CS & IT ENGINEERING

Theory of Computation

Push Down Automata

Lecture No. 02

By- DEVA Sir

TOPICS TO BE COVERED

01 CFG

02

03

04

05

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

RG

LLG

$\boxed{V \to VT^* \quad \text{OR} \quad V \to T^*}$

RLG

$\boxed{V \to T^*V \quad \text{OR} \quad V \to T^*}$

CFG

It represents CFL

$$V \to \underbrace{(V \cup T)}_{\text{Any sequence}}{}^*$$

I) $\boxed{S} \rightarrow SaAbS \mid a$

$\boxed{A} \rightarrow SAb \mid \varepsilon$

CFG ✓

II) $\boxed{S} \rightarrow aA$

$\boxed{aA} \rightarrow b$

CFG ✗

Note: i) Every RG is CFG

ii) CFG need not be RG

iii) Every Reg lang is CFL

iv) CFL is may or may not be Regular

CFG

- Definition ✓
- Derivation of a string
  - LMD (Left Most Derivation)
  - RMD (Right Most Derivation)
  - Parse Tree (Derivation Tree)
- Types of CFGs (In compiler)
  - Ambiguous CFG
  - Unambiguous CFG
- *** CFG Vs CFL
- Inherently ambiguous CFL

Example :

CFG

$$\boxed{\begin{aligned} S &\rightarrow AB \mid \dots \\ A &\rightarrow a \mid \dots \\ B &\rightarrow b \mid \dots \end{aligned}}$$

String : ab

Derivation :

① LMD

$$S \Rightarrow AB \xRightarrow{A \to a} aB \xRightarrow{B \to b} ab$$

No. of steps = 3

Left Sentential forms : AB, aB, ab

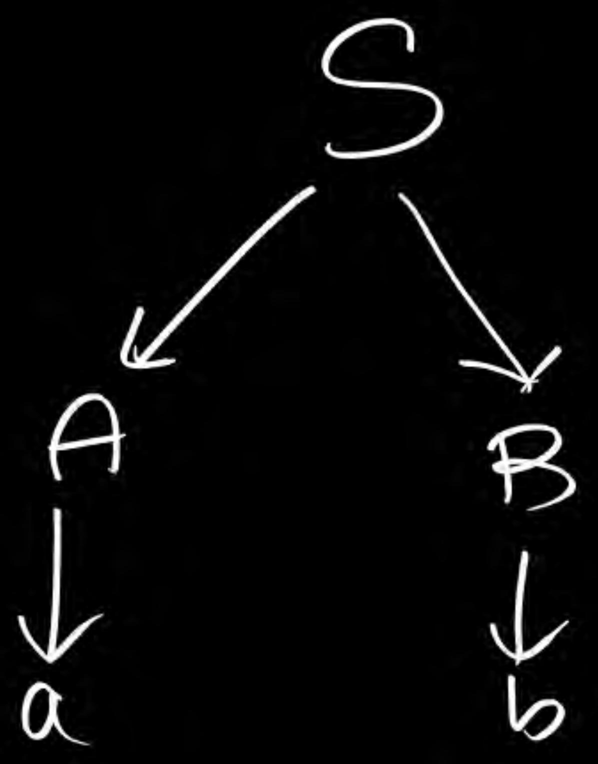
② RMD

$$S \Rightarrow AB \xRightarrow{B \to b} Ab \Rightarrow ab$$

No. of steps = 3

Right sentential forms : AB, Ab, ab

③ Parse Tree



No. of derivative steps

=

No. of substitution in LMD/RMD

Note: In general, LMD and RMD are need not be same

$S \to Aa \mid AB$

$A \to c$

$B \to d$

$\underbrace{w = ca}$

$\Downarrow$

LMD & RMD are same

$$\boxed{\begin{array}{c} S \\ \Downarrow \\ Aa \\ \Downarrow \\ ca \end{array}}$$ LMD RMD

$\underbrace{w = cd}$

$\Downarrow$

LMD & RMD are different

LMD	RMD
S	S
$\Downarrow$	$\Downarrow$
AB	AB
$\Downarrow$	$\Downarrow$
cB	Ad
$\Downarrow$	$\Downarrow$
cd	cd

LMD:

