Aditya Dhanaraj Kundu

Formal Language & Automata

YCS 4004

HL

$\downarrow$ $\leftarrow$ Finite Automata $\leftarrow$ Regular grammar (g_1)

Token

$\downarrow$ $\leftarrow$ Push down Automata $\leftarrow$ Content free grammar (g_2)

Parse tree

$\downarrow$ $\leftarrow$ Lower Bound Automata $\leftarrow$ Content Sensitive grammar (g_3)

Code generation

$\downarrow$ $\leftarrow$ Turing Machine $\leftarrow$ Universal grammar (g_4)

ML

$$g_1 \subseteq g_2 \subseteq g_3 \subseteq g_4$$

Alphabet $\rightarrow$ Alphabet $\Rightarrow$ $\Sigma = \{a, b, c, \text{----}, 0, 1\}$

(In case of computer only binary 0 and 1)

$\downarrow$ $\quad\quad$ $\downarrow$

Word $\rightarrow$ Substring

$\downarrow$ $\quad\quad$ $\downarrow$

Sentence $\rightarrow$ String

$\downarrow$ $\quad\quad$ $\downarrow$

language $\rightarrow$ languages Eg: It is a language that accept even no of 0's.

$$L = \Sigma = \{d, c\}$$

$$L_0 = \Sigma = \{1, 0\}$$

09/Jan/2023

[ ] $\rightarrow$ machine

format of instruction

$\downarrow$

Language

$\downarrow$

Symbol, Grammar

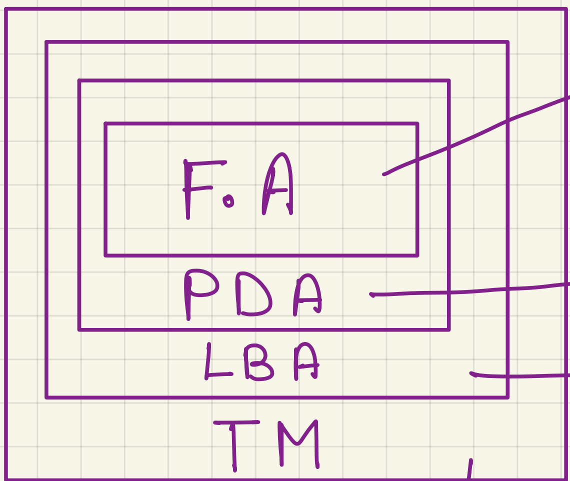
Σ = Set of alphabet ie $\{0, 1\}$

Q = Finite Set of states

q = Initial State

δ = Transition Function.

Types of Automata :-



Regular language {Finite Automata}

Content free Language {Push Down Automata}

Content Sensitive language {Lower Bound Auto..}

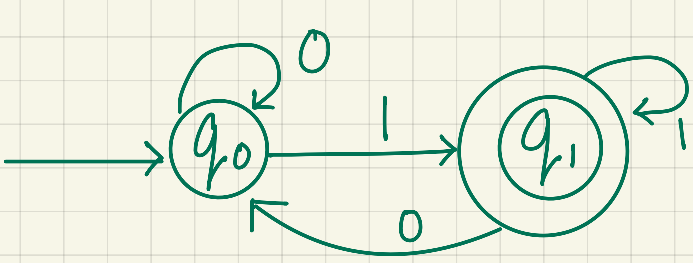
Universal Language. {Turing Machine}

Finite Automata :-

F. A → Finite State Machine

Suppose there is a machine M that can perform a task by 2 states q_0 & q_1. Machine can understand the language that has two alphabet. {0,1}. So the input of the machine will be 0,1.

So for state transition, we have to use these two inputs.



the final state is bounded by two circle.

1, 1 0 1, 1 0 0, 1 0 1 0

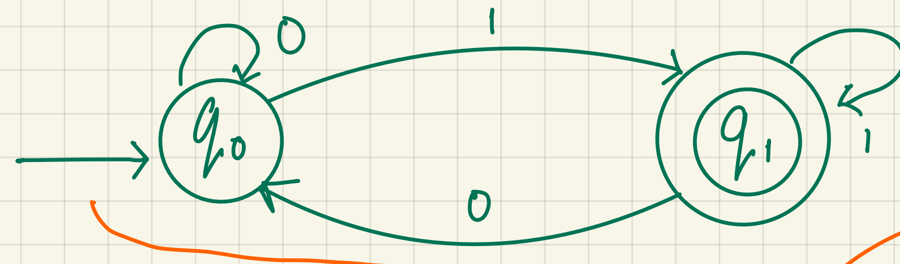
q_1 q_1 q_0 q_1 q_1 q_0 q_0 q_1 q_0 q_1 q_0

accepted by automata   not accepted by automata

REMEMBER

if the last bit is not in the final state then it is not accepted by automata.

10/Jan/2023



this arrow symbol signifies the initial state

① q_0 is the initial state, $\longrightarrow$ (q_0)

② q_1 is the final state, signified by double circle enclosement ie ◎

③ Input symbol = 0, 1 denoted by Σ

④ Transition function are denoted by δ

⑤ δ = ① $\delta(q_0, 0) \rightarrow \underline{\underline{q_0}}$ (current state: q_0; input: 0) $\longrightarrow$ next state

② $\delta(q_1, 1) \rightarrow q_1$

③ $\delta(q_0, 1) \rightarrow q_1$

④ $\delta(q_1, 0) \rightarrow q_0$

$$\frac{0}{q_0}\;\frac{1}{q_1}\;\frac{1}{q_1}\;\frac{1}{q_1} \longrightarrow \text{state transition}$$

For representing a finite automata we need five tuples that are, F.A = $\{q_0, Q, \Sigma, \delta, F\}$

$q_0 \rightarrow$ initial state

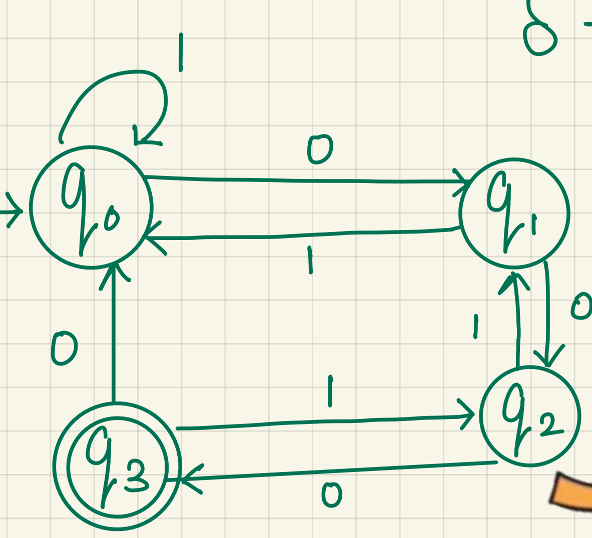
$Q \rightarrow$ set of states

$\Sigma \rightarrow$ set of input symbols

$F \rightarrow$ set of final states

$\delta \rightarrow$ transition function

M_1



$q_0 \rightarrow$ q_0 is initial state.

$Q \rightarrow \{q_0, q_1, q_2, q_3\}$

$\Sigma \rightarrow \{0, 1\}$ $F = \{q_3\}$

$\delta =$ $\delta(q_0, 0) \rightarrow q_1$

$\delta(q_0, 1) \rightarrow q_0$

$\delta(q_1, 0) \rightarrow q_2$

$\delta(q_1, 1) \rightarrow q_0$

$\delta(q_2, 0) \rightarrow q_3$

$\delta(q_2, 1) \rightarrow q_1$

$\delta(q_3, 0) \rightarrow q_0$

$\delta(q_3, 1) \rightarrow q_2$

- **Input Symbols :-**

Current state	0	1
q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_3	q_1
q_3	q_0	q_2

State transition table

$q^n \rightarrow \underline{q \times n}$ ie there are 4 states from q_0 to q_3 and 2 input symbol.
$\therefore$ $4 \times 2 = 8$ transition state.

