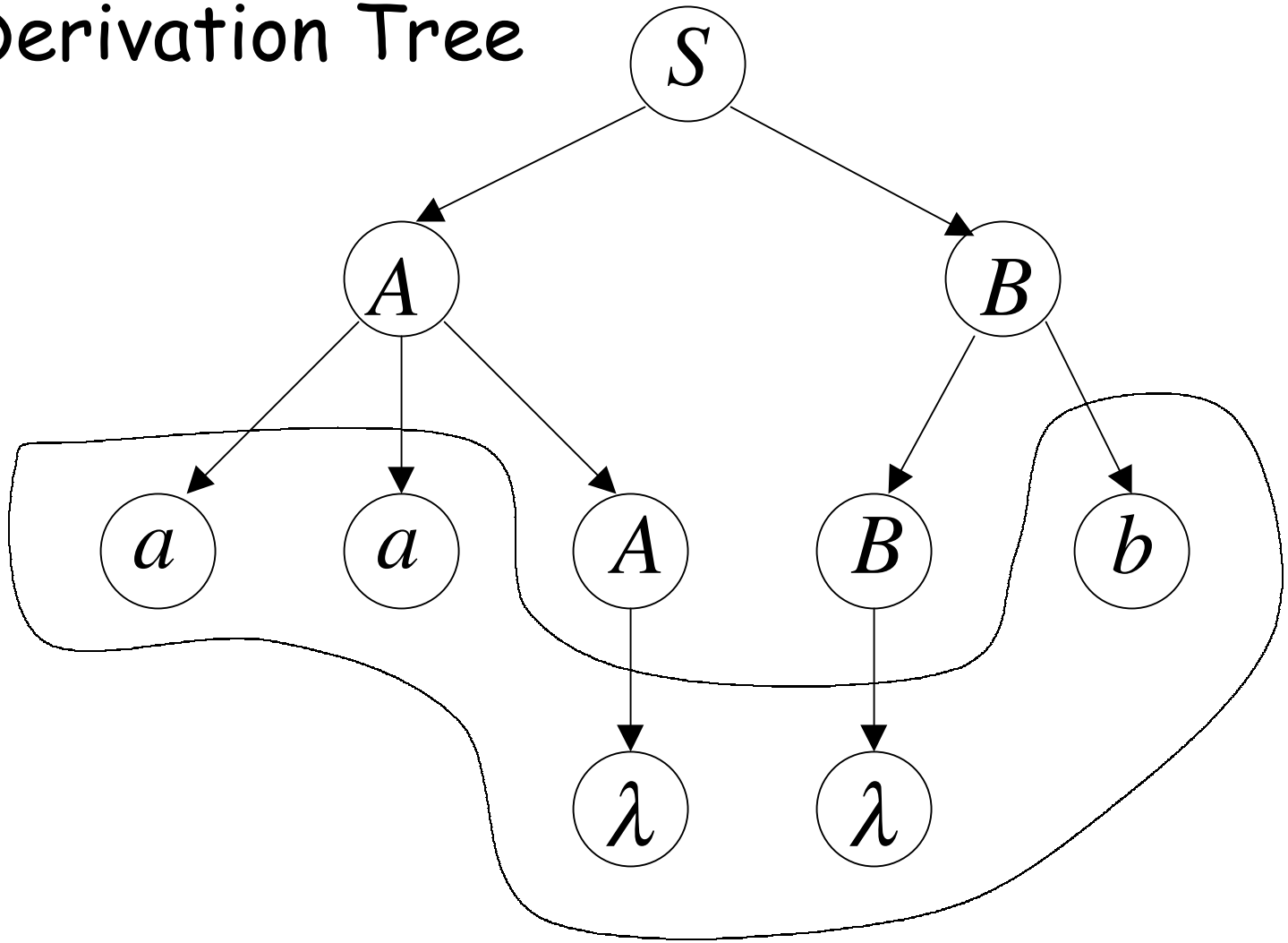
Derivation Tree



$$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaABb \Rightarrow aaBb \Rightarrow aab$$

