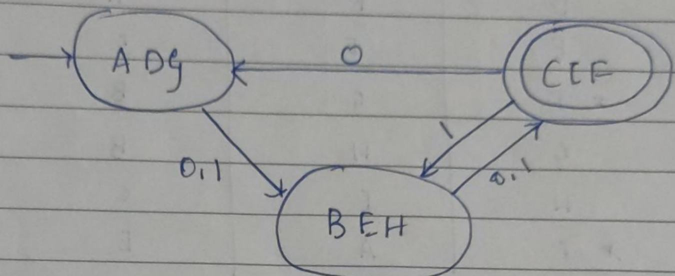
BE → EH → HB

CF → FI → IC

AD → DG → GA

AD
AG
BE
BH
CF
CI
DF
DG
EH
FI

BEH
CIF
ADG



Pumping Lemma for Regular Languages:

Let $M = \{Q, \Sigma, \delta, q_0, F\}$ be the deterministic finite automata & has 'n' no of states. Let L be the language accepted by above DFA. Assume that we have a string 'x' whose length is m, if $x \in L$ then there exists a count (No of strings) such that length of $x' >= n$ i.e the length of the string (w) is larger than the no of states (n) $m > n$.

Now, if the string x is decomposed into 3 sub strings u, v & w such that
$|uv| <= n$ then $x = uv^i w \in L$
& $|v| >= 1$ for all $i = 0, 1, 2, \ldots$

$u = a_i$ to a_j = prefix of string x

$v = a_{j+1}$ to a_k = loop string

$w = a_{k+1}$ to a_m = suffix of string

(e)

w^r = reverse of w __/__/__

(1) $L = \{ww^r \mid w \in (a+b)^*\}$

Assume that the language is regular & 'n' be the states of DFA.

Select the string 'x' such that $|x| >= n$ for some +ve 'n' & divide the string x into 3 sub-strings u, v & w, so that $x = uvw$ & length of $|uv| <= n$ & $|v| >= 1$

$w = a^n b^n$

$w^r = b^n a^n$

$x = ww^r = a^n b^n b^n a^n$ $|x| = 4n$

$= \underbrace{a^{n-1}}_{u} \underbrace{a}_{v} \underbrace{b^n b^n a^n}_{w}$

$|u| = n-1$ $|v| = 1$ $|uv| = n-1+1 = n$

$\therefore x = a^{n-1} a^i b^n b^n a^n \in L$

for : $i = 0, 1, 2 \ldots$

$x = a^{n-1} b^n b^n a^n$

$|u| = n-1$ $|v| = 0$ $|uv| = n-1$

$|v| \not>= 1$

$\therefore x \notin L$ for $i = 0$

$\therefore$ The language is not regular.

(2) $L = \{w \mid n_a(w) < n_b(w), w \in (a+b)^*\}$

(1) Assume that language is regular & 'n' be the states of DFA

(2) Select 'x' such that $|x| >= n$ & divide x into 3 parts u, v, & w such that $|uv| <= n$ & $|v| >= 1$

$a^{n-1} b^n$

$w = a^{n-1} b^n$ $\quad |x| = 2n-1 \geq n$

$x = a^{n-1} \; b^1 \; b^{n-1}$
$\quad\;\; \downarrow \quad \downarrow \quad \downarrow$
$\quad\;\; u \quad v \quad w$

$|uv| = n-1+1 = n$ $\quad |v| = 1$

$x = a^{n-1} (b)^i \, b^{n-1} \in L$ for $i = 0, 1, 2 \ldots$

for $i = 0$
$x = a^{n-1} b^{n-1}$ $\quad \notin L$
$|v| = 0$

$\therefore$ Assumption is wrong hence language is not regular.

(3) $L = \{ w \mid n_a(w) > n_b(w), \; w \in (a+b)^* \}$

$w = a^n b^{n-1}$ $\quad |w| = 2n-1 \geq n$

$x = a^{n-1} \; a \; b^{n-1}$
$\quad\;\; \downarrow \quad \downarrow \quad \downarrow$
$\quad\;\; u \quad v \quad w$

$|uv| = n-1+1 = n$ $\quad |v| = 1$

$x = a^{n-1} a^i b^{n-1} \in L$ for $i = 0, 1, 2 \ldots$

for $i = 0$
$x = a^{n-1} b^{n-1}$ $\quad$ there are equal no of a's & b's
$|v| = 0$

$\therefore x \notin L$

$\therefore$ Our Assumption is wrong hence the language is not regular.

_/_/_

(4) $L = \{ (ab)^n a^k \mid n > k, \ k >= 0 \}$

→ Assume that language is regular & 'n' be states of DFA
→ select x such that $|x| >= n$ & divide into 3 parts u, v & w $|uv| <= n$ & $|v| >= 1$

$w = (ab)^n (a)^{n-1}$ $\qquad n > k$

$x = \underbrace{(ab)^{n-1}}_{u} \ \underbrace{ab}_{v} \ \underbrace{(a)^{n-1}}_{w}$

$|uv| = n-1+1 = n \qquad |v| = 1$

$x = (ab)^{n-1} (ab)^i (a)^{n-1} \in L$ for $i = 0, 1, 2 ---$

for $i = 0$

$x = (ab)^{n-1} (a)^{n-1}$ here $n = k$

∴ Contradicts the ~~as~~ language. Hence our ~~langu~~ assumption is wrong & our language is not regular.

