Formal Languages

Context-Free Languages

Module – 3	
CONTEXT-FREE LANGUAGES AND SIMPLIFICATION OF CONTEXT-FREE GRAMMARS AND NORMAL FORMS: Context-Free grammars, Parsing and Ambiguity, Methods for Transforming Grammars, Two important Normal Forms. **Text Book 1:** Chapter 5: 5.1 -5.2, Chapter 6: 6.1 – 6.2	**05 Hours**

Module – 3	
CONTEXT-FREE LANGUAGES AND SIMPLIFICATION OF CONTEXT-FREE GRAMMARS AND NORMAL FORMS: Context-Free grammars, Parsing and Ambiguity, Methods for Transforming Grammars, Two important Normal Forms. **Text Book 1:** Chapter 5: 5.1 -5.2, Chapter 6: 6.1 – 6.2	**05 Hours**

Context-Free Grammars

- The productions in a regular grammar are restricted in two ways:

  → The left side must be a single variable,

  → The right side has a special form.

- To create grammars that are more powerful, we must relax some of these restrictions.

- By retaining the restriction on the left side, but permitting anything on the right, we get

"context-free grammars"

Regular Grammar – It is either Right Linear or Left Linear

A grammar $G = (V, T, S, P)$ is said to be **right-linear** if all productions are of the form

$$A \rightarrow xB,$$

$$A \rightarrow x,$$

where $A, B \in V$, and $x \in T^*$. A grammar is said to be **left-linear** if all productions are of the form

$$A \rightarrow Bx,$$

or

$$A \rightarrow x.$$

A regular grammar is one that is either right-linear or left-linear.

Context-free Grammars

Definition: A grammar $G = (V, T, S, P)$ is said to be context-free if all production rules in P have the form

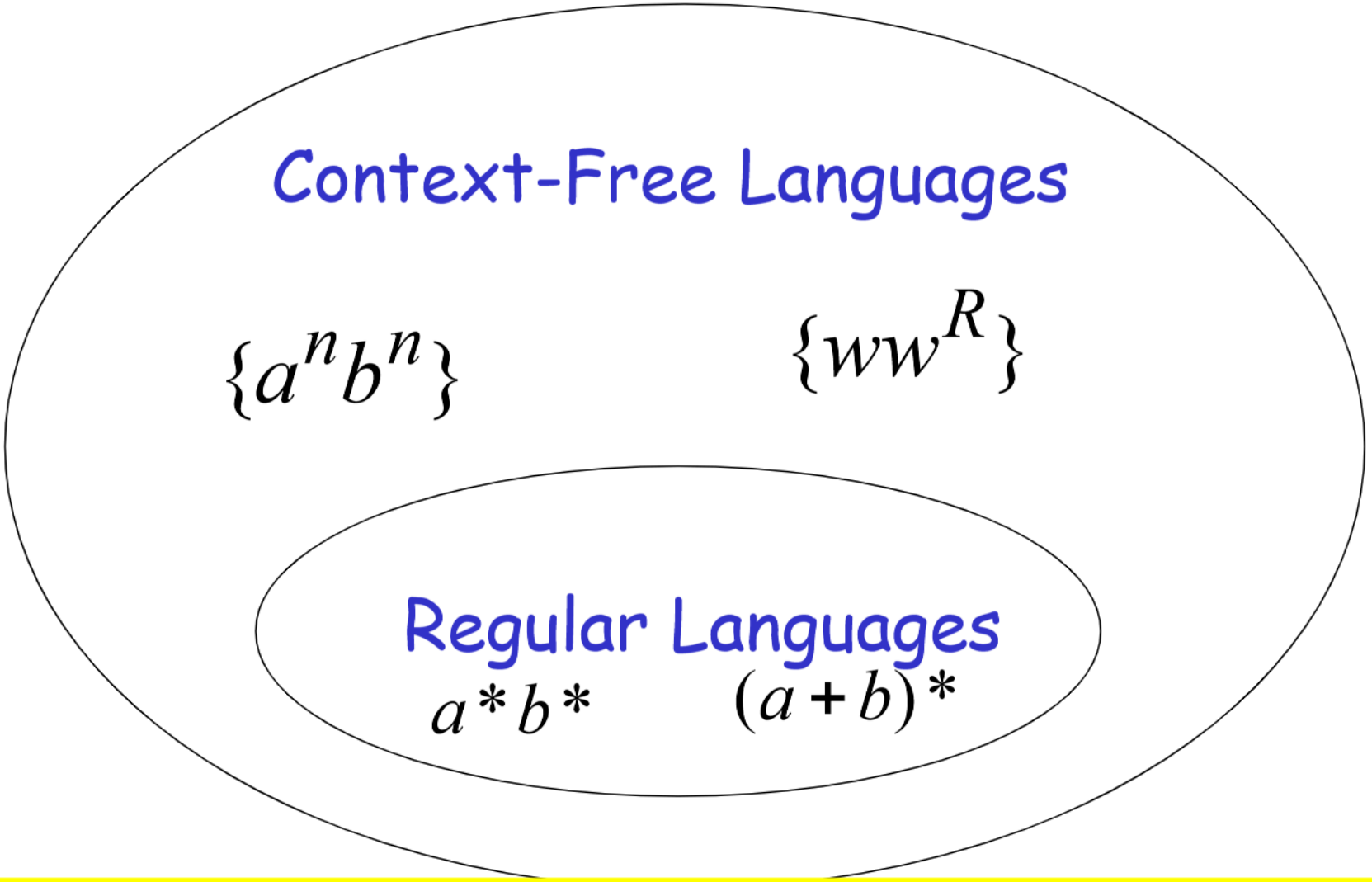
$$A \rightarrow x$$

where $A \in V$ and $x \in (V \cup T)^*$

A language is said to be context-free iff there is a context free grammar G such that $L = L(G)$.

Context-Free Languages

All Regular Grammar is Context Free



Every regular grammar is Context Free…so reg lang is also context free

- Non regular languages like $\{a^n b^n\}$ and $\{ww^R\}$ can be generated by Context Free languages.

- Family of regular languages is a proper subset of family of context free languages.

Example of Context Free Languages

Example 1:

The grammar $G = (\{S\}, \{a, b\}, S, P)$, with productions

$$S \to aSa,$$
$$S \to bSb,$$
$$S \to \lambda,$$

is context-free. A typical derivation in this grammar is

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa.$$

This, and similar derivations, make it clear that

$$L(G) = \{ww^R : w \in \{a, b\}^*\}.$$

Example of Context Free Languages

Grammar: $S \rightarrow aSb$

$S \rightarrow \lambda$

Derivation of sentence ab:

$$S \Rightarrow aSb \Rightarrow ab$$

$S \rightarrow aSb$ $\qquad$ $S \rightarrow \lambda$

Grammar: $S \to aSb$

$S \to \lambda$

Derivation of sentence $aabb$:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$$

$S \to aSb$ $\quad$ $S \to \lambda$

Other derivations:

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

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb$$
$$\Rightarrow aaaaSbbbb \Rightarrow aaaabbbb$$

Language of the grammar

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

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

More Notation

Grammar $G = (V, T, S, P)$

V: Set of variables

T: Set of terminal symbols

S: Start variable

P: Set of Production rules

Example

Grammar G : $S \rightarrow aSb$

$S \rightarrow \lambda$

$G = (V, T, S, P)$

$V = \{S\}$ $T = \{a, b\}$

$P = \{S \rightarrow aSb,\ S \rightarrow \lambda\}$

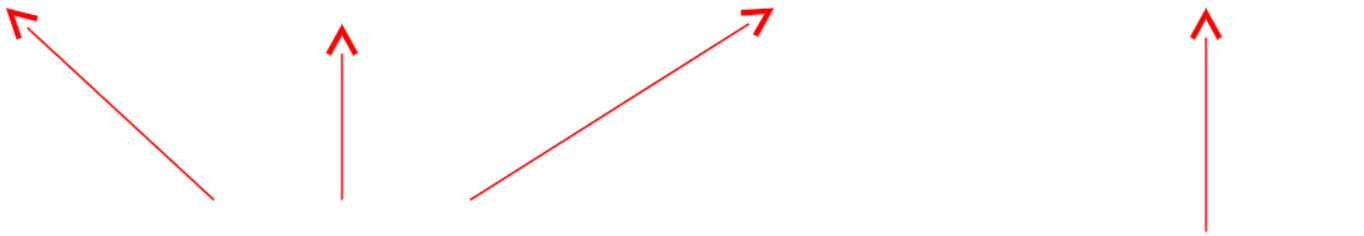
More Notation

Sentential Form:

A sentence that contains variables and terminals

Example:

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



Sentential Forms

sentence

We write: $S \overset{*}{\Rightarrow} aaabbb$

Instead of:

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

In general we write: $w_1 \overset{*}{\Rightarrow} w_n$

If: $w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \cdots \Rightarrow w_n$

Example

Grammar

$$S \rightarrow aSb$$

$$S \rightarrow \lambda$$

Derivations

$$S \overset{*}{\Rightarrow} \lambda$$

$$S \overset{*}{\Rightarrow} ab$$

$$S \overset{*}{\Rightarrow} aabb$$

$$S \overset{*}{\Rightarrow} aaabbb$$

Another Grammar Example

Grammar G :

$$S \rightarrow Ab$$

$$A \rightarrow aAb$$

$$A \rightarrow \lambda$$

Derivations:

$$S \Rightarrow Ab \Rightarrow b$$

$$S \Rightarrow Ab \Rightarrow aAbb \Rightarrow abb$$

$$S \Rightarrow Ab \Rightarrow aAbb \Rightarrow aaAbbb \Rightarrow aabbb$$

More Derivations

$$S \Rightarrow Ab \Rightarrow aAbb \Rightarrow aaAbbb \Rightarrow aaaAbbbb$$

$$\Rightarrow aaaaAbbbbb \Rightarrow aaaabbbbb$$

$$S \overset{*}{\Rightarrow} aaaabbbbb$$

$$S \overset{*}{\Rightarrow} aaaaaabbbbbbb$$

$$S \overset{*}{\Rightarrow} a^n b^n b$$

Example

For grammar G :

$$S \rightarrow Ab$$

$$A \rightarrow aAb$$

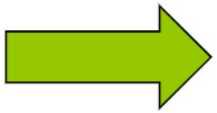
$$A \rightarrow \lambda$$

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

Since: $S \overset{*}{\Rightarrow} a^n b^n b$

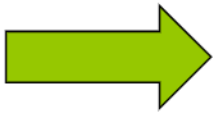
A Convenient Notation

$$A \rightarrow aAb$$
$$A \rightarrow \lambda$$



$$A \rightarrow aAb \mid \lambda$$

$$\langle article \rangle \rightarrow a$$
$$\langle article \rangle \rightarrow the$$



$$\langle article \rangle \rightarrow a \mid the$$

Example

A context-free grammar G:

$$S \to aSa$$

$$S \to bSb$$

$$S \to \lambda$$

A derivation:

$$S \Rightarrow aSa \Rightarrow abSba \Rightarrow abba$$

A context-free grammar G:

$$S \to aSa$$

$$S \to bSb$$

$$S \to \lambda$$

Another derivation:

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

$$S \rightarrow aSa$$

$$S \rightarrow bSb$$

$$S \rightarrow \lambda$$

$$L(G) = \{ww^R : \quad w \in \{a,b\}^*\}$$

Example

A context-free grammar G:

$$S \to aSb$$

$$S \to SS$$

$$S \to \lambda$$

Find out the language generated by this grammar…

$$S \rightarrow aSb$$

$$S \rightarrow SS$$

$$S \rightarrow \lambda$$

Language generated by the production

$$L = \{w \in \{a,b\}^* : n_a(w) = n_b(w) \text{ and } n_a(v) \geq n_b(v), \text{ where } v \text{ is any prefix of } w\}.$$

The Language $L = \{a^n b^m : n \neq m\}$ is context free

Case 1: n = m :

$$S \rightarrow aSb$$
$$S \rightarrow \lambda$$

Case 2: n > m :

$$S \rightarrow AS_1,$$
$$S_1 \rightarrow aS_1b|\lambda,$$
$$A \rightarrow aA|a.$$

Case 3: n < m :

$$S \rightarrow S_1B,$$
$$S_1 \rightarrow aS_1b|\lambda,$$
$$B \rightarrow bB|b.$$

Case 2 and 3 can be combined as

$$S \rightarrow AS_1|S_1B,$$
$$S_1 \rightarrow aS_1b|\lambda,$$
$$A \rightarrow aA|a,$$
$$B \rightarrow bB|b.$$

Definition: Context-Free Grammars

Grammar $G = (V, T, S, P)$

Variables

Terminal symbols

Start variable

Productions of the form:

$$A \rightarrow x$$

Variable

String of variables and terminals

Definition: Context-Free Languages

A language L is context-free

if and only if

there is a context-free grammar G

with $L = L(G)$ where $G = (V, T, S, P)$

$$L(G) = \{w: \ S \overset{*}{\Rightarrow} w, \ \ w \in T^*\}$$

Leftmost and Rightmost Derivations

In the grammar….that is not linear,

→ Derivation may involve sentential forms with more than one variable.