$S \to AB \mid \dots$

$A \to a \mid \dots$

$B \to b \mid \dots$

$S \Rightarrow ab$: $\boxed{S \Rightarrow AB \Rightarrow aB \Rightarrow ab}$ is LMD

$\xrightarrow[\text{LMD order}]{S,\ A,\ B}$

LMD

S

$\Downarrow$

$\boxed{AB}$

$\Downarrow$

$\boxed{aB}$

$\Downarrow$

$\boxed{ab}$ string

Left sentential forms:

AB, aB, ab

LMD: To derive a string,

in every sentential form

"left most non-terminal" is

Substituted

left most non-term

$a\,\boxed{A}\,b\,B$

Left most

Q1) Left most symbol = a

Q2) left most terminal = a

Q3) left most non-terminal = A

RMD: To deriva a string,

in every sentential form

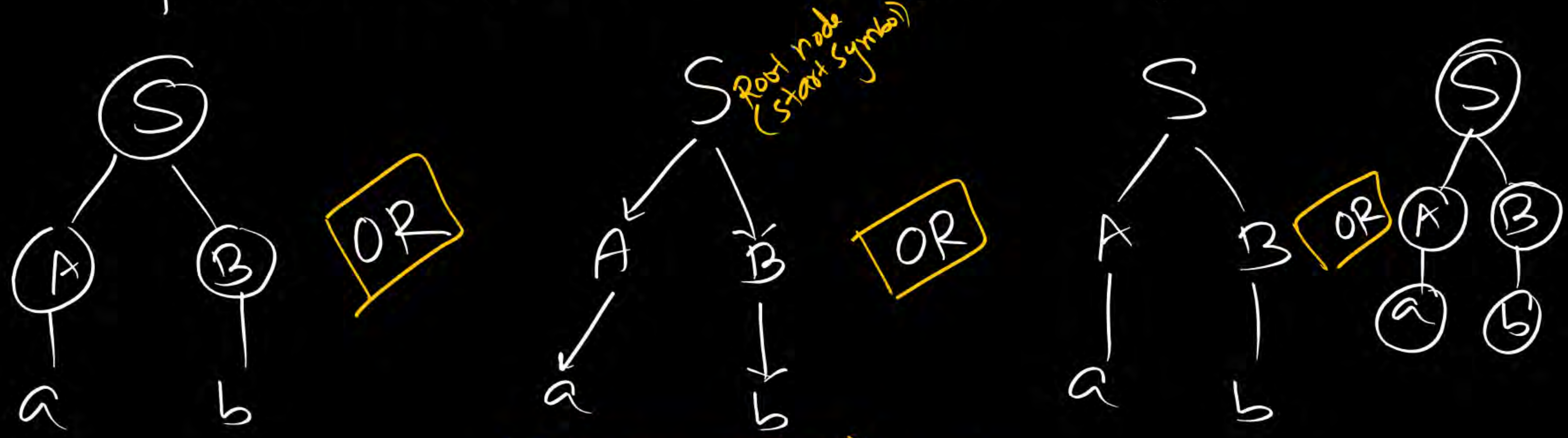
"Right most non-terminal" is

Substituted.

$$S \Downarrow \boxed{\text{Sentential form}}$$

Any sequence derived from S

Parse Tree (Derivation Tree) (Abstract Syntax Tree)

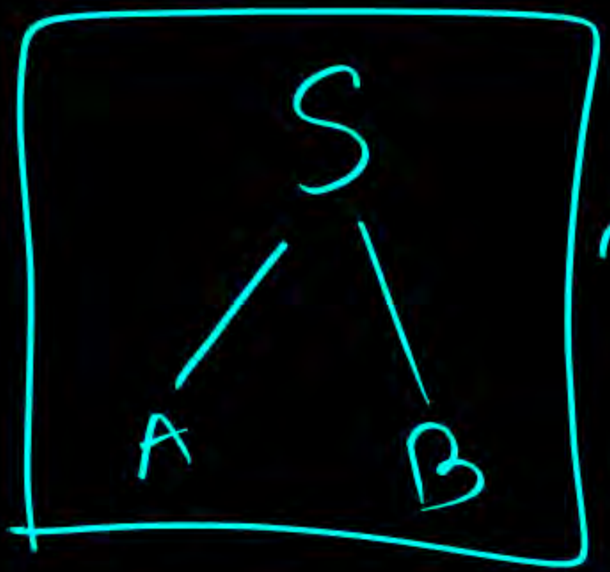


collect all leaf nodes = ab

Leaf node : Terminal or ϵ

Non leaf node (intermediate node) : Variable (non terminal)

$$\underbrace{S \longrightarrow AB}_{\text{production rule}}$$



Production Tree

A is child of S

B is child of S

S is parent of A&B

For a given String: (If only one derivation exist)

