جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

قسم الامن السيبراني

Department of Cyber Security

Subject: Computation Theory
Class: 3rd

Lecturer: Msc :Muntather AL-mussawee

Lecture: (6)
Language Grammar
Derivation Trees

Derivation Trees

Derivation tree is a graphical representation for the derivation of the given production rules for a given CFG. It is the simple way to show how the derivation can be done to obtain some string from a given set of production rules.

✍ The derivation tree is also called **parse tree** or **syntax tree** or **phrase tree** or **generation tree** or **production tree** or **derivation tree**.

The properties of parse tree are:

1- Root: The root node is always a node indicating start symbols.
2- PSG: The interior nodes are always the non-terminal nodes.
3- Leaves: The leaf node is always terminal nodes.
4- Links: collection of connections.

Phrase Structure Grammar (PSG) وهي القواعد التي ظهرت في الشككككش الشككككجري بشككككش العقد الوسككككطية التي تتحول في النهاية الى الرمول النهائية Terminal، وتتكون ايضككككان من اربعة مجامي
.PSG = (N, T, P, S)

Derivation

Derivation is a sequence of production rules. It is used to get the input string through these production rules. During parsing, we have to take two decisions. These are as follows:

- We have to decide the non-terminal which is to be replaced.
- We have to decide the production rule by which the non-terminal will be replaced.

We have two options to decide which non-terminal to be placed with production rule.

1. Leftmost Derivation:

In the leftmost derivation, the input is scanned and replaced with the production rule from left to right. So in leftmost derivation, we read the input string from left to right.

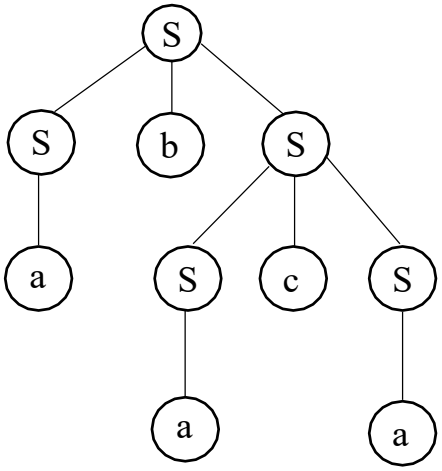
N ⟶ t | Nt - Leftmost Derivation

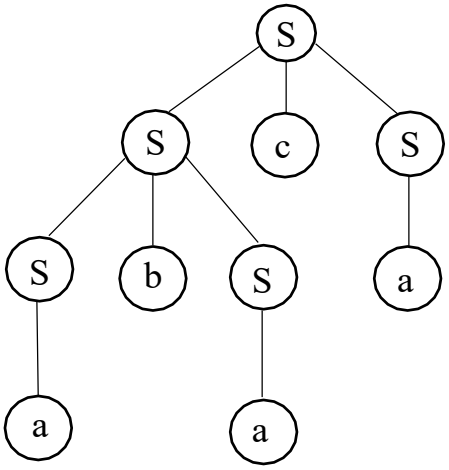
Example: Let G(L) = ({S}, {a, b, c}, P, S), where P is:
S → SbS | ScS | a

Find the string "abaca"

S → SbS
→ abS
→ abScS
→ abacS
→ abaca Accept

S → ScS
→ SbScS
→ abScS
→ abacS
→ abaca Accept





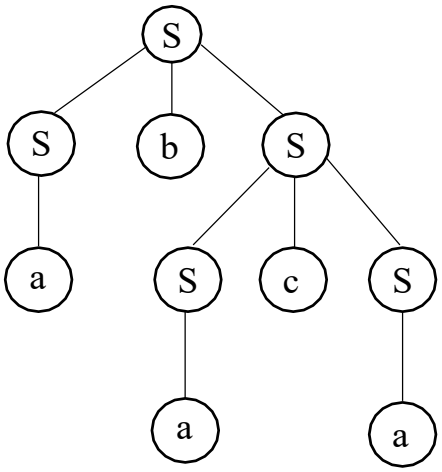
2. Rightmost Derivation:

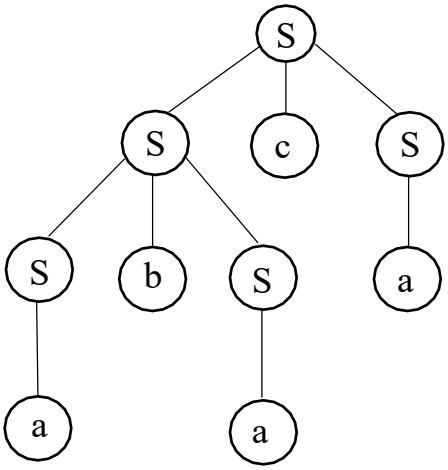
In rightmost derivation, the input is scanned and replaced with the production rule from right to left. So in rightmost derivation, we read the input string from right to left.

N ⟶ t | t<u>N</u> - Rightmost Derivation

S → S<u>bS</u>
→ SbS<u>cS</u>
→ Sb<u>Sca</u>
→ Sbaca
→ abaca Accept

S → S<u>cS</u>
→ Sca
→ SbSca
→ Sbaca
→ abaca Accept





✍ في المثال السككككككابق نلاحظ ان الحش الثاني هو نفس جهة الحش الاول، متكون من اشككككككتقاق من جهة اليمين واشتقاق من جهة اليسارولهُ نفس الشجرة.

نسككككتنتج من علك عندما يوجد امثر من حش من جهة اليسككار او امثر من حش من جهة اليمين تسككككمى القاعد غامضة ambiguous grammar.

Examples of Derivation:

<u>Example 1:</u>

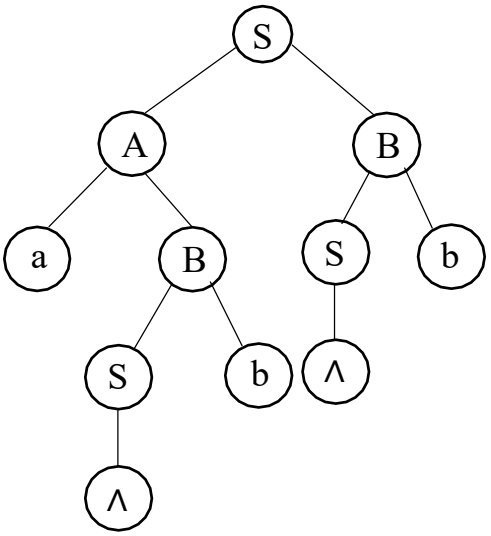
Derive the string "abb" for leftmost derivation and rightmost derivation using a CFG given by,

S ⟶ AB | ∧
A ⟶ aB
B ⟶ Sb

Solution:

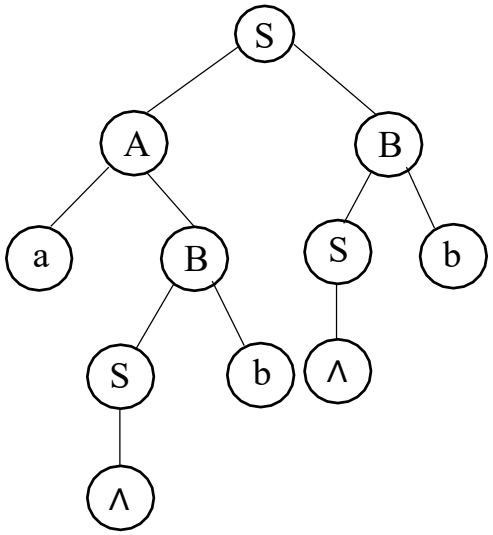
Leftmost derivation:

S ⟶ AB
→ aBB
→ aSbB
→ a∧bB
→ abSb
→ ab∧b
→ abb



Rightmost derivation:

