

Eg:  $A \rightarrow aB|a$   
 $B \rightarrow aB|bB|a|b$

} Right Linear Grammar  
 ↓  
 Regular Grammar

Eg:  $A \rightarrow Ba|a$   
 $B \rightarrow Ba|Bb|b|a$

} Left Linear Grammar  
 ↓  
 Regular Grammar

Eg:  $A \rightarrow Ba|a$        $\rightarrow$  LLG  
 $B \rightarrow aB|a$        $\rightarrow$  RLG

} Not a Regular Grammar  
 (Type 3 Grammar)  
 bcz it is a combination LLG &  
 RLG

### Type 2:

Context free Grammar

Productions of the form:

$$A \rightarrow \alpha$$

$$\begin{array}{l} A \in V \\ \alpha \in (V \cup T)^* \end{array}$$

Eg:  $A \rightarrow aAb|ab | Bc$   
 $B \rightarrow Aab$

Variable      Any combination of variables & terminals

$$\begin{array}{ll} \underbrace{Aa}_{\text{Variable}} \rightarrow bCd & \times \\ AB \rightarrow bd & \times \end{array}$$

### Ambiguity

$$E \rightarrow E+E | E^*E | id$$

$$\begin{array}{l} V = \{E\} \\ T = \{+, ^*, id\} \\ S = \{E\} \end{array}$$

String:  $id + id * id$

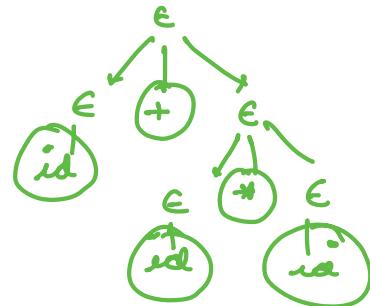
Leftmost Derivation

$$\begin{aligned} E &\rightarrow \underline{E+E} \\ &\rightarrow id + \underline{E} \\ &\rightarrow id + \underline{E+E^*E} \\ &\rightarrow id + id + \underline{id^*E} \\ &\rightarrow id + id + id \end{aligned}$$

Rightmost Derivation

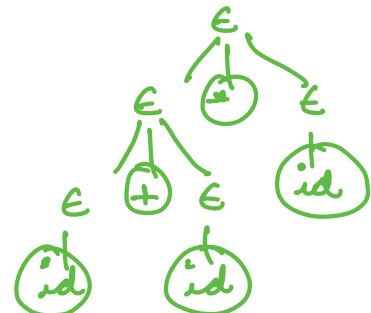
$$\begin{aligned} E &\rightarrow E + \underline{E} \\ &\rightarrow E + E + \underline{E^*E} \\ &\rightarrow E + E + \underline{E^*id} \\ &\rightarrow E + id + \underline{id^*id} \\ &\rightarrow id + id + \underline{id^*id} \end{aligned}$$

Parse Tree



$$\begin{aligned} E &\rightarrow E^*E \\ &\rightarrow E+E^*E \\ &\rightarrow id+E^*E \\ &\rightarrow id+id+E^*E \\ &\rightarrow id+id+id^*id \end{aligned}$$

$$\begin{aligned} E &\rightarrow E^*E \\ &\rightarrow E^*id \\ &\rightarrow E+E^*id \\ &\rightarrow E+id+E^*id \\ &\rightarrow id+id+E^*id \end{aligned}$$