(5) $L = \{ a^{n!} \mid n >= 0 \}$

→ Assume that language is regular & n be states of DFA
→ select x such that $|x| >= n$ & divide into 3 parts u, v, w such that $|uv| <= n$ & $|v| >= 1$

$w = \underbrace{a^j}_{u} \ \underbrace{a^k}_{v} \ \underbrace{a^{n!-j-k}}_{w}$ $\qquad |u| = j \qquad |v| = k$

$|uv| = j+k < = n$

$w = a^j (a^k)^i a^{n!-j-k} \in L$ for $i = 0, 1, 2 ..$

for $i = 0$

$w = a^j \ a^{n!-j-k} \approx a^{n!-k}$

for any $k > 0$ $w \notin L$ as it contradicts the language

$\therefore$ Our assumption is wrong & our language is not regular

(6) $L = \{a^n \mid n \text{ is prime}\}$

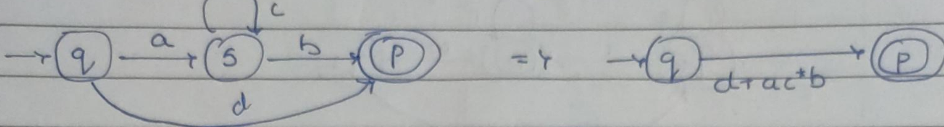
$w = \underbrace{a^j}_{u} \underbrace{a^k}_{v} \underbrace{a^{n-j-k}}_{w}$

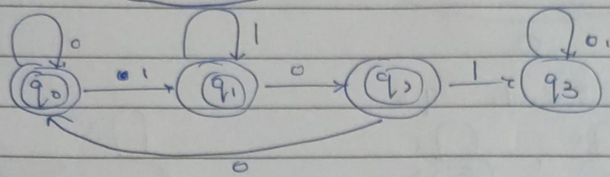
$|u| = j$

$|v| = k$

$|uv| = j + k <= n$

STATE ELIMINATION METHOD:



① 

$\phi \cdot R = \phi$

$\epsilon + R = R$

$(\epsilon + R)^* = R^*$

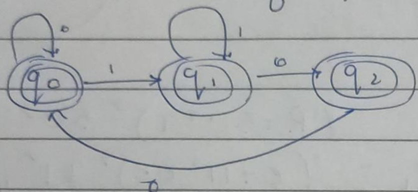
$(\epsilon + R) \cdot R^* = R^*$

$\epsilon \cdot R = R$

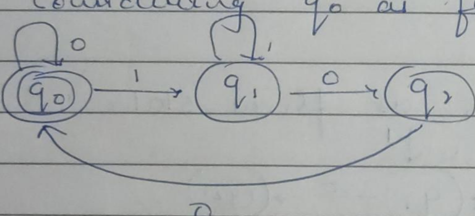
$\epsilon^* = \epsilon$

$\phi^* = \epsilon$

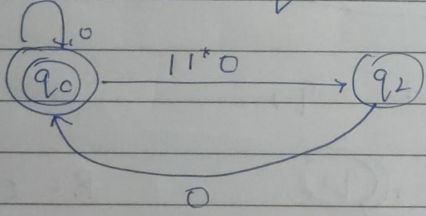
i) elimination of q_3



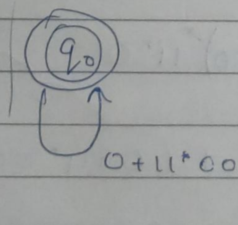
ii) Considering q_0 as final state:



→ eliminate q_1

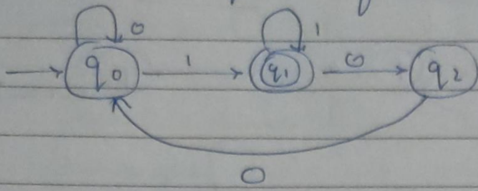


→ eliminate q_2

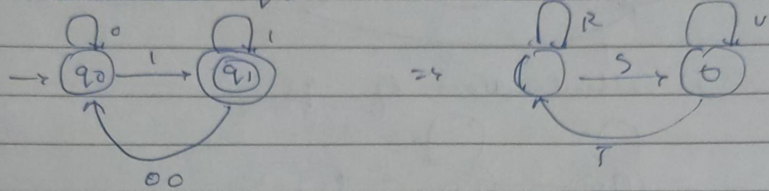


$R_0 = (0 + 11^*00)^*$

iii) Consider q_1 as final state



→ eliminate q_2



$R = 0$

$S = 1$

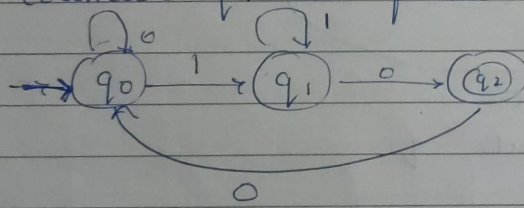
$U = 1$

$T = 00$

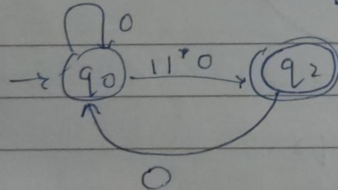
$(R + SU^*T)^* SU^*$

$R_1 = (0 + 11^*00)^* 11^*$ //

iv) Consider q_2 as final state



→ eliminate q_1



$R = 0 \quad S = 11^*0$

$T = 0 \quad U = \phi$

$R_2 = (0 + 11^*00)^* 11^*0$

$R = (0 + 11^*00)^* + (0 + 11^*00)^* 11^* + (0 + 11^*00)^* 11^*0$

19/12/22

UNIT-3 - CONTEXT FREE GRAMMARS & LANGUAGES:

Hierarchy

- Type 0 — Structured
- Type 1 — Context sensitive
- Type 2 — Content free
- Type 3 — Regular

Context free grammar: CFG is grammar which is used to generate all the possible patterns of strings in a given formal- language.