No. of derivative steps = No. of steps in LMD

= No. of steps in RMD

= No. of nonleaf nodes in parse tree

Example :

$S \rightarrow AB \mid ab$

$A \rightarrow a \mid c$

$B \rightarrow b \mid d$

$w = ab$

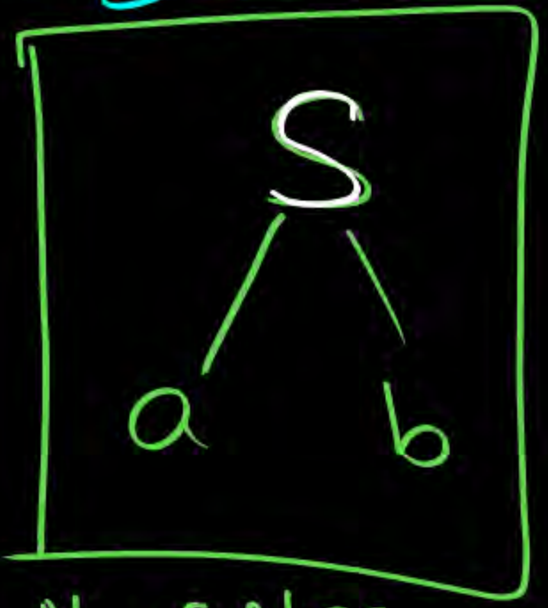
How many parse trees ? = 2

(derivations)

(no. of LMDs)

(no. of RMDs)

Derivation 1



No. of steps = 1

and

Derivation 2

No. of steps = 3

= No. of RMDs
= No. of LMDs
= No. of Parse trees

No. of derivations ⇒ How many ways string can be derived?

$\neq$

$\boxed{\text{No. of derivative steps}}$ ⇒ For a particular derivation, How many steps?

→ It is possible to answer if we have only one derivation.

→ No. of substitutions in LMD
No. of " in RMD
No. of Non leaf nodes in Parse tree

Types of CFGs:

I) **Ambiguous CFG**

- Some string (atleast one) has more than 1 derivation

$S \rightarrow AB \mid ab$
$A \rightarrow a \mid c$
$B \rightarrow b \mid d$

$cd \Rightarrow 1$ PT
$ab \Rightarrow 2$ PTs

II) **Unambiguous CFG** (not ambiguous)

- Every string derived from CFG has only one derivation

$S \rightarrow a \mid AB$
$A \rightarrow a$
$B \rightarrow b$

$\underbrace{w_1 = a}_{\text{only 1 PT}}$ $\underbrace{w = ab}_{\text{only 1 PT}}$

① $S \to a$

Unambiguous CFG

② $S \to a \mid A$

$A \to a \mid b$

$w = a$

$$\begin{array}{c} S \\ | \\ a \end{array} \qquad \begin{array}{c} S \\ | \\ A \\ | \\ a \end{array}$$

Amb CFG

③ $S \to AB \mid cd$

$A \to a$

$B \to b$

Unamb CFG

Easy to check Ambiguous CFG

↳ you have to find one string which has >1 PT

CFG Vs CFL :

