



BLM2502

Theory of

Computation

Spring 2015

BLM2502 Theory of Computation

» Course Outline

» Week	Content
» 1	Introduction to Course
» 2	Computability Theory, Complexity Theory, Automata Theory, Set Theory, Relations, Proofs, Pigeonhole Principle
» 3	Regular Expressions
» 4	Finite Automata
» 5	Deterministic and Nondeterministic Finite Automata
» 6	Epsilon Transition, Equivalence of Automata
» 7	Pumping Theorem
» 8	April 10 - 14 week is the first midterm week
» 9	Context Free Grammars
» 10	Parse Tree, Ambiguity,
» 11	Pumping Theorem
» 12	Turing Machines, Recognition and Computation, Church-Turing Hypothesis
» 13	Turing Machines, Recognition and Computation, Church-Turing Hypothesis
» 14	May 22 – 27 week is the second midterm week
» 15	Review
» 16	Final Exam date will be announced



Context-Free Languages

»

Context-Free Languages

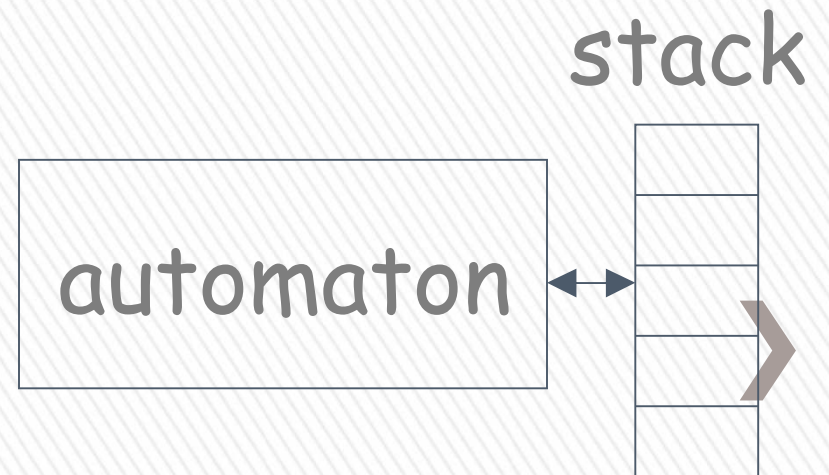
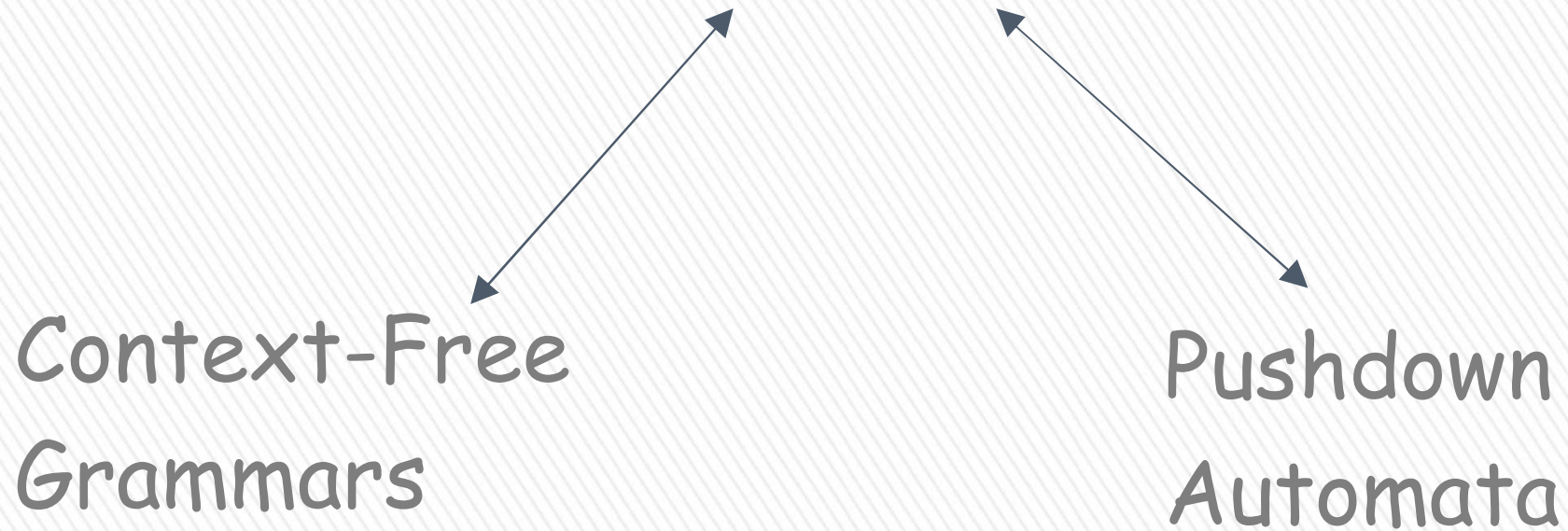
$$\{\underline{a}^n b^n : n \geq 0\} \quad \{\underline{w} w^R\}$$

Regular Languages

$$a^* b^* \quad (a + b)^*$$



Context-Free Languages





Context-Free Grammars

Grammars

- » Grammars express languages
- » Example: **the English language grammar**

$\langle \textit{sentence} \rangle \rightarrow \langle \textit{noun_phrase} \rangle \langle \textit{predicate} \rangle$

$\langle \textit{noun_phrase} \rangle \rightarrow \langle \textit{article} \rangle \langle \textit{noun} \rangle$

$\langle \textit{predicate} \rangle \rightarrow \langle \textit{verb} \rangle$

inglese grammi



$\langle \textit{article} \rangle \rightarrow a$

$\langle \textit{article} \rangle \rightarrow \textit{the}$

$\langle \textit{noun} \rangle \rightarrow \textit{cat}$

$\langle \textit{noun} \rangle \rightarrow \textit{dog}$

$\langle \textit{verb} \rangle \rightarrow \textit{runs}$

$\langle \textit{verb} \rangle \rightarrow \textit{sleeps}$



» Derivation of string “the dog walks” :

$\langle sentence \rangle \Rightarrow \langle noun_phrase \rangle \langle predicate \rangle$
 $\Rightarrow \langle noun_phrase \rangle \langle verb \rangle$
 $\Rightarrow \langle article \rangle \langle noun \rangle \langle verb \rangle$
 $\Rightarrow the \langle noun \rangle \langle verb \rangle$
 $\Rightarrow the \ dog \langle verb \rangle$
 $\Rightarrow the \ dog \ sleeps$



» Derivation of string “a cat runs” :

$\langle sentence \rangle \Rightarrow \langle noun_phrase \rangle \langle predicate \rangle$
 $\Rightarrow \langle noun_phrase \rangle \langle verb \rangle$
 $\Rightarrow \langle article \rangle \langle noun \rangle \langle verb \rangle$
 $\Rightarrow a \langle noun \rangle \langle verb \rangle$
 $\Rightarrow a \ cat \langle verb \rangle$
 $\Rightarrow a \ cat \ runs$



» Language of the grammar:

$$L = \{ \text{"a cat runs"}, \\ \text{"a cat sleeps"}, \\ \text{"the cat runs"}, \\ \text{"the cat sleeps"}, \\ \text{"a dog runs"}, \\ \text{"a dog sleeps"}, \\ \text{"the dog runs"}, \\ \text{"the dog sleeps"} \}$$


Productions

Sequence of
Terminals (symbols)

$\langle \text{noun} \rangle \rightarrow \text{cat}$

$\langle \text{sentence} \rangle \rightarrow \langle \text{noun_phrase} \rangle \langle \text{predicate} \rangle$

Variables
└─> non-terminal

Sequence of Variables



Another Example

Sequence of
terminals and variables

Grammar:

$$S \rightarrow \overbrace{aSb} \mid \varepsilon$$

Handwritten notes: aSb and ε are circled in red.

$$S \rightarrow \varepsilon$$

Handwritten notes: aSb and aSb are underlined in red. aSb is crossed out with a red X.

Variable

The right side
may be ε



» Grammar:

$$S \rightarrow aSb$$

$$S \rightarrow \varepsilon$$

Handwritten notes:
 aS
 aSb
 ab (circled and crossed out)

» Derivation of string : ab

$$S \Rightarrow aSb \Rightarrow ab$$

$$S \rightarrow aSb$$

$$S \rightarrow \varepsilon$$



» Grammar:

$$S \rightarrow aSb$$

$$S \rightarrow \varepsilon$$

» Derivation of string :

aabb

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



$$S \rightarrow aSb$$



$$S \rightarrow \varepsilon$$



Grammar: $S \rightarrow aSb$

$S \rightarrow \varepsilon$

Other derivations:

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbbb$

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



Grammar:

$$S \rightarrow aSb$$

$$S \rightarrow \varepsilon$$

Language of the grammar:

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



A Convenient Notation

» We write: $S \xRightarrow{*} aaabbb$

for one or more derivation steps

» Instead of:

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaasbbb \Rightarrow aaabbb$



In general we write:

$$w_1 \Rightarrow w_n$$

~~*~~ \Rightarrow all

If:

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

in zero or more derivation steps

Trivially:

$$w \Rightarrow w$$



Example Grammar

Possible Derivations

$$S \rightarrow aSb$$

$$S \rightarrow \varepsilon$$

$$S \xRightarrow{*} \varepsilon$$

$$S \xRightarrow{*} ab$$

$$S \xRightarrow{*} aaabbbb$$

$$S \xRightarrow{*} aaSbb \xRightarrow{*} aaaaaaSbbbbbb$$



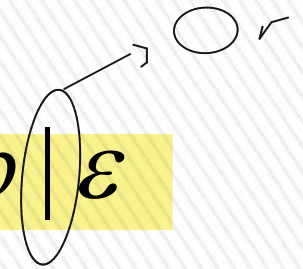
Another convenient notation:

$$S \rightarrow aSb \mid \varepsilon$$

$$S \rightarrow \varepsilon$$



$$S \rightarrow aSb \mid \varepsilon$$



$$\langle \text{article} \rangle \rightarrow a$$

$$\langle \text{article} \rangle \rightarrow the$$



$$\langle \text{article} \rangle \rightarrow a \mid the$$



Formal Definition

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

Gramerde
oldugu
bircisi
igin
hang
kural
kuralda
serketigini
baslamiz
bilmeniz
luzin