A CFG G can be defined as $G = (V, T, S, P)$
where,

V = set of variables / Non-terminals
T = Set of Terminals
S = Start symbol
P = set of Productions / rules.
(head $\in P \to \gamma^*$ (Body)

20/12/22

NOTATIONS:

Terminals:

- Lower case letters near the beginning of the alphabet ex. a, b ---
- digits & other charachters ex: +, (,) → paranthesis. punctuation , 0-9
- All keywords written in bold
- Lower-case letters near end of alphaet ex: ... x, y, z are strings of terminals.

VARIABLE / NON-TERMINALS:

- Upper case letters near the beginning of alphabet
  ex: A, B, C ---
-

→ Upper-case letters near end of alphabet ex: ... X Y Z are strings of NT or T

→ Greek letters ex: α, β are strings of NT or T

START SYMBOL:

- If the letter 'S' appears in the grammar, it is taken as start symbol. • if not first production head will be taken as start symbol.

PRODUCTIONS:

$$\underset{\text{head}}{P} \underset{\text{derives}}{\longrightarrow} \underset{\text{body}}{\varphi}$$

DERIVATIONS:

lmd : left most derivation
rmd : right most derivation.

DOMS Page No.
Date / /

① (a+b) * c - d / e

$E \overset{lm}{\Rightarrow} E - E$
$\Rightarrow E * E - E$
$\Rightarrow (E) * E - E$
$\Rightarrow (E + E) * E - E$
$\Rightarrow (id + E) * E - E$
$\Rightarrow (id + id) * E - E$
$\Rightarrow (id + id) * id - E$
$\Rightarrow (id + id) * id - E / E$
$\Rightarrow (id + id) * id - id / E$
$\Rightarrow (id + id) * id - id / id$ //

* Always process "only" one element per step.

22/12/22

② Conside a grammer
$S \rightarrow AS \mid \epsilon$
$A \rightarrow aa \mid bb \mid ba \mid ab$
write lm & rm derivation for sentence w = aabbaa also write a Parse tree.

Sol! w = aabbaa
$G = \{V, T, S, P\}$

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

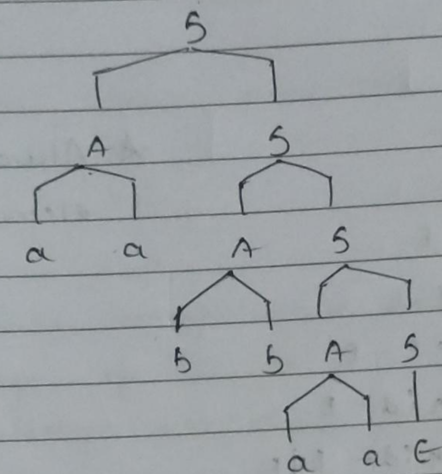
deriv^n steps prod^n app^n

$S \overset{lm}{\Rightarrow} AS$	$(S \rightarrow AS)$
$\Rightarrow aaS$	$(A \rightarrow aa)$
$\Rightarrow aaAS$	$(S \rightarrow AS)$
$\Rightarrow aabbS$	$(A \rightarrow bb)$
$\Rightarrow aabbAS$	$(S \rightarrow AS)$
$\Rightarrow aabbaaS$	$(A \rightarrow aa)$
$\Rightarrow aabbaa\epsilon$	$(S \rightarrow \epsilon)$
$\Rightarrow aabbaa$ //	

$S \overset{rm}{\Rightarrow} AS$	$(S \rightarrow AS)$
$\Rightarrow AAS$	$(S \rightarrow AS)$
$\Rightarrow AAAS$	$(S \rightarrow AS)$
$\Rightarrow AAA\epsilon$	$(S \rightarrow \epsilon)$
$\Rightarrow AAaa$	$(A \rightarrow aa)$
$\Rightarrow Abbaa$	$(A \rightarrow bb)$
$\Rightarrow aabbaa$ //	$(A \rightarrow aa)$

Page No.
Date / /

PARSE TREE



H.W

(3) $S \to SS+ \mid SS* \mid a$

$w = aa+a*$

~~$S \overset{lm}{\Rightarrow} SS*$ $(S \to SS*)$~~

~~$\Rightarrow aa+S*$ $(S \to aa+)$~~

~~$\Rightarrow aa+a*$ $(S \to a)$~~

$S \overset{lm}{\Rightarrow} \underline{S}S*$ $(S \to SS*)$

$\Rightarrow \underline{S}S+S*$ $(S \to SS+)$

$\Rightarrow a\underline{S}+S*$ $(S \to a)$

$\Rightarrow aa+\underline{S}*$ $(S \to a)$

$\Rightarrow aa+a*$ $(S \to a)$

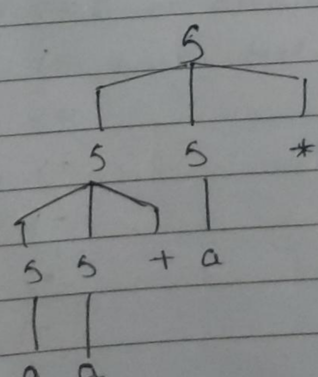
$S \overset{rm}{\Rightarrow} S\underline{S}*$ $(S \to SS*)$