There are two type of finite automata — i) Deterministic ii) Non deterministic

- **Deterministic :-** (DFA)

In this automata state transition for all the inputs is specific, that means for 1 input 1 state can go to only one state not to the multiple state.

- **Non-deterministic :-** (NFA)

This automata 1 state can go to multiple state for a single input.
Equivalent DFA can be generated from a NFA

In case of DFA single input, single state $\delta = Q \times \Sigma \rightarrow Q$

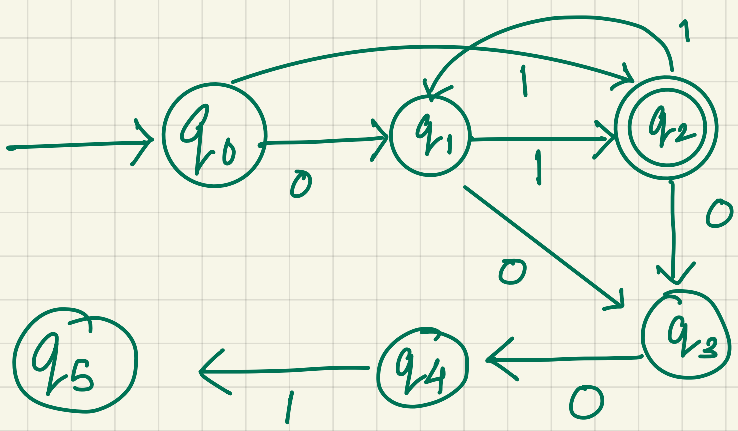
In case of NDFA single input, more than one state $\delta = Q \times \Sigma \rightarrow 2^Q$

Accessibility of string by Finite Automata :-

Starting from the initial state and after consuming all the input string if we will reach to the final state then automata accept the string.

$$(q_0, X0) \Rightarrow (q_2, 0) \Rightarrow (q_3, X)$$

$$(q_0, \cancel{0}10) \Rightarrow (q_1, X0) \Rightarrow (q_2, \cancel{0}) \Rightarrow (q_3, X)$$



- $\delta(q_0, \cancel{0}11001)$

$\Rightarrow \delta(q_1, \cancel{1}1001)$

$\Rightarrow \delta(q_2, \cancel{1}001)$

$\Rightarrow \delta(q_1, \cancel{0}01)$

$\Rightarrow \delta(q_3, \cancel{0}1) \Rightarrow \delta(q_4, 1) \Rightarrow \delta(q_5, X)$

- $\delta(q_0, \cancel{0}11101)$

$\Rightarrow \delta(q_1, \cancel{1}1101) \Rightarrow \delta(q_2, \cancel{1}101) \Rightarrow \delta(q_1, \cancel{1}01) \Rightarrow \delta(q_2, \cancel{0}1) \Rightarrow \delta(q_3, 1)$

haulting problem of automata

Present State	0	1
q_0	q_1	q_2
q_1	q_3	q_2
q_2	q_3	q_1
q_3	q_4	—
q_4	—	q_5
q_5	—	—

$$(q_0, 1) \cup (q_1, 1)$$

$$= q_0 q_1$$

NDFA to DFA :-

Present State	0	1
→ q_0	q_1	q_2
q_1	q_0	$q_0 q_1$
q_2	q_2	q_1

⇒

Present State	0	1
→ $[q_0]$	$[q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_0 q_1]$
$[q_2]$	$[q_2]$	$[q_1]$
→ $[q_0 q_1]$	$[q_0 q_1]$	$[q_0 q_1 q_2]$
→ $[q_0 q_1 q_2]$	$[q_0 q_1 q_2]$	$[q_0, q_1, q_2]$

Both initial & final state

$(q_0, 1) \cup (q_1, 1)$

$= q_2 \cup q_0 q_1$

$= [q_0 q_1 q_2]$

New State

Present state	0	1
→ q_0	q_0	q_2
q_1	q_2	q_1
q_2	q_3	$q_0 q_1$
q_3	q_3	$q_2 q_1$

	0	1
→ $[q_0]$	$[q_0]$	$[q_2]$
$[q_2]$	$[q_3]$	$[q_0 q_1]$
$[q_3]$	$[q_3]$	$[q_2 q_1]$
→ $[q_0 q_1]$	$[q_0 q_2]$	$[q_1 q_2]$
$[q_1 q_2]$	$[q_2, q_3]$	$[q_0, q_1]$
→ $[q_0 q_2]$	$[q_0, q_3]$	$[q_0 q_1, q_2]$
$[q_2 q_3]$	$[q_3]$	$[q_0 q_1 q_2]$
→ $[q_0 q_3]$	$[q_0 q_3]$	$[q_1, q_2]$
→ $[q_0 q_1 q_2]$	$[q_0 q_2 q_3]$	$[q_0 q_1 q_2]$
→ $[q_0 q_2 q_3]$	$[q_0 q_3]$	$[q_0 q_1 q_2]$

Mealy and Moore Machine :-

Mealy machine gives the output from the present states and present inputs.
Moore machine produces outputs from present state only.

$$(\underbrace{Q}_{\text{set of states}}, \underbrace{\Sigma}_{\text{set of input alphabet}}, \underbrace{\Delta}_{\text{set of output alphabet}}, \underbrace{\delta}_{\text{transition function}}, \underbrace{q_0}_{\text{initial state}}, \underbrace{\lambda}_{\text{output function}})$$

Present State	Next State $a=0$	$a=1$	λ
q_1	q_1	q_2	0
q_2	q_2	q_3	1
q_3	q_3	q_1	1

$Q \rightarrow \Delta$

Present State	Next State $a=0$		$a=1$	
	State	λ	State	λ
q_1	q_2	$0/Z_1$	q_3	$0/Z_1$
q_2	q_1	$1/Z_2$	q_2	$1/Z_2$
q_3	q_2	$1/Z_3$	q_1	$1/Z_3$

$Q \times \Sigma \longrightarrow \Delta$

Present State	$a=0$		$a=1$	
	State	Output	State	Output
q_1	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

$q_2 \rightarrow 0$

Conversion from Mealy to Moore :-

Present State	a = 0		a = 1	
	State	λ	State	λ
q_1	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}	q_{41}	1	q_3	0

Conversion from Mealy to Moore.

Present State	a = 0	a = 1	
	State	State	λ
q_{01}	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	1

Q.

P.S	a = 0		a = 1	
	State	λ	State	λ
q_1	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	1	q_3	1
q_4	q_3	0	q_1	1

P.S	a = 0		a = 1	
	State	λ	State	λ
q_1	q_1	1	q_{20}	0
q_{20}	q_4	1	q_4	1
q_{21}	q_4	1	q_4	1
q_{30}	q_{21}	1	q_{31}	1
q_{31}	q_{21}	1	q_{31}	1
q_4	q_{30}	0	q_1	1

P.S	a = 0	a = 1	λ
q_1	q_1	q_{20}	1
q_{20}	q_4	q_4	0
q_{21}	q_4	q_4	1
q_{30}	q_{21}	q_{31}	0
q_{31}	q_{21}	q_{31}	1
q_4	q_{30}	q_1	1

Conversion from Mealy to Moore.

Moore machine to Mealy Machine :~

09/Feb/2023

P.S	a=0	a=1	λ
q_0	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

$\rightarrow$

PS	a=0	λ	a=1	λ
q_0	q_3	0	q_1	1
q_1	q_1	1	q_2	0
q_2	q_2	0	q_3	0
q_3	q_3	0	q_0	0

Q.1.

P.S	a=0	a=1	λ
q_1	q_1	q_2	0
q_2	q_1	q_3	0
q_3	q_1	q_3	1

$\rightarrow$

PS	a=0	λ	a=1	λ
q_1	q_1	0	q_2	0
q_2	q_1	0	q_3	1
q_3	q_1	0	q_3	1

Q.2.