① $S \rightarrow AB$

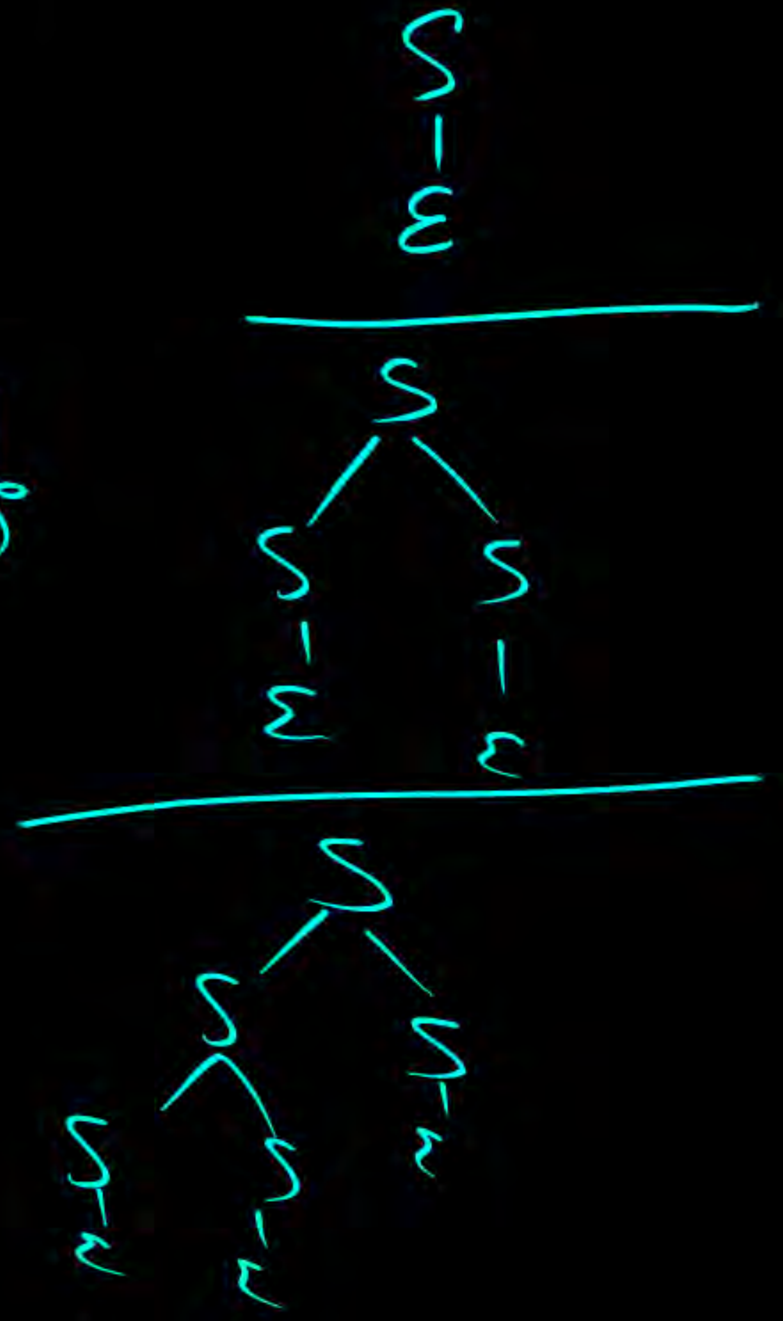
$A \rightarrow \varepsilon$

$B \rightarrow b$

$L = \{b\}$

② $S \rightarrow SS \mid \varepsilon$

$L = \{\varepsilon\}$



③ $S \to Sa \mid \varepsilon$

$L = a^*$

④ $S \to aS \mid \varepsilon$

$L = a^*$

⑤ $S \to Sa \mid a$

$L = a^+$

⑥ $S \to aS \mid a$

$L = a^+$

⑦ $S \rightarrow aS \mid bS \mid \varepsilon$

$L = (a+b)^*$

⑧ $S \rightarrow aS \mid bS \mid a \mid b$

$L = (a+b)^+$

⑨ $S \rightarrow Sa \mid Sb \mid \varepsilon$

$L = (a+b)^*$

⑩ $S \rightarrow Sa \mid Sb \mid a \mid b$

$L = (a+b)^+$

⑪ $S \rightarrow aaS \mid \varepsilon$

$L = (aa)^*$

⑫ $S \rightarrow aS \mid bS \mid cS \mid \varepsilon$

$L = (a+b+c)^*$

⑬ $S \rightarrow abS \mid aaS \mid \varepsilon$

$L = (ab+aa)^*$

⑭ $S \rightarrow aS \mid bS \mid b$

$L = (a+b)^* b$

$$(\underbrace{ab+aa})^*$$

$$= \left(a(b+a) \right)^*$$

$$= \{\varepsilon, aa, ab, aaaa, aaab, abab, abaa, \ldots\}$$

$$\boxed{\neq a(a+b)^*}$$

(15)