sentence

Derivation Tree



yield

$aa\lambda\lambda b$

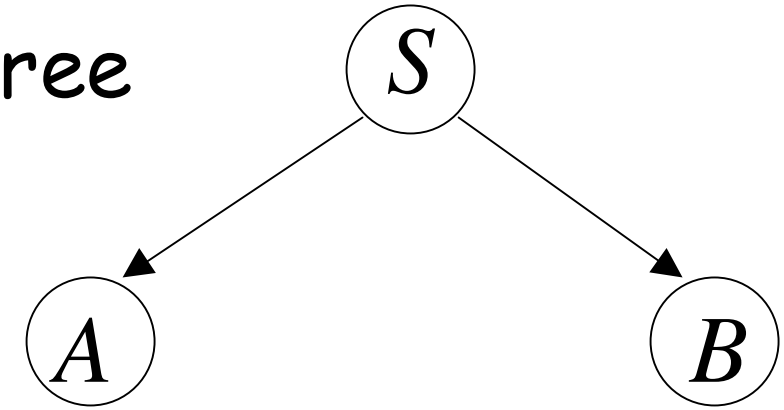
$= aab$

Partial Derivation Trees

$S \to AB$ $\quad A \to aaA \mid \lambda$ $\quad B \to Bb \mid \lambda$

$$S \Rightarrow AB$$

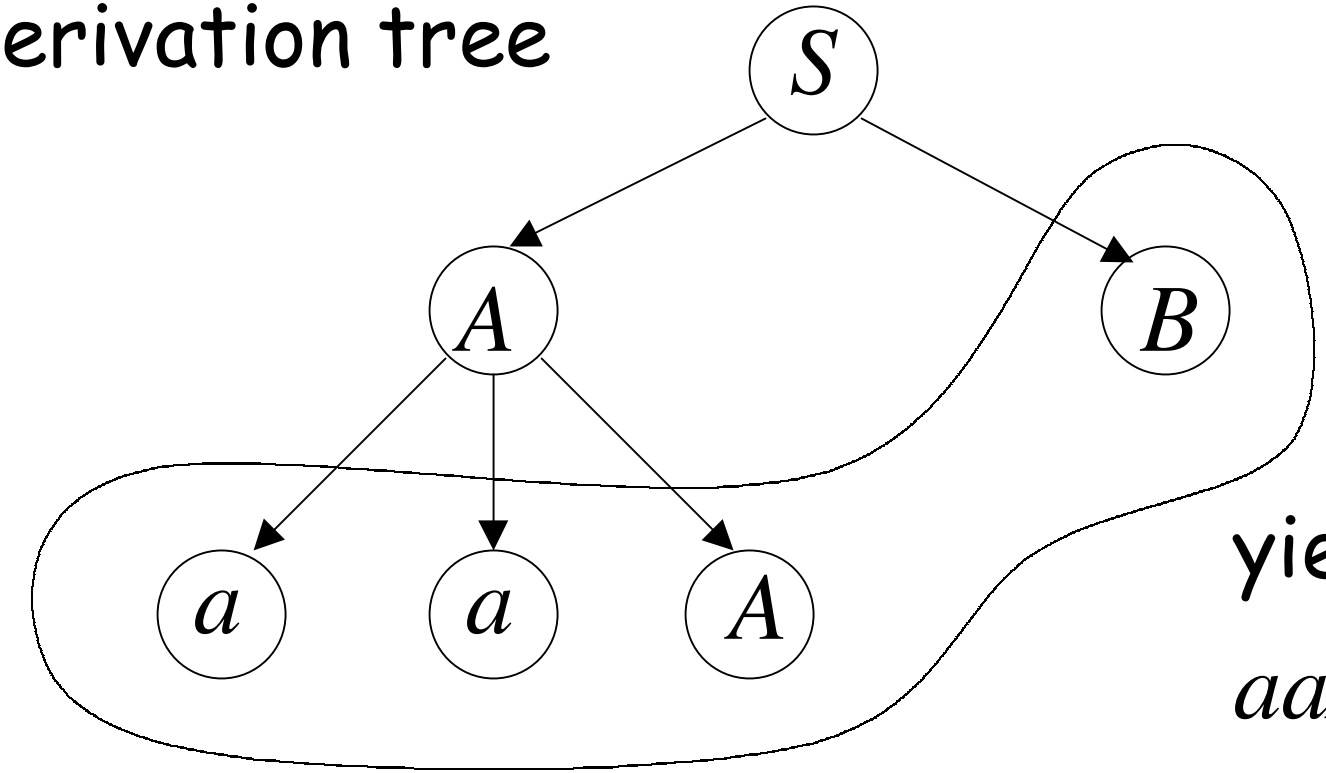
Partial derivation tree



$$S \Rightarrow AB \Rightarrow aaAB$$

sentential form

Partial derivation tree



yield

$aaAB$

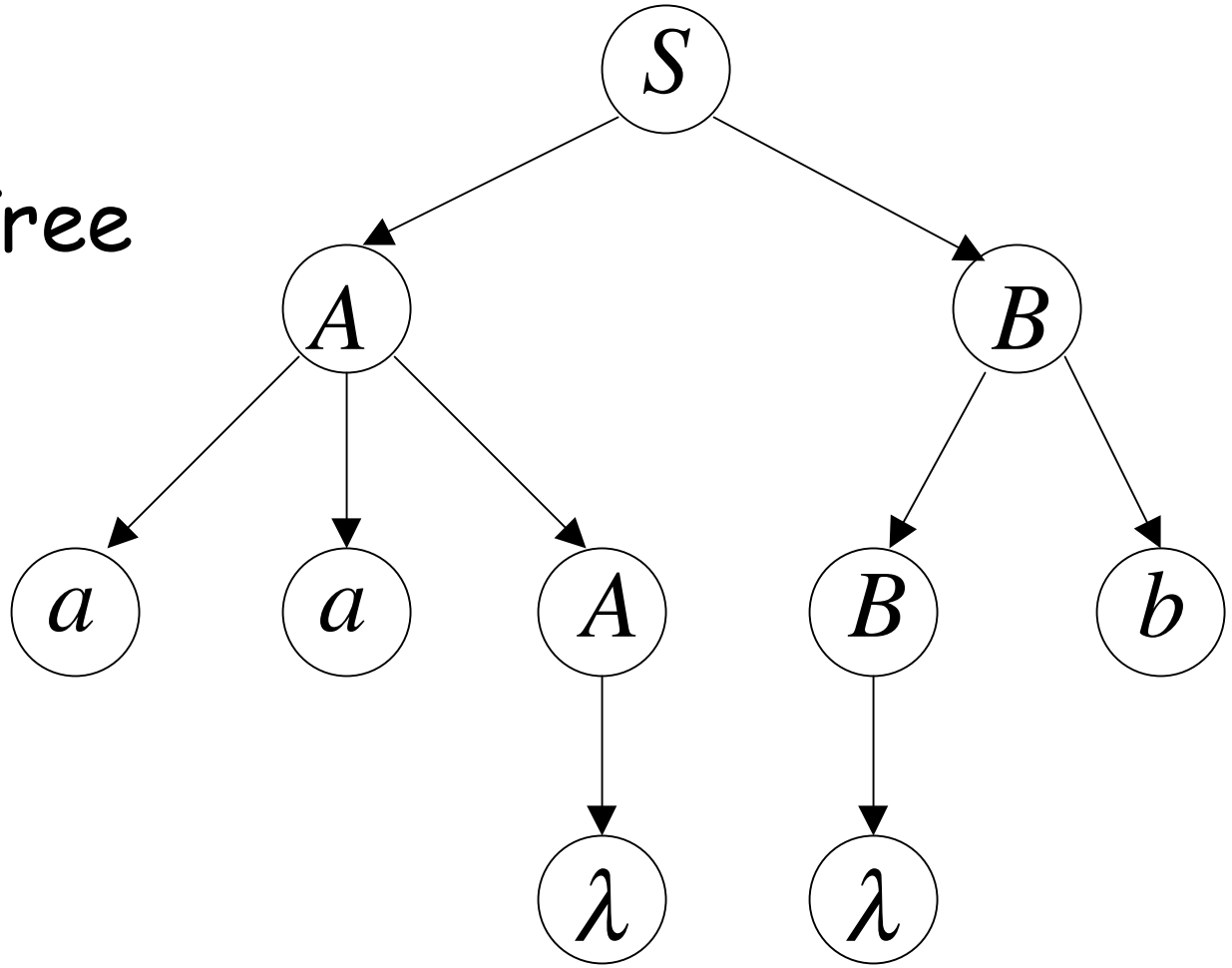
Often, derivation order doesn't matter

$$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaB \Rightarrow aaBb \Rightarrow aab$$

$$S \Rightarrow AB \Rightarrow ABb \Rightarrow Ab \Rightarrow aaAb \Rightarrow aab$$