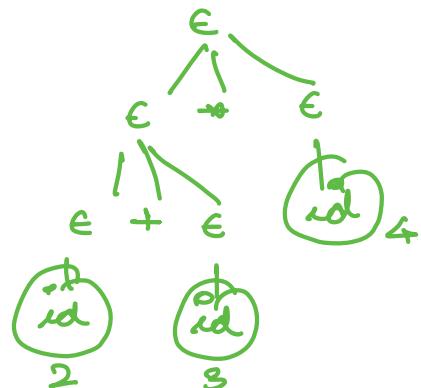
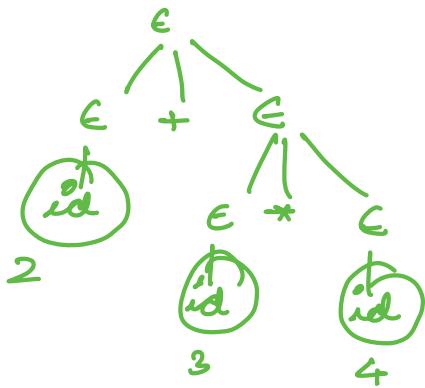
If you have more than 1 LHD or more than 1 RHD or  
more than 1 PT



Grammar is ambiguous.

Eg:  $2+3*4$

int a = 2+3\*4;



$2+12=14$

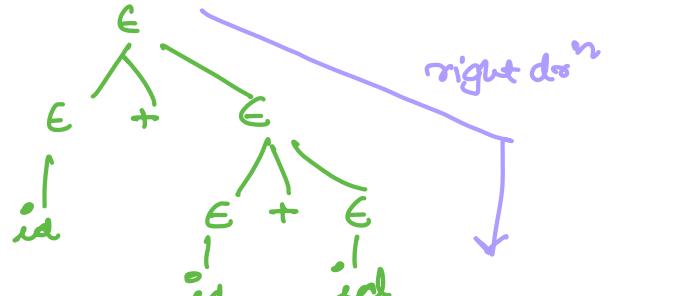
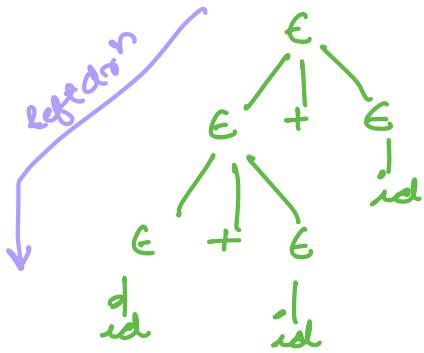
20

Confusion?

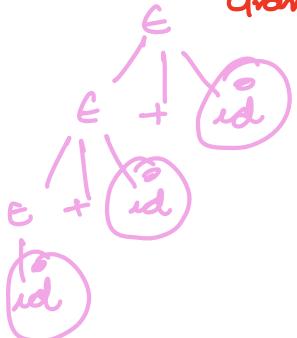
## Disambiguation:

$$E \rightarrow E^* E \mid E + E \mid id$$

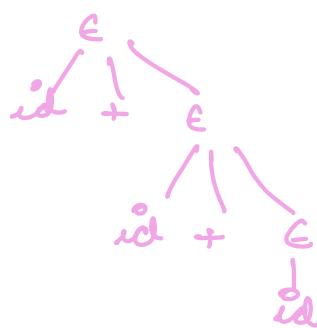
Case 1:  $id + id + id$



$E \rightarrow E + E \mid id$   
 $E \rightarrow E + id \mid id$   
 left Recursive Grammar



$E \rightarrow id + E \mid id$   
 Right Recursive Grammar



operator  
 $2 + 3 * 4$   
 operand  
 -  
 operand -

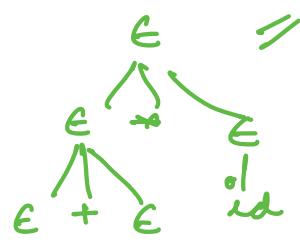
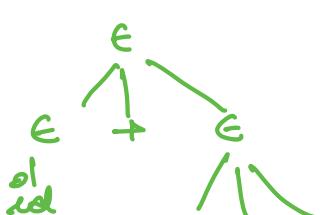
$2 + 3 + 5$   
 $(2+3)+5$   
 left  
 $2 + (3+5)$   
 right

$= power$   
 $2^3^5$   
 $2^{3^5}$

$2^{(3^5)}$

Case 2: Precedence Rules

$$id + id * id$$



$$\begin{aligned}
 C &\rightarrow E + T \mid \\
 T &\rightarrow R * R \mid id \\
 R &\rightarrow id
 \end{aligned}$$