Set of
variables

Set of
terminal
symbols

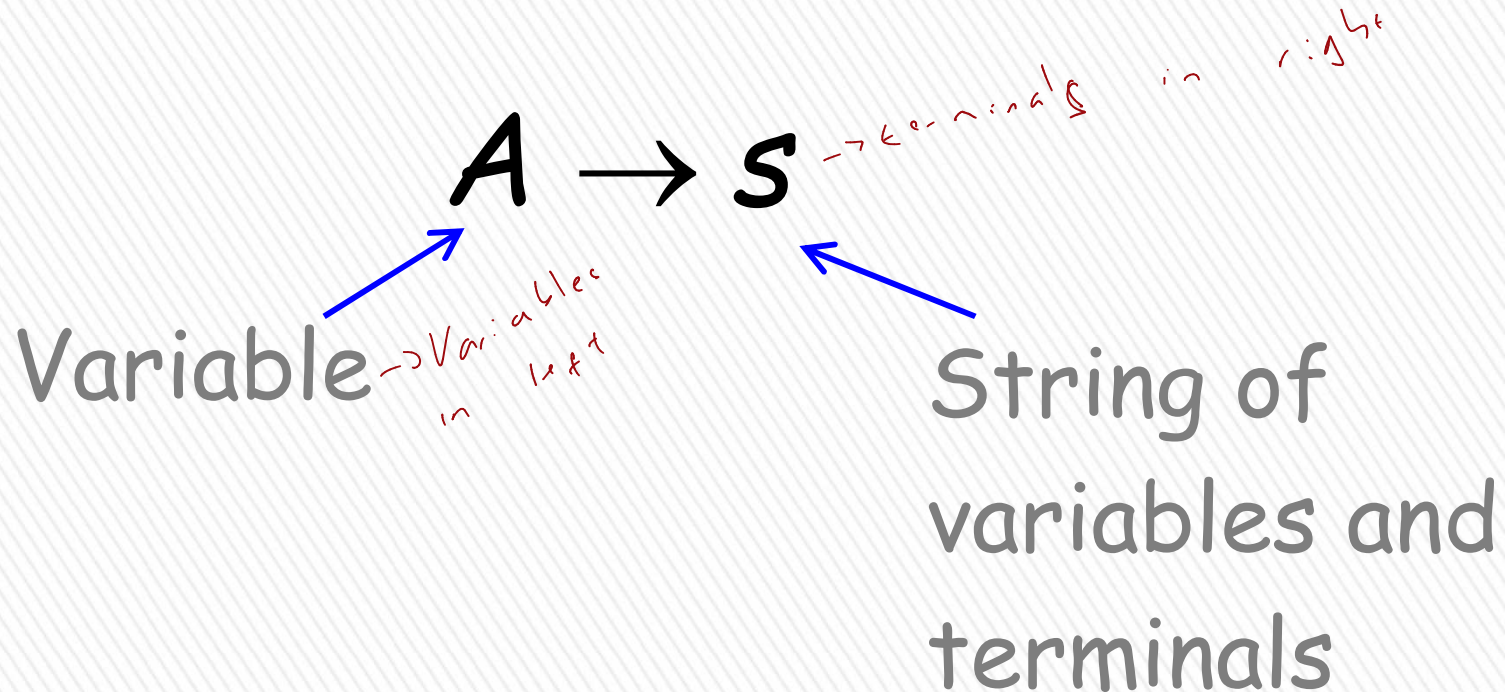
Start
variable

Set of
productions



Context-Free Grammar: $G = (V, T, S, P)$

All productions in P are of the form



Example for Context-Free Grammar

$$S \rightarrow aSb \mid \varepsilon$$

productions

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

$$V = \{S, L, P\}$$

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

$$S = L$$

$$P = \{S \rightarrow aSL, S \rightarrow aSaL, S \rightarrow \varepsilon\}$$

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

$$V = \{S\}$$

variables

begin
symbol
half

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

terminals

start variable



Language of a Grammar:

» For a grammar G with start variable S

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

**
terminaller*

String of terminals or ε



Example:

context-free grammar $G : \boxed{S \rightarrow aSb \mid \varepsilon}$

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

Since, there is derivation

$$S \xRightarrow{*} a^n b^n \quad \text{for any } n \geq 0 \rangle$$

$$L = \{0^*1^*\}$$

Context-Free Language:

regular
Language
Context
Language
all
business
S → ε | 0 S | 1 S

- » A language L is context-free
- » if there is a context-free grammar G
- » with $L = L(G)$



Example:

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

is a context-free language

since context-free grammar G :

$$S \rightarrow aSb \mid \varepsilon$$

generates $L(G) = L$



Another Example

Context-free grammar G : → Palindrome

$$S \rightarrow \underline{aSa} \mid \underline{bSb} \mid \underline{\varepsilon} \mid a \mid b$$

Example derivations:

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

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

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

Palindromes of even length



Another Example

Context-free grammar G :

$$S \rightarrow aSb \mid SS \mid \varepsilon$$

$\rightarrow \varepsilon$ -eulerle



aSb
 $aSSb$
 $aSSSb$

Example derivations:

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

$$S \Rightarrow SS \Rightarrow aSbS \Rightarrow abS \Rightarrow abaSb \Rightarrow abab$$

$$L(G) = \{w : n_a(w) = n_b(w),$$

$$\text{and } n_a(v) \geq n_b(v)$$

Describes
matched

in any prefix v

parentheses:

$() ((())) (())$ $a = ($, $b =)$



Derivation Trees

Derivation Order

Consider the following example grammar with 5 productions:

- | | | |
|-----------------------|--------------------------------|--------------------------------|
| 1. $S \rightarrow AB$ | 2. $A \rightarrow aaA$ | 4. $B \rightarrow Bb$ |
| | 3. $A \rightarrow \varepsilon$ | 5. $B \rightarrow \varepsilon$ |

aaA \nearrow S



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

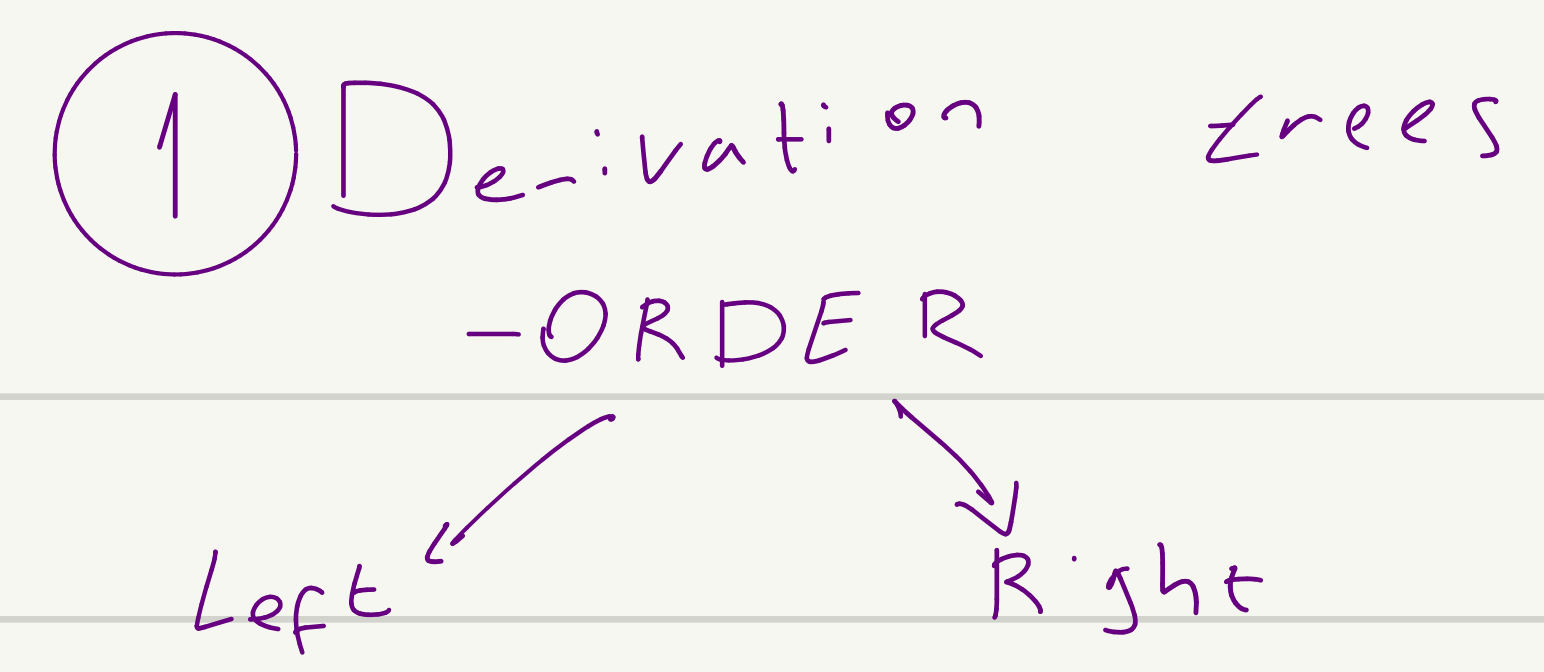
soldan başlıyoruz

Leftmost derivation order of string aab:

$$\begin{array}{ccccccccc}
 1 & & 2 & & 3 & & 4 & & 5 \\
 S & \Rightarrow & AB & \Rightarrow & aaAB & \Rightarrow & aaB & \Rightarrow & aaBb \Rightarrow aab
 \end{array}$$

At each step, we substitute the leftmost variable





$$\begin{array}{lll}
 1. S \rightarrow AB & 2. A \rightarrow aaA & 4. B \rightarrow Bb \\
 & 3. A \rightarrow \varepsilon & 5. B \rightarrow \varepsilon
 \end{array}$$

Rightmost derivation order of string *aab*:

$$\begin{array}{cccccc}
 1 & & 4 & & 5 & & 2 & & 3 \\
 S \Rightarrow AB \Rightarrow ABb \Rightarrow Ab \Rightarrow aaAb \Rightarrow aab
 \end{array}$$