Same

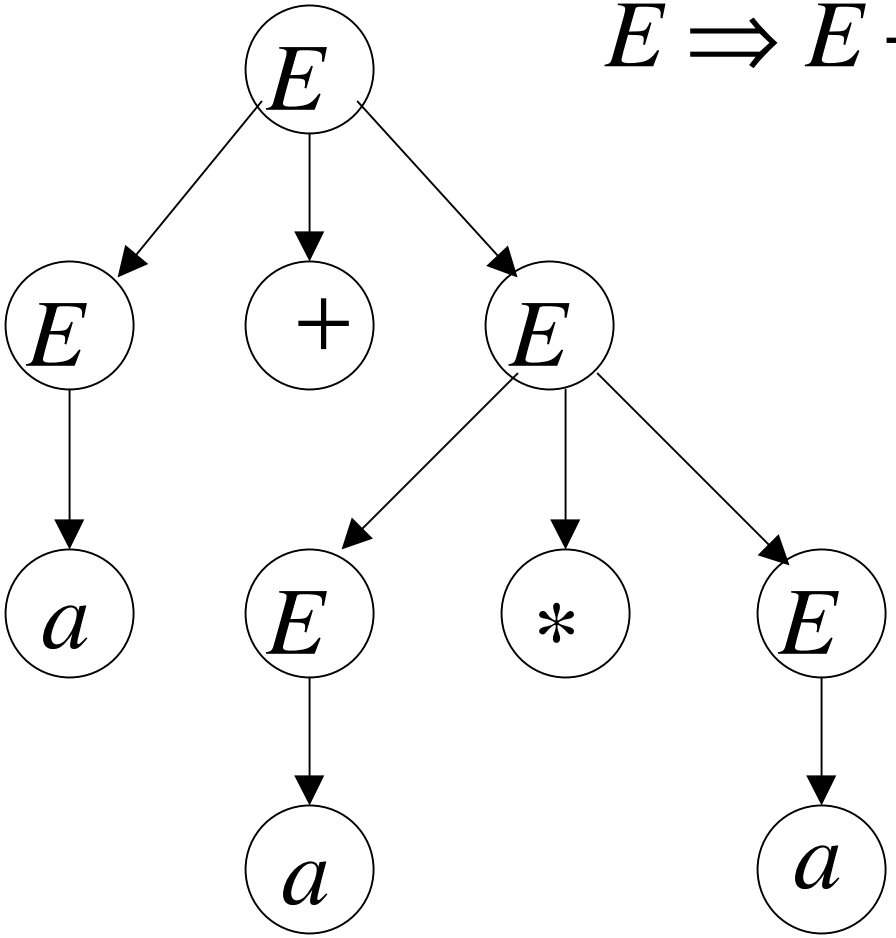
Derivation Tree



Ambiguity

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

$$a + a * a$$



$$E \Rightarrow E + E \Rightarrow a + E \Rightarrow a + E * E$$

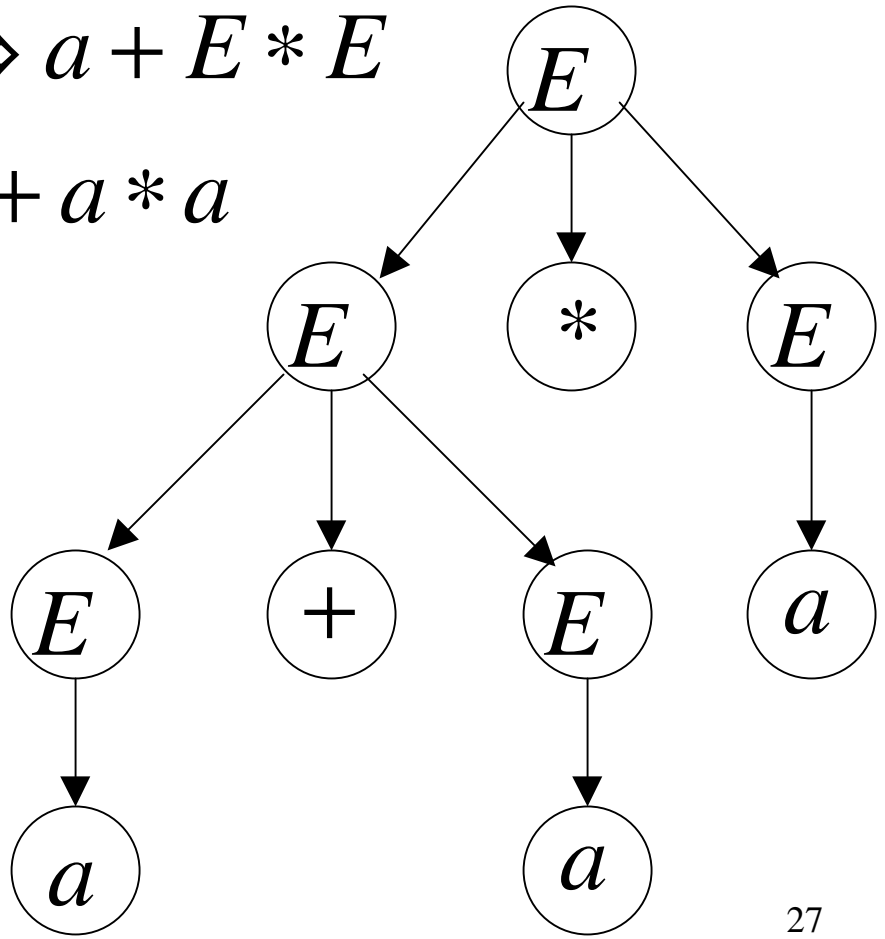
$$\Rightarrow a + a * E \Rightarrow a + a * a$$

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

$$a + a * a$$

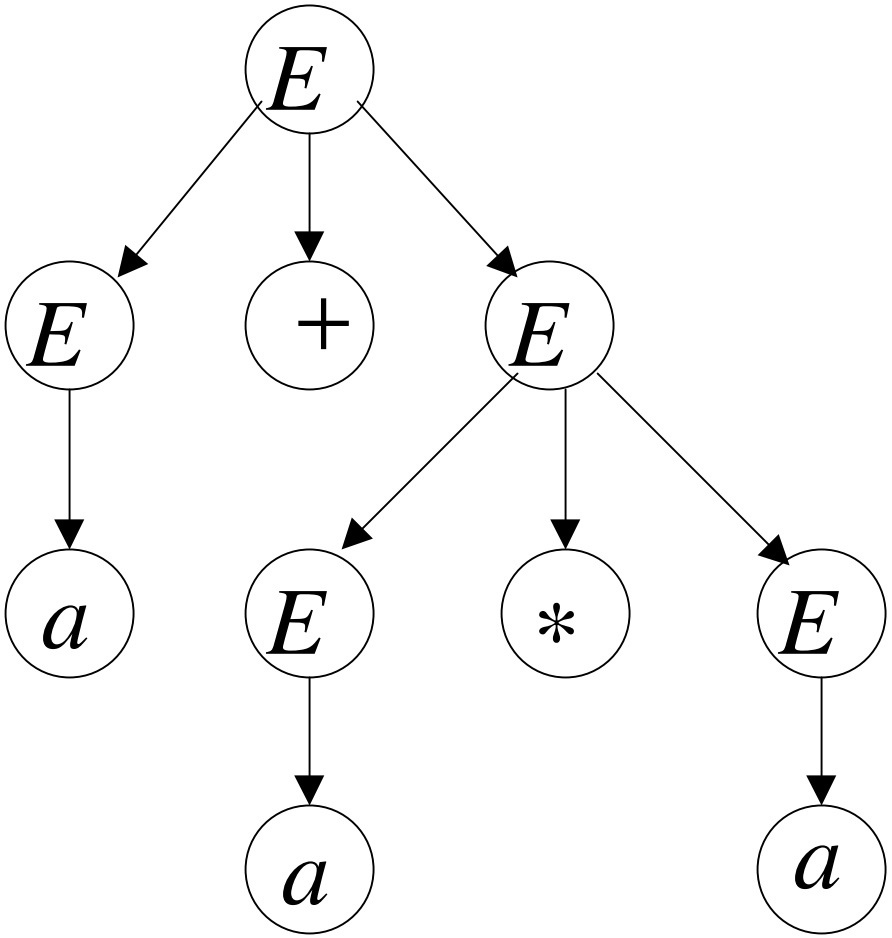
$$E \Rightarrow E * E \Rightarrow E + E * E \Rightarrow a + E * E$$