$\epsilon \rightarrow \epsilon^* \epsilon$   
 $\text{id} \quad \text{id}$

$\text{id} \quad \text{id}$

$\text{id} + (\text{id} * \text{id})$



same level  
 $\epsilon \rightarrow \epsilon^* \epsilon \mid \epsilon + \epsilon \mid \text{id}$



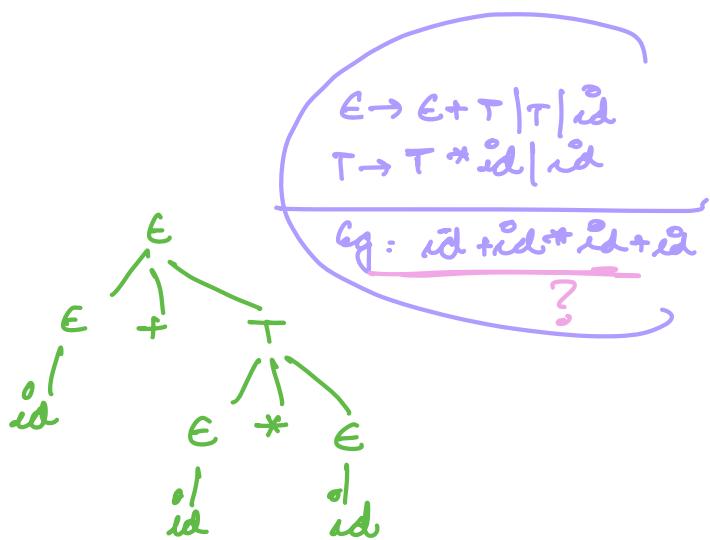
$\epsilon \rightarrow \epsilon + T \mid T \mid \text{id}$

$T \rightarrow \epsilon^* \epsilon \mid \text{id}$

$(\text{id} + \text{id})^* \text{id}$



Should not get generated



Ambiguous Situation

→ Associativity  
→ Precedence

→ left Rec OR right Rec

↓  
higher precedence move far from Start Symbol

## Simplification of CFG:

### ① Remove $\epsilon$ productions:

- We will not be able to remove all  $\epsilon$  productions, if  $\epsilon$  is present in language.
- If language does not contain  $\epsilon$  then we can remove all  $\epsilon$  productions.

Eg:  $S \rightarrow aSb \mid aAb$   
 $A \rightarrow \epsilon$   
    \epsilonpsilonion

1. find out all nullable variables.



- if a variable can directly give  $\epsilon$  like  $A \rightarrow \epsilon$
- if a variable can generate  $\epsilon$  after some no. of steps

$$A \rightarrow (\ ) \rightarrow (\ ) \rightarrow \epsilon$$

$$\text{nullable variable} = \{A\}$$

2. goto RHS of every production and wherever nullable variable is present write with it and without it.

$$\begin{array}{l} S \rightarrow aSb \mid aAb \\ A \rightarrow \epsilon \end{array} \longrightarrow \begin{array}{l} S \rightarrow aSb \mid a\underline{Ab} \mid ab \\ \underline{A \rightarrow \epsilon} \\ \text{eliminate it} \end{array}$$

3. now in production A is no longer present. So remove aAb

$$S \rightarrow aSb \mid ab$$

Eg:

$$\begin{array}{l} S \rightarrow AB \\ A \rightarrow aAA \mid \epsilon \\ B \rightarrow bBB \mid \epsilon \end{array}$$

$$\text{nullable variable} = \{A, B, S\}$$

$$S \rightarrow \underline{AB} \rightarrow B \rightarrow \epsilon$$

$$\begin{aligned} S &\rightarrow AB \mid B \mid A \mid \epsilon \\ A &\rightarrow AAA \mid AA \mid a \\ B &\rightarrow BBB \mid BB \mid b \end{aligned}$$

Eg:

$$\begin{aligned} S &\rightarrow AbaC \\ A &\rightarrow BC \\ B &\rightarrow b \mid \epsilon \\ C &\rightarrow D \mid \epsilon \\ D &\rightarrow d \end{aligned}$$

Start Symbol is a nullable variable that means  $\epsilon$  is present in language.