At each step, we substitute the
rightmost variable



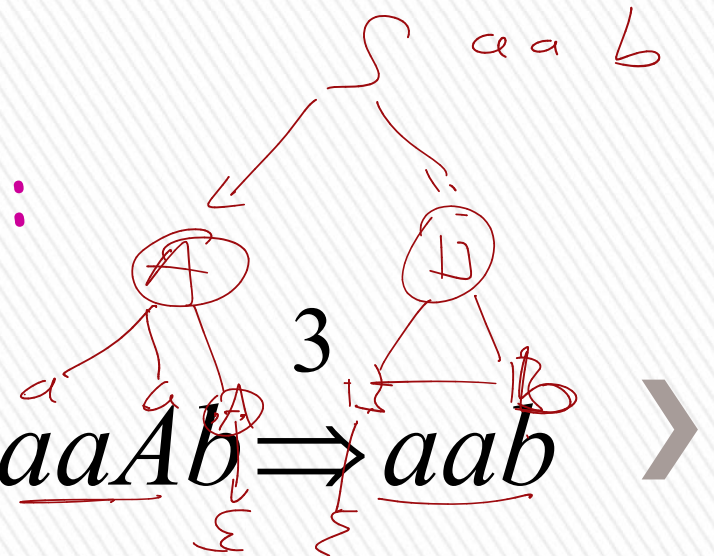
- | | | |
|-----------------------|--|--|
| 1. $S \rightarrow AB$ | 2. $A \rightarrow aaA$ | 4. $B \rightarrow \underline{Bb}$ |
| | 3. $A \rightarrow \underline{\varepsilon}$ | 5. $B \rightarrow \underline{\varepsilon}$ |

Leftmost derivation of aab :

$$\begin{array}{ccccccccc}
 1 & & 2 & & 3 & & 4 & & 5 \\
 S & \Rightarrow & \underline{A}B & \Rightarrow & aa\underline{A}B & \Rightarrow & aa\underline{B} & \Rightarrow & aa\underline{Bb} & \Rightarrow & aab
 \end{array}$$

Rightmost derivation of aab :

$$\begin{array}{ccccccccc}
 1 & & 4 & & 5 & & 2 & & 3 \\
 S & \Rightarrow & \underline{A}B & \Rightarrow & A\underline{Bb} & \Rightarrow & \underline{A}b & \Rightarrow & aa\underline{A}b & \Rightarrow & aab
 \end{array}$$



Consider the same example grammar:

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

And a derivation of *aab*:

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

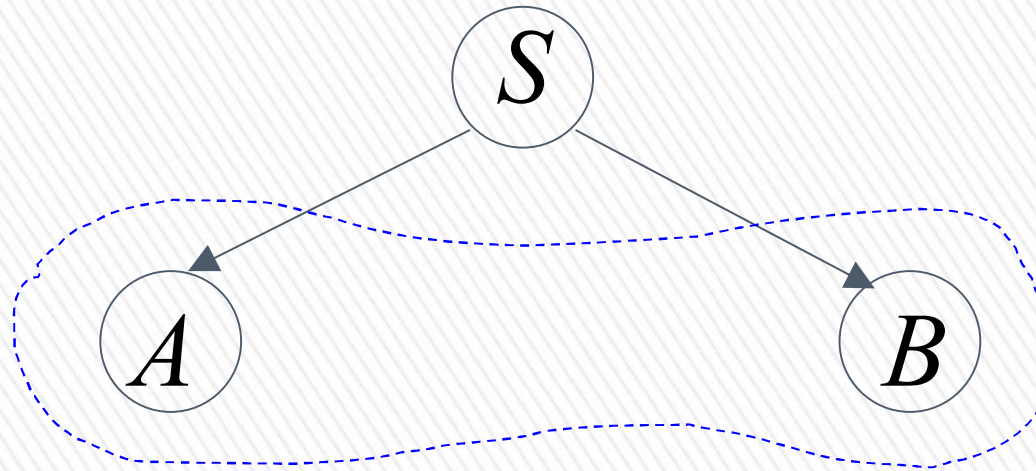


Derivation Tree

$$S \rightarrow AB$$

$$A \rightarrow aaA \mid \varepsilon \quad B \rightarrow Bb \mid \varepsilon$$

$$S \Rightarrow AB$$



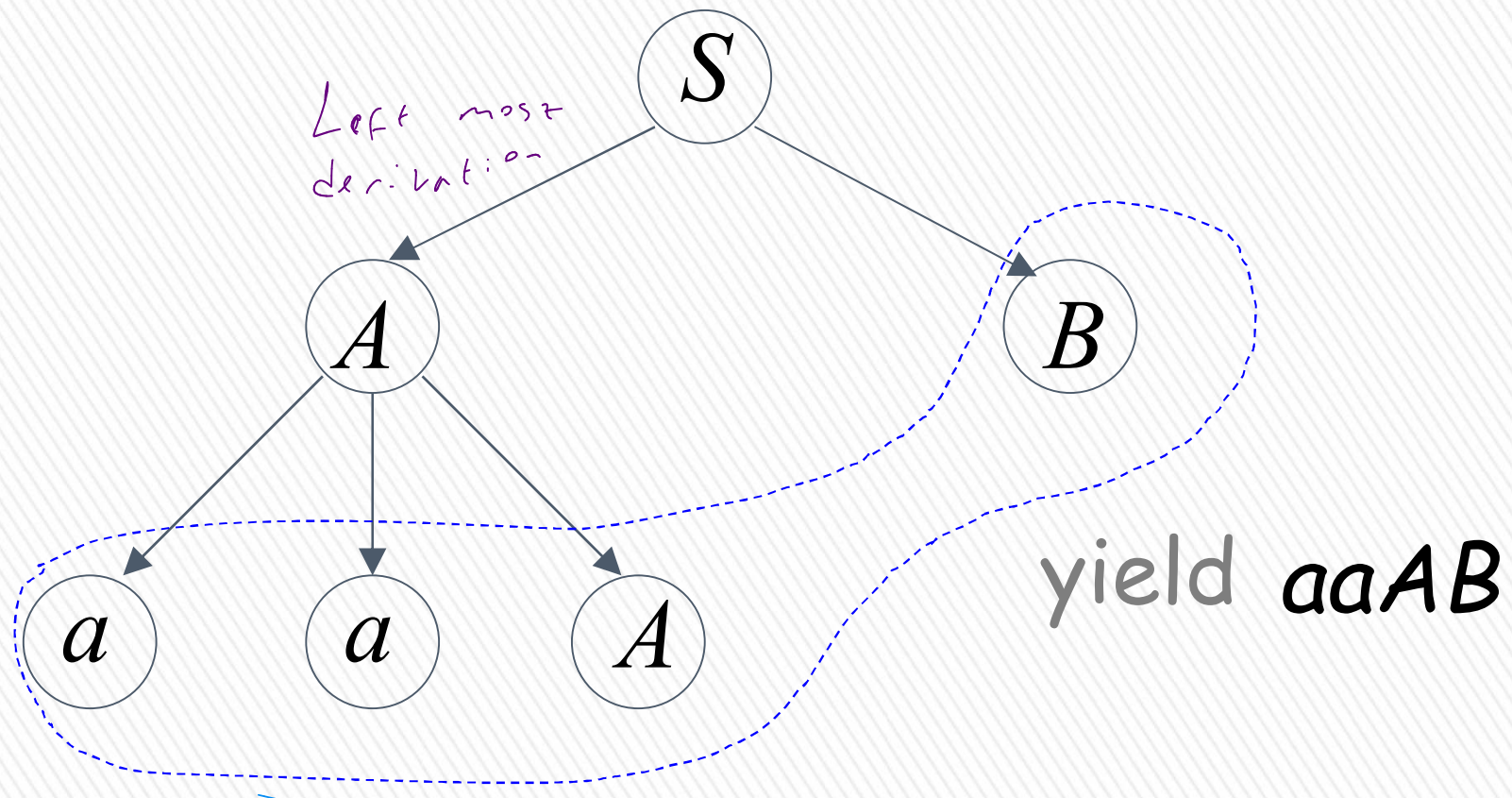
yield AB



$$S \rightarrow AB$$

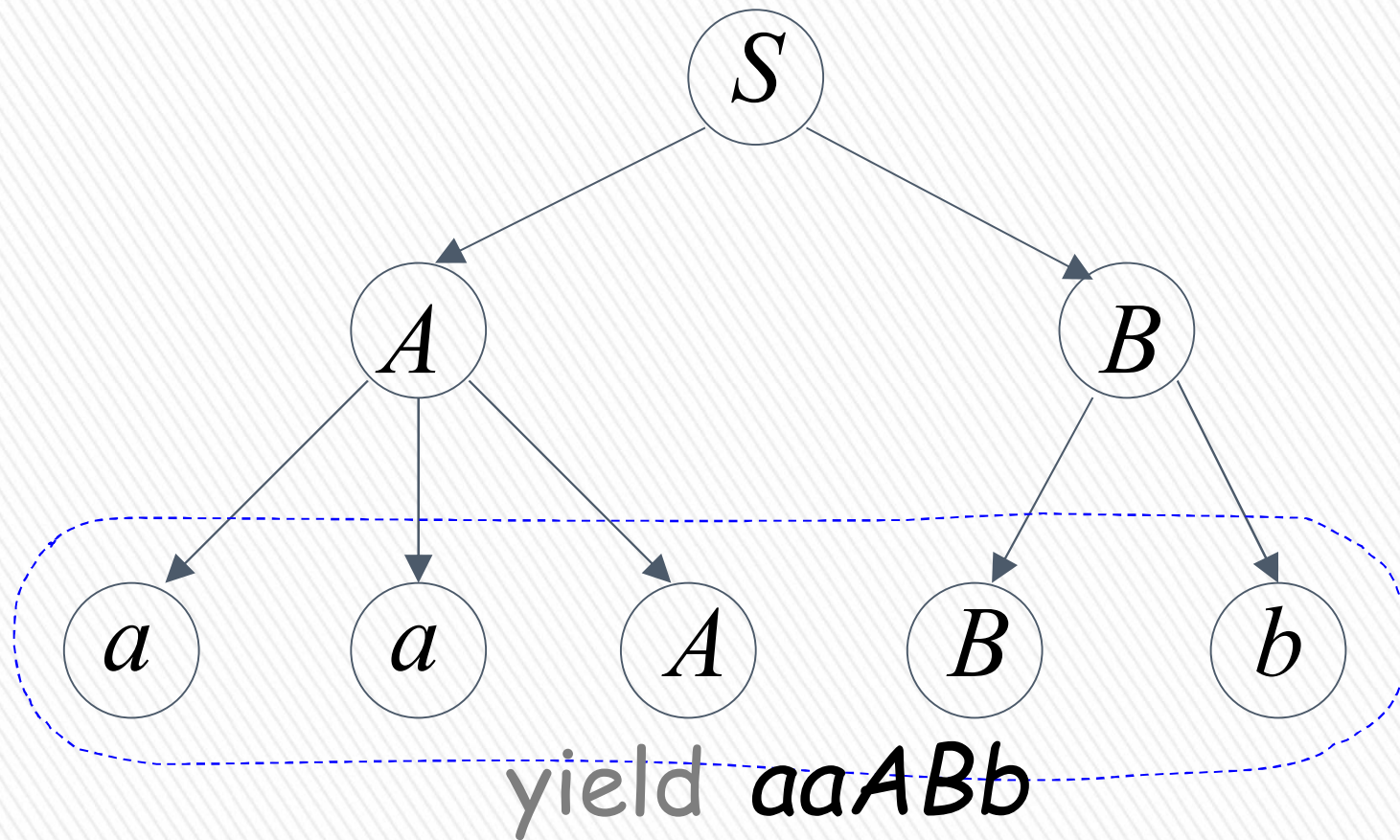
$$A \rightarrow aaA \mid \varepsilon \quad B \rightarrow Bb \mid \varepsilon$$