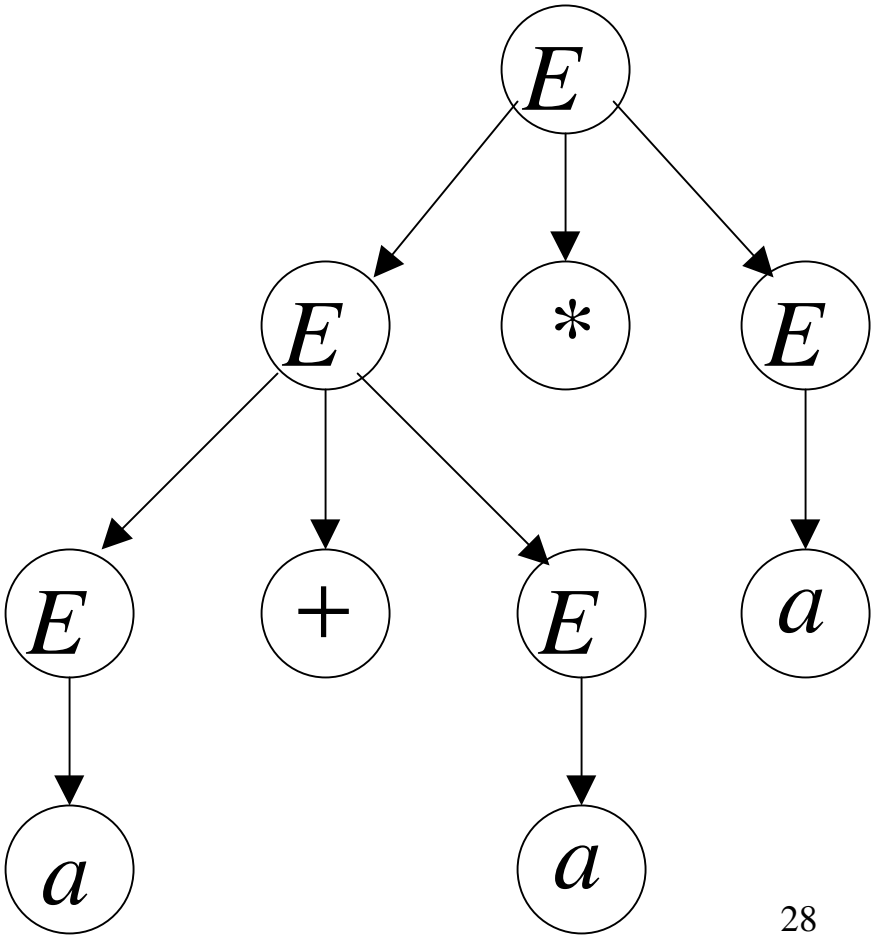
$$\Rightarrow a + a * E \Rightarrow a + a * a$$



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

$$a + a * a$$





The grammar

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

is **ambiguous**:

string $a + a * a$ has two derivation trees

The grammar $E \to E + E \mid E * E \mid (E) \mid a$

is **ambiguous**:

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

$$E \Rightarrow E + E \Rightarrow a + E \Rightarrow a + E * E$$
$$\Rightarrow a + a * E \Rightarrow a + a * a$$

$$E \Rightarrow E * E \Rightarrow E + E * E \Rightarrow a + E * E$$
$$\Rightarrow a + a * E \Rightarrow a + a * a$$

In general:

A context-free grammar G is **ambiguous**

if some string $w \in L(G)$ has:

two or more derivation trees

In other words:

A context-free grammar G is **ambiguous**

if some string $w \in L(G)$ has:

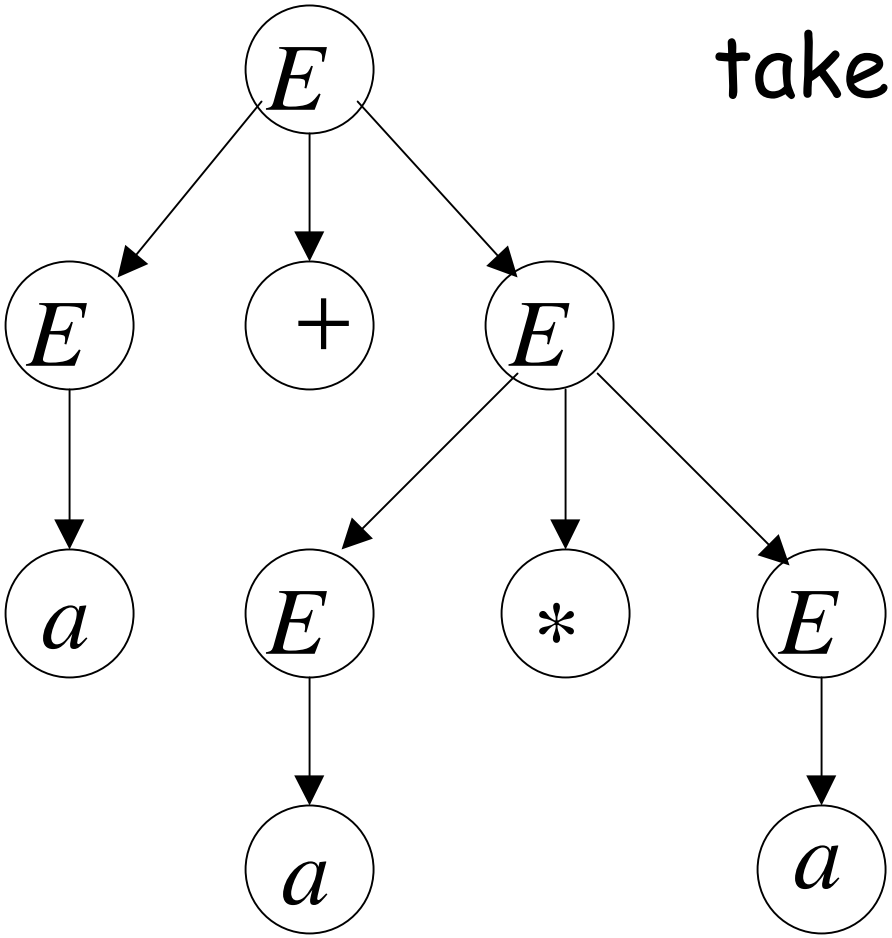
two or more leftmost derivations

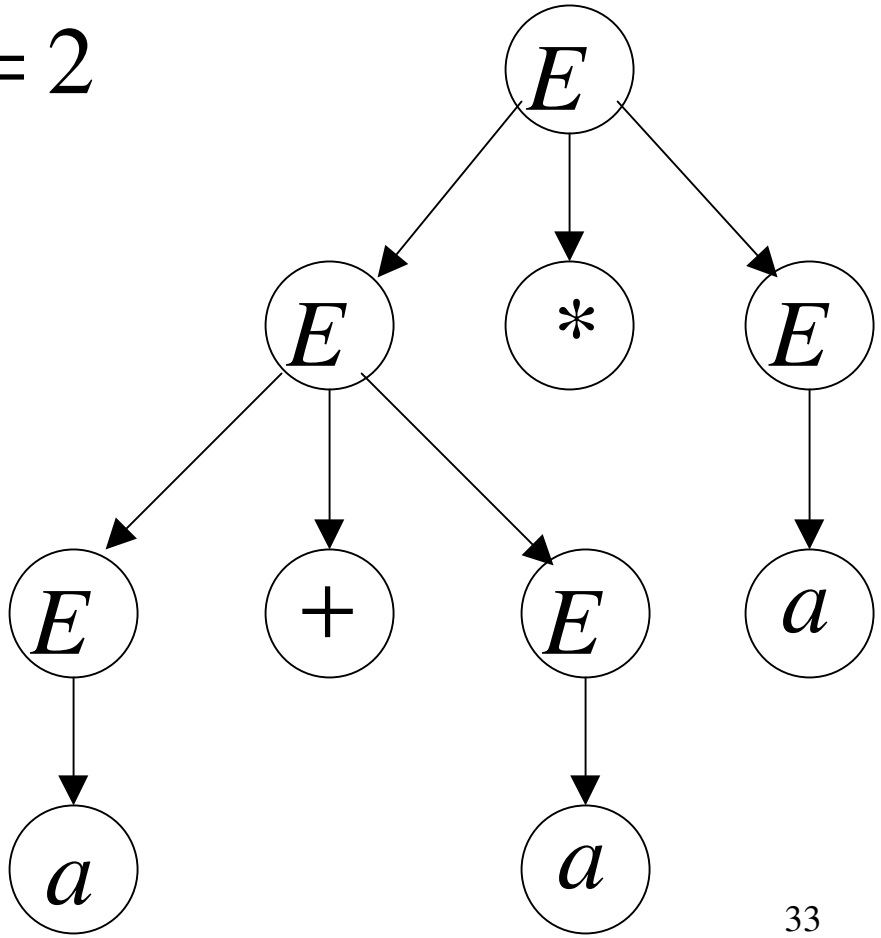
rightmost

Why do we care about ambiguity?