$\Rightarrow \underline{S}a*$ $(S \to a)$

$\Rightarrow S\underline{S}+a*$ $(S \to SS+)$

$\Rightarrow \underline{S}a+a*$ $(S \to a)$

$\Rightarrow aa+a*$ $(S \to a)$



(4) $S \to aB \mid bA$

$A \to a \mid aS \mid bAA$

$B \to b \mid bS \mid aBB$

$w = aaabbabbba$

$\underline{S} \overset{lm}{\Longrightarrow} a\underline{B}$ $\quad (S \to aB)$

$\Rightarrow aa\underline{B}B$ $\quad (B \to aBB)$

$\Rightarrow aaa\underline{B}BB$ $\quad (B \to aBB)$

$\Rightarrow aaab\underline{B}B$ $\quad (B \to b)$

$\Rightarrow aaabb\underline{B}$ $\quad (B \to b)$

$\Rightarrow aaabba\underline{B}B$ $\quad (B \to aBB)$

$\Rightarrow aaabbab\underline{B}$ $\quad (B \to b)$

$\Rightarrow aaabbabb\underline{S}$ $\quad (B \to bS)$

$\Rightarrow aaabbabbb\underline{A}$ $\quad (S \to bA)$

$\Rightarrow aaabbabbba$ // $\quad (A \to a)$

$S \overset{rm}{\Longrightarrow} a\underline{B}$ $\quad (S \to aB)$

$\Rightarrow aaB\underline{B}$ $\quad (B \to aBB)$

$\Rightarrow aaBb\underline{S}$ $\quad (B \to bS)$

$\Rightarrow aaBbb\underline{A}$ $\quad (S \to bA)$

$\Rightarrow aa\underline{B}bba$ $\quad (A \to a)$

$\Rightarrow aaa\underline{B}Bbba$ $\quad (B \to aBB)$

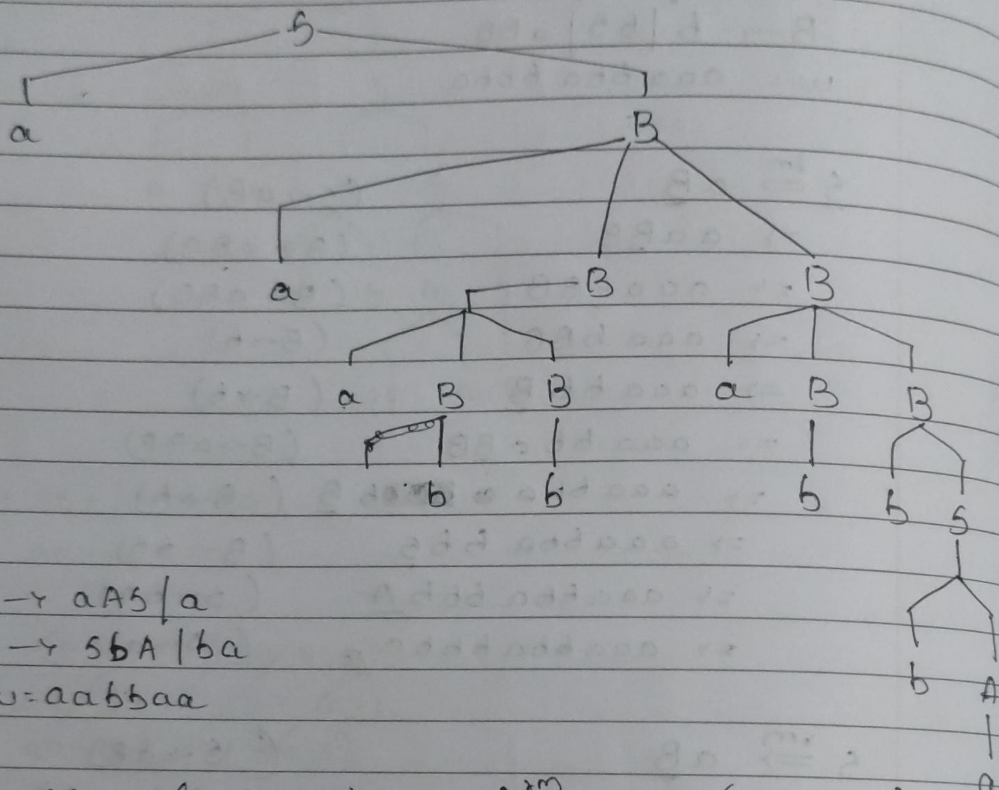
$\Rightarrow aaa\underline{B}bbba$ $\quad (B \to b)$

$\Rightarrow aaab\underline{S}bbba$ $\quad (B \to bS)$

$\Rightarrow aaabb\underline{A}bbba$ $\quad (S \to bA)$

$\Rightarrow aaabbabbba$ // $\quad (A \to a)$

PARSE TREE:



(5) $S \rightarrow aAS \mid a$

$A \rightarrow SbA \mid ba$

$w = aabbaa$

$S \xRightarrow{lm} a\underline{A}S$ $(S \rightarrow aAS)$

$\Rightarrow a\underline{S}bAS$ $(A \rightarrow SbA)$

$\Rightarrow aab\underline{A}S$ $(S \rightarrow a)$

$\Rightarrow aabba\underline{S}$ $(A \rightarrow ba)$

$\Rightarrow aabbaa$ $(S \rightarrow a)$

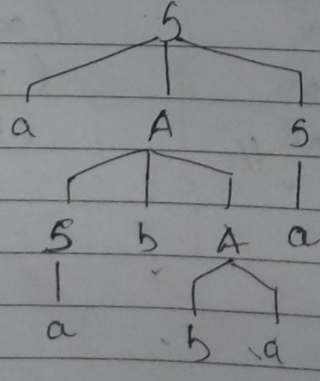
$S \xRightarrow{rm} aA\underline{S}$ $(S \rightarrow aAS)$