$$S \Rightarrow AB \Rightarrow aaAB$$



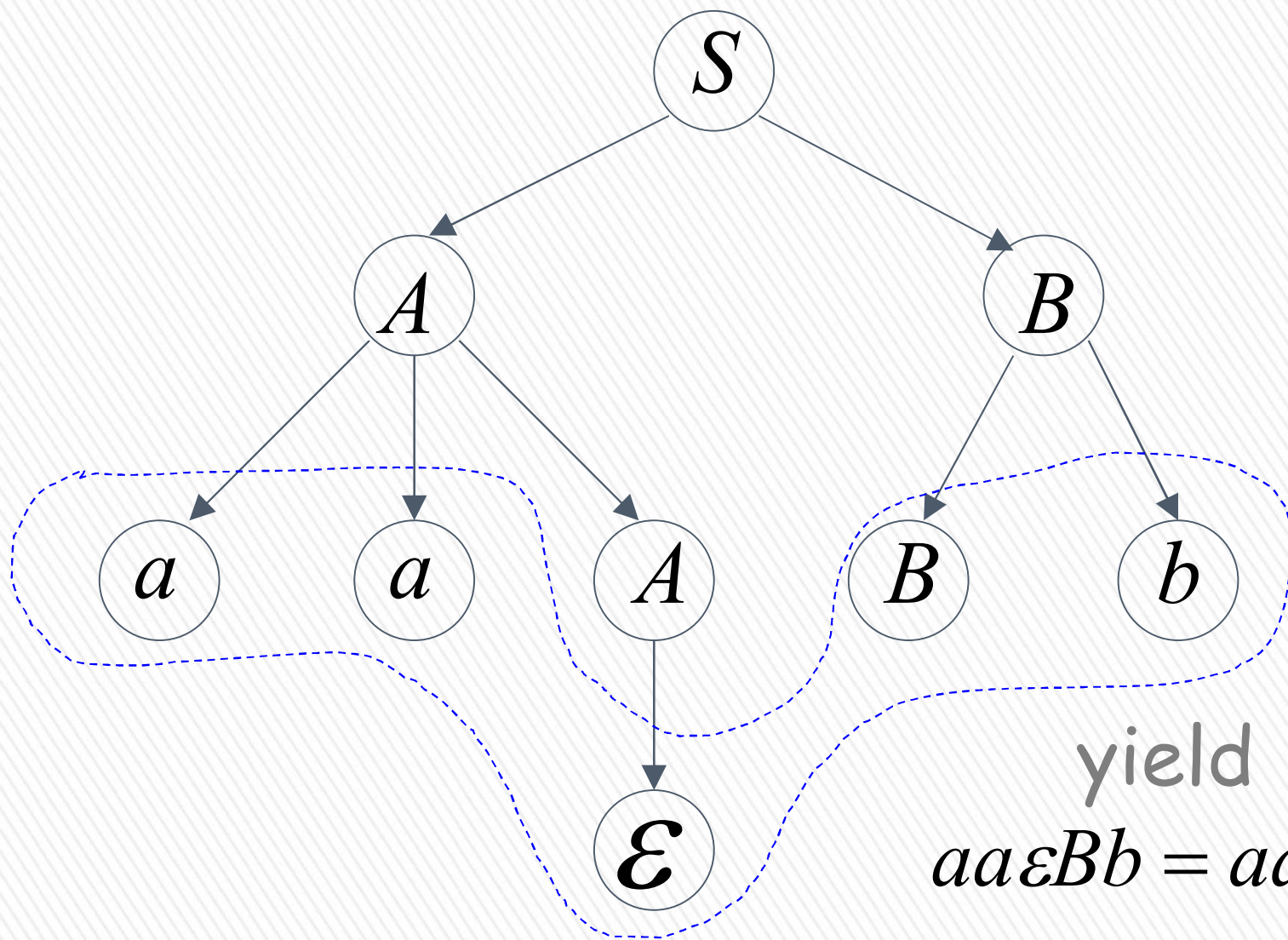
$$S \rightarrow AB \qquad A \rightarrow aaA \mid \varepsilon \qquad B \rightarrow Bb \mid \varepsilon$$

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



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

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



$$aa\varepsilon Bb = aaBb$$

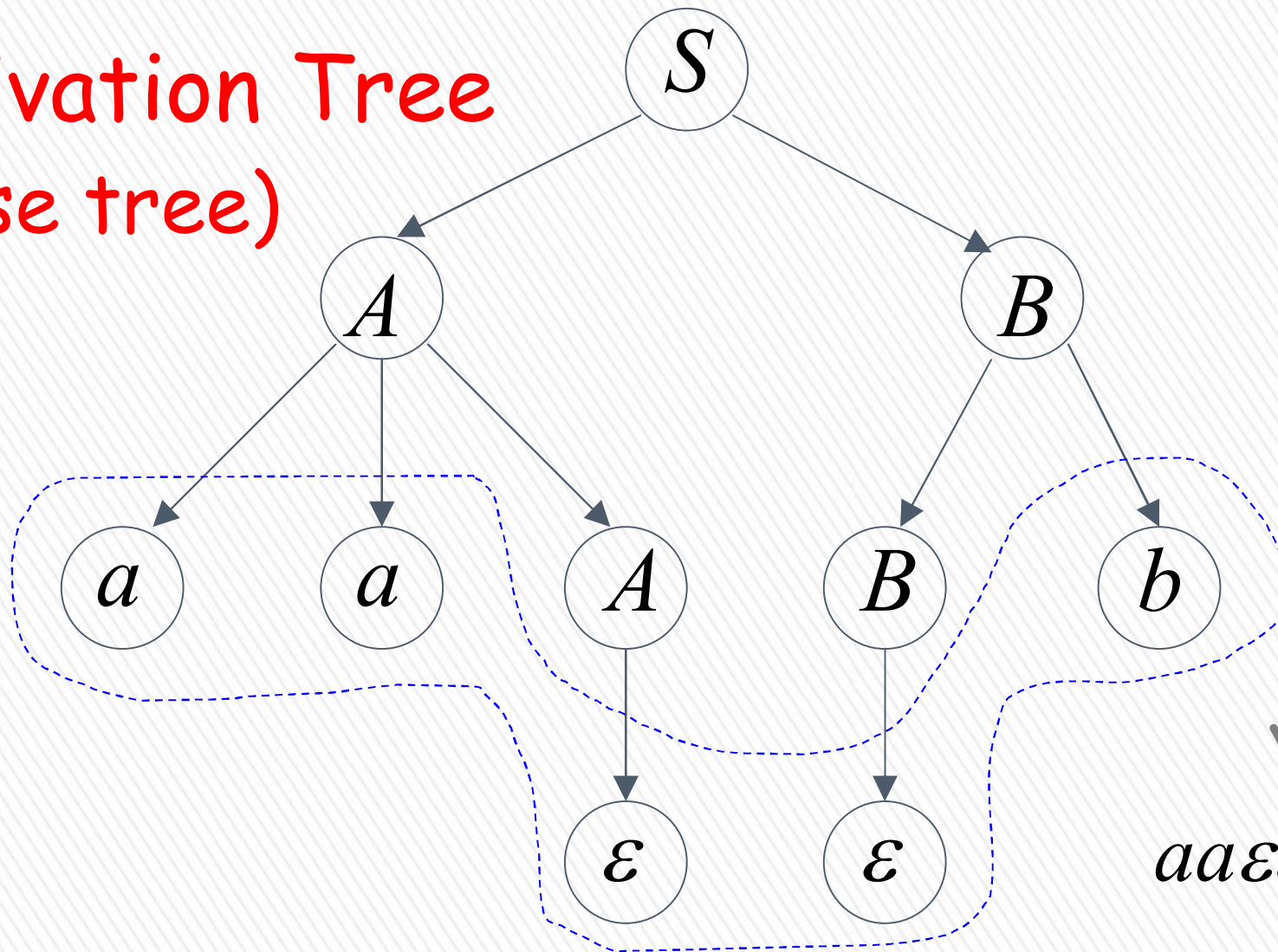


$$S \rightarrow AB$$

$$A \rightarrow aaA \mid \varepsilon \quad B \rightarrow Bb \mid \varepsilon$$

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

Derivation Tree
(parse tree)