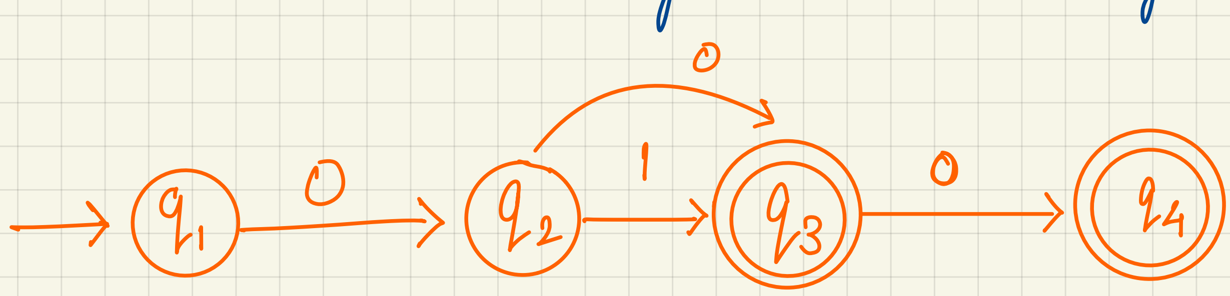
P.S	a=0	a=1	λ
q_0	q_1	q_2	1
q_1	q_3	q_2	0
q_2	q_2	q_1	1
q_3	q_0	q_3	1

$\rightarrow$

P.S	a=0	λ	a=1	λ
q_0	q_1	0	q_2	1
q_1	q_3	1	q_2	1
q_2	q_2	1	q_1	0
q_3	q_0	1	q_3	1

Minimisation of Automata :

Two states q_1 and q_2 are equivalent if & only if (iff) ie (q_1, x) and (q_2, x) reach to same final state or non final state.



$(q_1, 0) \rightarrow$ N.F
$(q_3, 0) \rightarrow$ F

Nonequivalent

Non final state $\rightarrow$ $\pi_0 = (q_1, q_2)$

Final state $\longrightarrow$ (q_3, q_4)

π_0 = Set of single type of group.

P.S	0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
q_2	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

$\pi_0 = (\{q_2\} \{q_0\ q_1\ q_3\ q_4\ q_5\ q_6\ q_7\})$

$\pi_1 = \{\{q_2\} \{q_0\ q_4\ q_6\} \{q_1\ q_7\} \{q_3\ q_5\}\}$

$\pi_2 = (\{q_2\} \{q_0 q_4\} \{q_1 q_7\} \{q_3 q_5\}$

15/Feb/2023

$G = \{N, T, P, S\} \longrightarrow 1 \xrightarrow{i} 2 \xrightarrow{n} \bigcirc \xrightarrow{t} \circledcirc$

T = Terminal state

S = Start Symbol.

P = production rule

N = Non terminal state

1. P: $S \rightarrow AB$
   $B \rightarrow aB$ → Production Rule
   $C \rightarrow AC$
   $\{(A, B, C, S), \{a\}, S\}$

2. $S \rightarrow AB$
   $A \rightarrow BC$
   $B \rightarrow ab$
   $aB \rightarrow Ba$
   $\{(A, B, C, S), \{a, b\}, S\}$

Type-0 → Phrase Structure — Turing Machine

Type-1 → Context Sensitive — Linear Bound Automata

Type-2 → Context free — Pushdown Automata

Type-3 → Regular Grammar → Finite

Type-①: $\alpha \to \beta$, $|\alpha| \leq |\beta|$

$AS \to AB$

$AB \to SD$

$BA \to aB$ → Type 0

$aB \to aCD$

$S \to AB$

$AB \to SD$ → Type 1

$A \to aB$

$aB \to aB$

$S \to AB$

$A \to BC$ → Type 1

$B \to aA$

$C \to ab$

(what is the highest form of the grammar?)

Type-②: Type 2 grammar always have a single variable in left hand side of production.

$A \to \alpha$

$S \to AB$

$aA \to BC$ → Type 1 but not type 2

$B \to aA$

$C \to ab$

$L(G) = a^n b^n , n > 1$

Type-③: $A \to aB$

$A \to Ba$

$A \to a$

$L(G) \Rightarrow$ Set of all terminal string generated from the start symbol by the grammar is known as the Language of the Grammar.

$S \to ab$

$S \to aabb$

$S \to aSb$

$\{ab, aabb, aaabbb \ldots \}$

$S \rightarrow aSb$

$S \rightarrow aabb \Rightarrow aSb \Rightarrow aaSbb$

$\left\{ \begin{array}{l} S = ab \\ S \rightarrow aSb \end{array} \right\} \longrightarrow$ grammar

$L(G) = a^n, \quad n > 1$

$\{ a, aa, aaa, \ldots \}$

$S \rightarrow a$

$S \rightarrow aa \Rightarrow aS$

$S \rightarrow aaa \Rightarrow aaS$

$\left. \begin{array}{l} S \rightarrow a \\ S \rightarrow aS \end{array} \right\}$ grammar

$L(G) = a^i b^n c^n, \quad n \geq 1 \quad i \geq 0$

$n = 1, i = 0 \Rightarrow S \rightarrow bc$

$\{ bc, abbcc, aabbbccc \ldots$

In, $n = 1, i = 1 \Rightarrow S \rightarrow abc$

$S \rightarrow aS$

1) $S \rightarrow bc$

In, $n = 2, i = 1,$

2) $S \rightarrow aS$

$S \rightarrow abbcc$

$S \rightarrow absc$

for $a^n b^n$

$S \rightarrow ab$

$S \rightarrow aSb$

so, for $b^n c^n$, $S \rightarrow bc$, $S \rightarrow bSc$

1) $S \to bc$
2) $S \to ac$
3) $S \to bcc$

$n = 3, i = 2$

$S \to aa\,bbbccc$

~~S→aabbbccc~~

$S \to aS \to abSc \to$

*) $\boxed{S \to aSc \quad S \to \lambda \text{ (NULL)}}$ → Production Rule
P:- What is the L.G of the given production rule?