$\Rightarrow a\underline{A}a$ $(S \rightarrow a)$

$\Rightarrow aSb\underline{A}a$ $(A \rightarrow SbA)$

$\Rightarrow a\underline{S}bbaa$ $(A \rightarrow ba)$

$\Rightarrow aabbaa$ $(S \rightarrow a)$



DOMS	Page No.
Date / /	

(6) $E \rightarrow ET+ \mid T$
$T \rightarrow TF* \mid F$
$F \rightarrow FP\uparrow \mid P$
$P \rightarrow E \mid id$

i) id id id * + id +
ii) id id id ↑ id * id * +

(i) $E \xRightarrow{lm} ET+$ $(E \rightarrow ET+)$
$\Rightarrow ET+T+$ $(E \rightarrow ET+)$
$\Rightarrow TT+T+$ $(E \rightarrow T)$
$\Rightarrow FT+T+$ $(T \rightarrow F)$
$\Rightarrow PT+T+$ $(F \rightarrow P)$
$\Rightarrow id\,T+T+$ $(P \rightarrow id)$
$\Rightarrow id\,TF*+T+$ $(T \rightarrow TF*)$
$\Rightarrow id\,FF*+T+$ $(T \rightarrow F)$
$\Rightarrow id\,PF*+T+$ $(F \rightarrow P)$
$\Rightarrow id\,id\,F*+T+$ $(P \rightarrow id)$
$\Rightarrow id\,id\,P*+T+$ $(F \rightarrow P)$
$\Rightarrow id\,id\,id*+T+$ $(P \rightarrow id)$
$\Rightarrow id\,id\,id*+F+$ $(T \rightarrow P)$
$\Rightarrow id\,id\,id*+P+$ $(F \rightarrow P)$
$\Rightarrow id\,id\,id*+id+$ // $(P \rightarrow id)$

$E \xRightarrow{rm} ET+$ $(E \rightarrow ET+)$
$\Rightarrow EF+$ $(T \rightarrow P)$
$\Rightarrow EP+$ $(F \rightarrow P)$
$\Rightarrow E\,id+$ $(P \rightarrow id)$
$\Rightarrow ET+id+$ $(E \rightarrow ET+)$
$\Rightarrow ETF*+id+$ $(T \rightarrow TF*)$
$\Rightarrow ETP*+id+$ $(F \rightarrow P)$

$\Rightarrow E\underline{T}\ id * + id +$ $(P \rightarrow id)$

$\Rightarrow E\underline{F}\ id * + id +$

$\Rightarrow E\underline{P}\ id * + id +$

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

$\Rightarrow \underline{T}\ id\ id * + id +$

$\Rightarrow \underline{F}\ id\ id * + id +$

$\Rightarrow \underline{P}\ id\ id * + id +$

$\Rightarrow id\ id\ id * + id +$

ii) $E \overset{lm}{\Rightarrow} \underline{E}T +$ $(E \rightarrow ET+)$

$\Rightarrow \underline{T}\ T +$ $(E \rightarrow T)$

$\Rightarrow \underline{F}\ T +$ $(T \rightarrow F)$

$\Rightarrow \underline{P}\ T +$ $(F \rightarrow P)$

$\Rightarrow id\ \underline{T} +$ $(P \rightarrow id)$

$\Rightarrow id\ \underline{T}F * +$ $(T \rightarrow TF*)$

$\Rightarrow id\ \underline{T}F * F * +$ $(T \rightarrow TF*)$

$\Rightarrow id\ \underline{F}F * F * +$ $(T \rightarrow F)$

$\Rightarrow id\ \underline{F}P \uparrow F * F * +$ $(F \rightarrow FP\uparrow)$

$\Rightarrow id\ \underline{P}P \uparrow F * F * +$ $(F \rightarrow P)$

$\Rightarrow id\ id\ \underline{P} \uparrow F * F * +$ $(P \rightarrow id)$

$\Rightarrow id\ id\ id \uparrow \underline{F} * F * +$ $(P \rightarrow id)$

$\Rightarrow id\ id\ id \uparrow \underline{P} * F * +$ $(F \rightarrow P)$

$\Rightarrow id\ id\ id \uparrow id * \underline{F} * +$ $(P \rightarrow id)$

$\Rightarrow id\ id\ id \uparrow id * \underline{P} * +$ $(F \rightarrow P)$

$\Rightarrow id\ id\ id \uparrow id * id * +$ $(P \rightarrow id)$

$E \xrightarrow{rm} ET+ \quad (E \to ET+)$

$\Rightarrow ETF*+ \quad (T \to TF*)$

$\Rightarrow ETP*+ \quad (F \to P)$

$\Rightarrow ET\,id*+ \quad (P \to id)$

$\Rightarrow ETF*id*+ \quad (T \to TF*)$

$\Rightarrow ETP*id*+ \quad (F \to P)$

$\Rightarrow ET\,id*id*+ \quad (P \to id)$

$\Rightarrow EF\,id*id*+ \quad (T \to F)$

$\Rightarrow EFP\uparrow id*id*+ \quad (F \to FP\uparrow)$

$\Rightarrow EF\,id\uparrow id*id*+ \quad (P \to id)$

$\Rightarrow EP\,id\uparrow id*id*+ \quad (F \to P)$

$\Rightarrow E\,id\,id\uparrow id*id*+ \quad (P \to id)$

$\Rightarrow T\,id\,id\uparrow id*id*+ \quad (E \to T)$

$\Rightarrow F\,id\,id\uparrow id*id*+ \quad (T \to F)$

$\Rightarrow P\,id\,id\uparrow id*id*+ \quad (T \to P)$

$\Rightarrow id\,id\,id\uparrow id*id*+ \quad (P \to id)$

(7) $S \to AaAb \mid BbBa$
$A \to aAa \mid bAB \mid d$
$B \to aB \mid bB \mid a$

(i) $w = aabbaa$
(ii) $w = badbabaadb$

(i) $S \xrightarrow{lm} BbBa \quad (S \to BbBa)$

$\Rightarrow aBbBa \quad (B \to aB)$

$\Rightarrow aabBa \quad (B \to a)$

$\Rightarrow aabbBa \quad (B \to bB)$

$\Rightarrow aabbaa \quad (B \to a)$

$S \overset{rm}{\Rightarrow} BbBa \quad (S \rightarrow BbBa)$

$\Rightarrow BbbBa \quad (B \rightarrow bB)$

$\Rightarrow Bbbaa \quad (B \rightarrow a)$

$\Rightarrow aBbbaa \quad (B \rightarrow aB)$

$\Rightarrow aabbaa_{//} \quad (B \rightarrow a)$

(ii)

