Eg: $A \to aB \mid a$
$B \to aB \mid bB \mid a \mid b$
} Right linear Grammar ⇓ Regular Grammar

Eg: $A \to Ba \mid a$
$B \to Ba \mid Bb \mid b \mid a$
} Left linear Grammar ⇓ Regular Grammar

Eg: $A \to Ba \mid a \longrightarrow$ LLG
$B \to aB \mid a \longrightarrow$ RLG
} Not a Regular Grammar (Type 3 Grammar)

bcz it is a combination LLG & RLG

Type 2:

Context free Grammar

Productions of the form:

$$A \longrightarrow \alpha$$

$A \in V$
$\alpha \in (V \cup T)^*$

Eg: $A \to aAb \mid ab \mid BC$
$B \to Aab$

"variable" — "Any combination of variables & terminals"

$Aa \to bCd$ ✗
$AB \to bd$ ✗

Ambiguity

$E \to E+E \mid E*E \mid id$

$V = \{E\}$
$T = \{+, *, id\}$
$S = \{E\}$

String: id + id * id

Leftmost Derivation

$E \rightarrow \underline{E}+E$

$\rightarrow id+\underline{E}$

$\rightarrow id+E*\underline{E}$

$\rightarrow id+\underline{E}*id$

$\rightarrow id+id*id$

Rightmost Derivation

$E \rightarrow E+\underline{E}$

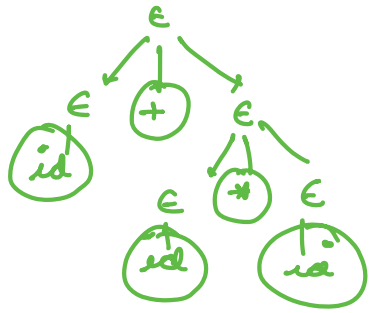
$\rightarrow E+\underline{E}*E$

$\rightarrow E+E*id$

$\rightarrow \underline{E}+id*id$

$\rightarrow id+id*id$

Parse Tree



$E \rightarrow E*E$

$\rightarrow E+E*E$

$\rightarrow id+E*E$

$\rightarrow id+id*E$

$\rightarrow id+id*id$

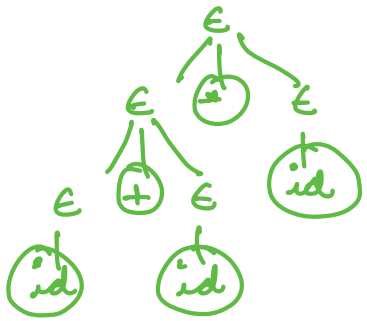
$E \rightarrow E*E$

$\rightarrow E*id$

$\rightarrow E+E*id$

$\rightarrow E+id*id$

$\rightarrow id+id*id$



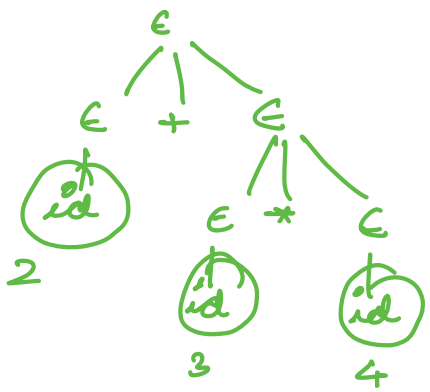
If you have more than 1 LMD or more than 1 RMD or more than 1 PT

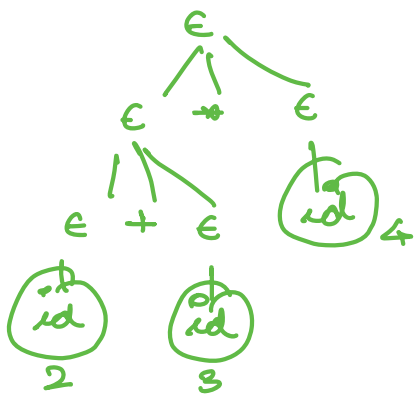
⇓

Grammar is ambiguous.