$S \to AB$

$A \to aA \mid \epsilon$

$B \to Bb \mid \epsilon$

$B = b^*$

$A = a^*$

It is not regular grammar

But

It generates a regular language

$L = a^* b^*$

16

Not RG

But generates Regular lang

$S \to aS \mid Sb \mid \varepsilon$

$a^* S b^* \Rightarrow a^* b^*$

ε ✓

a ✓

b ✓

aa ✓

bb ✓

ab ✓

~~ba~~

$a^* b^*$

(17) $S \to bS \mid Sa \mid \varepsilon \Rightarrow b^* a^*$

$b^* S a^*$

(18) $S \to aS \mid Sb \mid a \Rightarrow L = a^* a b^* = a^+ b^*$

$a^* S b^*$

(19) $S \to aS \mid Sb \mid b \Rightarrow L = a^* b b^* = a^* b^+$

(20) $S \to aS \mid Sb \mid a \mid b \Rightarrow L = a^*(a+b)b^*$

$= a^+ b^* + a^* b^+$

$a^* S b^*$

(21) $S \rightarrow aS \mid Sb \mid ab$

$L = a^* ab\, b^*$

$= a^+ b^+$

Note:

$$X \rightarrow \alpha X \mid X\beta \mid \gamma$$

$$\boxed{L = \alpha^* \gamma \beta^*}$$

$\left.\begin{matrix}\gamma \\ \beta \\ \gamma\end{matrix}\right\}$ terminal sequence $\in T^*$

*** (22) $S \rightarrow aSb \mid \varepsilon$

same count

$L = a^n S b^n$

$= a^n b^n$

$= \{a^n b^n \mid n \geq 0\}$

$= \{a^* b^* \mid \#a's = \#b's\}$

ε ✓

ab ✓

$aabb$ ✓

$aaabbb$ ✓

$a^n \quad b^n$

(23) $S \rightarrow bSa \mid \varepsilon$ $L=\{b^n a^n \mid n \geq 0\}$

(24) $S \rightarrow aSb \mid a$ $L=\{a^n a b^n \mid n \geq 0\}$
$=\{a^{n+1} b^n\}$

(25) $S \rightarrow aSb \mid b$ $L=\{a^n b^{n+1} \mid n \geq 0\}$

(26) $S \rightarrow aaSbb \mid \varepsilon$ $L=\{a^{2n} b^{2n}\}$

(27) $S \rightarrow aaaSbbb \mid \varepsilon$ $L=\{a^{3n} b^{3n}\}$

(28) $S \rightarrow aSbb \mid \varepsilon$ $L=\{a^n b^{2n} \mid n \geq 0\}$

(29) $S \rightarrow aaSb \mid \varepsilon$

$L=\{a^{2n} b^n\}$
$=\{a^m b^n \mid m=2n\}$

(30) $S \rightarrow aaSbbb \mid \varepsilon$

$L=\{a^{2n} b^{3n} \mid n \geq 0\}$
$=\{a^i b^j \mid i=2n, j=3n\}$

(31) $S \to aSb \mid ab$

$L = \{ab, a^2b^2, a^3b^3, \ldots\}$

$=$

a^+b^+

$L = \{a^n ab\, b^n \mid n \geq 0\}$

$= \{a^{n+1} b^{n+1} \mid n \geq 0\}$

$= \{a^n b^n \mid n \geq 1\}$

$= \{a^{n-1} b^{n-1} \mid n \geq 2\}$

$$a^m b^n = a^* b^* \neq a^n b^n$$

$a^* b^*$	$a^n b^n$
ε ✓	ε ✓
a ✓	ab ✓
b ✓	aabb ✓
ab ✓	aaabbb ✓
aa ✓	⋮
bb ✓	
aab ✓	
abb ✓	
⋮	