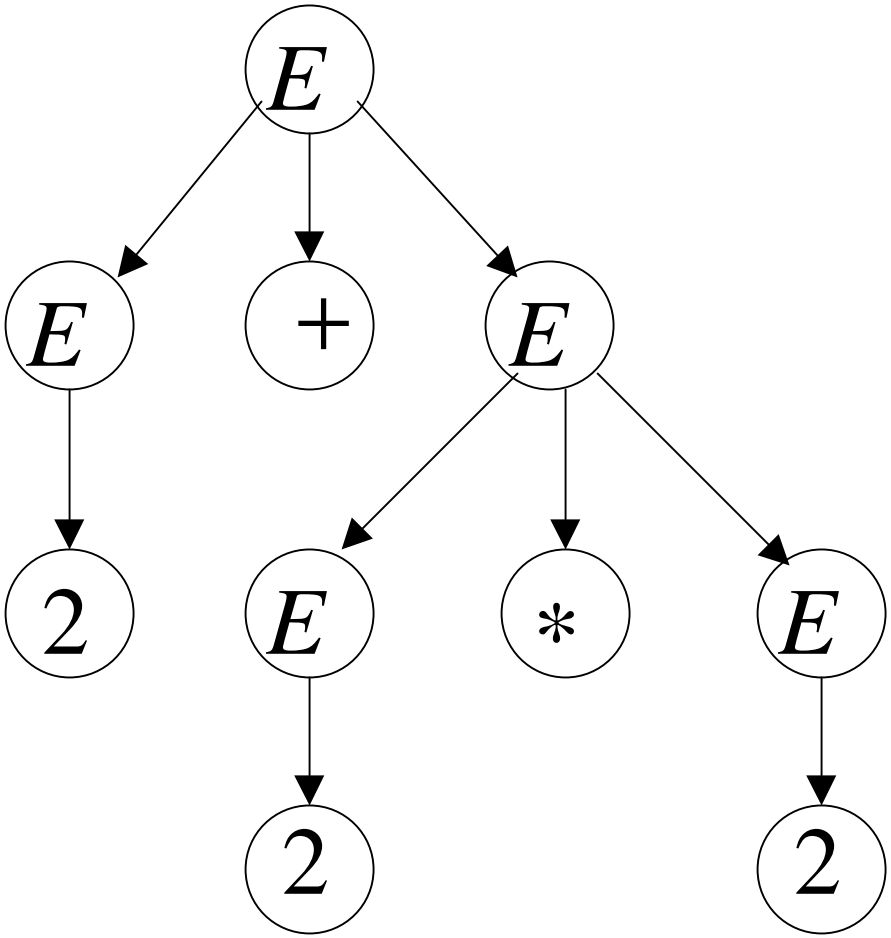
$$a + a * a$$

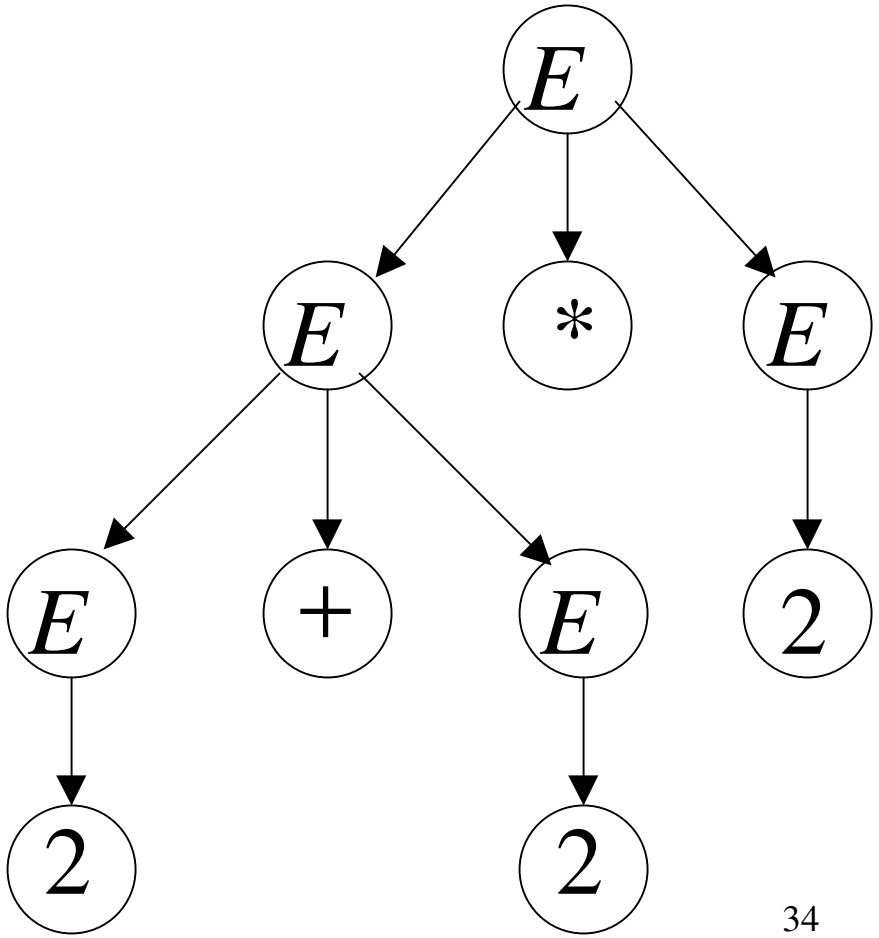
take $a = 2$



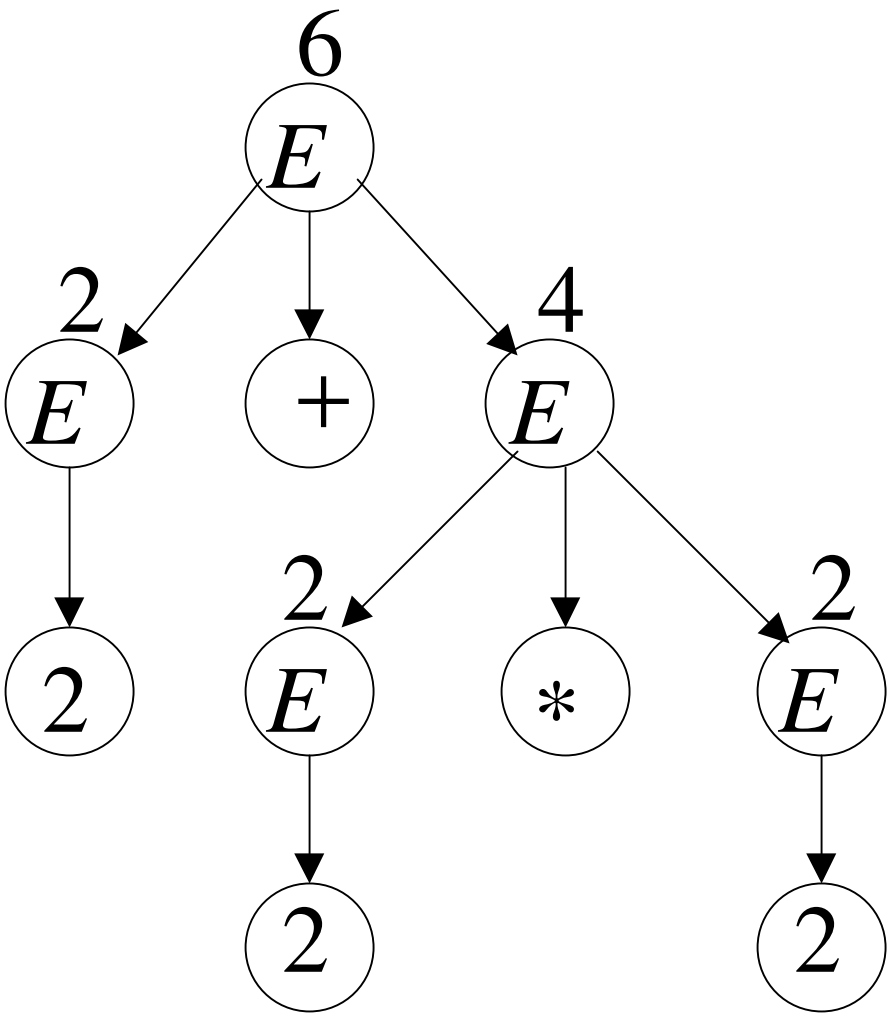


$2+2*2$

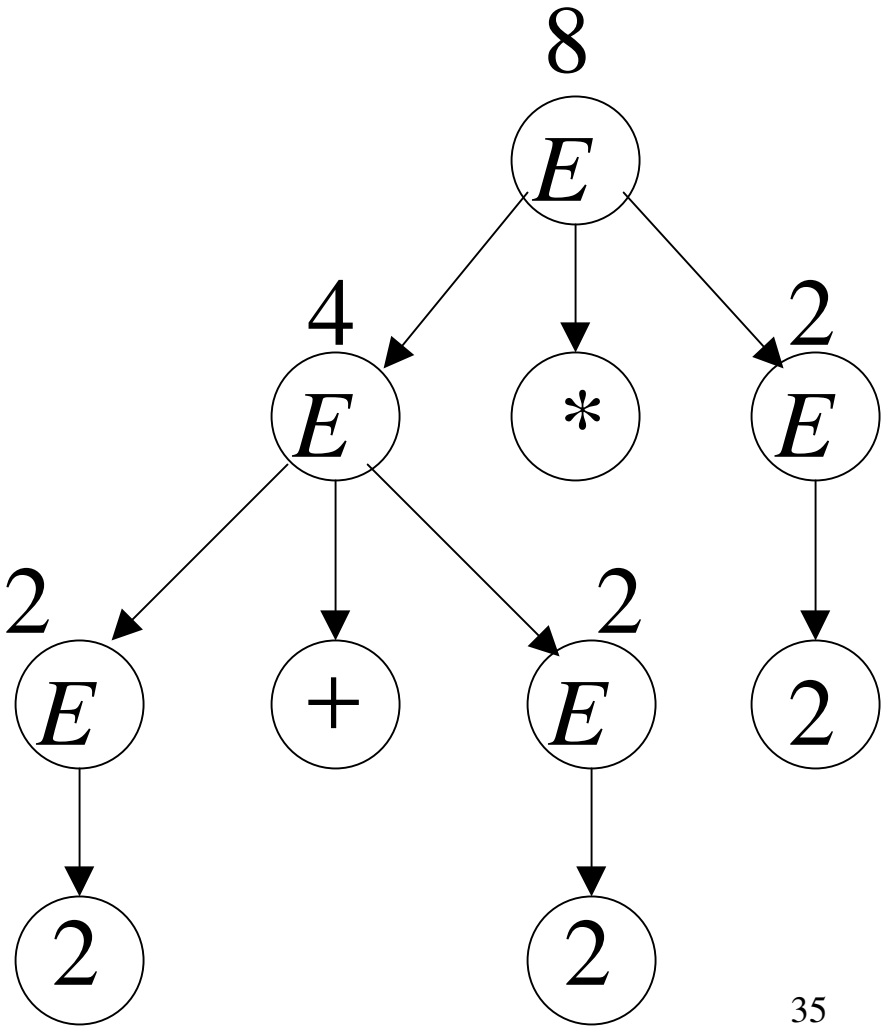




$2+2*2=6$

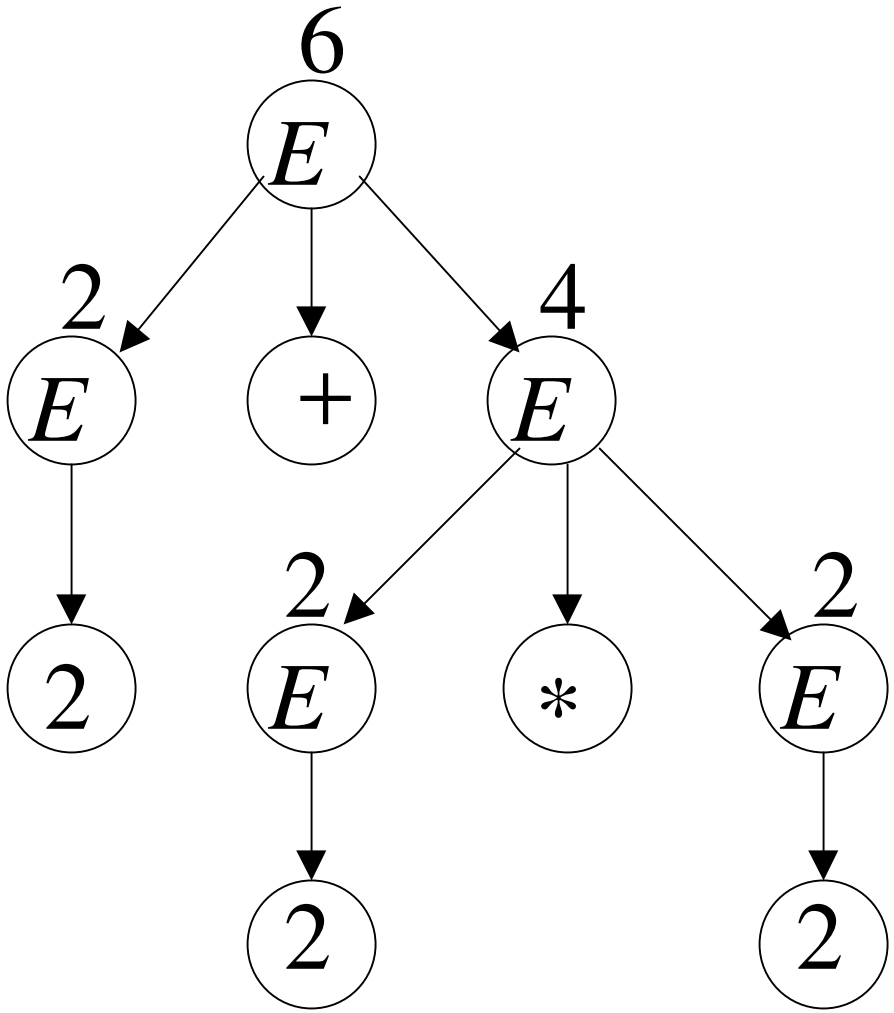


$2+2*2=8$



The right solution:

$$2+2*2=6$$



Therefore:

Ambiguity is bad for programming languages

We want to remove the ambiguity

We fix the ambiguous grammar:

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

New grammar:

$$E \to E + T$$

$$E \to T$$

$$T \to T * F$$

$$T \to F$$

$$F \to (E)$$

$$F \to a$$

$$E \Rightarrow E+T \Rightarrow T+T \Rightarrow F+T \Rightarrow a+T \Rightarrow a+T*F$$
$$\Rightarrow a+F*F \Rightarrow a+a*F \Rightarrow a+a*a$$