Eg: 2+3*4

int a = 2+3*4;





2+12=14

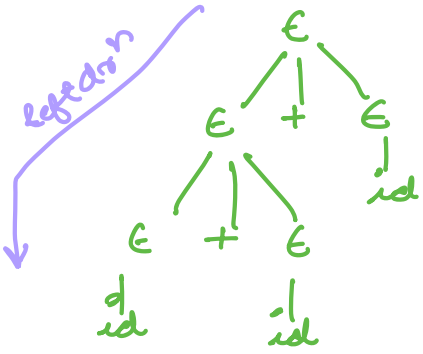
20

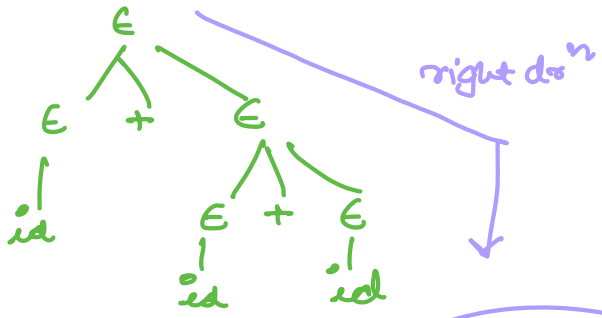
Confusion ?

Disambiguous:

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

Case 1: $id + id + id$

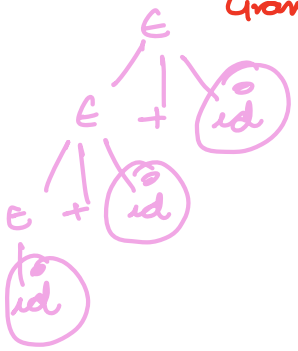




$E \rightarrow E + E \mid id$

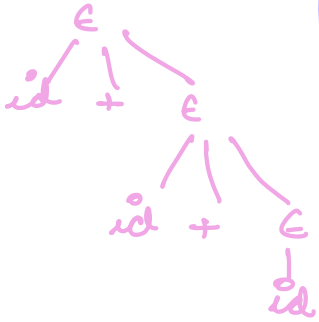
$E \rightarrow E + id \mid id$

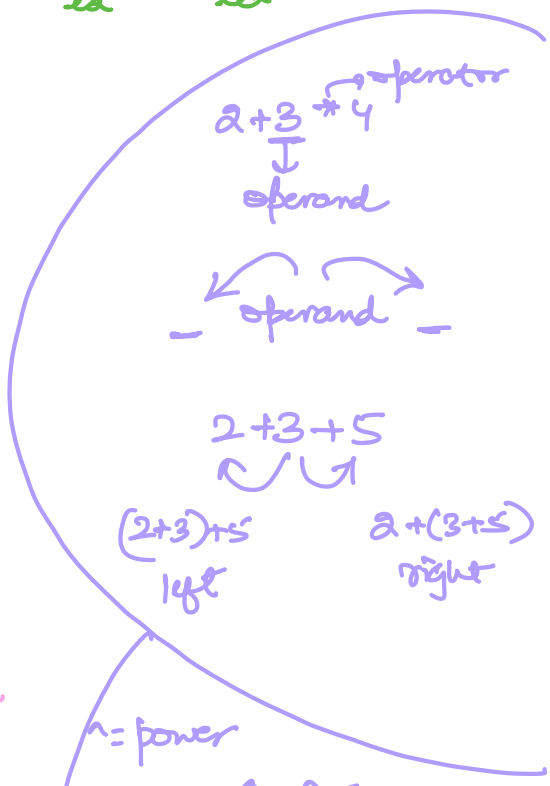
Left Recursive Grammar

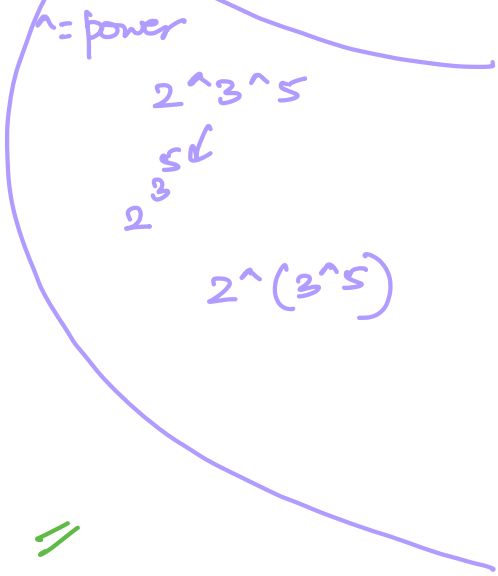


$E \rightarrow id + E \mid id$

Right Recursive Grammar

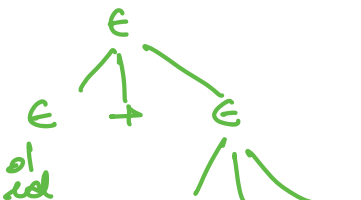


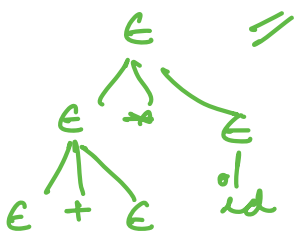




Case 2: Precedence Rules

$id + id * id$





$E \rightarrow E + T \mid$

$T \rightarrow R * R \mid id$

$R \rightarrow id$

$id + (id * id)$ ✓

$(id + id) * id$

↓

Shouldnot get generated

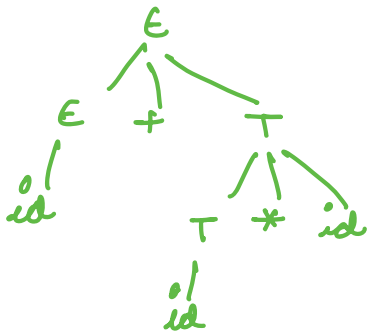
same level

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

⇓

$E \rightarrow E + T \mid T \mid id$

$T \rightarrow T * id \mid id$



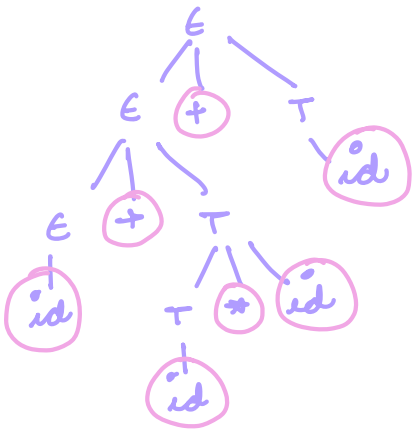
Ambiguous Situation → Associativity → Left Rec OR Right Rec

→ Precedence → higher precedence move far from Start Symbol

Eg: $id + id * id + id$

?

$(id + (id * id)) + id$



Simplification of CFG:

1) Remove ε productions:

- We willnot be able to remove all ε productions, if ε is present in language.
- If language doesnot contain ε then we can remove all ε productions.

Eg:
$$S \to aSb \mid aAb$$
$$A \to \varepsilon$$
(ε → epsilon)

1. find out all nullable variables.
   - ⊙ if a variable can directly give ε like $A \to \varepsilon$
   - ⊙ if a variable can generate ε after some no. of steps

$$A \to ( ) \to ( ) \to \varepsilon$$

nullable variable = {A}

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

$$S \to aSb \mid aAb \qquad A \to \varepsilon$$
$$\longrightarrow$$
$$S \to aSb \mid aAb \mid ab$$
~~$A \to \varepsilon$~~ eliminate it

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

$$S \to aSb \mid ab$$

Eg:
$$S \to AB$$
$$A \to aAA \mid \varepsilon$$
$$B \to bBB \mid \varepsilon$$

nullable variable = {A, B, S}

$S \to AB \to B \to \varepsilon$

Start Symbol is a nullable variable that means ε is present in language.

$S \to AB \mid BA \mid \varepsilon$
$A \to aAA \mid aA \mid a$
$B \to bBB \mid bB \mid b$

L = Even length strings
$\Sigma = \{a, b\}$
$L = \{\varepsilon, aa, ab, ba, bb, \ldots\}$

Eg:

$S \to AbaC$
$A \to BC$
$B \to b \mid \varepsilon$
$C \to D \mid \varepsilon$
$D \to d$

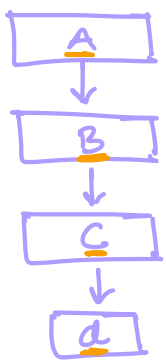
nullable variable = $\{A, B, C\}$

$S \to AbaC \mid baC \mid Aba \mid ba$
$A \to BC \mid B \mid C$
$B \to b$
$C \to D$
$D \to d$

In process of removing ε production, you might get unit production.