Sol. $S \to aSc \Rightarrow aaScc$

S. $\{\lambda, ac, aacc, \ldots\}$ $\quad \{S\}\ \{\lambda, a, c\}$

$L(G) = a^n c^n \quad n \geq 0$

*) $S \to a|b|aS|bS$ $\quad a^+ = \{a, aa, aaa, \ldots\}$

$\Rightarrow L(G) = (ab)^+$ $\quad a^* = \{\lambda, a, aa, aaa, \ldots\}$

$\{a, b, aa, ab \ldots$
$ba, bb \ldots\}$

*) $L(G) = WcW^T \quad W \in (a,b)^*$

c aca when W = a

Sol. $S \to c$ when null value

$S \to aSa$ when $W = a$ $\quad S \to abSba$

Q. $S \to aSa \mid bSb$

$S \Rightarrow bSb \Rightarrow basab$

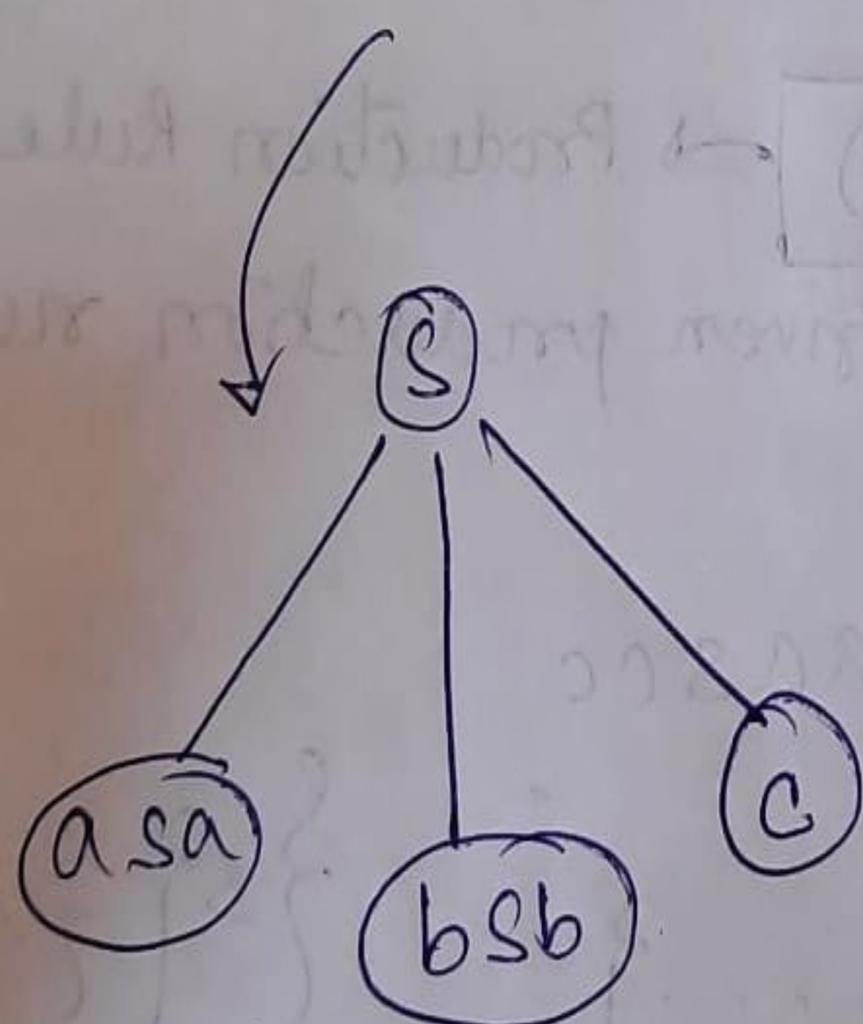
$S \Rightarrow wcw^T$

$S \rightarrow abcba$

$S \rightarrow aSa \rightarrow absba \rightarrow abcba$.

The grammar is :-

$S \rightarrow aSa \mid bSb \mid c$ (Ans :-)



16/Feb/2023

Type 2 : $A \to aB$
$B \to Ba$
$A \to a$

$A \to BC$

$AB \to C$

$L(G) = a^i b^n c^n \quad n \geq 1 \geq i \geq 0$

$= bc$

$n=1;\ i=0$

$S \to bc$

$a^n b^n$

$n=1;\ i=1$

$S \to ab$

$S \to abc$

$n=2;\ i=1$

$S \to aSb$

$S \to aS$

$n=2;\ i=2$

$S \Rightarrow aS \Rightarrow aGS$

$S \Rightarrow aS \Rightarrow abSc \Rightarrow abbcc$

$aabbcc$

<u>Problem</u> : $a \to aSc \qquad S \to \lambda$ (null)

$S \Rightarrow aSc \Rightarrow aaScc$
$\Rightarrow aacc$

$L = \{\lambda, ac, aacc \ldots\ldots\}$

$L(G) = a^n b^n \quad n \geq 0$

$a^+ = \{a, aa, aaa \ldots\ldots\}$

$L(G) = WCW^T \quad$ T is transpose.

↳ Write the language of this.

$a^* = \{\lambda, a, aa, aaa \_\_ . \}$

↳ with null value.

$W \in (a,b)^*$

$L(G) = aca$

$S \to c$

$S \to aSa \mid bSb$

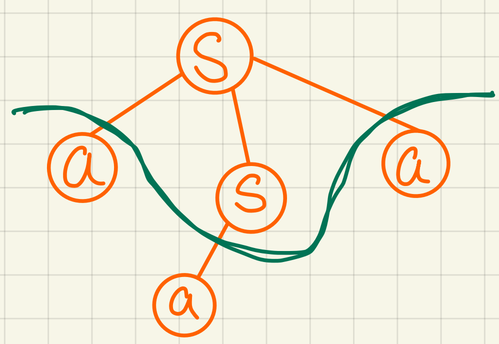
$S \Rightarrow aSa \Rightarrow abSba \Rightarrow abcba$

22/Feb/2023

Parse tree :-

- For access purpose link list is a linear data structure and for storage purpose link list is a non linear data structure.
- By using Parse tree we can.
- The start symbol of the grammar is the root node of the parse tree.
- leaf node is always the terminal string of the grammar.
- Variable of the grammar/non-terminal is the internal node of the tree.

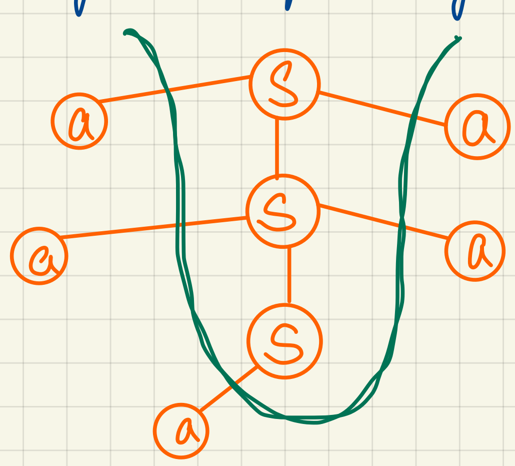
$S \rightarrow aSa / a$



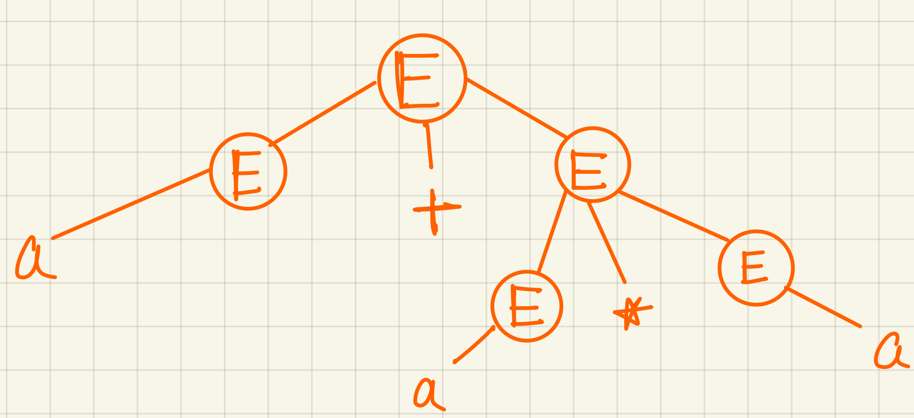
aaa → acceptable

aaaaa → acceptable

- Going from left to right if travelling all the leaf node, if we get the string then the given string is accepted by the grammar.



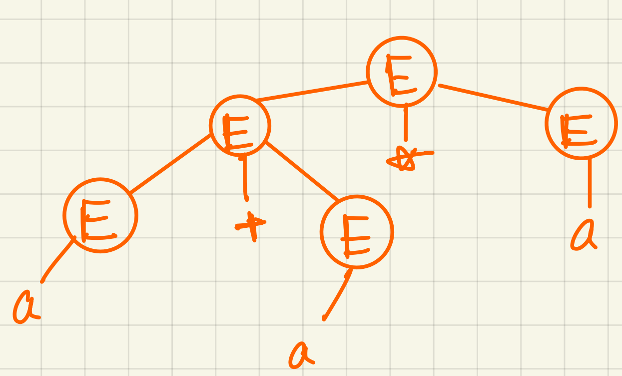
$E \rightarrow E+E / E*E / a$



a+a*a

Right most

$E \overset{lm}{\Rightarrow} \underline{E}*E \overset{lm}{\Rightarrow} \underline{E}+E*E \overset{lm}{\Rightarrow} a+\underline{E}*E \overset{lm}{\Rightarrow} a+a*E \Rightarrow a+a*a$



Left most

$E \overset{rm}{\Rightarrow} E * \underline{E} \overset{rm}{\Rightarrow} \underline{E} * a \overset{rm}{\Rightarrow} E + \underline{E} * a \overset{rm}{\Rightarrow} \underline{E} + a * a \overset{rm}{\Rightarrow} a + a * a$

(2) $E \overset{rm}{\Rightarrow} E + \underline{E} \overset{rm}{\Rightarrow} E + E * \underline{E} \overset{rm}{\Rightarrow} E + \underline{E} * a \overset{rm}{\Rightarrow} \underline{E} + a * a \overset{rm}{\Rightarrow} a + a * a$

- **Ambiguity of the grammar :-**

When the grammar has the left most and right most derivation for the same input string then the grammar have ambiguity.