→ In such cases, we should have a choice in the order in which variables are replaced.

Leftmost and Rightmost Derivations

Example 1:

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

A derivation is said to be **leftmost** if in each step the leftmost variable in the sentential form is replaced.

If in each step the rightmost variable is replaced, we call the derivation **rightmost**.

Derivation Order

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

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:

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

$$L(G) = \{a^{2n}b^m : n \geq 0, m \geq 0\}$$

Example 2:

$$S \rightarrow aAB$$

$$A \rightarrow bBb$$

$$B \rightarrow A \mid \lambda$$

Language ???

ab^{2n} : n>=1

Leftmost derivation:

$$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB$$

$$\Rightarrow abbbbB \Rightarrow abbbbb$$

Rightmost derivation:

$$S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb$$

$$\Rightarrow abbBbb \Rightarrow abbbbb$$

Derivation Trees

Derivation Tree

- A second way of showing derivations, independent of the order in which productions are used, is by a **derivation** or **parse tree**.

- A derivation tree is an ordered tree in which **nodes** are labeled with the **left sides** of productions .

- The **children** of a **node** represent its corresponding **right sides**.

Let $G = (V, T, S, P)$ be a context-free grammar. An ordered tree is a derivation tree for G if and only if it has the following properties.

1. The root is labeled S.

2. Every leaf has a label from $T \cup \{\lambda\}$.

3. Every interior vertex (a vertex that is not a leaf) has a label from V.

4. If a vertex has label $A \in V$, and its children are labeled (from left to right) $a_1, a_2, \ldots, a_n$, then P must contain a production of the form

$$A \rightarrow a_1 a_2 \cdots a_n.$$

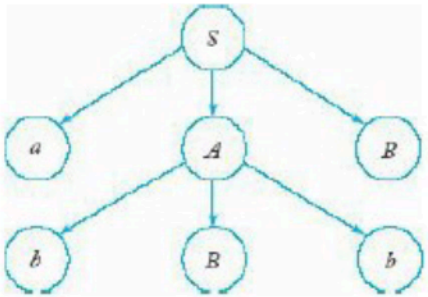
5. A leaf labeled λ has no siblings, that is, a vertex with a child labeled λ can have no other children.

If every leaf has a label from $V \cup T \cup \{\lambda\}$, is said to be a **partial derivation tree.**

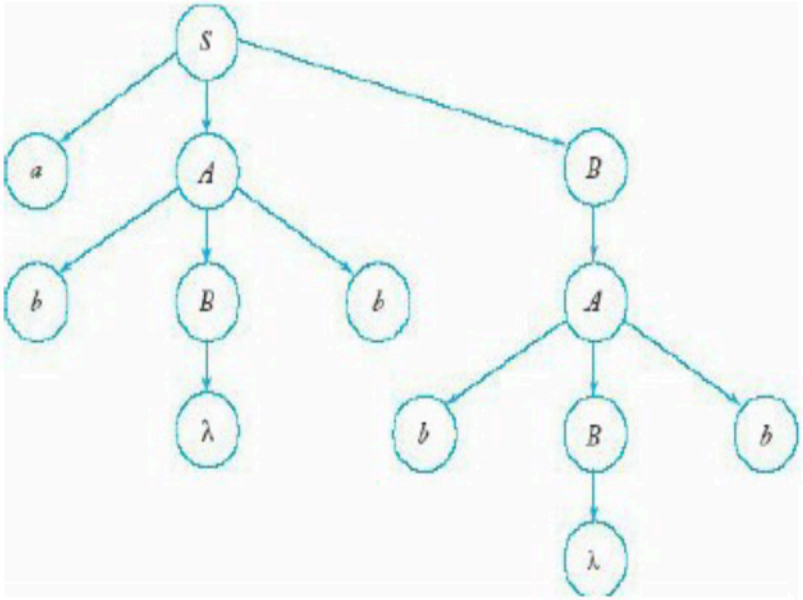
For the CFG...

$$S \rightarrow aAB,$$
$$A \rightarrow bBb,$$
$$B \rightarrow A|\lambda.$$

Partial Derivation tree

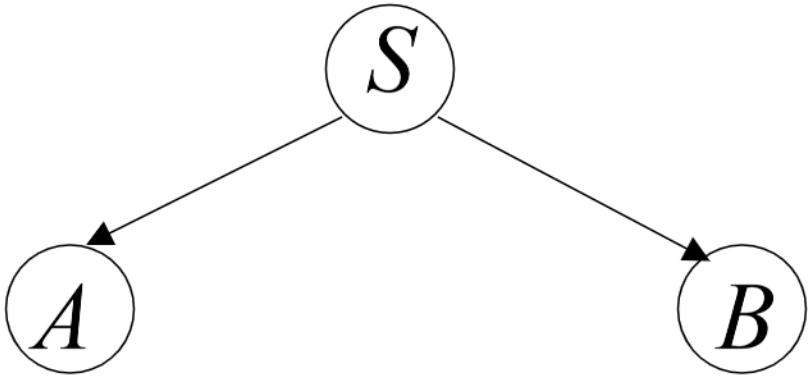


Derivation tree



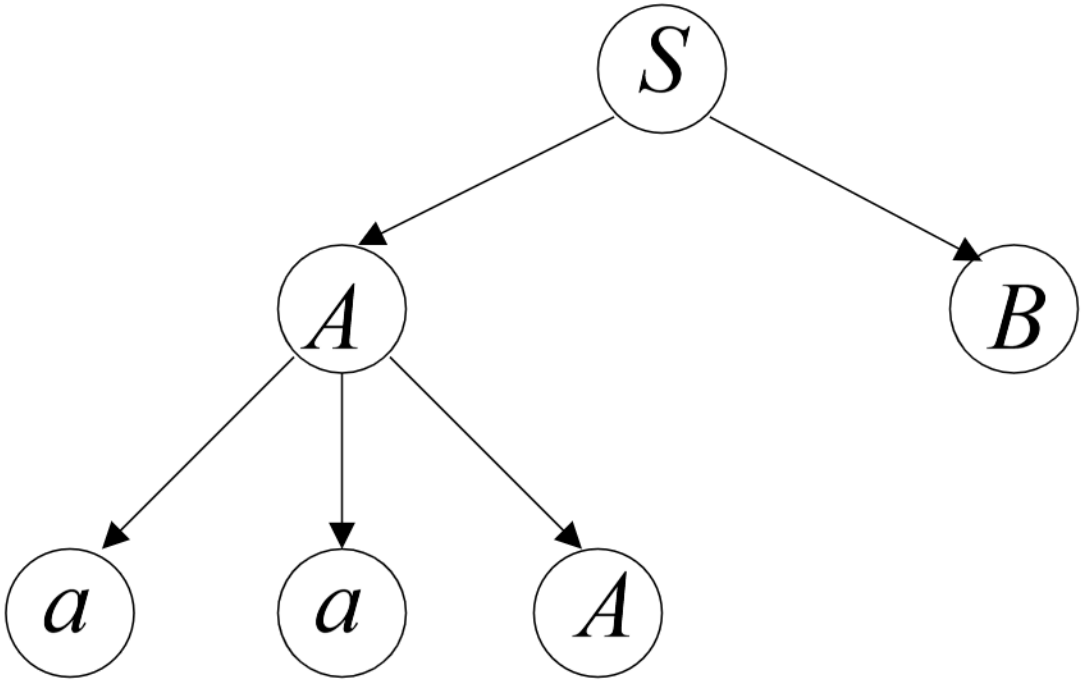
$S \rightarrow AB$ $A \rightarrow aaA \mid \lambda$ $B \rightarrow Bb \mid \lambda$

$S \Rightarrow AB$



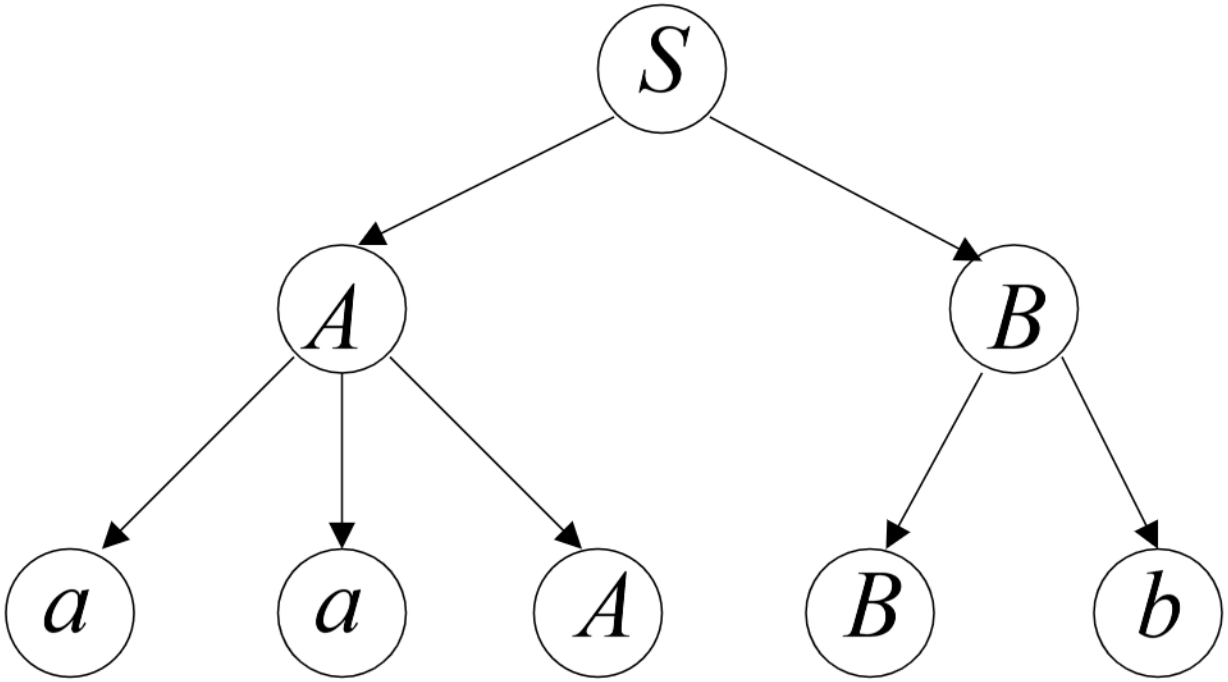
$S \rightarrow AB \qquad A \rightarrow aaA \mid \lambda \qquad B \rightarrow Bb \mid \lambda$