$A \to B$ ‖ unit
$B \to C$
$C \to d$

$A \to d$
$B \to d$
$C \to d$



Variable → Variable
length 1 length 1

length is not changing.

wastage of steps
$A \to d$

2

Elimination of unit productions:

Eg:

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

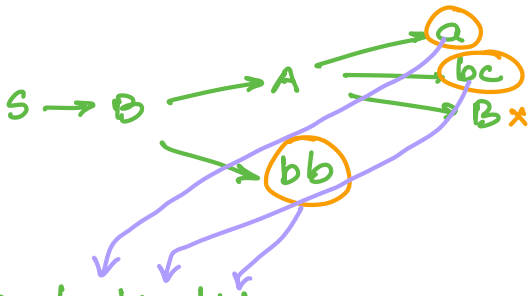
After removal of unit productions, language shouldnot be affected.

Write grammar without unit production.

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

Look at every unit production one by one

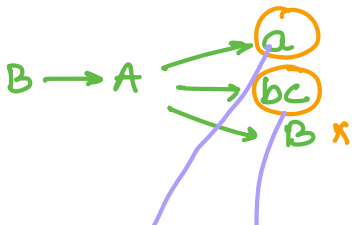
$S \rightarrow B$



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

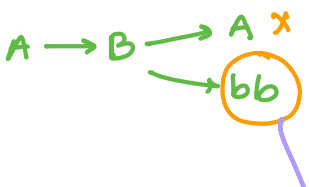
$B \rightarrow bb$
$A \rightarrow a \mid bc$

$B \rightarrow A$



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

$A \rightarrow B$



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

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

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

Eg:

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow C \mid b$

$C \rightarrow D$

$D \rightarrow \epsilon$

$\epsilon \rightarrow a$

Grammar without unit production

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

$\epsilon \rightarrow a$

$B \rightarrow C$

$B \rightarrow C \rightarrow D \rightarrow \epsilon \rightarrow a$

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b \mid a$

$\epsilon \rightarrow a$

$C \rightarrow D$

$C \rightarrow D \rightarrow \epsilon \rightarrow a$

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b \mid a$

$\epsilon \rightarrow a$

$C \rightarrow a$

$D \rightarrow \epsilon$

$D \rightarrow \epsilon \rightarrow a$

$S \to AB$
$A \to a$
$B \to b \mid a$
$E \to a$
$C \to a$
$D \to a$

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.

$A \longrightarrow \cdots \longrightarrow$ terminals

- Variable should be reachable from start state.

} useful

Eg:

$S \to AB \mid a$
$A \to BC \mid b$
$B \to aB \mid C$
$C \to aC \mid B$

$T = \{a, b\}$
$V = \{S, A, B, C\}$

- All terminals are useful symbols $\{a, b\}$

- RHS of production contains only terminals, then LHS of production will also be useful.

$S \to AB \mid a$
$A \to BC \mid b$
$B \to aB \mid C$
$C \to aC \mid B$

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

- 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.

$B \to aB \mid C$ not useful
$C \to aC \mid B$

B & C are useless symbols.

$B \rightarrow aB \rightarrow aaB \rightarrow aaaB \ldots$

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

~~$S \rightarrow AB \mid a$~~
~~$A \rightarrow BC \mid b$~~
~~$B \rightarrow aB \mid C$~~
~~$C \rightarrow aC \mid B$~~

$S \rightarrow a$
$A \rightarrow b$

- Symbol which is unreachable, remove it.

$S \rightarrow a$
~~$A \rightarrow b$~~

$\longrightarrow \quad S \rightarrow a$

Eg:

$S \rightarrow AB \mid AC$
$A \rightarrow aAb \mid bAa \mid a$
$B \rightarrow bbA \mid aaB \mid AB$
$C \rightarrow abCA \mid aDb$
$D \rightarrow bD \mid ac$

$T = \{a, b\}$

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

- Terminals are useful $\{a, b\}$
- Variables which are giving terminals $\{a, b, A\}$
- $B \rightarrow bbA \mid aaB \mid AB$ $\{a, b, A, B\}$

(bbA: useful; B: useful)

$S \rightarrow \underline{AB} \mid AC$

useful ⇐ useful

{a, b, A, B, S}

C & D are not useful.