Q. Without the input string can you say that the grammar is ambiguous ? (GATE)

Ans: No, without the input string the grammar is not ambiguous.

Simplification of grammar :-

$S \rightarrow AB$, $A \rightarrow a$, $B \rightarrow b$, $C \rightarrow D$, $D \rightarrow E$, $E \rightarrow x$ Useless symbol/production

$(S, A, B, C, D, E)\ (a, b, x)$

$L(G) = ab \quad S \Rightarrow AB \Rightarrow ab$

Unit production : Single variable derived the single variable

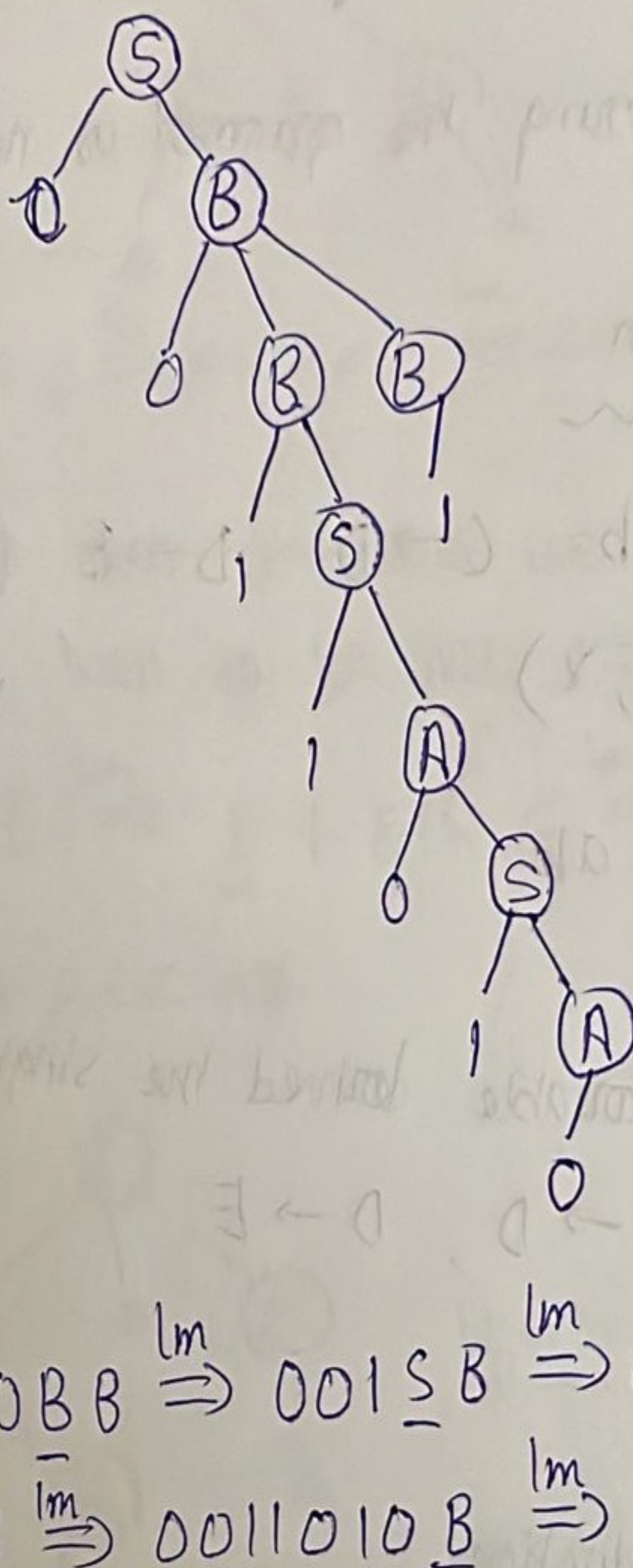
e.g. $C \rightarrow D$, $D \rightarrow E$

1. Reduce
2. Removal of unit production
3. Removal of null production.

$S \rightarrow 0B \mid 1A \quad A \rightarrow 0 \mid 0S \mid 1AA \quad B \rightarrow 1 \mid 1S \mid 0BB$

00110101

(S, A, B) (0, 1)



$S \overset{lm}{\Rightarrow} 0\underline{B} \overset{lm}{\Rightarrow} 0\underline{B}B \overset{lm}{\Rightarrow} 001\underline{S}B \overset{lm}{\Rightarrow} 0011\underline{A}B \overset{lm}{\Rightarrow} 00110\underline{S}B$
$\overset{lm}{\Rightarrow} 001101\underline{A}B \overset{lm}{\Rightarrow} 0011010\underline{B} \overset{lm}{\Rightarrow} 00110101$

$S \overset{rm}{\Rightarrow} 0\underline{B} \overset{rm}{\Rightarrow} 0B\underline{B} \overset{rm}{\Rightarrow} 00B1\underline{S} \overset{rm}{\Rightarrow} 00B10\underline{A} \overset{rm}{\Rightarrow} 00B101\underline{S}$
$\overset{rm}{\Rightarrow} 00B1010\underline{B} \overset{rm}{\Rightarrow} 00\underline{B}10101 \overset{rm}{\Rightarrow} 00110101$

$S \rightarrow AB \quad A \rightarrow a \quad B \rightarrow b \quad \cancel{B \rightarrow C} \quad E \rightarrow c$

discard

Step 1

In W we can keep those variable which generate terminal string.

$W_0 = \{A, B, E\}$

$W_1 = \{S, A, B, E\}$

Find the variable which generate the element of the W_0

Find the variable which generate the element of W_1

$W_2 = \{S, A, B, E\}$

$V_n' = \{S, A, B, E\}$

$S \rightarrow AB \quad A \rightarrow a \quad B \rightarrow b \quad E \rightarrow c$ ← new set of productio

Step 2

$W_0 = \{S\}$, In W_0 put start symbol

$W_1 = (S, A, B)$, In W_1 see the variable & terminal string generated by the ~~element~~ set of the W_0

$W_2 = (S, A, B, a, b)$
$W_3 = (S, A, B, a, b)$ same ⇒ stop

$V_n'' = (S, A, B)$, $T = (a, b)$

$S \rightarrow AB \quad A \rightarrow a \quad B \Rightarrow b$

$S \to AB/CA \quad B \to 0 \quad A \to 1 \quad B \to BD \quad D \to 0$

Step 1

$W_0 = \{A, B, D\}$

$W_1 = \{$

$S \to \cancel{AB}/CA \quad B \to \cancel{BC}/AB \quad A \to a \quad C \to \cancel{aB}/b$