23/12/22

Ambigous Grammar:

→ When there are more than 1 lmd / rmd then that grammar is called ambigous grammer

→ If the grammar is ambiguous iff there exists more than 1 lmd or rmd for atleast 1 sentence correspondingly Parse trees are diff

Ex: $E \rightarrow E + E$

$E \rightarrow E * E$ $\quad\quad w = a + b * c$

$E \rightarrow (E)$

$E \rightarrow id$

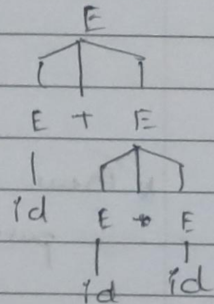
$E \overset{lm}{\Rightarrow} \underline{E} + E \quad (E \rightarrow E + E)$

$\Rightarrow id + \underline{E} \quad (E \rightarrow id)$

$\Rightarrow id + \underline{E} * E \quad (E \rightarrow E * E)$

$\Rightarrow id + id * \underline{E} \quad (E \rightarrow id)$

$\Rightarrow id + id * id \quad (E \rightarrow id)$



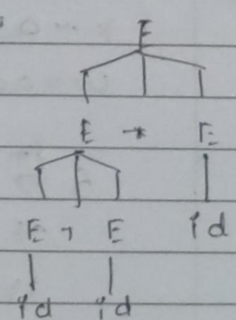
$E \overset{lm}{\Rightarrow} \underline{E} * E \quad (E \rightarrow E * E)$

$\Rightarrow \underline{E} + E * E \quad (E \rightarrow E + E)$

$\Rightarrow id + E * E \quad (E \rightarrow id)$

$\Rightarrow id + id * E \quad (E \rightarrow id)$

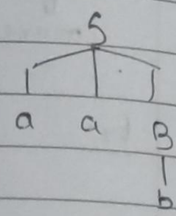
$\Rightarrow id + id * id \quad (E \rightarrow id)$



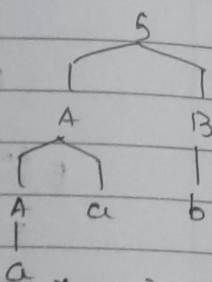
Ex: Consider the grammar

$S \to AB \mid aaB$
$A \to a \mid Aa$
$B \to b$
$w = aab$

$S \overset{lm}{\Rightarrow} aa\underline{B}$ $(S \to aaB)$
$\Rightarrow aab$ $(B \to b)$



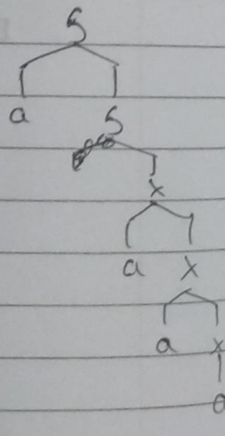
$S \overset{lm}{\Rightarrow} \underline{A}B$ $(S \to AB)$
$\Rightarrow \underline{A}aB$ $(A \to Aa)$
$\Rightarrow aa\underline{B}$ $(A \to a)$
$\Rightarrow aab$ $(B \to b)$



There exists 2 lmds, corresponding to 2 diff parse trees ∴ grammar is ambiguous.

(1) $S \to aS \mid X$
$X \to aX \mid a$
$w = aaaa$

$S \overset{lm}{\Rightarrow} a\underline{S}$ $(S \to aS)$
$\Rightarrow aa\underline{X}$ $(S \to aX)$
$\Rightarrow aaa\underline{X}$ $(X \to aX)$
$\Rightarrow aaaX$ $(X \to aX)$
$\Rightarrow aaaa$ $(X \to a)$



$S \overset{lm}{\Rightarrow} \underline{X}$ $(S \to X)$
$S \Rightarrow a\underline{X}$ $(X \to aX)$
$S \Rightarrow aa\underline{X}$ $(X \to aX)$

DOMS	Page No.
Date / /	

$\Rightarrow aaa\underline{X}$ $(\cancel{X} \rightarrow aX)$

$\Rightarrow aaa\bar{a}$ $(X \rightarrow a)$

There exists 2 lmd, therefore grammar is ambiguous.