$E \rightarrow E+T$

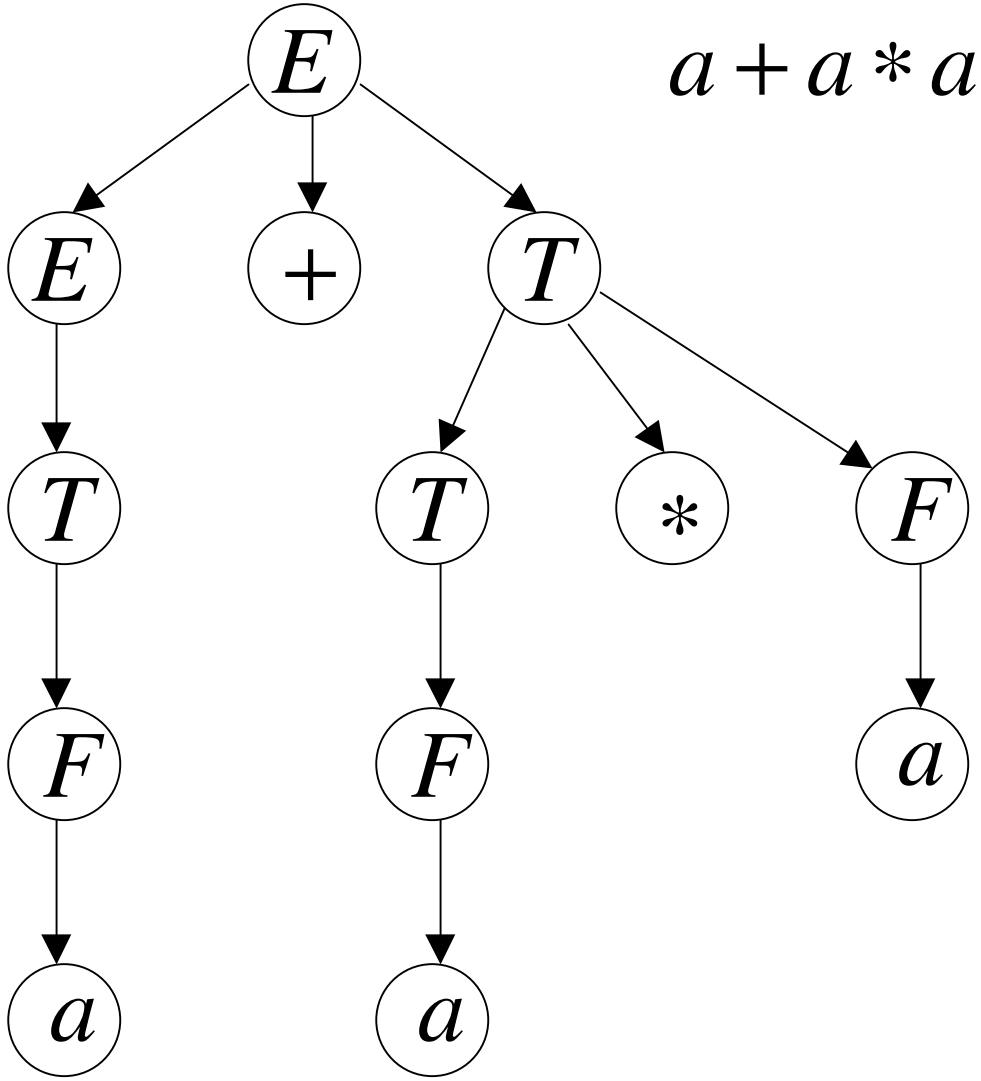
$E \rightarrow T$

$T \rightarrow T*F$

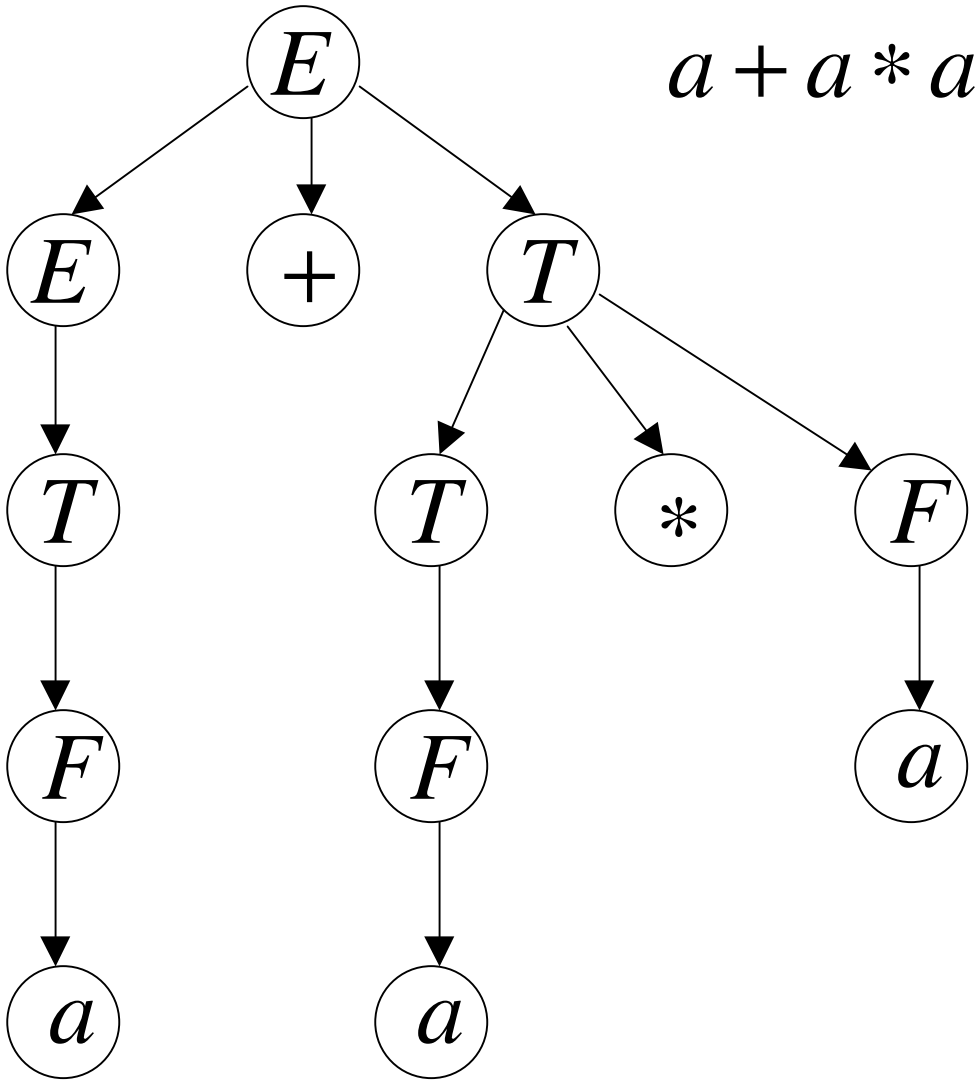
$T \rightarrow F$

$F \rightarrow (E)$

$F \rightarrow a$



Unique derivation tree



$a + a * a$

The grammar

$$E \to E + T$$

$$E \to T$$

$$T \to T * F$$

$$T \to F$$

$$F \to (E)$$

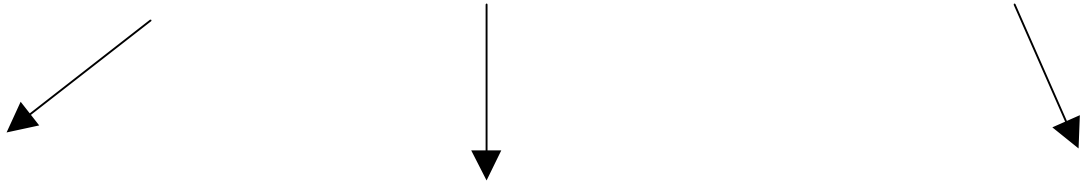
$$F \to a$$

Is not ambiguous

Inherent Ambiguity

Some context free languages have only ambiguous grammars

Example: $L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\}$



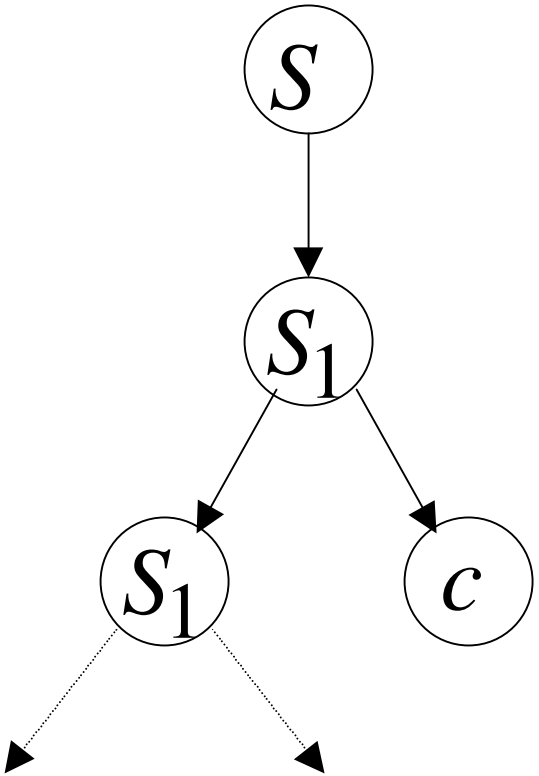
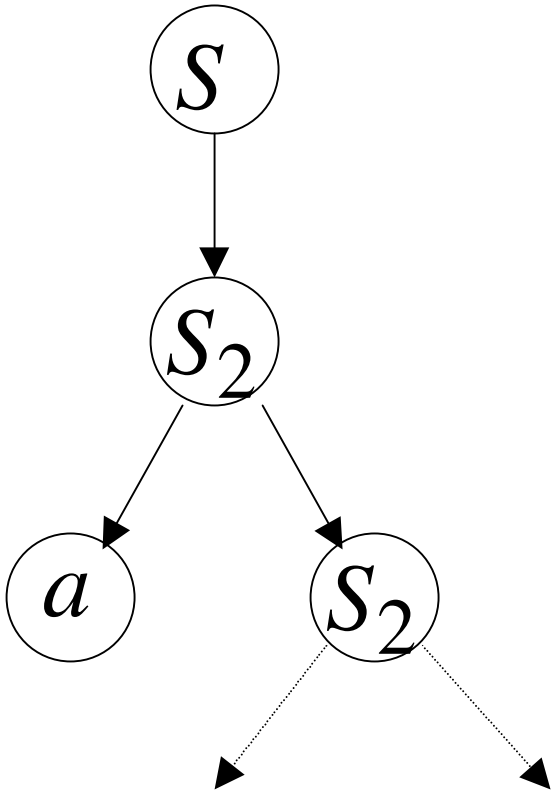
$S \to S_1 \mid S_2$

$S_1 \to S_1 c \mid A$

$A \to aAb \mid \lambda$

$S_2 \to aS_2 \mid B$

$B \to bBc \mid \lambda$

The string $a^n b^n c^n$

has two derivation trees