$L = \text{Even length strings}$   
 $\Sigma = \{a, b\}$   
 $L = \{\epsilon, aa, ab, ba, bb, \dots\}$

nullable variable = {A, B, C}

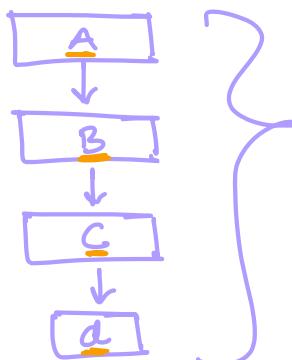
$$\begin{aligned} S &\rightarrow AbaC \mid bac \mid Aba \mid ba \\ A &\rightarrow BC \mid B \mid C \\ B &\rightarrow b \\ C &\rightarrow D \\ D &\rightarrow d \end{aligned}$$

In process of removing  $\epsilon$  production, you might get unit production.

$$\begin{aligned} A &\rightarrow B \\ B &\rightarrow C \\ C &\rightarrow d \end{aligned}$$



$$\begin{aligned} A &\rightarrow d \\ B &\rightarrow d \\ C &\rightarrow d \end{aligned}$$



$$\frac{\text{Variable} \rightarrow \text{Variable}}{\text{length 1} \quad \text{length 1}}$$

lengths is not changing.  
wastage of steps  
 $A \rightarrow d$

## ② Elimination of unit productions:

Eg:

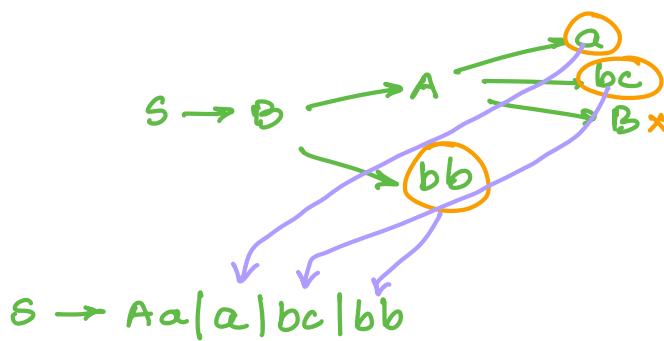
$$\begin{aligned} S &\rightarrow Aa \mid B \\ B &\rightarrow A \mid bb \\ A &\rightarrow a \mid bc \mid B \end{aligned}$$

Write grammar without unit production.

$$\begin{aligned} S &\rightarrow Aa \\ B &\rightarrow bb \\ A &\rightarrow a \mid bc \end{aligned}$$

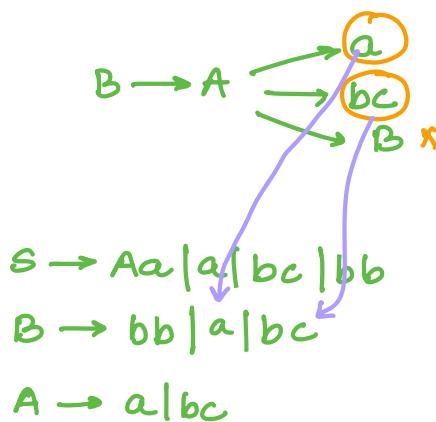
Look at every unit production one by one

$$S \rightarrow B$$



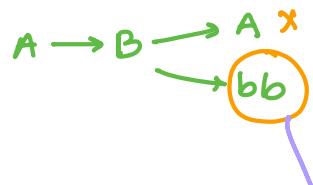
$$\begin{aligned} B &\rightarrow bb \\ A &\rightarrow a \mid bc \end{aligned}$$

$$B \rightarrow A$$



$$A \rightarrow a \mid bc$$

$$A \rightarrow B$$



After removal of unit production, language shouldn't be affected.