Step 1

$W_0 = \{A, C\}$

$W_1 = \{S\cancel{A}, A, C\}$

$W_2 = \{S, A, C\}$

Same $\Rightarrow$ stop

$V_n' = \{S, A, C\}$

$S \to CA \quad A \to a \quad C \to b$

Step 2

$W_0 = \{S\}$

$W_1 = \{S, A, C\}$

$W_2 = \{ S, A, C \cancel{B}, a, b \}$

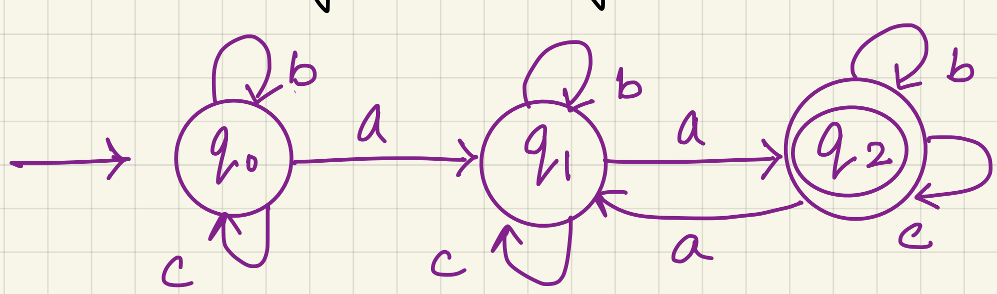
$V_n'' = \{ S, A, C \}$ $\quad T = \{ a, b \}$

$S \to AC, \; A \to a, \; C\cancel{B} \to b$

16/March/2023

R.K.J

• String Matching with Finite Automata :-



$\delta(q_0, c) = q_0$
$\delta(q_0, a) = q_1$
$\delta(q_0, b) = q_0$
$\delta(q_1, a) = q_2$
$\delta(q_1, b) = q_1$
$\delta(q_1, c) = q_1$
$\delta(q_2, c) = q_2$
$\delta(q_2, a) = q_1$
$\delta(q_2, b) = q_2$

String $\Rightarrow$ $W =$ bcaabcaaabac

$= q_0\,caabcaaabac$

$= q_0\,aabcaaabac$

$= q_1\,abcaaabac$

$= q_2\,bcaaabac$

$= q_2\,caaabac$

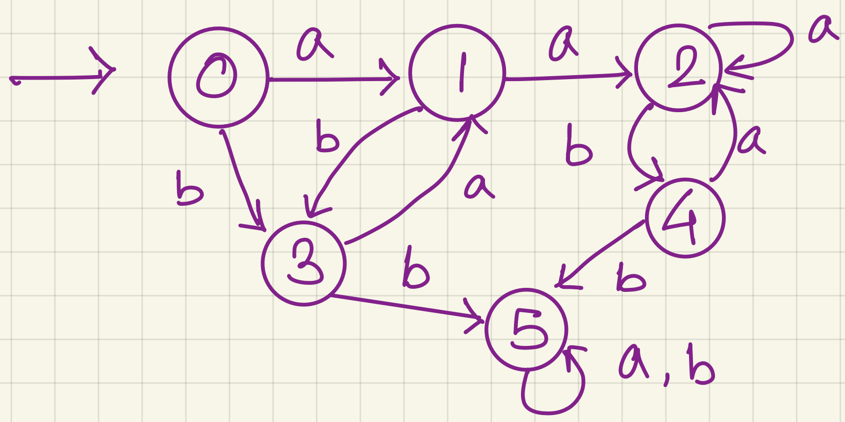
$= q_2\,aaabac = q_1\,aabac = q_2\,abac$

$= q_1\,bac$

$= q_1\,ac = q_2\,c$

$= q_2$

$\Rightarrow$ Automata accepted ✓



$W = bababaabaaabbaab$

$= 3ababaabaaabbaab$

$= 1babaabaaabbaab$

$= 3abaabaaabbaab$

$= 1baabaaabbaab$

$= 3aabaaabbaab$

$= 1abaaabbaab$

$= 2baaabbaab$

$= 4aaabbaab$

$= 2aabbaab$

$= 2bbaab$

$= 4baab = 5aab$

$= 5$ [Automata accepted]

22/March/2023

Removal of unit production :-

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

$W_0(S) = \{S\}$

$W_0(S) = \{S\}\{\phi\} = \{S\}$

$W_0(A) = \{A\}$

$W_0(B) = \{B\}$

$= \{B, C\}$

$= \{B, C, D\}$

$= \{B, C, D, E\}$ $\quad \because C \to a \Rightarrow B \to a$

$W_0(C) = \{C\}$
$= \{C, D\}$
$= \{C, D, E\}$

$\because D \to a \Rightarrow C \to a$

$W_0(D) = \{D\}$
$= \{D, E\}$

$\because E \to a \Rightarrow D \to a$

$W_0(E) = \{E\}$

Now we need to make changes where more than one variables are present.

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

CNF (Chomsky Normal Form)
GNF (Greibach Normal Form)

Same as SOP and POS

CNF $\Rightarrow$ $A \to a$ or $A \to BC$

GNF $\Rightarrow$ $A \to a\alpha \quad \alpha = (T \cup N)^*$

Terminal

Non terminal

$S \to aAD \quad A \to aB/bAB \quad B \to b \quad D \to d$

Step 1 : $\boxed{\text{CNF}}$

Remove the null and unit production of a given grammar.

Step 2 : Remove the terminal string by a variable in given grammar.

$P' = B \to b \quad D \to d \quad C_a \to a \quad A \to C_aB \quad C_b = b$

Now $S \to C_aAD \qquad \boxed{C_a \to a}$

$A \to C_aB / C_bAB \qquad \boxed{C_b \to b}$

Step-3 : Restrict the variable in the form of CNF

$S \rightarrow C_a AD$ $\quad$ $\boxed{\begin{array}{l} S \rightarrow C_a C_1 \\ A \rightarrow C_b C_2 \end{array}}$ $\quad$ $\boxed{\begin{array}{l} C_1 \rightarrow AD \\ C_2 \rightarrow AB \end{array}}$

$A \rightarrow C_b AB$

<u>New set of production rule</u> :

$P'' = B \rightarrow b \quad D \rightarrow d \quad C_a \rightarrow a \quad A \rightarrow C_a B \quad C_b \rightarrow b$

$S \rightarrow C_a C_1 \quad A \rightarrow C_b C_2 \quad C_1 \rightarrow AD \quad C_2 \rightarrow AB$

<u>New variable set</u> :

$V' = \{S, A, B, D, C_a, C_b, C_1, C_2\}$

Q. $S \rightarrow aAbB \quad A \rightarrow aA / a \quad B \rightarrow bB / b$

<u>Ans</u>: **Step 1 :** We remove the production rule.

Step 2 : $S \rightarrow C_a A C_b B$ $\quad$ $\boxed{C_a \rightarrow a}$

$\boxed{A \rightarrow C_a A / \cancel{C_a}}$ $\quad$ $\boxed{B \rightarrow C_b B / \cancel{C_b}}$ $\quad$ $\boxed{C_b = b}$

$P' = C_a \rightarrow a \quad C_b \rightarrow b \quad A \rightarrow C_a A / C_a \quad B \rightarrow C_b B / C_b$

Step-3 : S_3 : $S \rightarrow C_a A C_b B$

$S \rightarrow C_a A C_1 \quad\quad C_1 \rightarrow C_b B$

$C_2 \rightarrow A C_1$

S_4 : $S \rightarrow C_a A C_1$

$S \rightarrow C_a C_2$

<u>New Set of production rule</u> :

$P'' = C_a \rightarrow a \quad C_b \rightarrow b \quad A \rightarrow C_a A / \cancel{C_a} \quad B \rightarrow C_b B / \cancel{C_b} \quad C_1 \rightarrow C_b B \quad C_2 \rightarrow A C_1$