$S \Rightarrow AB \Rightarrow aaAB$



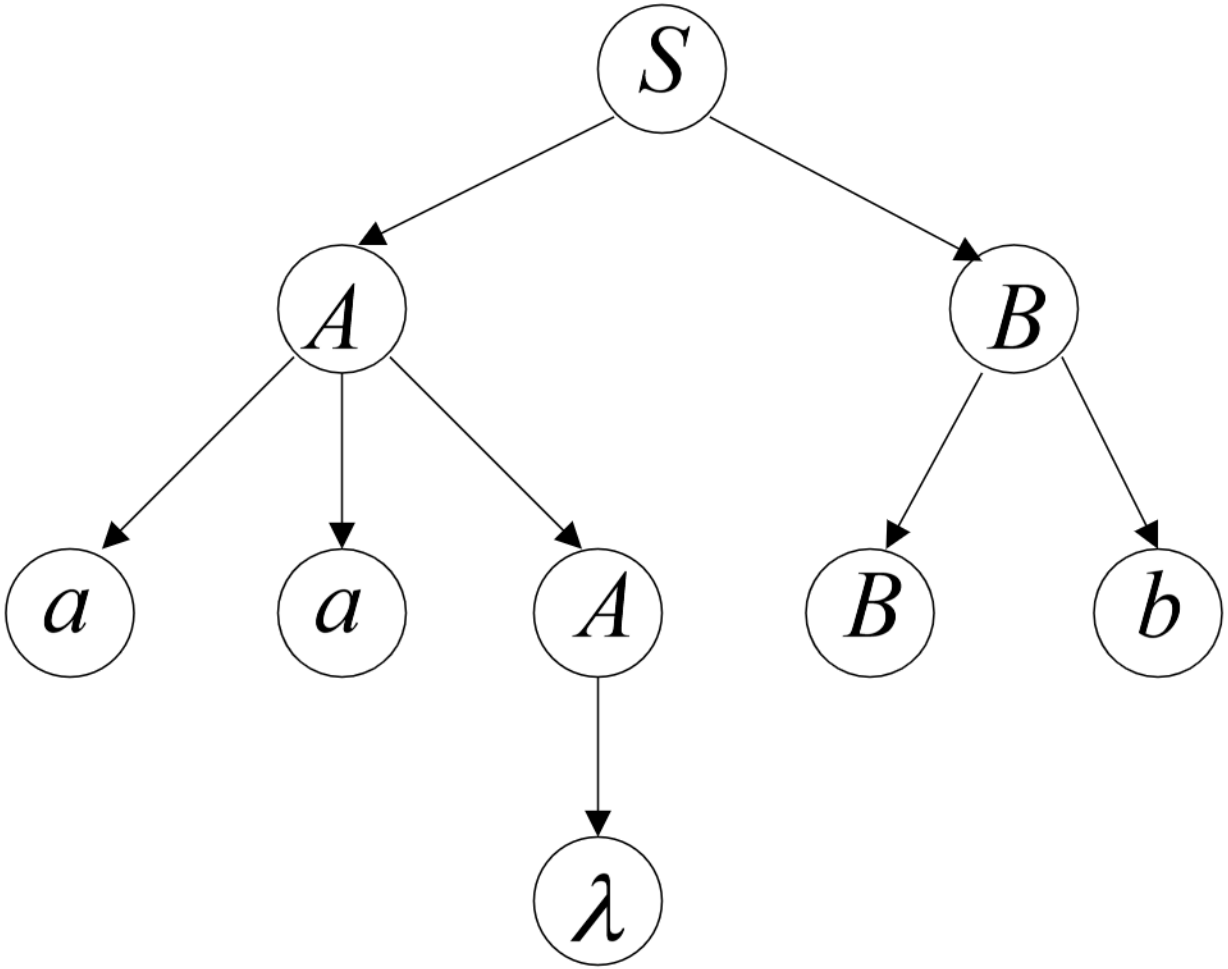
$S \rightarrow AB \qquad A \rightarrow aaA \mid \lambda \qquad B \rightarrow Bb \mid \lambda$

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



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

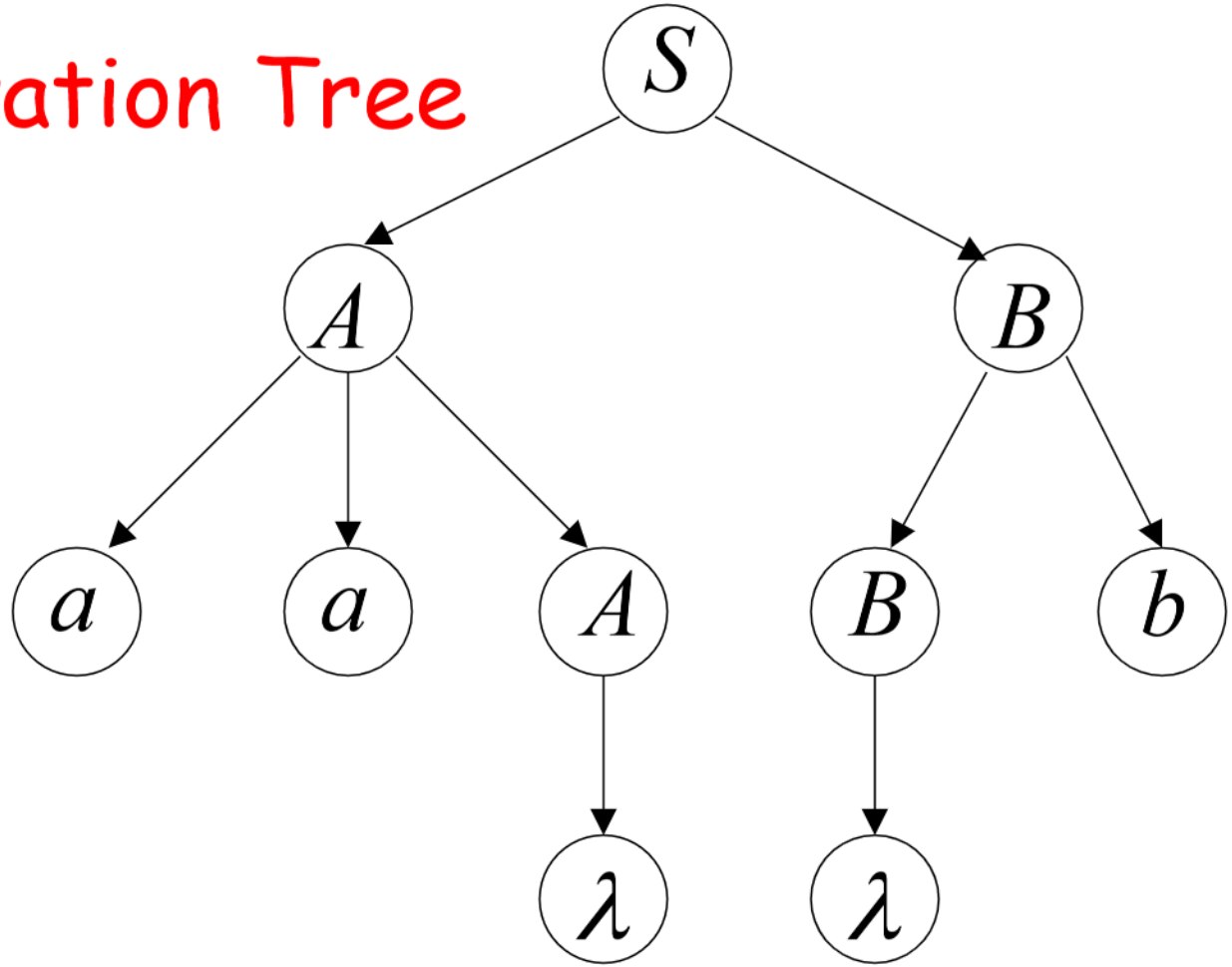
$$S \Rightarrow AB \Rightarrow aaAB \Rightarrow aaABb \Rightarrow aaBb$$