(2) $S \rightarrow ictS \mid ictSeS \mid a$ $\Rightarrow$ if | then | else

$C \rightarrow b$

$w = ibtibtaea$

(3) $S \rightarrow aSbS \mid bSaS \mid \epsilon$

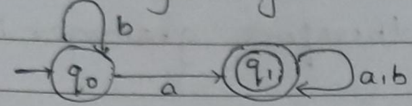
$w = aababb$

$RL \xrightarrow[\text{of}]{\text{sub set}} CFL$

30/12/22

- Writing a grammar for DFA
- Writing a grammar from RE } RG → RL
- Writing a grammar from other than DFA & RE

Ex1)



$G = \{V, T, S, P\}$

$V = \{S_1, S\}$

$T = \{a, b\}$

$S = S$

P = $S \to aS_1$, $S \to bS$ } $S \to aS_1 \mid bS$

$S_1 \to aS_1 \mid bS_1 \mid \epsilon$

w = abab

$S \xRightarrow{lm} aS_1$

$\Rightarrow abS_1$

$\Rightarrow abaS_1$

$\Rightarrow ababS_1$

$\Rightarrow abab\epsilon$

$\Rightarrow abab$ //

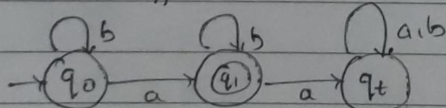
w = bbb

$S \xRightarrow{lm} bS$

$\Rightarrow bbS$

$\Rightarrow bbbS$

Ex2:



$G = \{V, T, S, P\}$

$V = \{S, S_1, S_2\}$

$T = \{a, b\}$

$S = S$

P = $S \to aS_1 \mid bS$

$S_1 \to aS_2 \mid bS_1 \mid \epsilon$

$S_2 \to aS_2 \mid bS_2$

DOMS Page No.
Date / /

$w = aabb$

$S \overset{lm}{\Rightarrow} aS_1$
$\Rightarrow aaS_2$
$\Rightarrow aabS_2$
$\Rightarrow aabbS_2$ //

WRITING GRAMMAR FROM RE:

→ break down R.E into smaller R.E.S until you get production

Ex: a^* → $S \to \epsilon | a | aS$ or $\epsilon | aS$ //

$(a+b)^*$ → $S \to \epsilon | a | b | aS | bS$ or $\epsilon | aS | bS$ //

$\underbrace{(a+b)^*}_{S_1} \underbrace{abb}_{S_2}$

$S \to S_1 S_2$
$S_1 \to \epsilon | aS_1 | bS_1$
$S_2 \to abb$ //

$\underset{S_1}{b^*} \underset{S_2}{a} \underset{S_1}{b^*}$

$S \to S_1 S_2 S_1$
$S_1 \to \epsilon | bS_1$
$S_2 \to a$

$\underset{S_1}{(a+b)^*} \underset{S_2}{a} \underset{S_1}{(a+b)^*}$

$S \to S_1 S_2 S_1$
$S_1 \to \epsilon | aS_1 | bS_1$
$S_2 \to a$

Page No.
Date / /

03/01/23

SIMPLIFICATION OF CONTEXT FREE GRAMMARS:-

* Removal of Redundant info.
* Normal form - CNF / GNF

Removal of Redundant Info:

⇓

- ϵ - productions
- Unit productions
- Useless productions

Order of Simplification
(1)
(2)
(3)

Nullable variable:-

→ A variable is said to nullable if $\overbrace{A \rightarrow \epsilon}^{\epsilon\text{-production}}$ and $A \in V$
for which $A \overset{*}{\Rightarrow} \epsilon$ is possible.

REMOVAL OF ϵ - PRODUCTIONS:

Ex1)
$S \rightarrow ABaC$
$A \rightarrow BC$
$B \rightarrow b \mid \epsilon$
$C \rightarrow D \mid \epsilon$
$D \rightarrow d$

(1) Compute nullable variable V_N
$V_N = \{B, C, A\}$

(2) Write all prod^{n}s except ϵ-prod^{n}s
$S \rightarrow ABaC$ $\quad A \rightarrow BC$
$B \rightarrow b$
$C \rightarrow D$
$D \rightarrow d$

DOMS Page No.
Date / /

③ Apply substitution (ε) rule & rewrite all the prod^n s

$S \to ABaC \mid BaC \mid AaC \mid ABa \mid aC \mid Aa \mid Ba \mid a$
$A \to BC \mid B \mid C$
$B \to b$
$C \to D$
$D \to d$

Ex 2) $S \to aAa \mid bBb \mid \epsilon$
$A \to C \mid a$
$C \to CDE \mid \epsilon$
$D \to A \mid B \mid ab$

① $V_N = \{S, C, A, D\}$

② $S \to aAa \mid bBb$
$A \to C \mid a$
$C \to CDE$
$D \to A \mid B \mid ab$

③ $S \to aAa \mid bBb \mid aa$
$A \to C \mid a$
$C \to CDE \mid DE \mid CE \mid E$
$D \to A \mid B \mid ab$

Removal Of Unit-Productions:

→ In a ~~CGA~~ CFG, if $A \to B$ where $B, A \in V$ is called as Unit production.

Ex 1)

$S \to Aa \mid B$

$B \to A \mid bb$

$A \to a \mid bc \mid B$

Step ①:

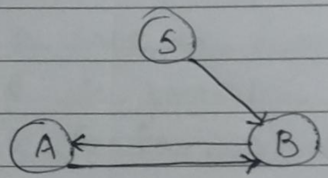
① Identify unit prodⁿs

$S \to B$

$B \to A$

$A \to B$

(ii) Write dependency graph



$S \stackrel{*}{\Rightarrow} B$

$S \stackrel{*}{\Rightarrow} A$

$B \stackrel{*}{\Rightarrow} A$

$A \stackrel{*}{\Rightarrow} B$

② Rewrite prodⁿ excluding unit prodⁿ

$S \to Aa \mid \cancel{bb}$

$B \to bb$

$A \to a \mid bc$

Add all prodⁿs possible

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

$B \to bb \mid a \mid bc$

$A \to a \mid bc \mid bb$

05|1|23

REMOVAL OF USELESS PRODUCTIONS:-

→ A variable which takes part in the derivatⁿ steps atleast once and derives terminal string is a useful variable
→ If a variable is not reachable from start symbol it is called Useless production / variable.
→ If a variable takes part in derivatⁿ step/but doesn't derive terminal string is a useless variable.