$S \rightarrow C_a C_2$

<u>New variable set</u> :-

$V' = \{S, A, B, C_a, C_b, C_1, C_2\}$

Q. $S \to ABb / a \quad A \to aaA \quad B \to bAB$

Ans: Step-1 : Remove the production rule

$S \to ABb \quad A \to aaA \quad B \to bAb$

Step-2 : $S \to ABC_b$

$A \to C_a C_a A$

$B \to C_b A C_b$

$\boxed{C_b \to b}$

$\boxed{C_a \to a}$

$P' = C_b \to b \quad C_a \to a$

Step-3 : $S \to ABC_b$

$S \to AC_1 \qquad C_1 \to BC_b$

$A \to C_a C_a A$

$A \to C_a C_2 \qquad C_2 \to C_a A$

$B \to C_b A C_b$

$B \to C_b C_3 \qquad C_3 \to AC_b$

New set of production rule :-

$P'' = C_b \to b \quad C_a \to a \quad C_1 \to BC_b \quad C_2 \to C_a A \quad C_3 \to AC_b$

$S \to AC_1 \quad A \to C_a C_2 \quad B \to C_b C_3$

New variable set :

$V' = \{S, A, B, C_a, C_b, C_1, C_2, C_3\}$

Lemma 1 $\quad L_1 \to A \to B\gamma \quad B \to aB / bA$

Lemma 2 $\quad L_2 \qquad \therefore A \to aB\gamma / bA\gamma$

- $A \to A\alpha_1 / A\alpha_2 / \ldots / B_1 / B_2 \ldots$

$S \to SA / SB / ab / ca$ $\qquad$ ($B \to$ doesn't start with A)

$\alpha_1 = A / B$

$B = ab / ca$

$A \rightarrow A\alpha_1 / A\alpha_2 / \ldots / \beta_1 / \beta_2 \ldots$

If the given production is in this form then,

$A \rightarrow \beta_1 / \beta_2 \ldots$

$A \rightarrow \beta_1 Z / \beta_2 Z \ldots$

$Z \rightarrow \alpha_1 / \alpha_2 / \ldots$

$Z \rightarrow \alpha_1 Z / \alpha_2 Z / \ldots$

Q. $A \rightarrow aBD / bDB / a \quad A \rightarrow AB / AD$

$A \rightarrow aBDZ / bDBZ / aZ$

$Z \rightarrow AB / AD$

$Z \rightarrow BZ / DZ$

$E \rightarrow E+E / E*E / a$

$\beta \rightarrow a, \ \alpha_1 \rightarrow +E \quad \alpha_2 \rightarrow *E$

$E \rightarrow aZ$

$Z \rightarrow +E / *E$

$Z \rightarrow +EZ / *EZ$

22/March/2023

① $S \rightarrow AA / a$

$A \rightarrow SS / b$

Rule ① for GNF : The production rule is in CNF

$S \rightarrow AA / a$

$S \downarrow A_1$, $A \downarrow A_2$

Rule ② : Rename with numeric value.

$A_1 \rightarrow A_2 A_2 / a$ — ①

$A_2 \rightarrow A_1 A_1 / b$ — ②

$A_i = A_j \gamma$

$j \geq i$

Lemma 1 we can put,

$$A_2 \rightarrow A_2 A_2 A_1 / aA_1 / b$$

after satisfying the condition of Lemma 2 we can apply Lemma 2,

$A = A\alpha_1 / A\alpha_2 / \ldots \beta_1 \beta_2 \ldots$ β does not

$A_2 \rightarrow aA_1 / b$ $\qquad \alpha = A_2 A_1$

$A_2 \rightarrow aA_1 Z / bZ$ $\qquad \beta_1 = aA_1$

$Z \rightarrow A_2 A_1$ $\qquad \beta_2 = b$

$Z \rightarrow A_2 A_1 Z$

put A_2 in $Z \rightarrow A_2 A_1$:

$\therefore Z \rightarrow aA_1 A_1 / bA_1$

$Z \rightarrow aA_1 Z A_1 / bZA_1$

Put A_2 in $Z \rightarrow A_2 A_1 Z$:

$Z \rightarrow aA_1 A_1 Z / bA_1 Z$

$Z \rightarrow aA_1 Z A_1 Z / bZA_1 Z$

Put A_2 in eqn ① :

$A_1 \rightarrow aA_1 A_2 / bA_2 / aA_1 Z A_2 / bZA_2 / a$

Q. Convert into GNF :

$S \to AB \quad A \to BS / b \quad B \to SA / a$

Rule ① : Convert into CNF,

$A_1 \to A_2 A_3$ —— ①

$A_2 \to A_3 A_1 / b$ —— ②

$A_3 \to A_1 A_2 / a$ —— ③

Rule ② : Rename

Griechbach Normal Form (GNF)

$A \rightarrow b$

$A \rightarrow b A_i \dots A_n$ (Any no. of variable followed by a terminal)

Step ①: Make sure the CFG is in Chomsky Normal form, so no null, unit and other useless productions, and if it is not in CNF convert it to CNF

Step ②: Replace the name of non terminal symbols with some A_i in ascending order of i.

eg:

$S \rightarrow CA \mid BB$
$B \rightarrow b \mid SB$
$C \rightarrow b$
$A \rightarrow a$

} this CFG is in CNF

Replacing the name of non terminals,

$S \rightarrow A_1$
$C \rightarrow A_2$
$A \rightarrow A_3$
$B \rightarrow A_4$

∴ Now the grammar is —

$A_1 \rightarrow A_2 A_3 \mid A_4 A_4$
$A_4 \rightarrow b \mid A_1 A_4$
$A_2 \rightarrow b \qquad A_3 \rightarrow a$

Step ③: Alter the rules, such that the non terminals are in ascending order. If the production is in form $A_i \rightarrow A_j X$, then always $i < j$

∴ From above productions, $A_4 \rightarrow b \mid A_1 A_4$ — 4 is not less than 1

Substitute A_1,

$A_4 \rightarrow b \mid A_2 A_3 A_4 \mid A_4 A_4 A_4$ — 4 not less than 2

Sub... A_2, — left recursion

$A_4 \rightarrow b \mid b A_3 A_4 \mid A_4 A_4 A_4$ — have to remove left recursion.

Step ⑤: Removing the left recursion, by introducing a new variable symbol

$$\underline{A_4} \rightarrow b \mid b A_3 A_4 \mid \underline{A_4 A_4 A_4}$$

left recursion.

introducing a new variable production,

$A_4 \rightarrow A_4 A_4 A_4$

replacing this one time with new variable and second time without anything.

$\therefore Z \rightarrow A_4 A_4 Z \mid A_4 A_4$

Now writing the recursive production one time with new variable and second time without.

$$A_4 \rightarrow bZ \mid bA_3A_4Z \mid b \mid bA_3A_4$$

Now the grammar is —

$\underline{A_1} \rightarrow A_2 A_3 \mid A_4 A_4 \longrightarrow$ not in GNF

$A_4 \rightarrow bZ \mid bA_3A_4Z \mid b \mid bA_3A_4 \longrightarrow$ is in GNF

$\underline{Z} \rightarrow A_4 A_4 Z \mid A_4 A_4 \longrightarrow$ Not in GNF

$A_2 \rightarrow b \qquad A_3 \rightarrow a$

Substituting A_2, A_4 for A_1:

$A_1 \rightarrow bA_3 \mid A_4 A_4$

in GNF:

$A_1 \rightarrow bA_3 \mid bZA_4 \mid bA_3A_4ZA_4 \mid bA_4 \mid bA_3A_4A_4$

Substituting A_4 for Z:

$Z \rightarrow bZA_4Z \mid bA_3A_4ZA_4Z \mid bA_4Z \mid bA_3A_4A_4Z \mid bZA_4 \mid bA_3A_4ZA_4 \mid bA_4 \mid bA_3A_4A_4$

Final Production in GNF:

$A_1 \rightarrow bA_3 \mid bZA_4 \mid bA_3A_4ZA_4 \mid bA_4 \mid bA_3A_4A_4$

$A_4 \rightarrow bZ \mid bA_3A_4Z \mid b \mid bA_3A_4$

$Z \rightarrow bZA_4Z \mid bA_3A_4ZA_4Z \mid bA_4Z \mid bA_3A_4A_4Z \mid bZA_4 \mid bA_3A_4ZA_4 \mid bA_4 \mid bA_3A_4A_4$