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

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

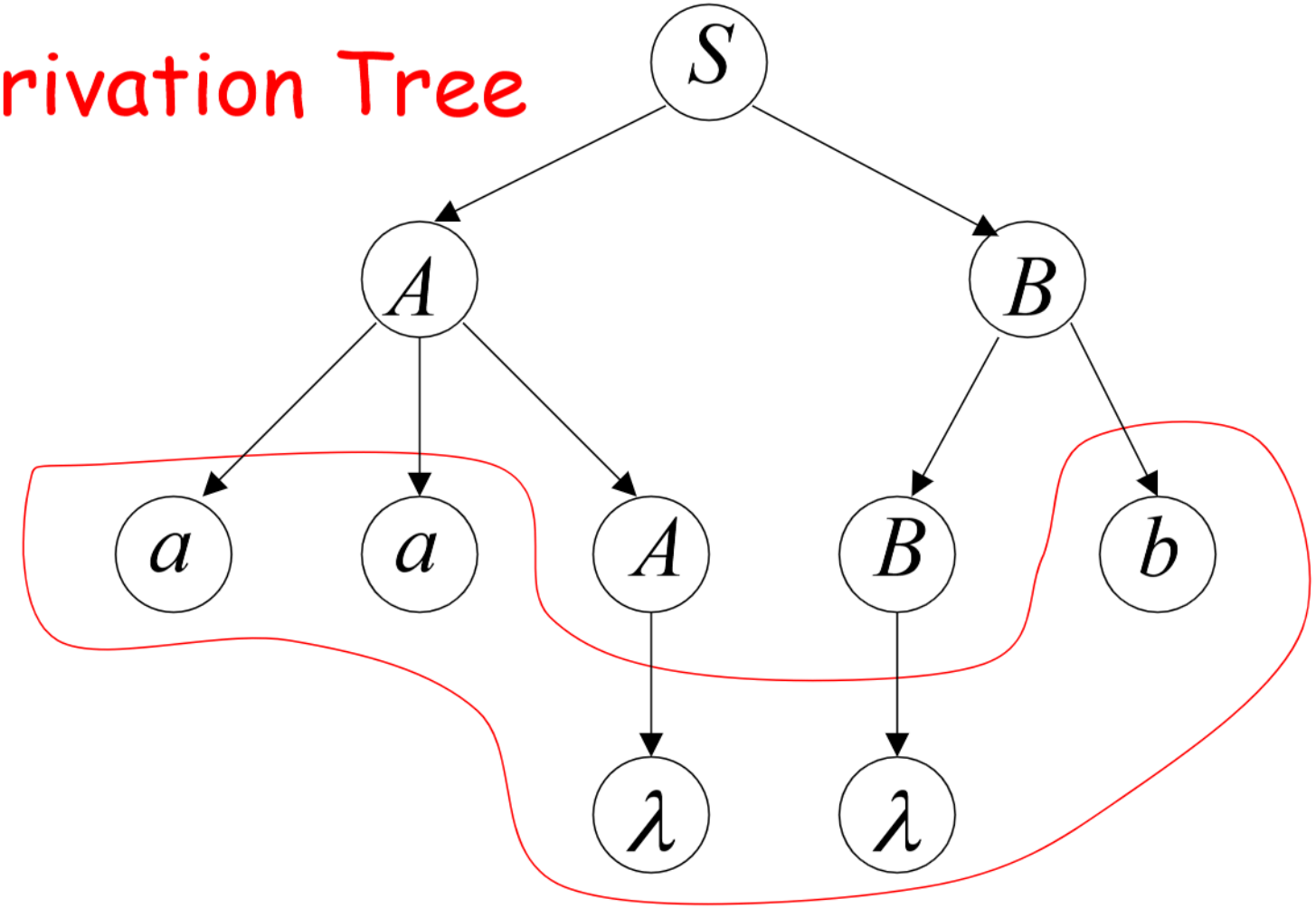
Derivation Tree



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

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

Derivation Tree



yield

$aa\lambda\lambda b$

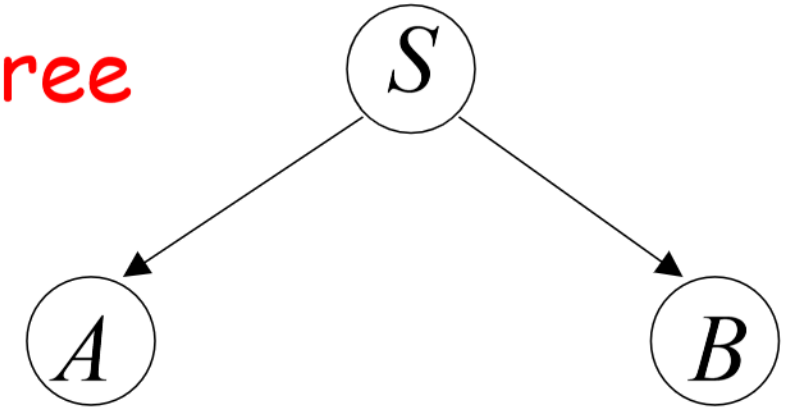
$= aab$

Partial Derivation Trees

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

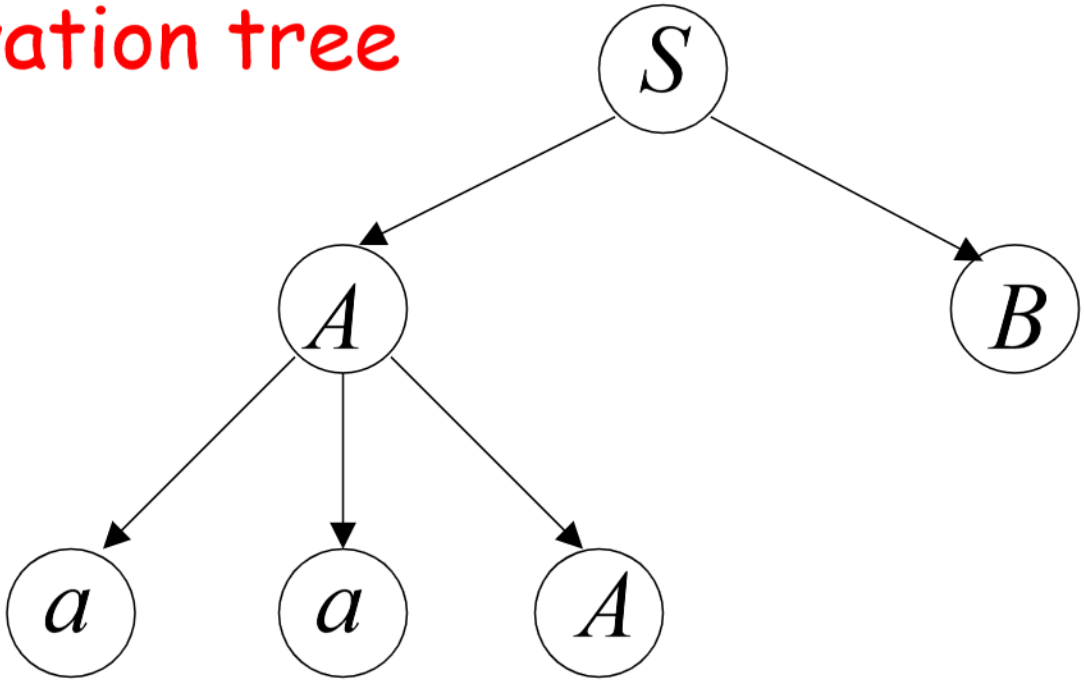
$S \Rightarrow AB$

Partial derivation tree



$$S \Rightarrow AB \Rightarrow aaAB$$

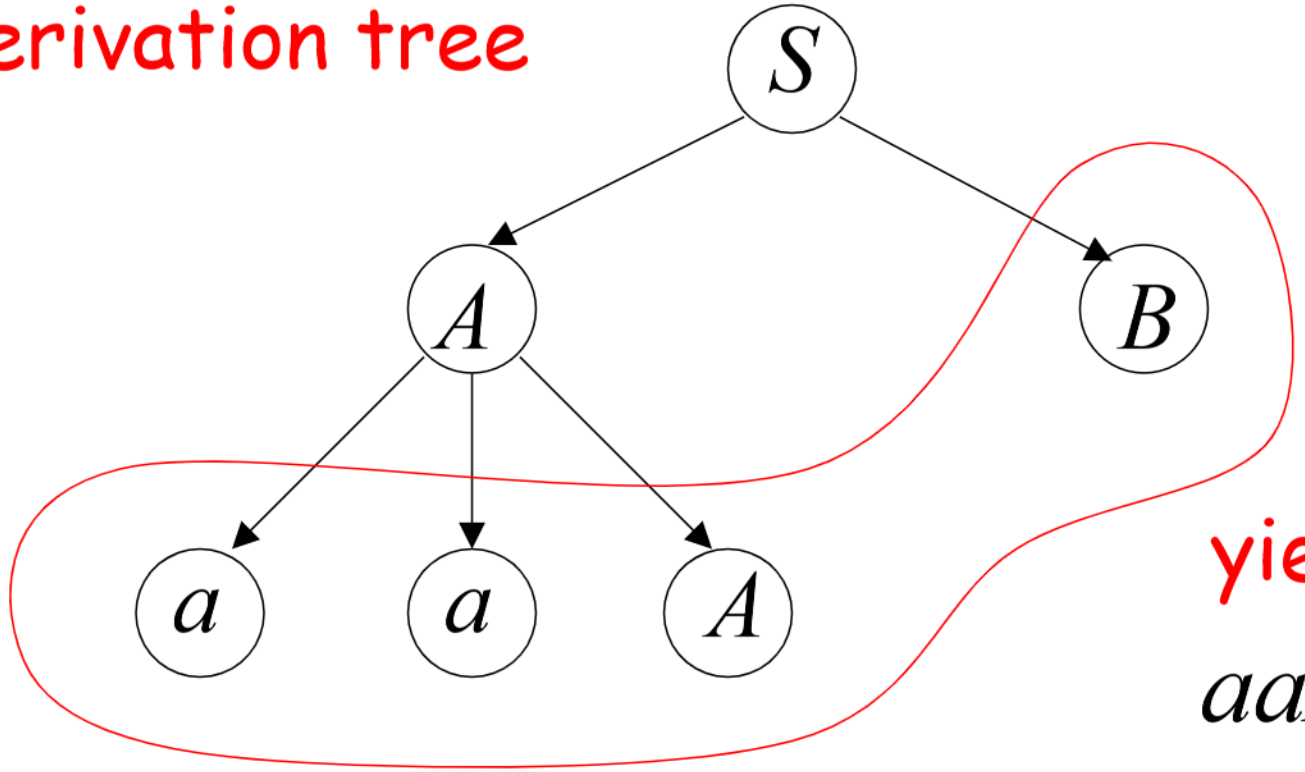
Partial derivation tree



$$S \Rightarrow AB \Rightarrow aaAB$$

sentential form

Partial derivation tree



yield

$aaAB$

Sometimes, derivation order doesn't matter

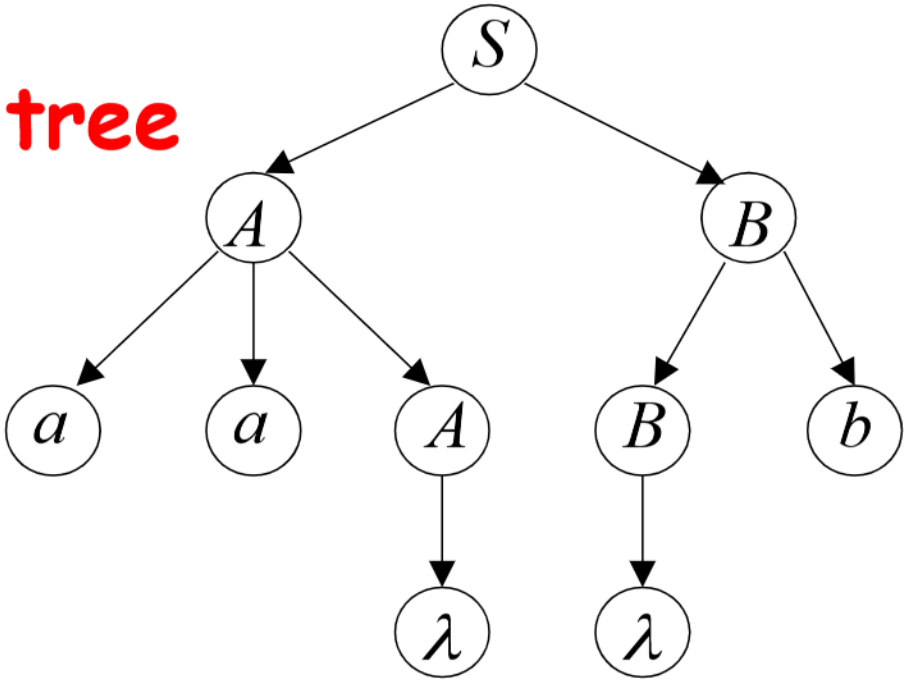
Leftmost:

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

Rightmost:

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

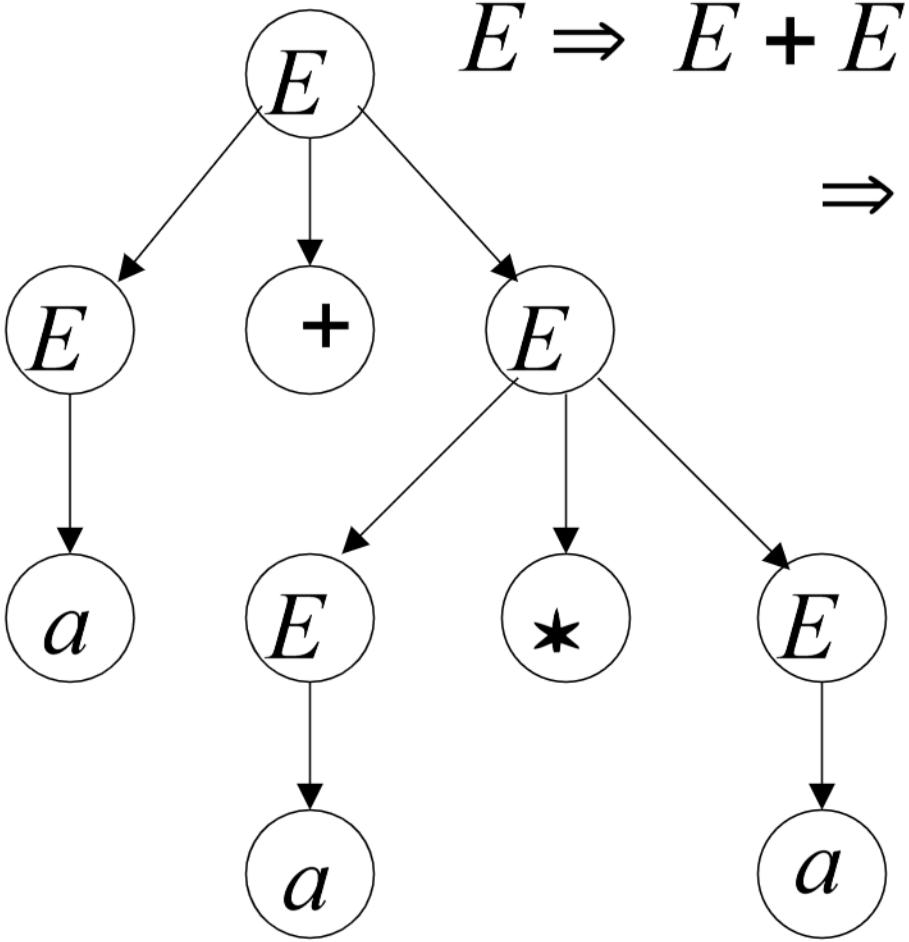
Same derivation tree



Ambiguity

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

$$a + a * a$$



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

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

leftmost derivation

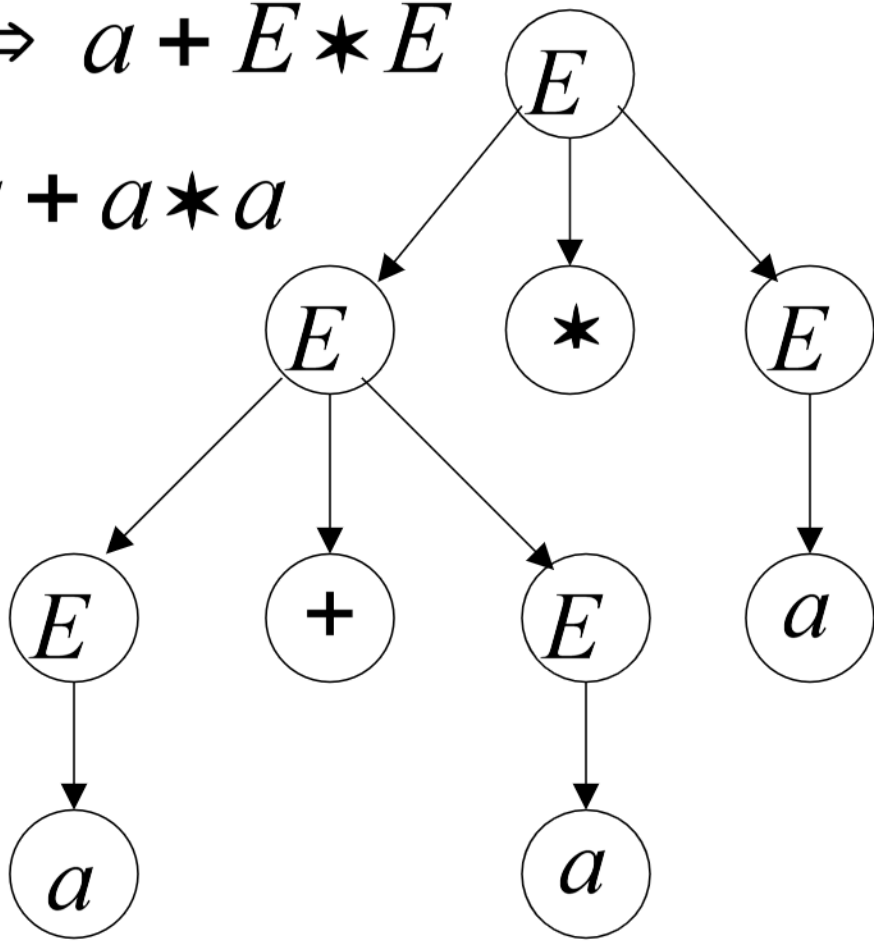
$$E \rightarrow E + E \mid E * E \mid (E) \mid a$$

$$a + a * a$$

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

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

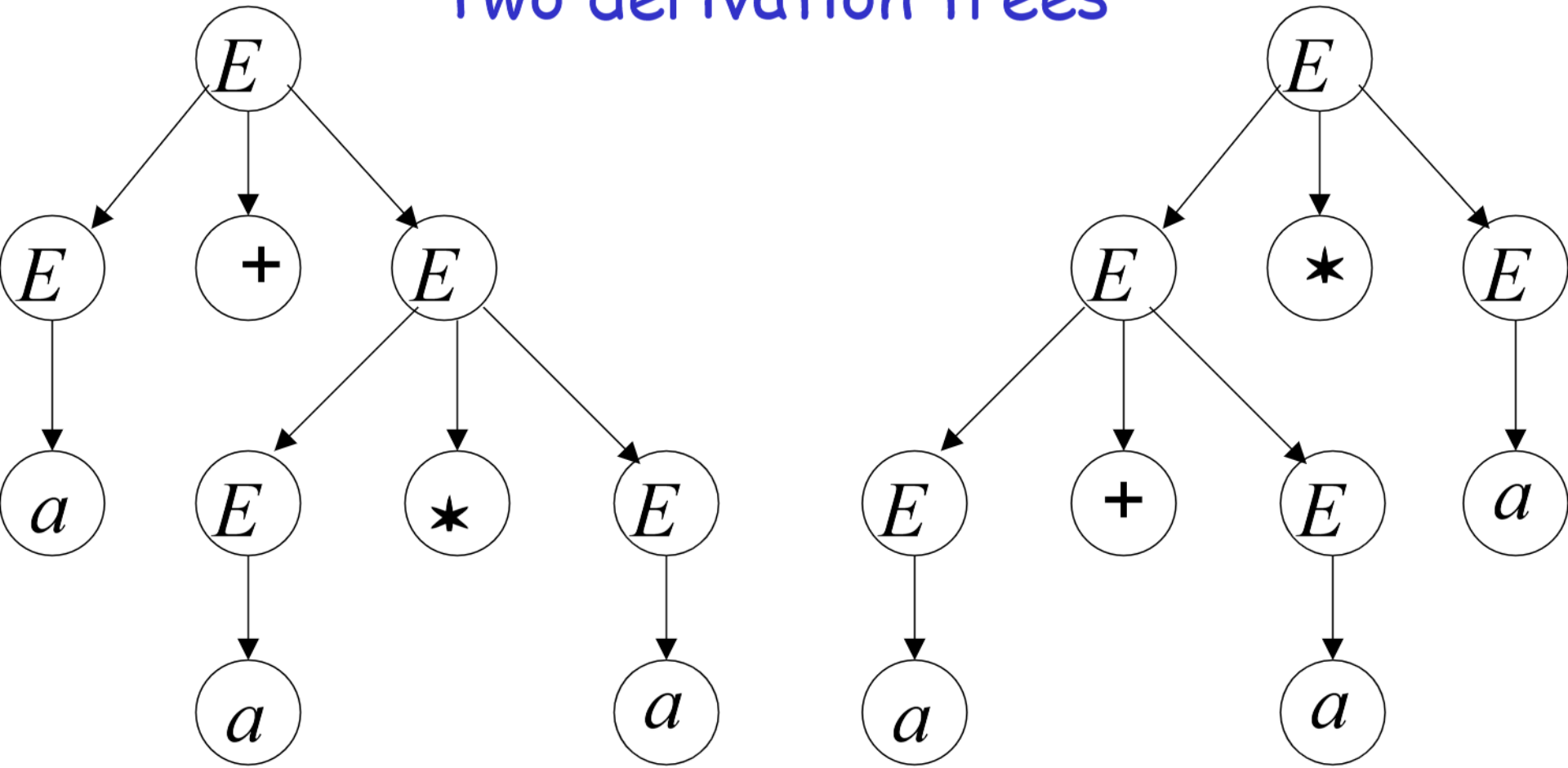
leftmost derivation



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

$$a + a * a$$

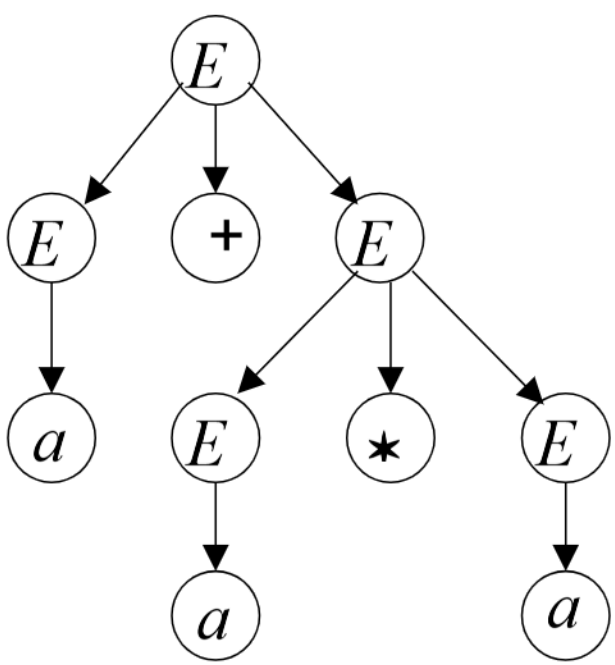
Two derivation trees

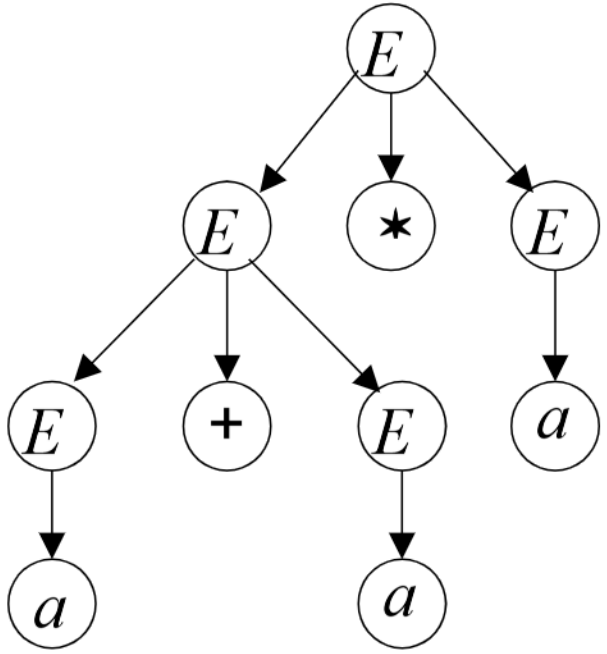


The grammar $E \rightarrow E + E \mid E * E \mid (E) \mid a$

is ambiguous:

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





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

is ambiguous:

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

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

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

Definition:

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:

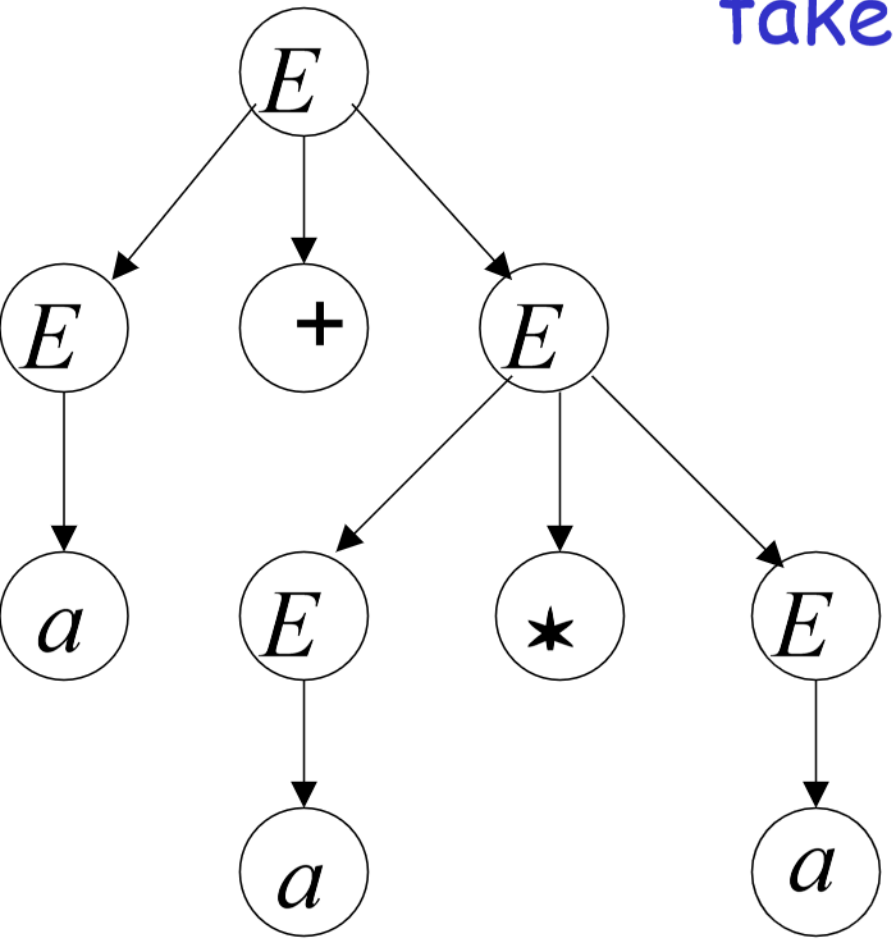
two or more leftmost derivations

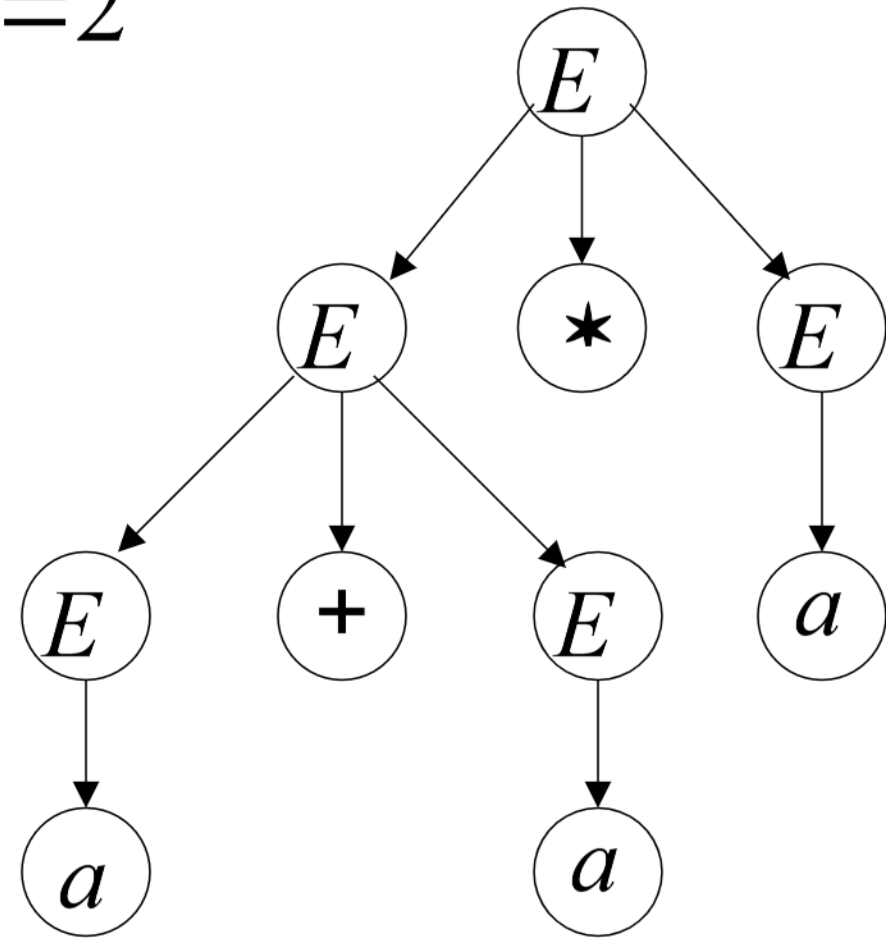
(or rightmost)

Why do we care about ambiguity?