yield \triangleright
 $aa\varepsilon\varepsilon b = aab$

Sometimes, derivation order doesn't matter

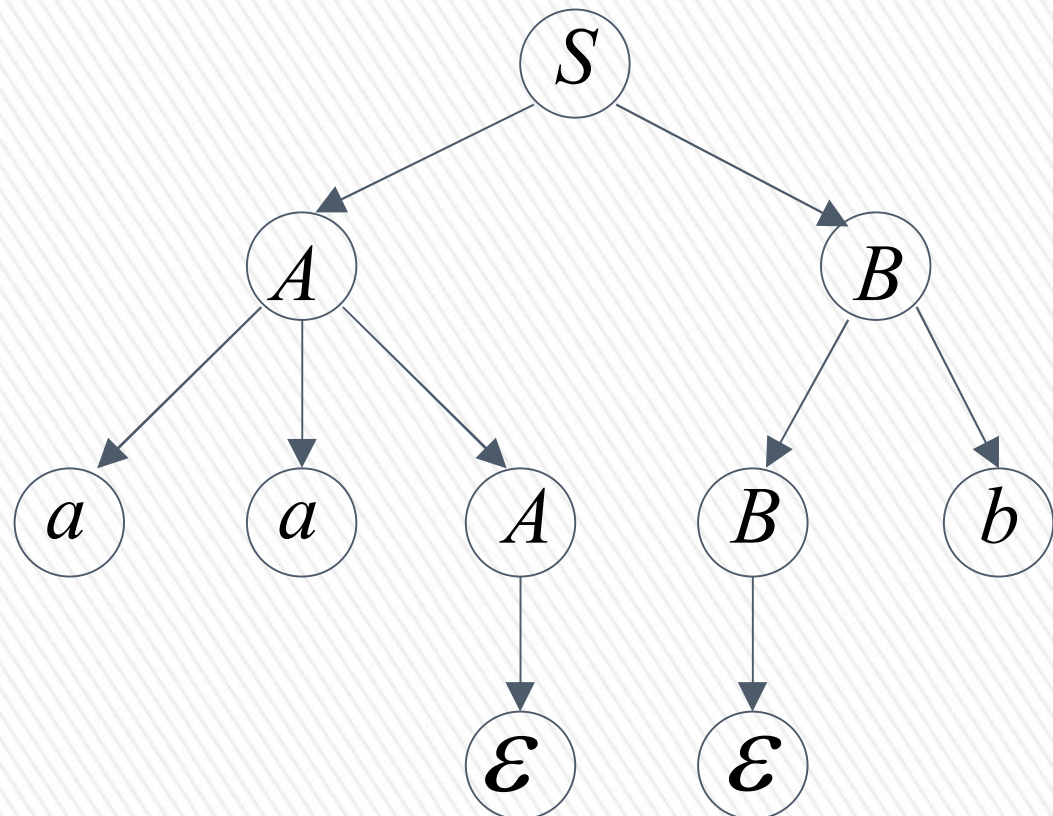
Leftmost derivation:

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

Rightmost derivation:

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

Give same
derivation tree





→ Ağaçlar benzer legilsin

Ambiguity

Grammar for mathematical expressions

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

Example strings:

$$(a + a) * a + (a + a * (a + a))$$



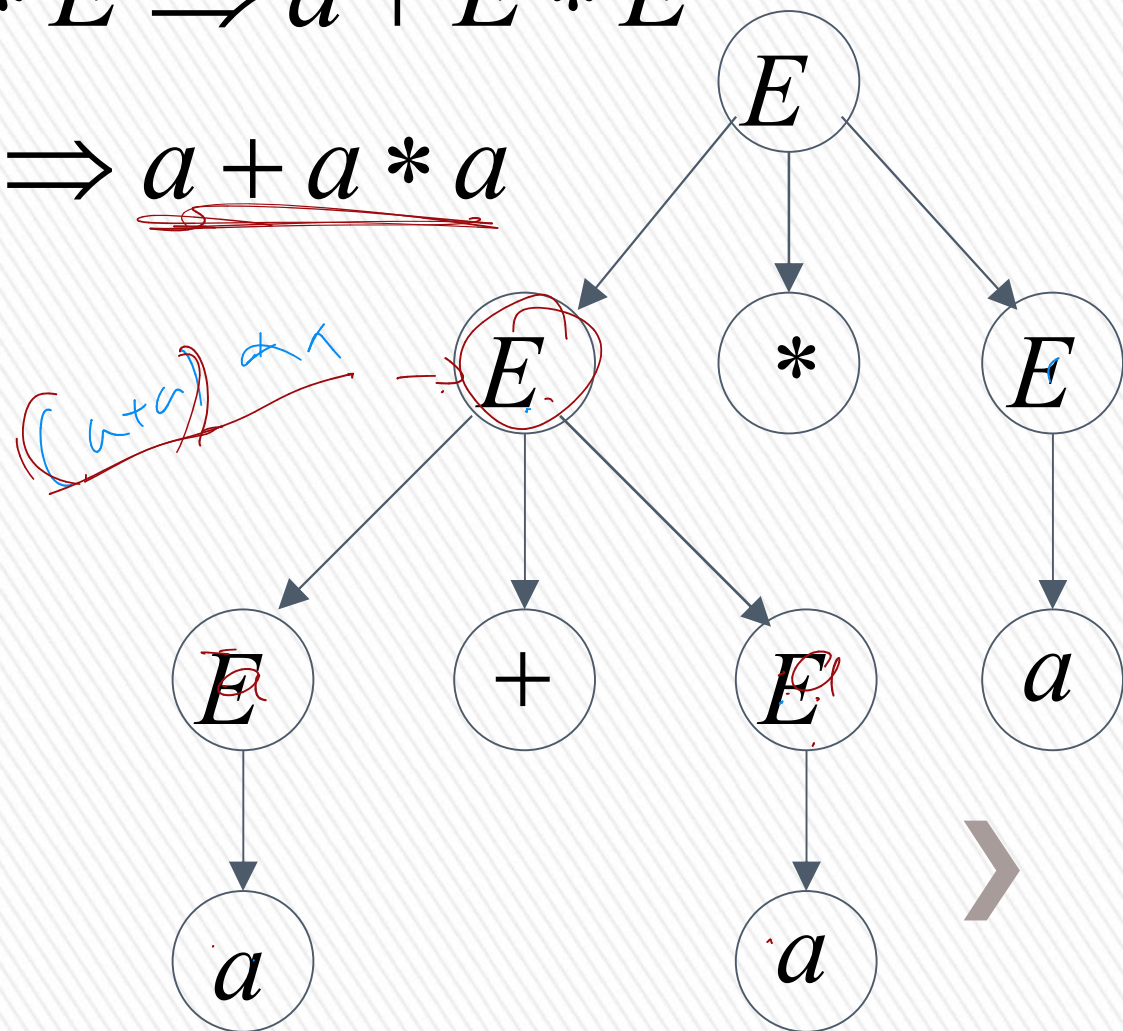
Denotes any number



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

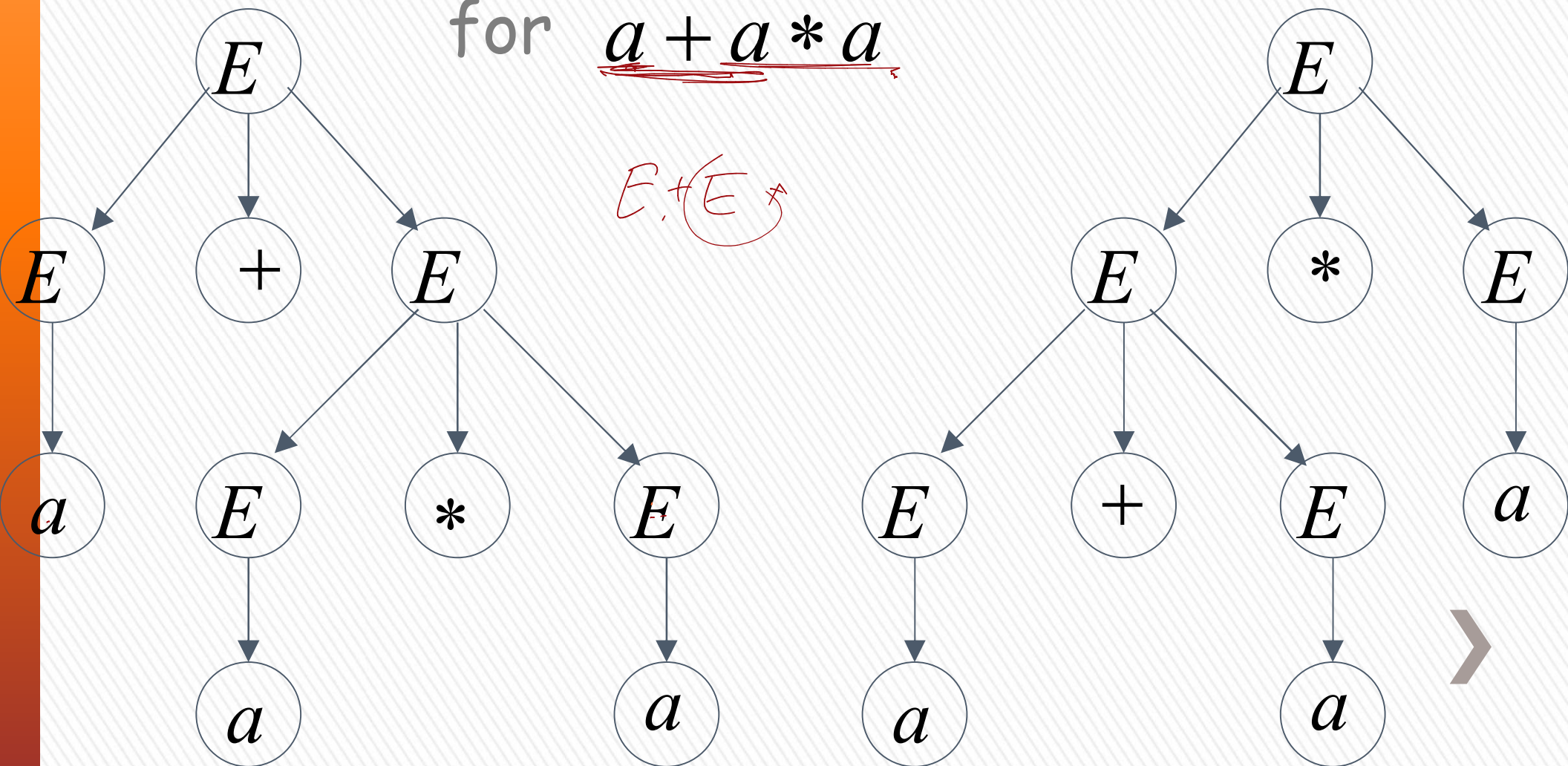
$$E \Rightarrow E * E \Rightarrow E + E * E \Rightarrow a + E * E \\ \Rightarrow a + a * E \Rightarrow \underline{a + a * a}$$