$$S \rightarrow Aa \mid a \mid bc \mid bb$$

$$B \rightarrow bb \mid a \mid bc$$

$$A \rightarrow a \mid bc \mid bb$$

Eq:

$$S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow C \mid b$$

$$C \rightarrow D$$

$$D \rightarrow E$$

$$E \rightarrow a$$

Grammar without unit production

$$S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow b$$

$$E \rightarrow a$$

$$B \rightarrow C$$

$$B \rightarrow C \rightarrow D \rightarrow E \rightarrow a$$

$$S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow b \mid a$$

$$E \rightarrow a$$

$$C \rightarrow D$$

$$C \rightarrow D \rightarrow E \rightarrow a$$

$$S \rightarrow AB$$

$$A \rightarrow a$$

$$B \rightarrow b \mid a$$

$$E \rightarrow a$$

$$C \rightarrow a$$

$$D \rightarrow E$$

$$D \rightarrow E \rightarrow a$$

$S \rightarrow AB$   
 $A \rightarrow a$   
 $B \rightarrow b/a$   
 $E \rightarrow a$   
 $C \rightarrow a$   
 $D \rightarrow a$

$C, D, E$  are not reachable

↳ useless symbol

### ③ Elimination of Useless Symbol:

- Variable which can derive a string of terminals is said to be useful.

$A \rightarrow \dots \rightarrow \text{terminals}$

- Variable should be reachable from start state.

Eg:

$S \rightarrow AB/a$   
 $A \rightarrow BC/b$   
 $B \rightarrow aB/c$   
 $C \rightarrow ac/B$

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

$$V = \{S, A, B, C\}$$

- All terminals are useful symbols  $\{a, b\}$
- Rhs of production contains only terminals, then lhs of production will also be useful.

$S \rightarrow AB/a$   
 $A \rightarrow BC/b$   
 $B \rightarrow aB/c$   
 $C \rightarrow ac/B$

$$\{a, b, S, A\}$$

- Whatever symbols have been proven to be useful, just use these symbols and find out if Rhs of production is made up of only these symbols.

$B \rightarrow aB/c$  not useful  
 $C \rightarrow ac/B$

B & C are useless symbols.

$$B \rightarrow aB \rightarrow aaB \rightarrow aaaB \dots$$

- Delete all productions having B & C on LHS & also the productions having B & C on RHS

$$\begin{array}{l} S \rightarrow AB | a \\ A \rightarrow BC | b \\ \cancel{B \rightarrow aB | c} \\ \cancel{C \rightarrow ac | b} \end{array}$$

$$\begin{array}{l} S \rightarrow a \\ A \rightarrow b \end{array}$$

- Symbol which is unreachable, remove it.

$$\begin{array}{l} S \rightarrow a \\ \cancel{A \rightarrow b} \end{array} \rightarrow S \rightarrow a$$

Eg:  $S \rightarrow AB | AC$

$$A \rightarrow aAb | bAa | a$$

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

$$B \rightarrow bba | aaB | AB$$

$$V = \{A, B, C, D, \Sigma\}$$

$$C \rightarrow abcA | aDb$$

$$D \rightarrow bD | ac$$

① Terminals are useful  $\{a, b\}$

② Variables which are giving terminals  $\{a, b, A\}$

③  $B \rightarrow \underline{bba} | aaB | AB$   
    ↓                  ↓  
    useful          useful.

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



C & D are not useful.