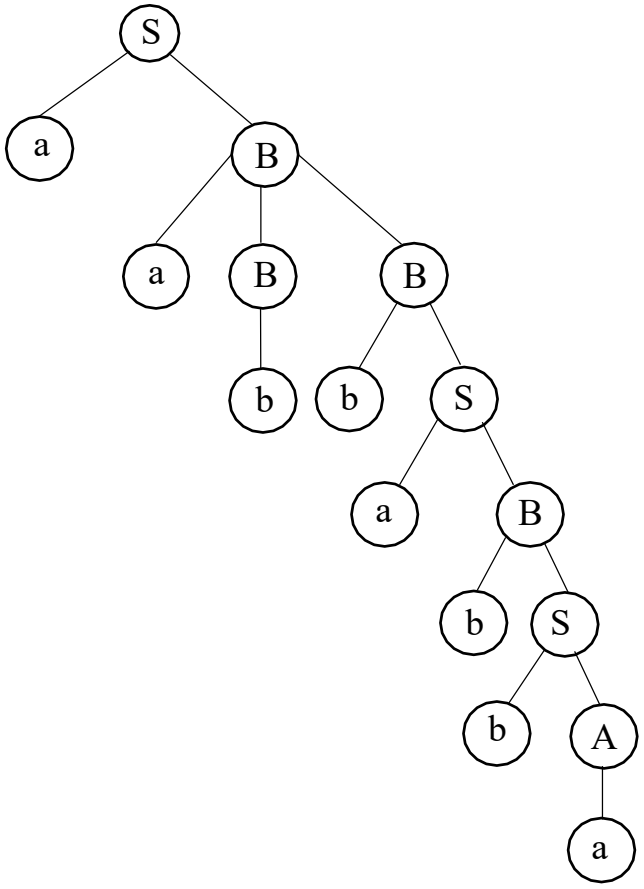
S → AB
→ ASb
→ A∧b
→ aBb
→ aSbb
→ a∧bb
→ abb



Example 2:

Derive the string "aabbabba" for leftmost derivation and rightmost derivation using a CFG given by,

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



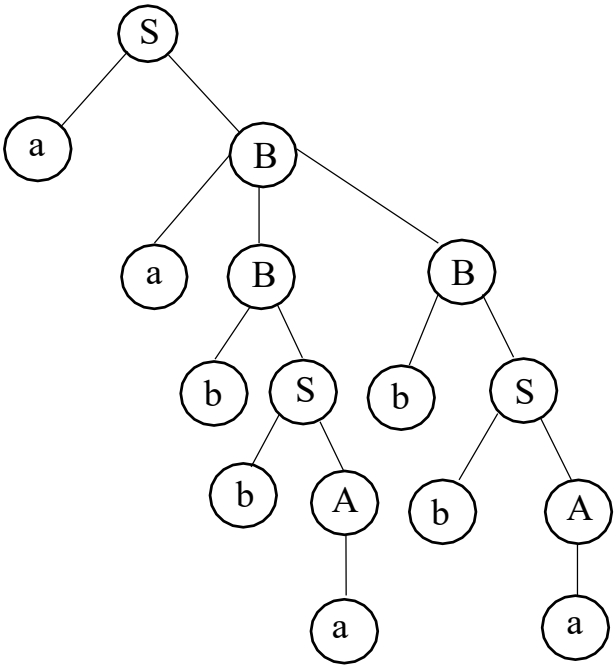
Solution:

Leftmost derivation:

S → aB
→ aaBB
→ aabB
→ aabbS
→ aabbaB
→ aabbabS
→ aabbabbA
→ aabbabba

Rightmost derivation:

S → aB
→ aaBB
→ aaBbS
→ aaBbbA
→ aaBbba
→ aabSbba
→ aabbAbba
→ aabbabba



Ambiguity in Grammar

A grammar is said to be ***ambiguous*** if there exists more than one leftmost derivation or more than one rightmost derivation or more than one parse tree for the given input string.
If it gives the same tree derivative from the leftmost derivation or from rightmost derivation the grammar is called ***unambiguous***.