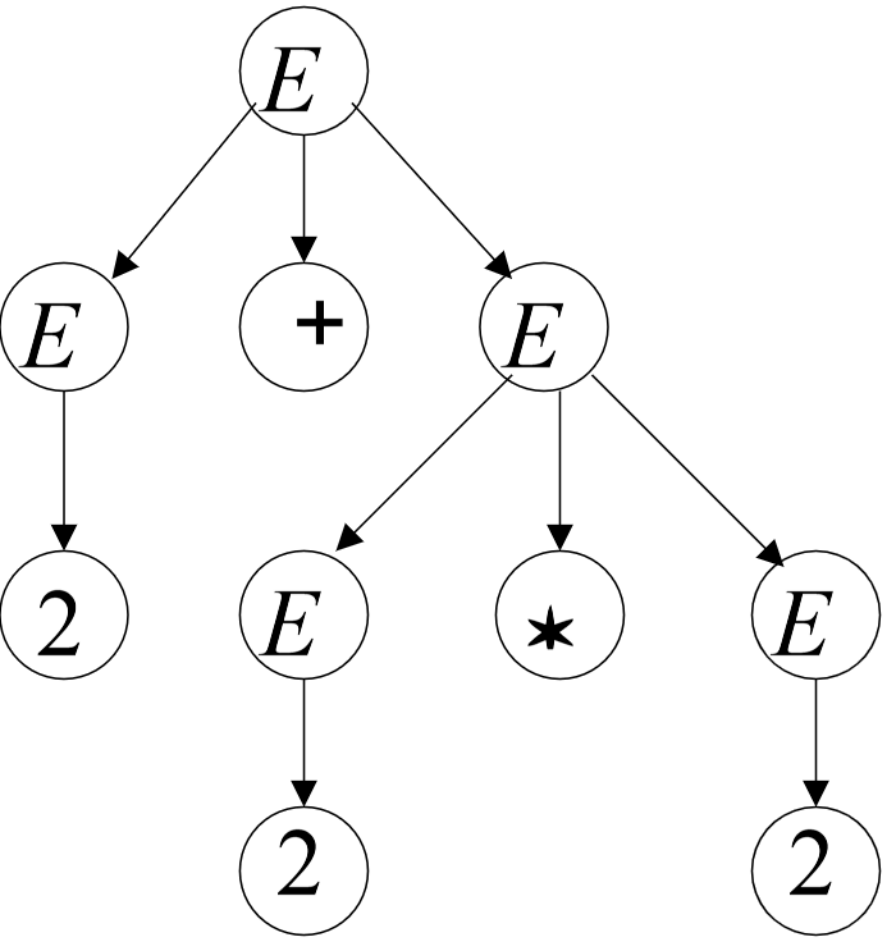
$$a + a * a$$

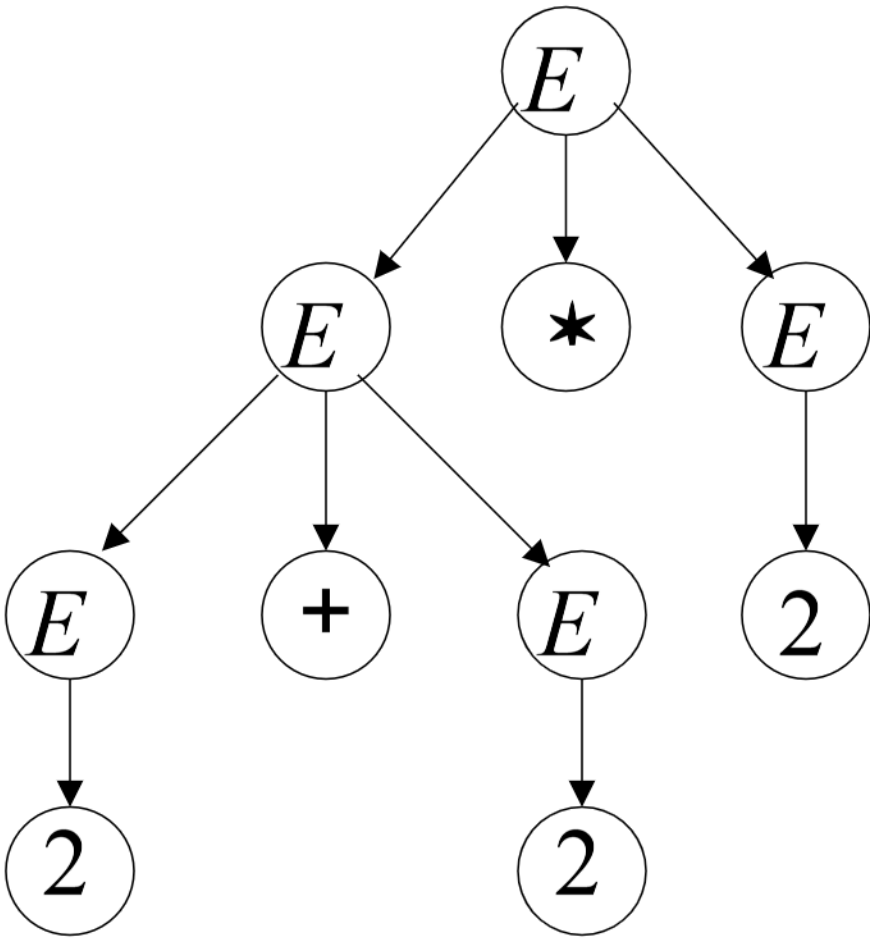
take $a = 2$



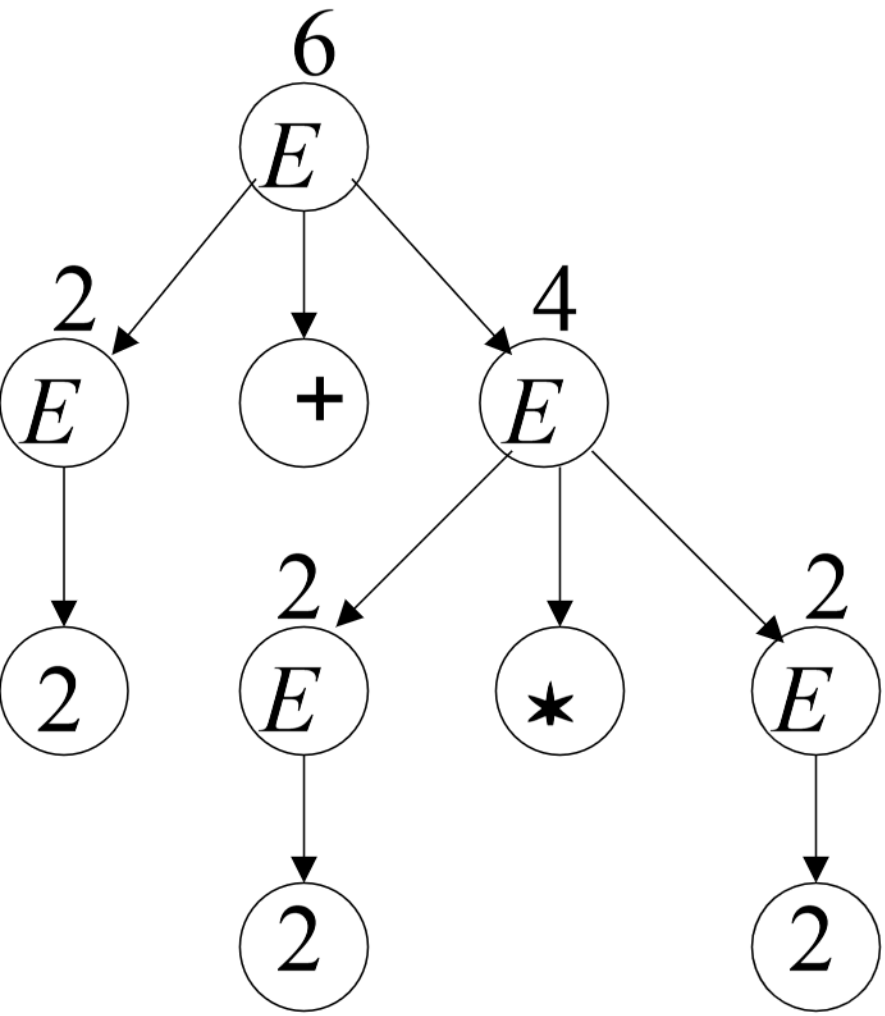


$2 + 2 * 2$

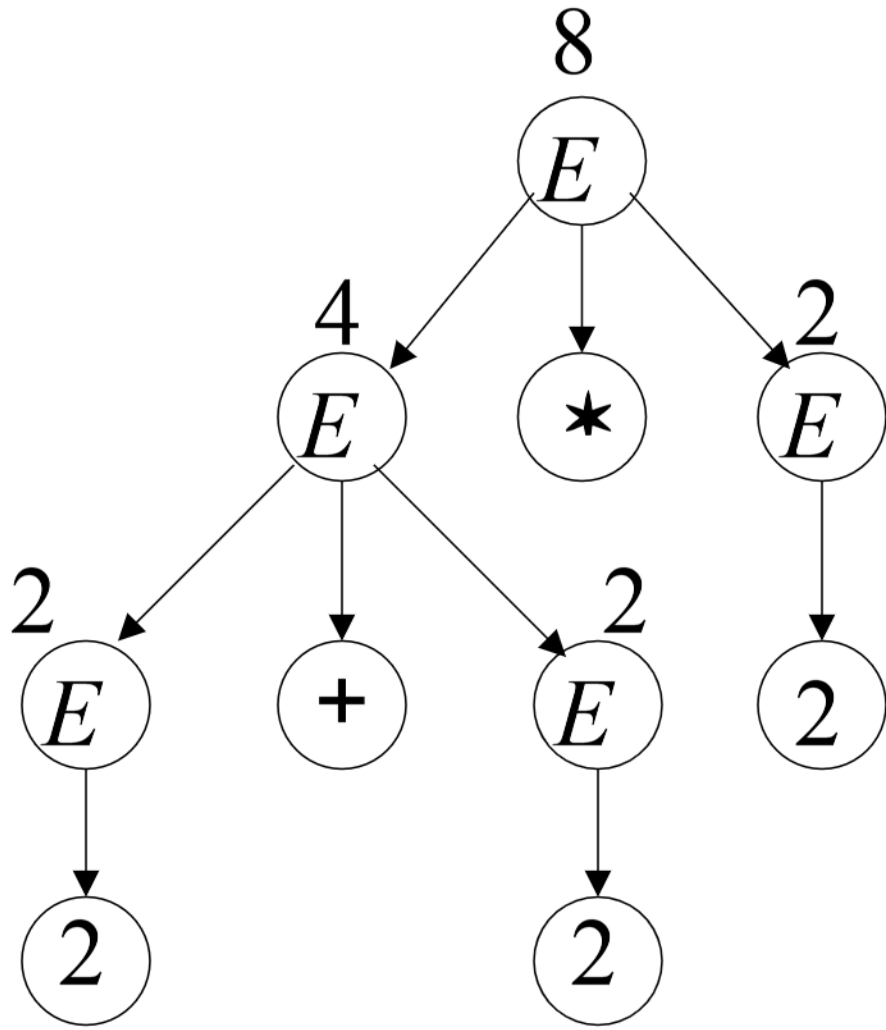




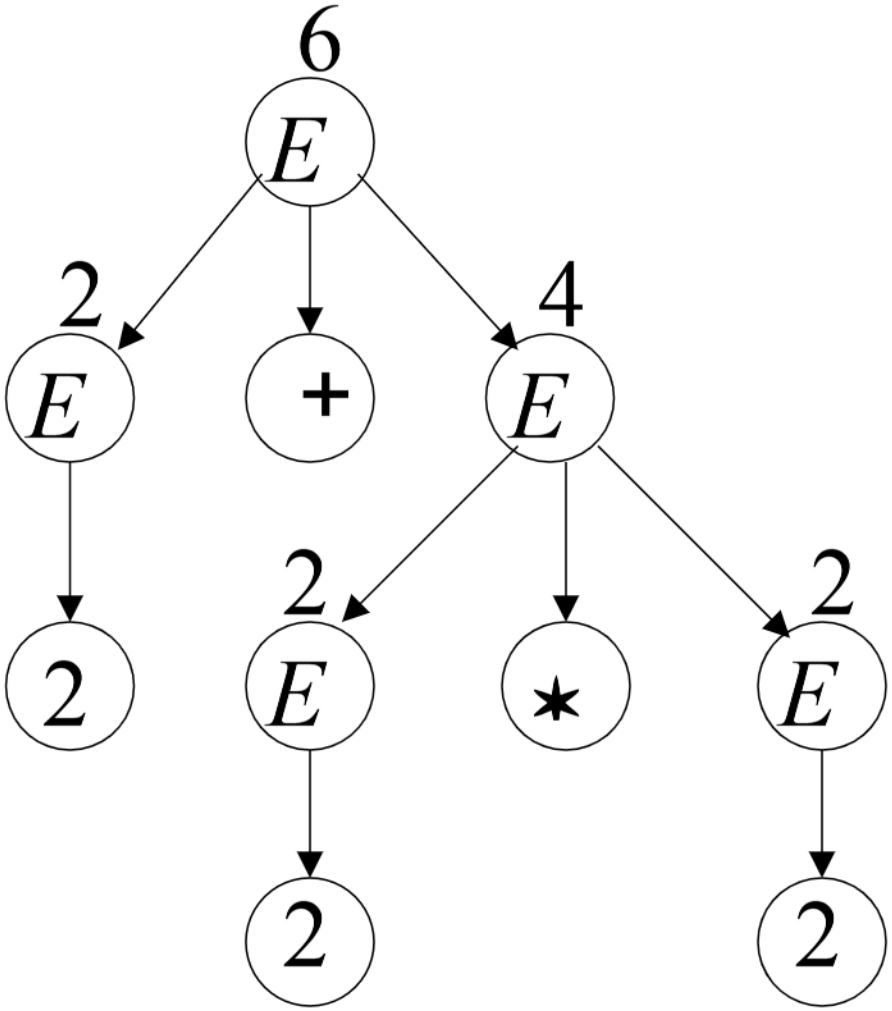
$2+2*2=6$



$2+2*2=8$



Correct result: $2+2*2=6$



- Ambiguity is **bad** for programming languages

- We want to remove ambiguity

We fix the ambiguous grammar:

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

New non-ambiguous grammar:

$$E \rightarrow E + T$$

$$E \rightarrow T$$

$$T \rightarrow T * F$$

$$T \rightarrow F$$

$$F \rightarrow (E)$$

$$F \rightarrow a$$

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

$E \rightarrow E+T$

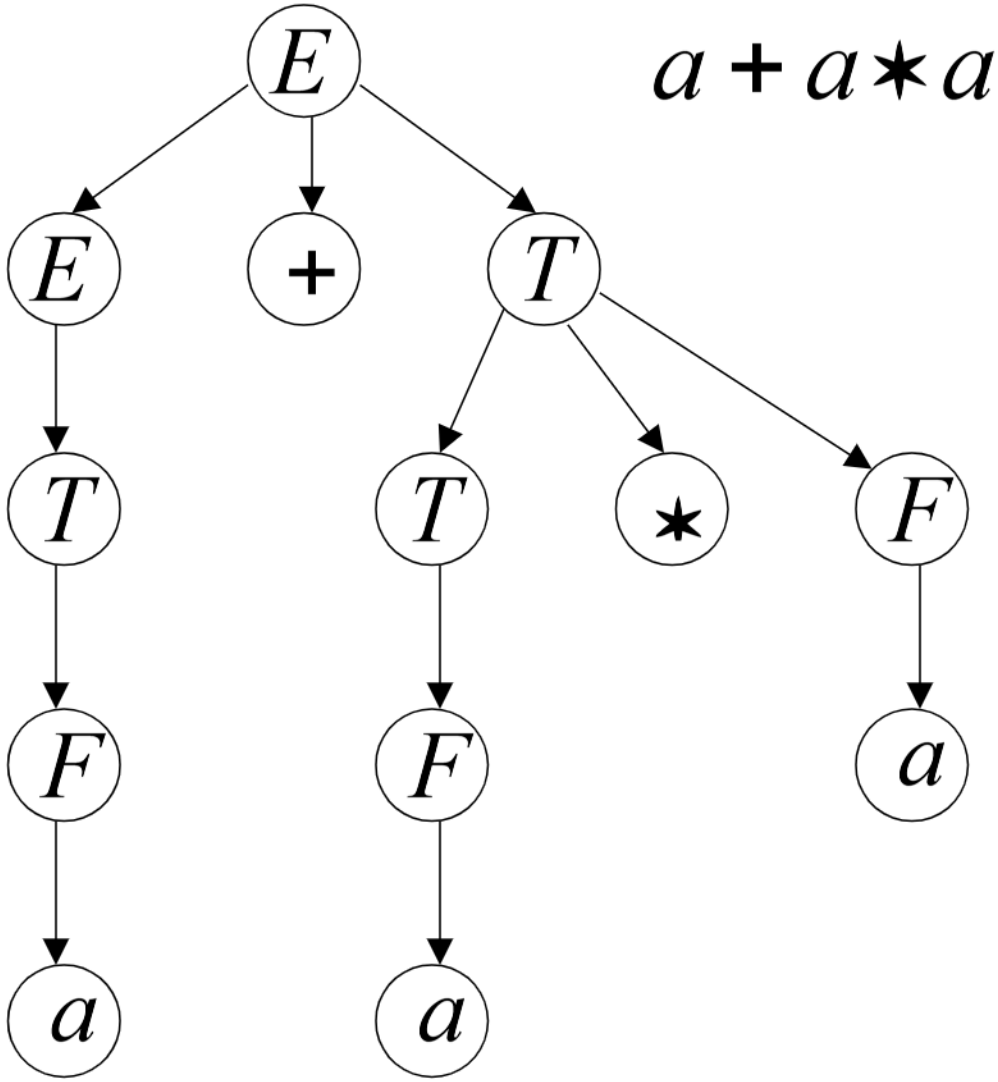
$E \rightarrow T$

$T \rightarrow T*F$

$T \rightarrow F$

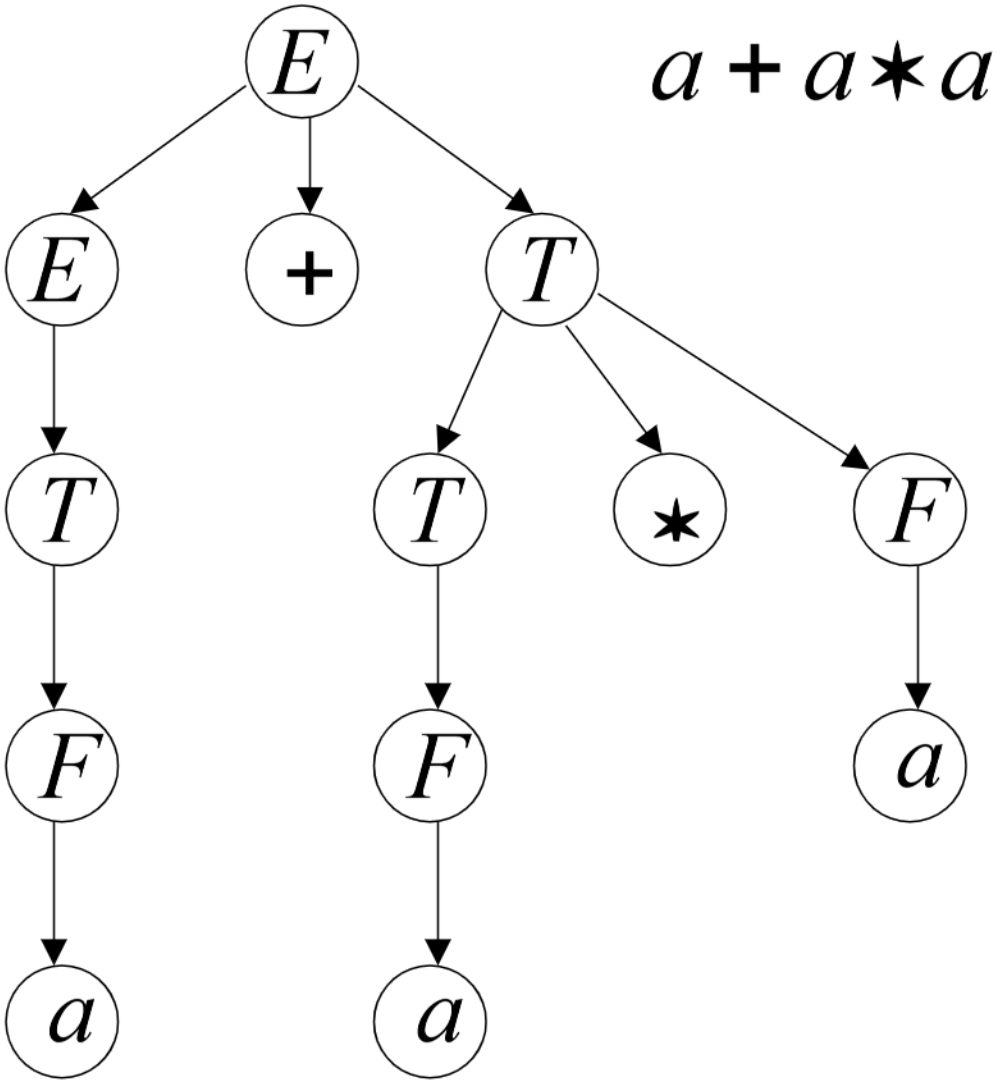
$F \rightarrow (E)$

$F \rightarrow a$



$a+a*a$

Unique derivation tree



The grammar G:

$$E \to E + T$$

$$E \to T$$

$$T \to T * F$$

$$T \to F$$

$$F \to (E)$$

$$F \to a$$

is non-ambiguous:

Every string $w \in L(G)$ has a unique derivation tree

Check for ambiguous grammar

Example 1: for string aaaa

S → aS | X
X → aX |a

Example 2: for string aabbab

S → aB | bA
A → aS |bAA|a
B → bS|aBB|b

Example 3: for string aababb

$S \rightarrow aSbS$

$S \rightarrow bSaS$

$S \rightarrow \lambda$

Example 4: for the strings aab and aaabb

$S \rightarrow aS \mid aSbS \mid \lambda$

Equivalent unambiguous grammar

$S \rightarrow aS \mid aAbS \mid \lambda$

$A \rightarrow aAbA \mid \lambda$

Find the unambiguous grammar for the following

$S \rightarrow S + S \mid a$

Let w = a+a+a

$S \rightarrow S+T \mid T$

$T \rightarrow a$

Parsing and Membership

- The term **parsing** describes finding a sequence of productions by which a $w \in L(G)$ is derived.
- An algorithm that can tell us whether w is in $L(G)$ is a membership algorithm
- We systematically construct all possible (say, leftmost) derivations and see whether any of them match w.
- We use **exhaustive search parsing** or **brute force parsing**. It is a form of **top-down parsing**, which we can view as the construction of a derivation tree from the root down.

Consider the grammar

$$S \rightarrow SS \,|aSb|\, bSa|\lambda$$

and the string $w = aabb$. Round one gives us

1. $S \Rightarrow SS$,
2. $S \Rightarrow aSb$,
3. $S \Rightarrow bSa$,
4. $S \Rightarrow \lambda$.

Here we remove 3 and 4 because string w cannot be derived from that productions.

Round 2:

$$
\begin{aligned}
S &\Rightarrow SS \Rightarrow SSS, \\
S &\Rightarrow SS \Rightarrow aSbS, \\
S &\Rightarrow SS \Rightarrow bSaS, \\
S &\Rightarrow SS \Rightarrow S,
\end{aligned}
$$

Round 3

$$
\begin{aligned}
S &\Rightarrow aSb \Rightarrow aSSb, \\
S &\Rightarrow aSb \Rightarrow aaSbb, \\
S &\Rightarrow aSb \Rightarrow abSab, \\
S &\Rightarrow aSb \Rightarrow ab.
\end{aligned}
$$

Round 4

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb.$$

Another example for parsing

- Try with

S → SS

S → aSb

S → bSa

S → Lambda

Check for the string w = abba

Parsing Continued…

See example 5.7….

Consider the grammar

$$S \rightarrow SS \,|aSb|\, bSa|\lambda$$

and the string $w = aabb$. Round one gives us

1. $S \Rightarrow SS$,
2. $S \Rightarrow aSb$,
3. $S \Rightarrow bSa$,
4. $S \Rightarrow \lambda$.

Again, several of these can be removed from contention. On the next round, we find the actual target string from the sequence

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb.$$

Therefore, *aabb* is in the language generated by the grammar under consideration.

See example 5.8….

EXAMPLE 5.8

The grammar

$$S \rightarrow SS \,|aSb|\, bSa \,|ab|\, ba$$

satisfies the given requirements. It generates the language in Example 5.7 without the empty string.

Given any $w \in \{a, b\}^+$, the exhaustive search parsing method will always terminate in no more than $|w|$ rounds. This is clear because the length of the sentential form grows by at least one symbol in each round. After $|w|$ rounds we have either produced a parsing or we know that $w \notin L(G)$.

Inherent Ambiguity

If every grammar that generates L is ambiguous - Inherently ambiguous

Some context free languages

have only ambiguous grammars

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

$S \rightarrow S_1 \mid S_2$

$S_1 \rightarrow S_1 c \mid A$

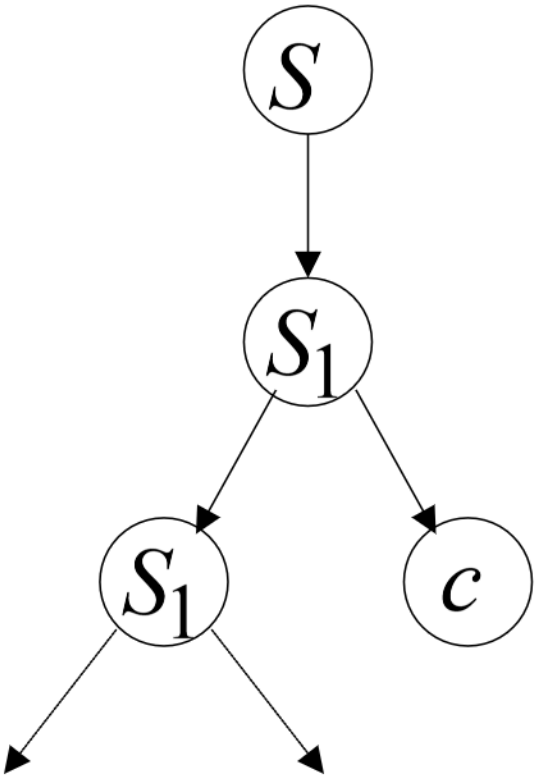
$A \rightarrow aAb \mid \lambda$

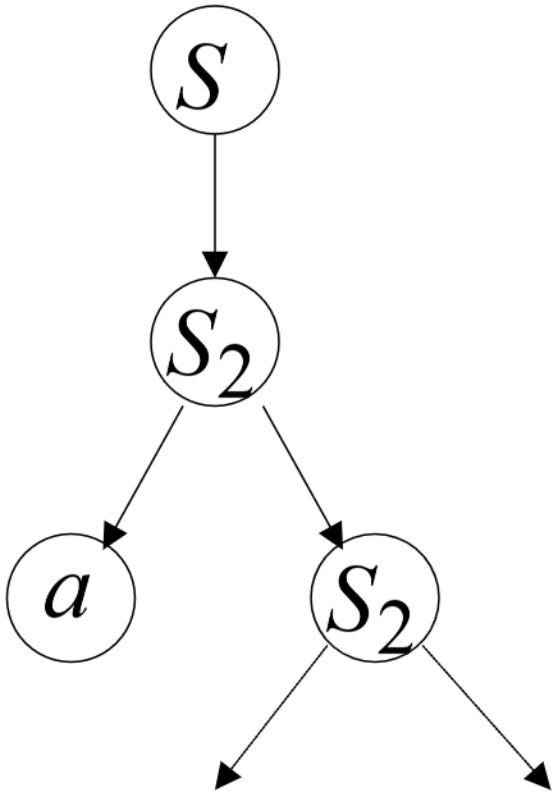
$S_2 \rightarrow aS_2 \mid B$

$B \rightarrow bBc \mid \lambda$

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

has two derivation trees





S Grammar

A context-free grammar $G = (V, T, S, P)$ is said to be a **simple grammar** or **s-grammar** if all its productions are of the form

$$A \rightarrow ax,$$

where $A \in V$, $a \in T$, $x \in V^*$, and any pair (A, a) occurs at most once in P.

The grammar

$$S \to aS \,|\, bSS \,|\, c$$

is an s-grammar. The grammar

$$S \to aS \,|\, bSS \,|\, aSS \,|\, c$$

is not an s-grammar because the pair (S, a) occurs in the two productions $S \to aS$ and $S \to aSS$.

Example for S grammar

1)

$S \rightarrow aABB$

$A \rightarrow aA \mid b$

$B \rightarrow bB \mid a$

2)

$S \rightarrow aAB$

$A \rightarrow aA \mid a$

$B \rightarrow b$

Find S grammar for aaa*b + b

3)

$S \rightarrow aA \mid b$

$A \rightarrow aB$

$B \rightarrow aB \mid b$

Find S grammar for $L = \{ a^n b^n \mid n >= 1 \}$

$S \rightarrow aA$
$A \rightarrow aAB \mid b$
$B \rightarrow b$

Yes..... This is S grammar

Example 2:

The Grammar with productions

$$S \rightarrow abB,$$
$$A \rightarrow aaBb,$$
$$B \rightarrow bbAa,$$
$$A \rightarrow \lambda,$$

Find out the language generated by the above language

Language generated is.....

$$L(G) = \{ab\,(bbaa)^n\,bba\,(ba)^n : n \geq 0\}$$

Regular grammars and linear grammars are context free but context free grammar are not necessarily linear.