Note: $L_1 = \{a^n \mid n \geq 0\} = a^*$

$L_2 = \{b^n \mid n \geq 0\} = b^*$

$\Downarrow$

$L_1 . L_2 = \{a^n\} . \{b^n\}$

$= \{\varepsilon, a^1, a^2, \ldots\} . \{\varepsilon, b, b^2, \ldots\}$

$= \{a^m b^n \mid m, n \geq 0\}$

$= \{a^{n_1} b^{n_2} \mid n_1, n_2 \geq 0\}$

A) $a^* b^*$ ✓

B) $\{a^n b^n\}$

C) $\{a^m b^n\}$ ✓

D) All

$$\{a^n b^n\} = \{a^n b^n \mid n \geq 0\}$$

(32) $S \to aSb \mid A$

$A \to aA \mid \varepsilon$

$\boxed{A = a^*}$ $\to$ (1)

$S \to aSb \mid a^*$

$L = \{a^n a^* b^n\}$

$= \{a^i b^j \mid i \geq j\}$

(33) $S \to aSb \mid A$

$\boxed{A \to bA \mid \varepsilon} \Rightarrow b^*$

$L = a^n b^* b^n$

$= \{a^n b^{m+n}\}$

$L = \{a^i b^j \mid i \leq j\}$

34

$S \to aSb \mid A$

$A \to bB \mid \varepsilon$

Useless

$\Rightarrow S \to aSb \mid \varepsilon$

$L = a^n b^n$

(35) $S \to SA \mid \varepsilon$

$A \to a$

$\boxed{L = A^* = a^*}$

(36) $S \to SA \mid \varepsilon$

$A \to aa$

$L = A^* = (aa)^*$

(37) $S \to SA \mid \varepsilon$

$A \to a \mid b$

$\boxed{L = A^* = (a+b)^*}$

***(38) $S \to SA \mid \varepsilon$

$A \to aAb \mid \varepsilon$

$L = \{a^n b^n\}^*$

(39) $S \to SA \mid \varepsilon$

$A \to bAa \mid \varepsilon$

$L = \{b^n a^n\}^*$

(40) $S \to SA \mid \varepsilon$

$A \to ab$

$L = (ab)^*$

(38) $S \to SA \mid \varepsilon$

$A \to aAb \mid \varepsilon$

$L = \{a^n b^n\}^*$

$= \{a^n b^n \mid n \geq 0\}^*$

ababab

aaabbbaaabbbaaabbbaaabbb

$A = \{a^n b^n \mid n \geq 0\}$

$L = A^* = \{a^n b^n \mid n \geq 0\}^* \neq \{(a^n b^n)^* \mid n \geq 0\}$

It is not CFL

abaaabbb ✓

abaaabbb ✗

$$\{a^n b^n\}^* = \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} a^{n_3} b^{n_3} \cdots a^{n_k} b^{n_k}\}$$

$$n_1, n_2, \ldots, n_k \geq 0$$

$$\{a^n b^n\}^0 = \varepsilon$$

$$\{a^n b^n\}^1 \longrightarrow \{a^{n_1} b^{n_1}\}$$

$$\{a^n b^n\}^2 \longrightarrow \{a^n b^n\}.\{a^n b^n\} = \{a^{n_1} b^{n_1} a^{n_2} b^{n_2}\}$$

$$\{a^n b^n\}^3 \longrightarrow \{a^n b^n\}.\{a^n b^n\}.\{a^n b^n\} = \{a^{n_1} b^{n_1}. a^{n_2} b^{n_2}. a^{n_3} b^{n_3}\}$$

New concept 41 $S \rightarrow aSa \mid bSb \mid \varepsilon$

42 $S \rightarrow aSa \mid bSb \mid a \mid b$

43 $S \rightarrow aSa \mid bSb \mid a \mid b \mid \varepsilon$

44 $S \rightarrow aSa \mid bSb \mid \#$

45. $S \rightarrow aSb \mid A$

$A \rightarrow aAc \mid \epsilon$

46. $S \rightarrow AB$

$A \rightarrow aAb \mid \epsilon$

$B \rightarrow cBd \mid \epsilon$

47. $S \rightarrow aSb \mid A$

$A \rightarrow cAb \mid \epsilon$

48. $S \rightarrow ABC$

$A \rightarrow aA \mid \epsilon$

$B \rightarrow Bb \mid \epsilon$

$C \rightarrow Cc \mid \epsilon$

49 $S \rightarrow SS \mid (S) \mid \varepsilon$

50 $E \rightarrow E+E \mid E*E \mid (E) \mid a$

functionality
(meaning)

Summary

→ CFG ✓


THANK YOU
GW
SOLDIERS !