If the grammar has ambiguity, then it is not good for compiler construction. No method can automatically detect and remove the ambiguity, but we can remove ambiguity by re-writing the whole grammar without ambiguity.

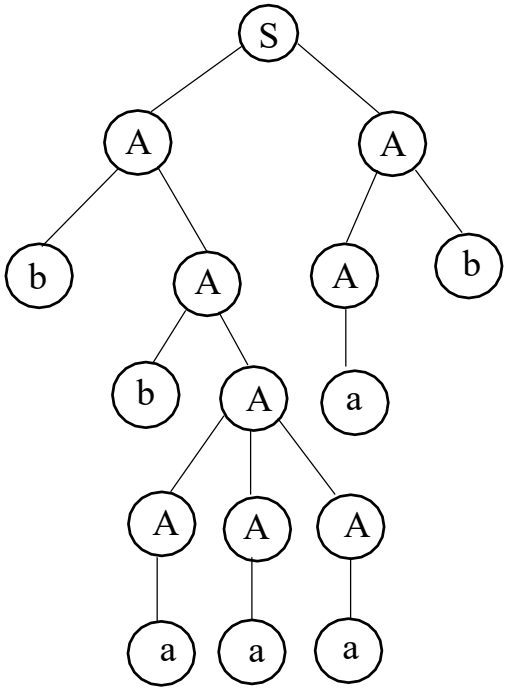
Example: Let G(L) = ({S, A}, {a, b}, P, S), where P is:

S → AA
A → AAA | bA | Ab | a

- Is the string "bbaaaab" Accept or not?
- Is the grammar ambiguous or not?

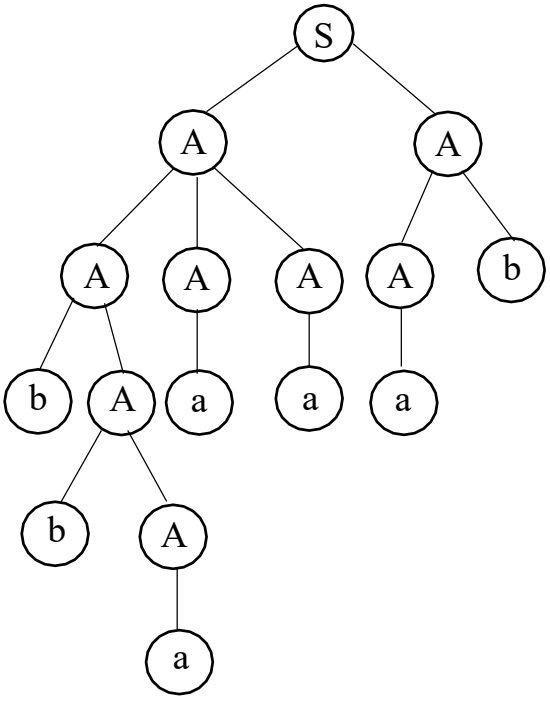
(Left) S → AA
→ bAA
→ bbAA
→ bbAAAA

→ bba<u>A</u>AA
→ bbaa<u>A</u>A
→ bbaaa<u>A</u>
→ bbaaa<u>A</u>b
→ bbaaaab Accept



(Left) S → <u>A</u>A
→ <u>A</u>AAA
→ b<u>A</u>AAA
→ bb<u>A</u>AAA
→ bba<u>A</u>AA
→ bbaa<u>A</u>A
→ bbaaa<u>A</u>
→ bbaaa<u>A</u>b
→ bbaaaab Accept