Another
leftmost derivation
for $a + a * a$



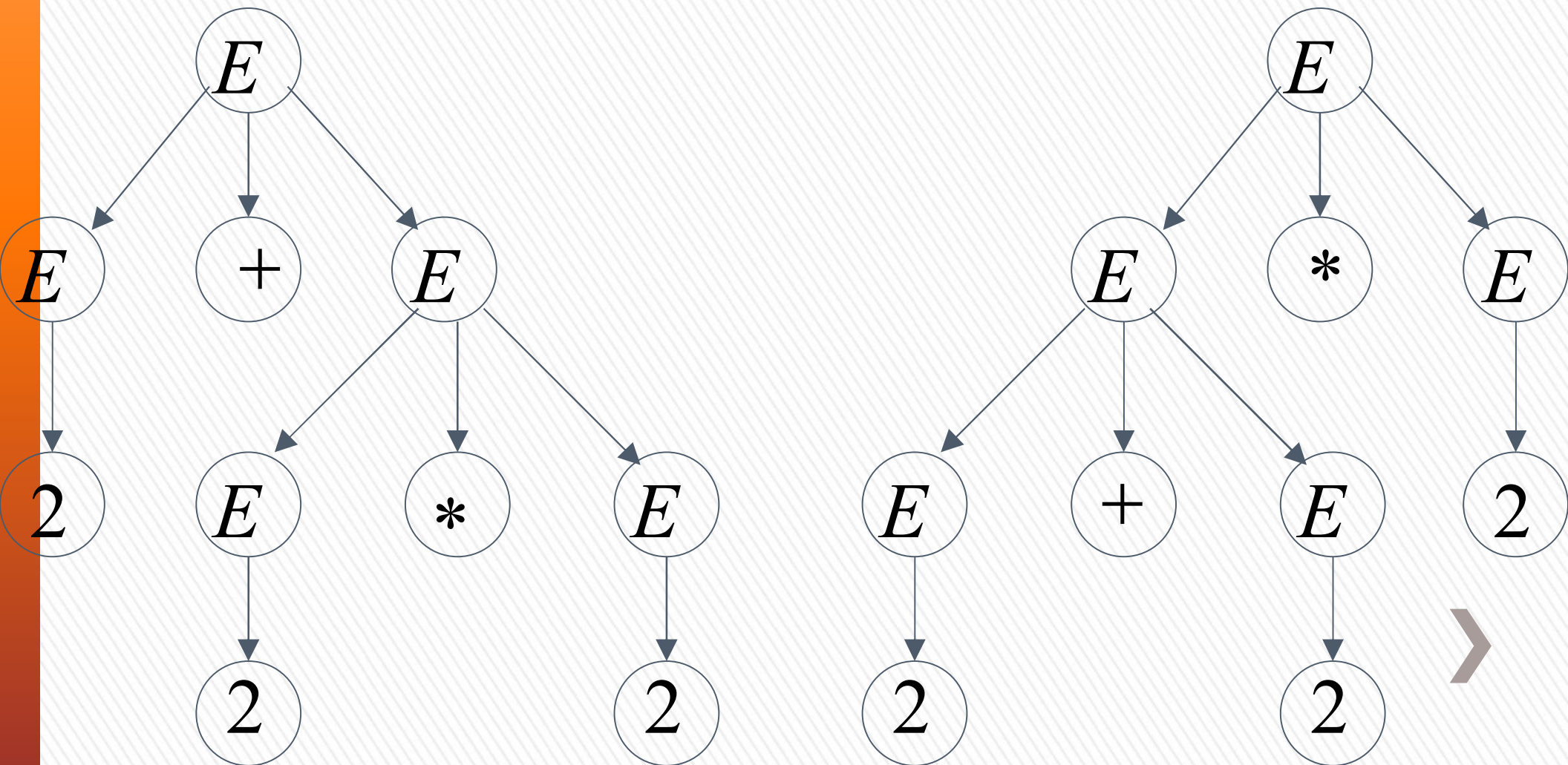
$$E \rightarrow \underline{E + E} \mid E * E \mid (E) \mid a$$

Two derivation trees
for $a + a * a$



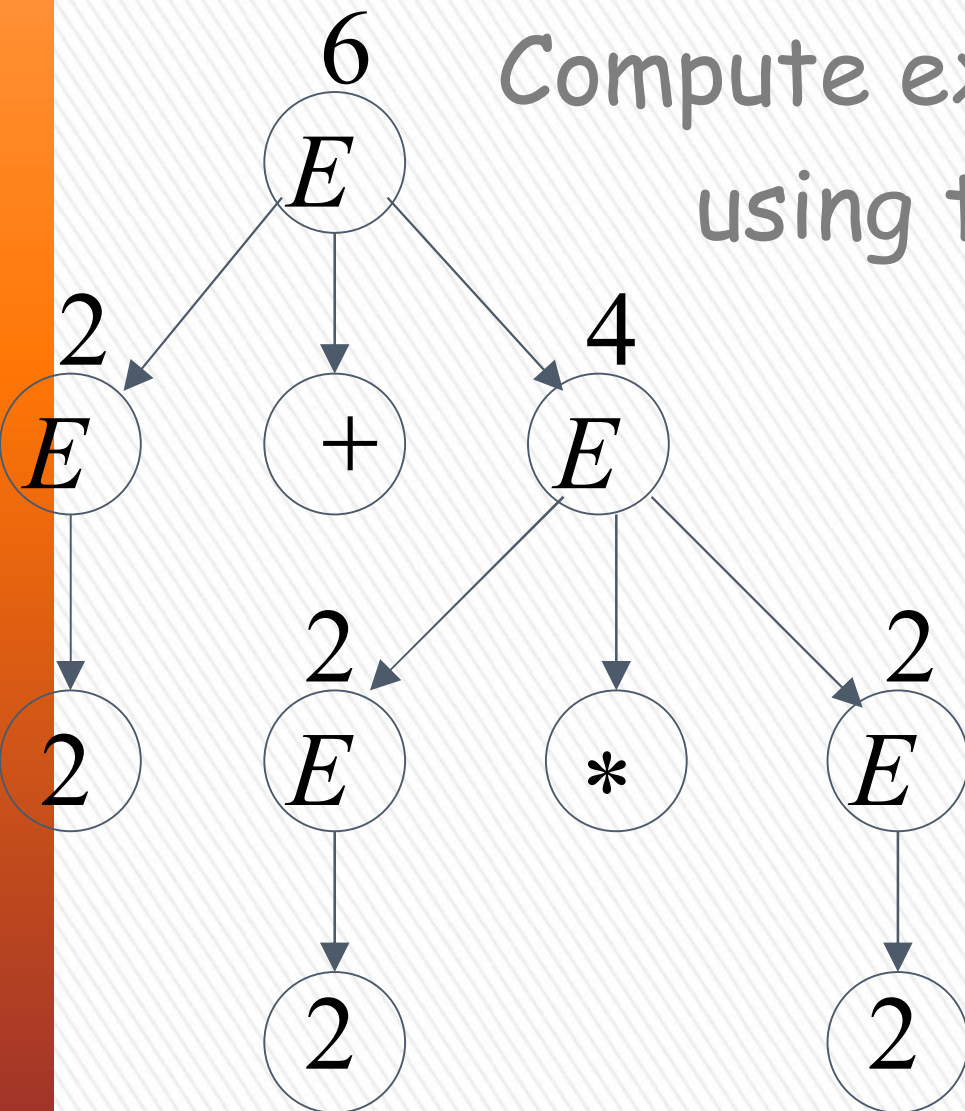
take $a = 2$

$$a + a * a = 2 + 2 * 2$$



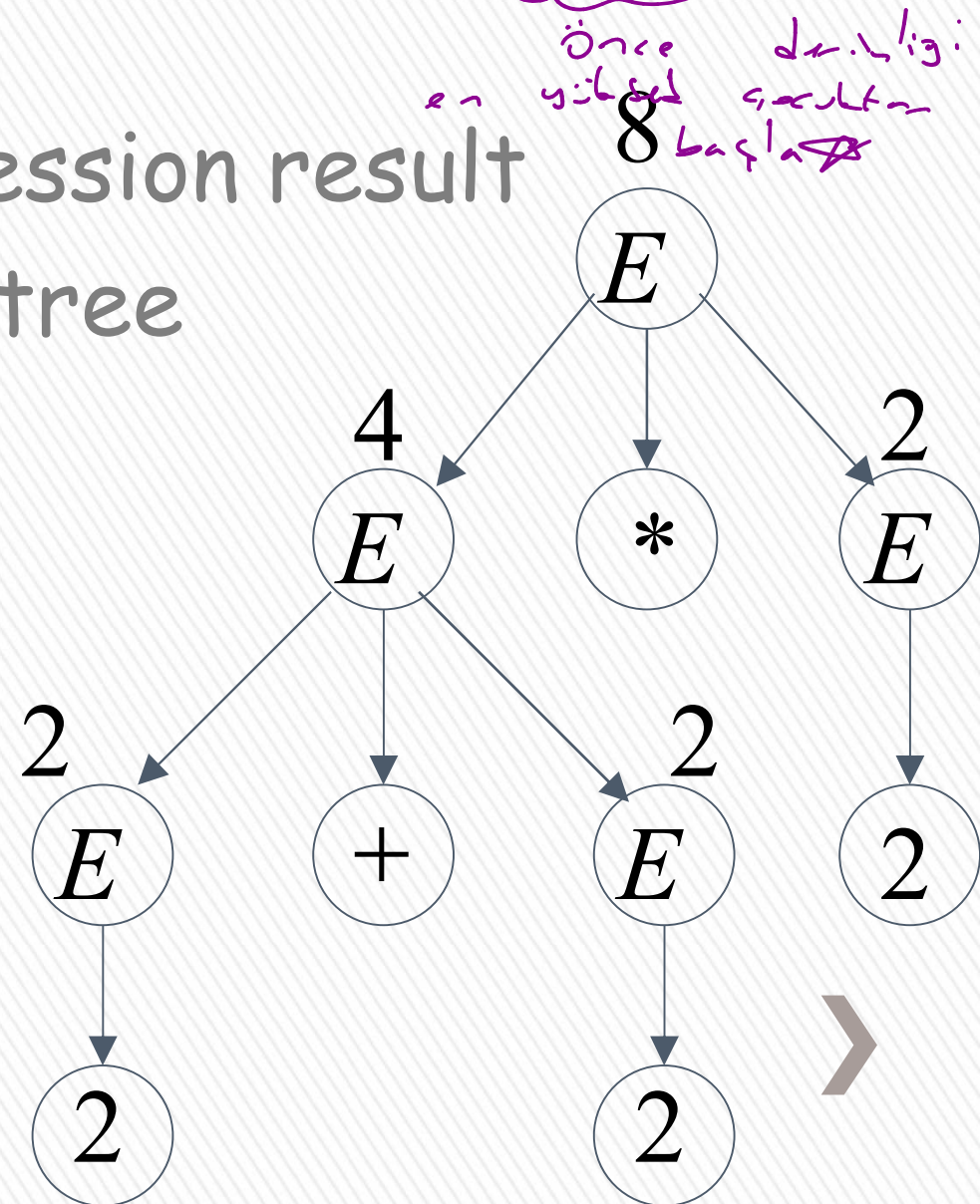
Good Tree

$$2 + 2 * 2 = 6$$



Bad Tree

$$2 + 2 * 2 = 8$$



Two different derivation trees
may cause problems in applications which
use the derivation trees:

- Evaluating expressions
- In general, in compilers
for programming languages



Ambiguous Grammar:

→ 2 farklı ağaç
Ambiguous

A context-free grammar G is ambiguous if there is a string $w \in L(G)$ which has:

two different derivation trees

or

two leftmost derivations

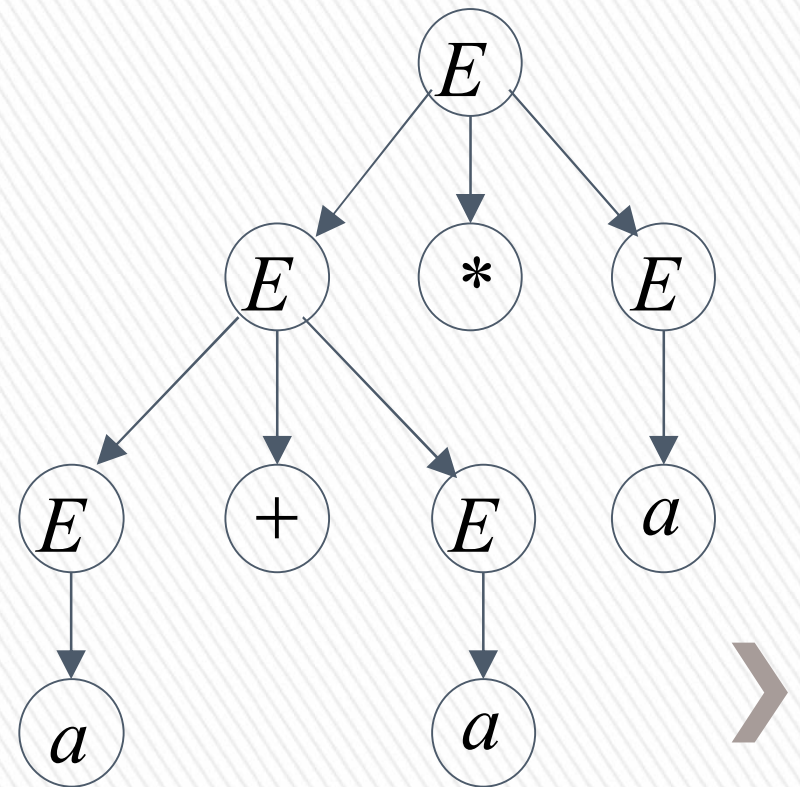
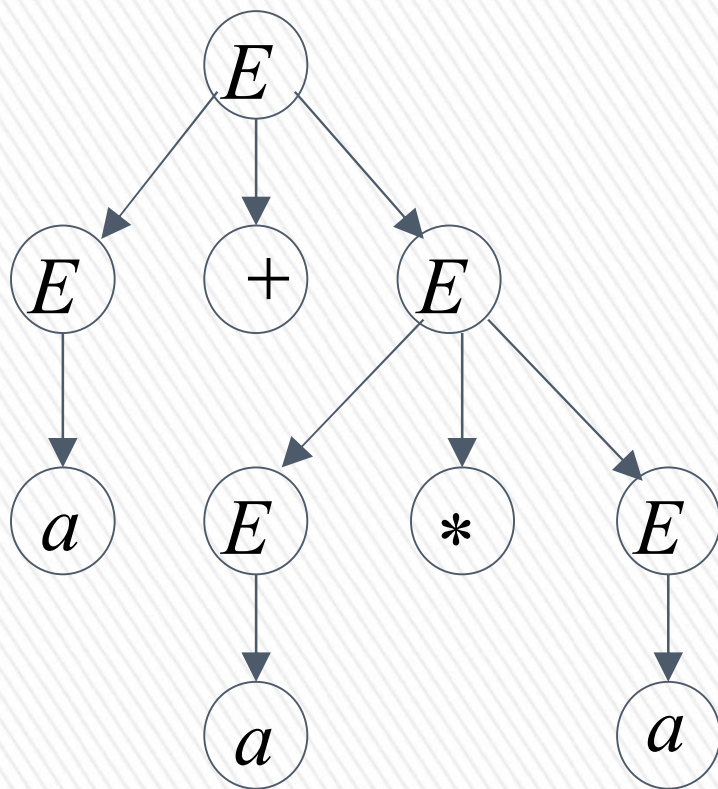
(Two different derivation trees give two different leftmost derivations and vice-versa)



Example:

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

this grammar is ambiguous since
string $a + a * a$ has two derivation trees



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

this grammar is ambiguous also because
string $a + a * a$ has two leftmost derivations

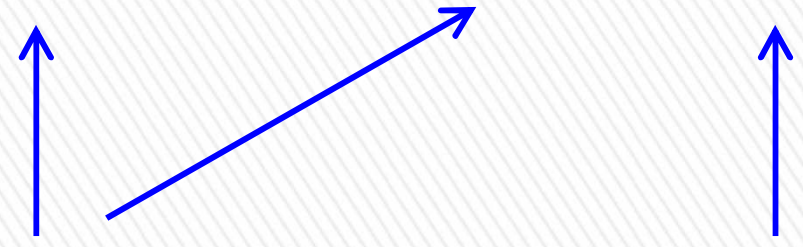
$$\begin{aligned} E &\Rightarrow E + E \Rightarrow a + E \Rightarrow a + E * E \\ &\Rightarrow a + a * E \Rightarrow a + a * a \end{aligned}$$

$$\begin{aligned} E &\Rightarrow E * E \Rightarrow E + E * E \Rightarrow a + E * E \\ &\Rightarrow a + a * E \Rightarrow a + a * a \end{aligned}$$



Another ambiguous grammar:

IF_STMT \rightarrow if EXPR then STMT
 | if EXPR then STMT else STMT

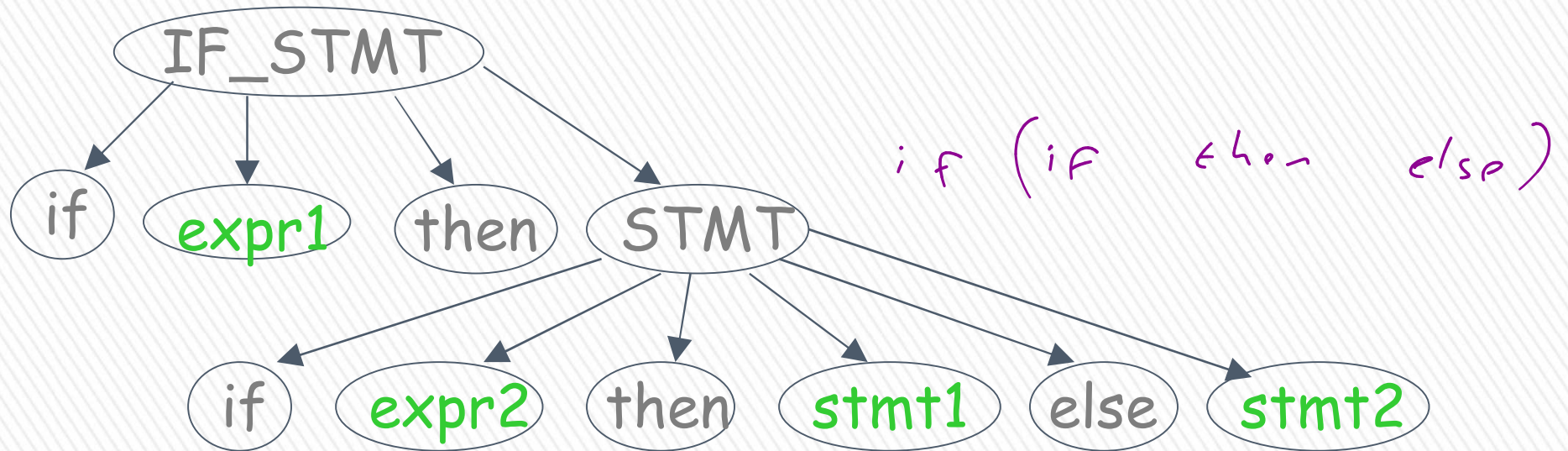


Variables Terminals

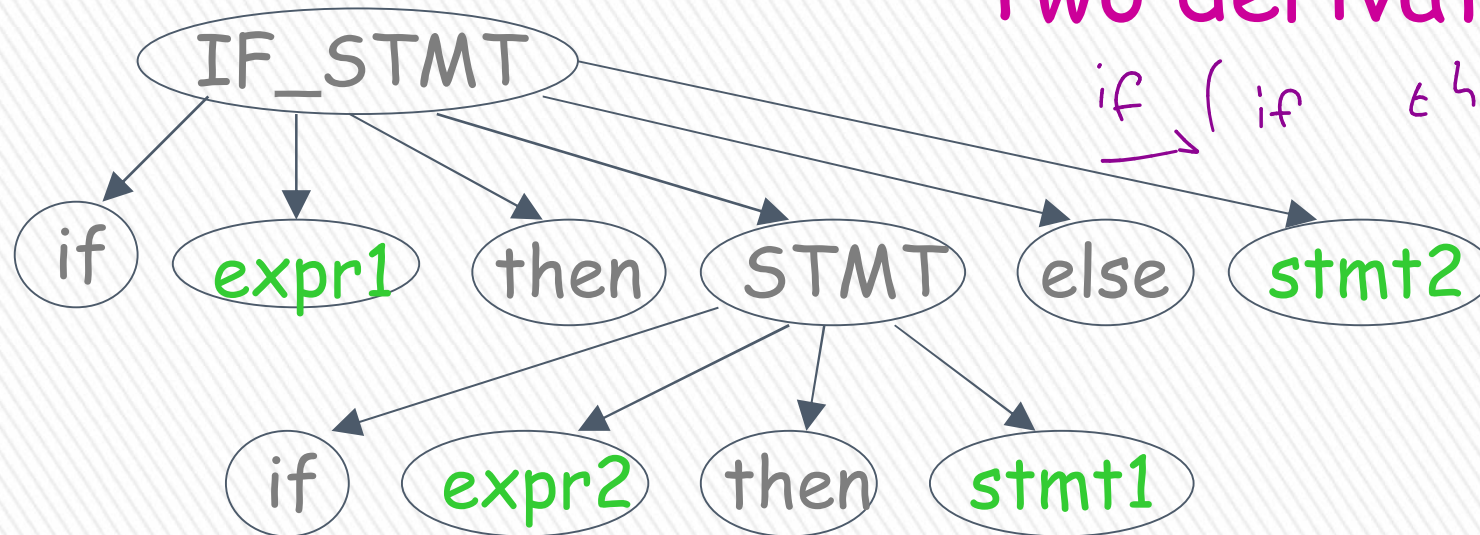
Very common piece of grammar
in programming languages