○

$S \rightarrow AB \mid A$ ~~C~~

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

$B \rightarrow bbA \mid aaB \mid AB$

~~$C \rightarrow abCA \mid aDb$~~

~~$D \rightarrow bD \mid aC$~~

○

$S \rightarrow AB$

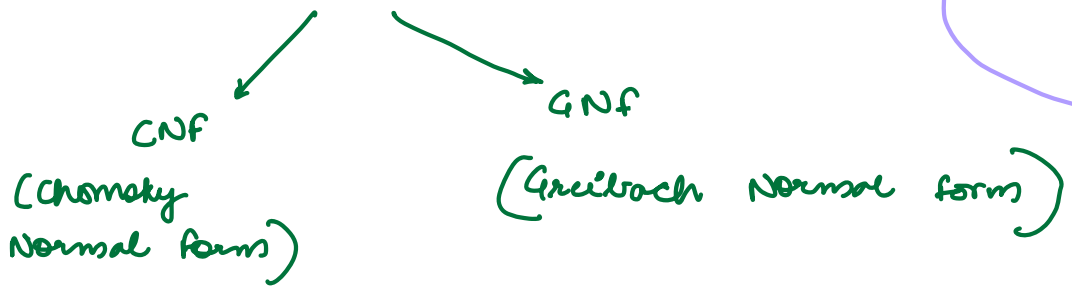
$A \rightarrow aAb \mid bAa \mid a$

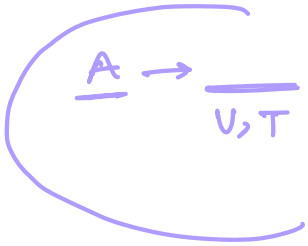
$B \rightarrow bbA \mid aaB \mid AB$

Simplify CFG :

1. ε productions
2. Unit productions
3. Useless Symbols

Normal forms of CFG

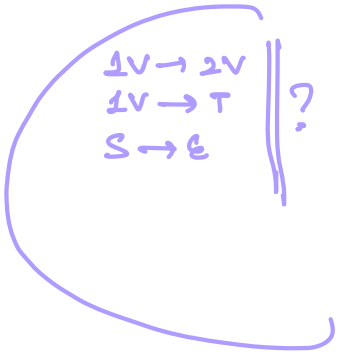




Chomsky Normal form (CNF):

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

- A non terminal generating a terminal ($A \to a$)
- A non terminal generating 2 non-terminals ($A \to BC$)
- Start symbol generating ε ($S \to \varepsilon$)



Eg:

$S \to a$
$S \to Az$
$A \to a$
$z \to b$

CFG is in CNF

$S \to a$
$S \to az$
$z \to b$

CFG is not in CNF.

Properties:

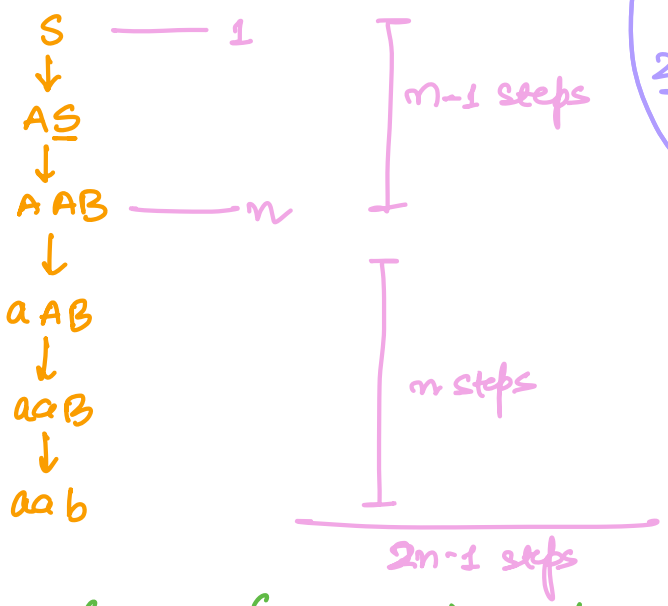
- For a given grammar, there can be more than 1 CNF
- CNF produces the same language as generated by CFG. (CFG → CNF)
- For generating a string of length n, you require $2n-1$ steps in CNF.

CNF: $S \to AB \mid AS$

$A \to a$

$B \to b$

"aab"



Grammar CNF String ? $2n-1$ steps

- Any CFG that donot have ε in its language has an equivalent CNF.

Conversion from CFG to CNF ?

1. Eliminate start symbol from RHS

$S \to SA$, $A \to a$ ⟶ $S' \to S$, $S \to SA$, $A \to a$ (new start symbol = S')

2. Eliminate null, unit & useless productions

3. $A \to B_1 B_2 B_3 \cdots B_n$ $\quad n > 2$

$A \to B_1 C$

$C \to B_2 B_3 \cdots B_n$

Repeat till 2 variables in RHS

$V \to VV$

$V \to T$

$S \to \varepsilon$

4. $A \to aB$

$A \to CB$

$C \to a$