①

$$S \rightarrow AB \mid A \cancel{X}$$

$$A \rightarrow aAb \mid bAa \mid a$$

$$B \rightarrow bBA \mid aAB \mid AB$$

~~$c \rightarrow abcA \mid acD$~~

~~$D \rightarrow BD \mid ac$~~

②

$$S \rightarrow AB$$

$$A \rightarrow aAb \mid bAa \mid a$$

$$B \rightarrow bBA \mid aAB \mid AB$$

Simplify CFG :

1.  $\epsilon$  productions
2. Unit productions
3. Useless Symbols

## Normal forms of CFG

**CNF**  
(Chomsky Normal form)

**GNF**

(Gratobach Normal form)

$$\underline{A} \rightarrow \underline{\underline{V, T}}$$

### Chomsky Normal form (CNF):

A CFG is in CNF if all production rules satisfy one of the following conditions:

- A non terminal generating a terminal ( $A \rightarrow a$ )
- A non terminal generating 2 non-terminals ( $A \rightarrow BC$ )
- Start symbol generating  $\epsilon$  ( $S \rightarrow \epsilon$ )

Eg:

$$\begin{aligned} S &\rightarrow a \\ S &\rightarrow AZ \\ A &\rightarrow a \\ Z &\rightarrow b \end{aligned}$$

CFG is in CNF

$$\begin{aligned} S &\rightarrow a \\ S &\rightarrow az \\ Z &\rightarrow b \end{aligned}$$

CFG is not in CNF.

$$\begin{aligned} 1V &\rightarrow 2V \\ 1V &\rightarrow T \\ S &\rightarrow E \end{aligned}$$

### Properties:

- for a given grammar, there can be more than 1 CNF
- CNF produces the same language as generated by CFG.
- for generating a string of length  $n$ , you require  $2n-1$  steps in CNF.

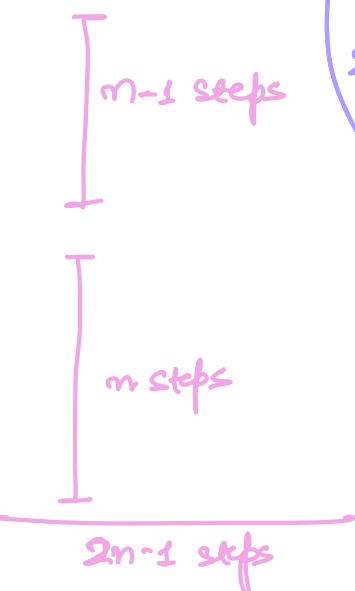
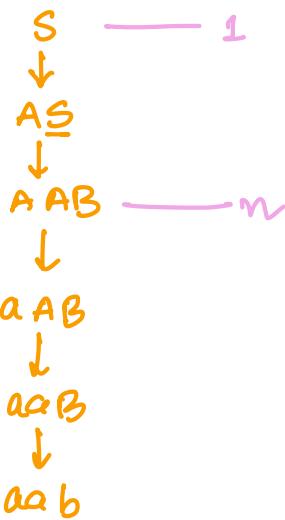
$$\text{CFG} \rightarrow \text{CNF}$$

Cnf:  $S \rightarrow AB \mid AS$

$A \rightarrow a$

$B \rightarrow b$

"aab"



Grammar  
CNF  
String ?

$2n-1$  steps

- Any CFG that doesn't have  $\epsilon$  in its language has an equivalent CNF.

Conversion from CFG to Cnf?

1. Eliminate start symbol from RHS

$$\begin{array}{ccc} S \rightarrow SA & \xrightarrow{\quad} & S' \rightarrow S \\ A \rightarrow a & & S \rightarrow SA \\ & & A \rightarrow a \end{array} \left. \begin{array}{l} \text{new} \\ \text{start} \\ \text{symbol} = S' \end{array} \right\}$$

2. Eliminate null, unit & useless productions

3.  $A \rightarrow B_1 B_2 B_3 \dots \dots B_n \quad n > 2$

$$A \rightarrow B_1 C$$

$$C \rightarrow B_2 B_3 \dots \dots B_n$$

Repeat till 2 variables in RHS

$V \rightarrow VV$   
 $V \rightarrow T$   
 $S \rightarrow E$

4.  $A \rightarrow aB$

$$\begin{array}{c} \curvearrowleft \\ A \rightarrow CB \\ C \rightarrow a \end{array}$$