If *expr1* then if *expr2* then *stmt1* else *stmt2*



Two derivation trees



In general, ambiguity is bad
and we want to remove it

Sometimes it is possible to find
a non-ambiguous grammar for a language

But, in general we cannot do so



A successful example:

Ambiguous
Grammar

$$\begin{aligned} E &\rightarrow E + E \\ E &\rightarrow E * E \\ E &\rightarrow (E) \\ E &\rightarrow a \end{aligned}$$

Equivalent

Non-Ambiguous
Grammar

$$\begin{aligned} E &\rightarrow E + T \mid T \\ T &\rightarrow T * F \mid F \\ F &\rightarrow (E) \mid a \end{aligned}$$

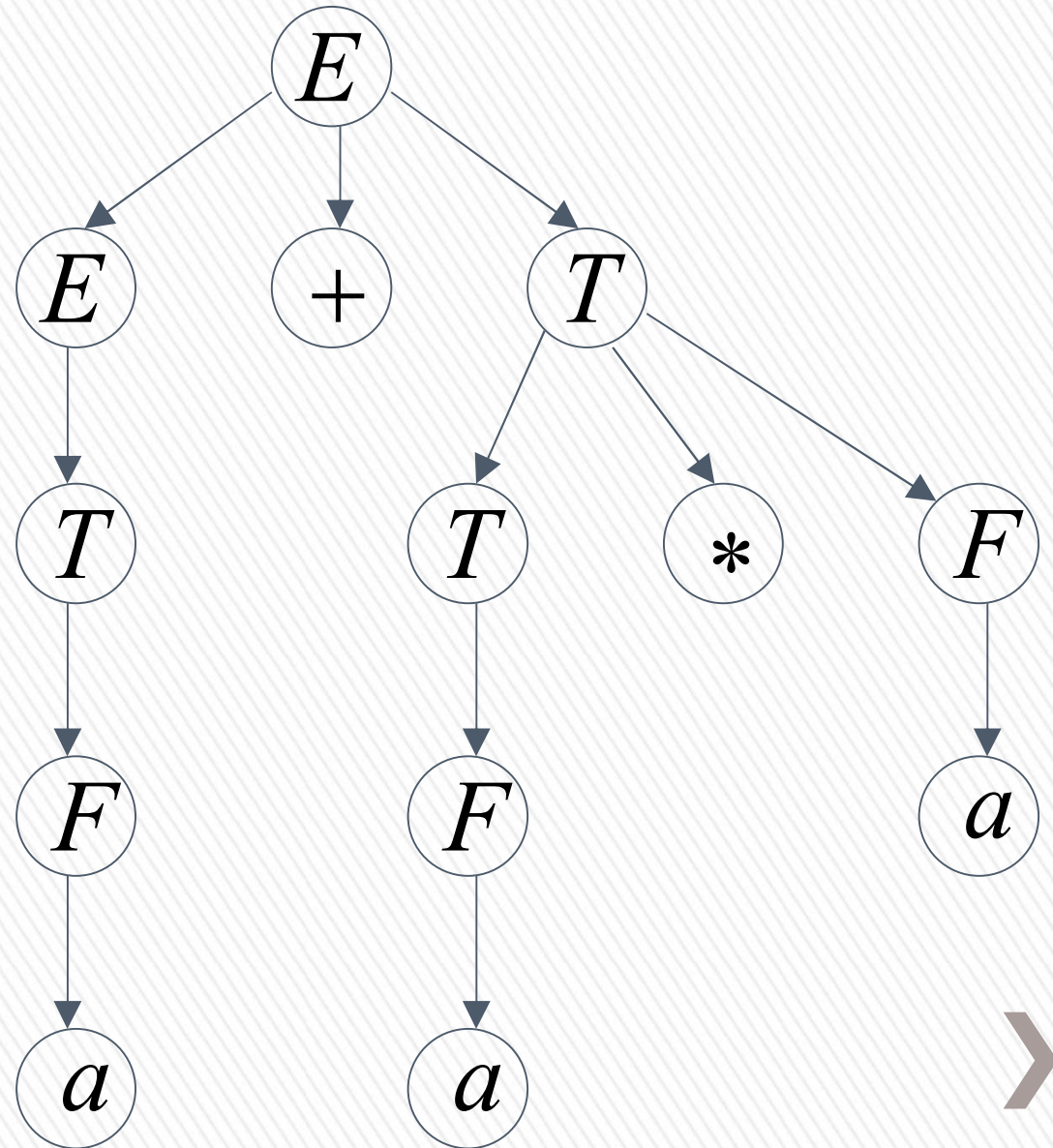
generates the same
language



$$\begin{aligned} E &\Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow a + T \Rightarrow a + T * F \\ &\Rightarrow a + F * F \Rightarrow a + a * F \Rightarrow a + a * a \end{aligned}$$

$$\begin{aligned} E &\rightarrow E + T \mid T \\ T &\rightarrow T * F \mid F \\ F &\rightarrow (E) \mid a \end{aligned}$$

Unique
derivation tree
for $a + a * a$



An un-successful example:

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

$$n, m \geq 0$$

L is inherently ambiguous: [دېڭىن ئۆزى ۋە دىلەر ambiguous،
birləşimi də ambiguous،

every grammar that generates this
language is ambiguous



Example (ambiguous) grammar for L :

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


$$S \rightarrow S_1 \mid S_2$$


$$S_1 \rightarrow S_1 c \mid A$$

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


$$S_2 \rightarrow aS_2 \mid B$$

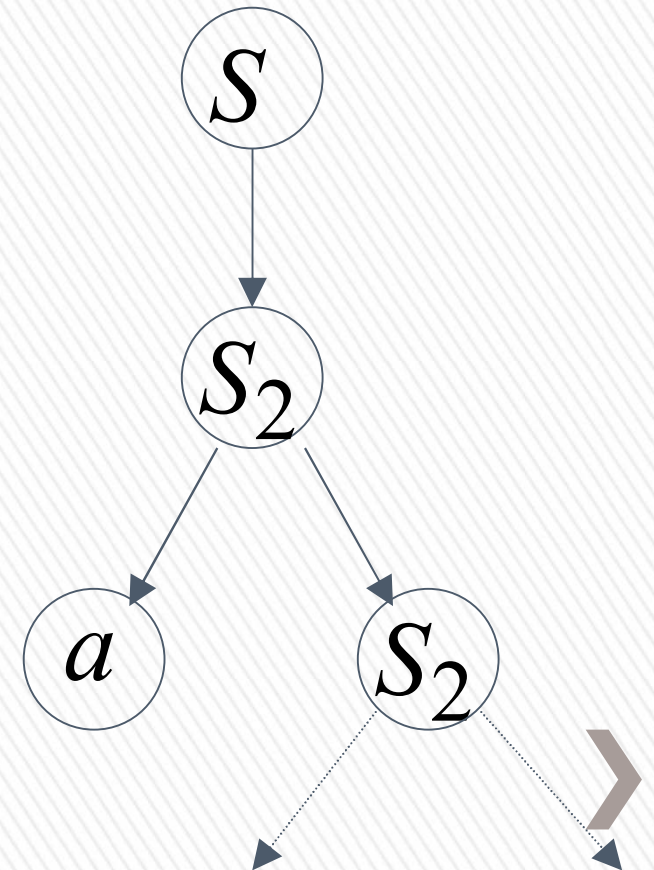
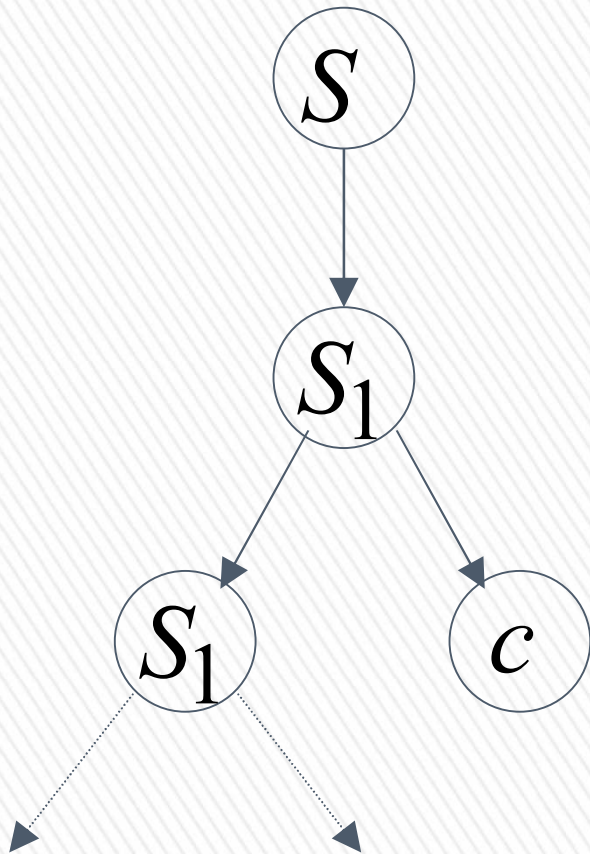
$$B \rightarrow bBc \mid \lambda$$



The string $a^n b^n c^n \in L$

has always two different derivation trees
(for any grammar)

For example



BLM2502 Theory of Computation