Formal defⁿ:

Let $G = \{V, T, S, P\}$ be a CFG a variable $A \in V$ is said to be useful iff, there is atleast one w $w \in L(G)$ such that $S \overset{*}{\Rightarrow} xAy \overset{*}{\Rightarrow} w$

Ex 1) $S \to aA \mid a \mid Bb \mid cC$
$A \to aB$
$B \to a \mid aA$
$C \to cCD$
$D \to ddd$

Step ①: Find useful variables that derive only terminal string.

	OV	NV	Prodⁿ
1	ϕ	S, B, D	$S \to a$ $B \to a$ $D \to ddd$
– –	–	–	– –

DOMS Page No. Date / /

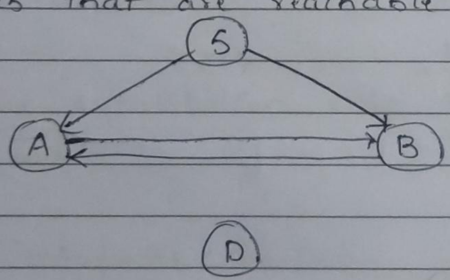
	OV	NV	Prodⁿs
2	S, B, D	S, B, D, A	S → Bb A → aB
3	S, B, D, A	S, B, D, A	S → aA B → aA
4	S, B, D, A	S, B, D, A	

'Consider prodⁿs where body contains combinaⁿ of terminals & useful variables

x — x - x — x — x END — x — x — x — x —

S → a | aA | Bb
A → aB
B → a | aA
D → ddd

Step ②: Find useful variables that can take part in derivⁿ steps, in order to do this we need to find variables that are reachable from the start symbol



⇒ Dependency Graph

→ D is not reachable from S and hence is considered to be useless variable.

S → a | aA | Bb
A → aB
B → a | aA.//

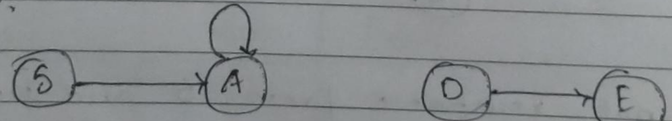
(2) $S \to aA \mid bB$
$A \to aA \mid a$
$B \to bB$
$D \to ab \mid Ea$
$E \to ac \mid Ad$

Step (1):

	OV	NV	Prod^n s
1	ϕ	A, E, D	$A \to a$ $D \to ab$ $E \to d$
2	A, E, D	A, E, S, D	$S \to aA$ $A \to aA$ $D \to Ea$
3	A, E, S, D	A, E, S, D	

$S \to aA$
$A \to a \mid aA$
$D \to ab \mid Ea$
$E \to d$

Step (2):



→ D & E are not reachable from S i.e. Useless
$S \to aA$
$A \to a \mid aA$ //

6/1/23

OVERALL SIMPLIFICATION OF GRAMMARS:

1. $S \to aAa \mid bBb \mid \epsilon$
   $A \to C \mid a$
   $B \to C \mid b$
   $C \to CDE \mid \epsilon$
   $D \to A \mid B \mid ab$

Step ①: Removal of ϵ-producⁿs

(i) Compute nullable variables
$V_N = \{S, A, B, C, D\}$

(ii) Write all prodⁿs except ϵ prodⁿs

$S \to aAa \mid bBb \mid$
$A \to C \mid a$
$B \to C \mid b$
$C \to CDE \mid$
$D \to A \mid B \mid ab$

(iii) Apply substitution rule (ϵ) to replace Nullable variable with ϵ & write down all the prodⁿs

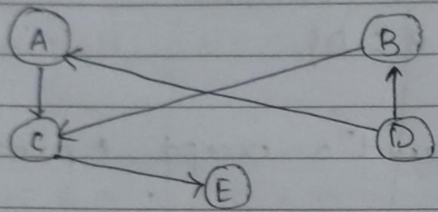
$S \to aAa \mid bBb \mid aa \mid bb$
$A \to C \mid a$
$B \to C \mid b$
$C \to CDE \mid DE \mid CE \mid E$
$D \to A \mid B \mid ab$

Step ②: Removal of Unit prodⁿs:

(i) Identify unit prodⁿs

$A \rightarrow C$
$B \rightarrow C$
$C \rightarrow E$
$D \rightarrow A \mid B$

(ii) Write a dependency graph



✓ $A \overset{*}{\Rightarrow} C$	$A \overset{*}{\Rightarrow} E$
✓ $B \overset{*}{\Rightarrow} C$	$B \overset{*}{\Rightarrow} E$
✓ $D \overset{*}{\Rightarrow} A$	✓ $D \overset{*}{\Rightarrow} C$
✓ $D \overset{*}{\Rightarrow} B$	$D \overset{*}{\Rightarrow} E$
✓ $C \overset{*}{\Rightarrow} E$	

(iii) Rewrite prodⁿs excluding unit prodⁿs & add all possible prodⁿs

$S \rightarrow aAa \mid bBb \mid aa \mid bb$
$A \rightarrow a \mid CDE \mid CE \mid DE$
$B \rightarrow b \mid CDE \mid CE \mid DE$
$C \rightarrow CDE \mid CE \mid DE$
$D \rightarrow ab \mid b \mid CDE \mid CE \mid DE \mid a$

Step ③: Removal of useless prodⁿs

(i) Identify vv that derive terminal string

	OV	NV	Prodⁿs
1	ϕ	S, A, B, D	$S \rightarrow aa \mid bb$ $A \rightarrow a$ $B \rightarrow b$ $D \rightarrow a \mid ab \mid b$