The grammar is ambiguous

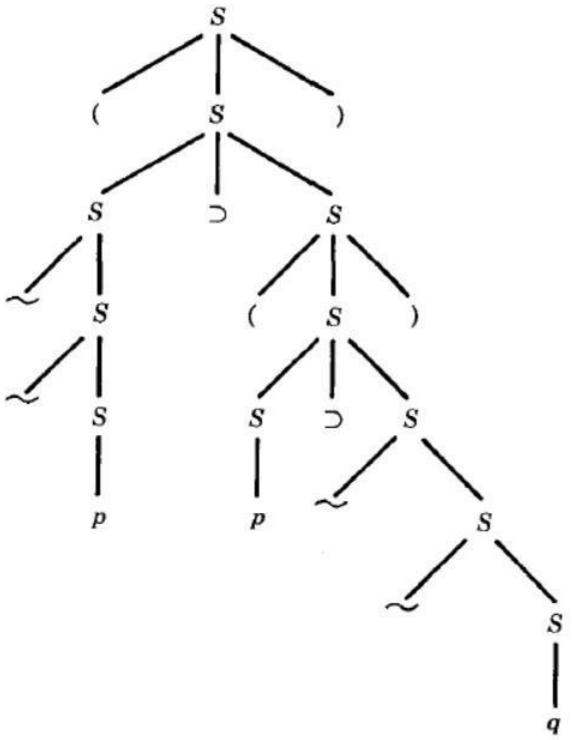


Example: Let the following grammar: G(L) = ({S}, {(,), ⊃, ~, p, q}, P, S}

S ⟶ (S) | S⊃S | ~S | p | q

- Is the string " (~ ~p ⊃ (p ⊃ ~ ~ q)) " Accept or not?
- Is the grammar ambiguous or not? **H.W**

(left) S ⟶ (S)
⟶ (S⊃S)
⟶ (~S⊃S)
⟶ (~ ~S⊃S)
⟶ (~ ~p⊃S)
⟶ (~ ~p⊃(S))
⟶ (~ ~p⊃(S⊃S))
⟶ (~ ~p⊃(p⊃S))
⟶ (~ ~p⊃(p⊃~S))
⟶ (~ ~p⊃(p⊃~ ~S))
⟶ (~ ~p ⊃(p ⊃ ~ ~ q)) Accept

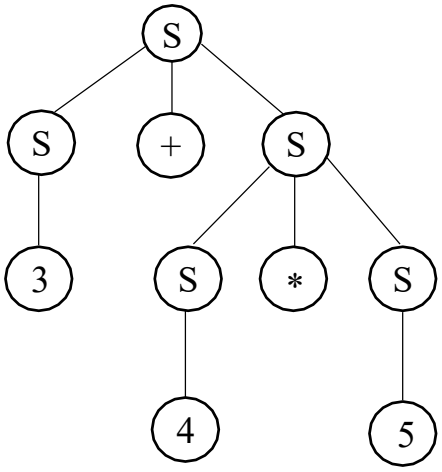


Example: Let the following grammar:

S → S +S | S ∗ S | digit

- Is the string " 3 + 4 ∗ 5 " Accept or not?
- Is the grammar ambiguous or not?

(Left) S ⟶ S + S
⟶ 3 + S
⟶ 3 + S ∗ S
⟶ 3 + 4 ∗ S
⟶ 3 + 4 ∗ 5 Accept



(Left) $S \longrightarrow S * S$

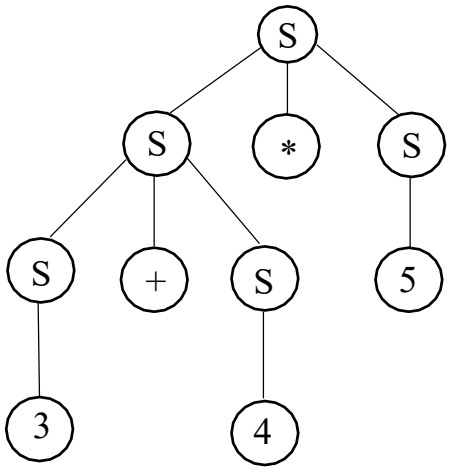
$\longrightarrow S + S * S$

$\longrightarrow 3 + S * S$

$\longrightarrow 3 + 4 * S$

$\longrightarrow 3 + 4 * 5$ Accept

The grammar is ambiguous



Homework: Let the following grammar:

$S \rightarrow aSa \mid bSb \mid a \mid b \mid \wedge$

- Is the string "aabaa" Accept or not?
- Is the grammar ambiguous or not?