$A_2 \rightarrow b$

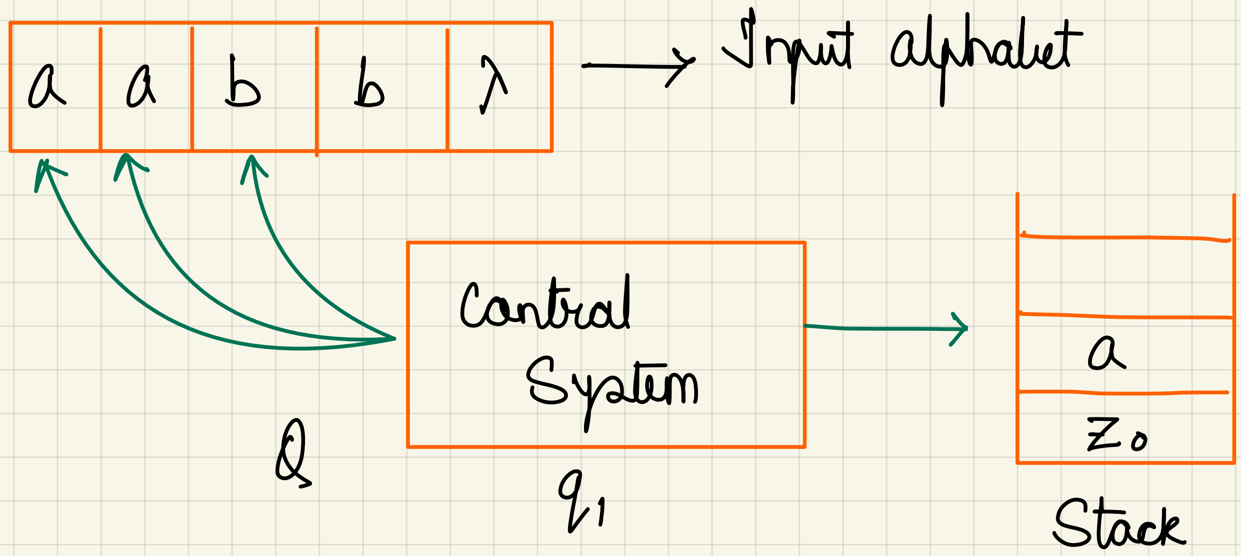
$A_3 \rightarrow a$

Ans

29/March/2023

Pushdown Automata :-

Type II ie Content Free Grammar is accepted by Pushdown Automata.



Seven tuples of Automata :-

$Q \rightarrow$ Finite Non empty set of State

$(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$

where,

$Q \rightarrow$ Set of state

$\Sigma \rightarrow$ Set of input alphabets

$\Gamma \rightarrow$ Stack alphabet

$q_0 \rightarrow$ initial state

$Z_0 \rightarrow$ initial stack symbol.

$\delta \rightarrow$ transition function

$F \rightarrow$ Final state

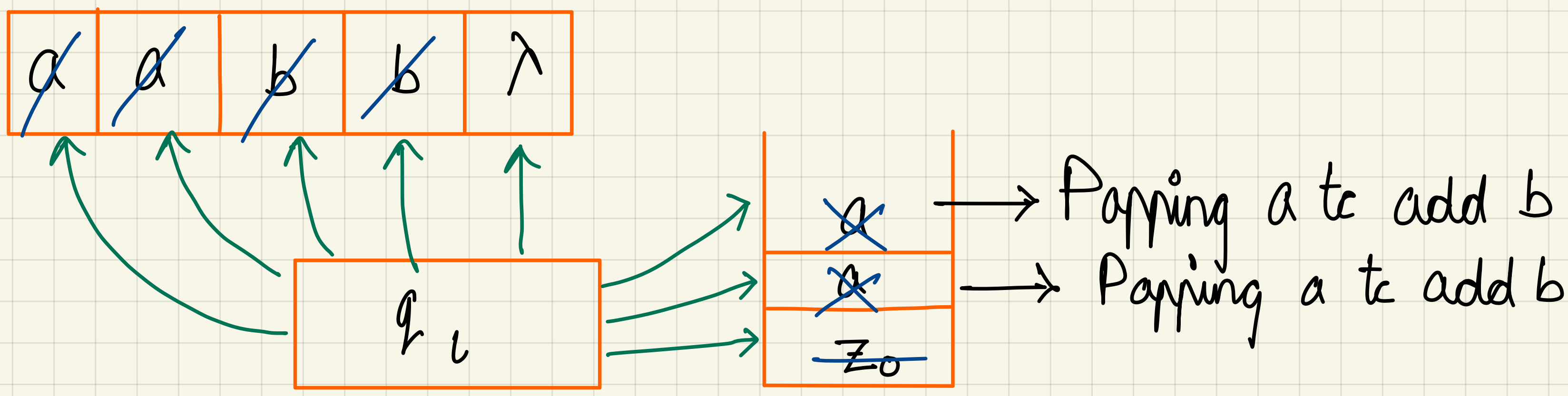
$Q \times \Sigma \times \Gamma \rightarrow Q \times \Gamma$

eg: $(q_0, a, Z_0) \rightarrow (q_1, a, Z_0)$

Acceptance by Pushdown Automata :-

- By final state $\rightarrow (q_0, w, Z_0) \rightarrow (q_f, \alpha)$ → Value may be present
- By empty store $\rightarrow \underbrace{(q_0, w, Z_0)}_{\text{Continue form}} \rightarrow (q_f, \lambda)$ → no value may be present.

$L(G) = a^n b^n, n \geq 1$



We will not change the state for a.

6) $\delta(q_0, a, Z_0) \Rightarrow (q_0, aZ_0)$

7) $\delta(q_0, a, a) \Rightarrow (q_0, aa)$

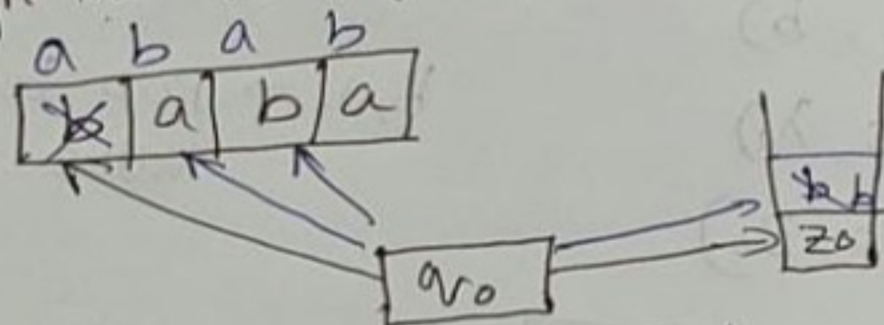
8) $\delta(q_0, b, a) \Rightarrow (q_1, \lambda)$

9) $\delta(q_1, b, a) \Rightarrow (q_f, \lambda)$

10) $\delta(q_f, \lambda, Z_0) \Rightarrow (q_f, \lambda) / (q_f, Z_0)$

Q. Design the Pushdown Automata for equal no of a and equal no of b.

Design the PDA for equal no. of a & equal no. of b



$\delta(q_0, b, Z_0) \Rightarrow (q_0, bZ_0)$

$\delta(q_0, a, b) \Rightarrow (q_1, \lambda)$

$\delta(q_1, b, Z_0) \Rightarrow (q_1, bZ_0)$

$\delta(q_1, a, b) \Rightarrow (q_f, \lambda)$

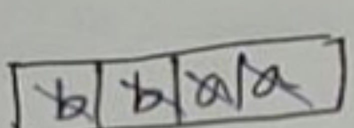
$\delta(q_0, a, Z_0) \Rightarrow (q_0, aZ_0)$

$\delta(q_0, b, a) \Rightarrow (q_1, \lambda)$

~~$\delta(q_1, a, b) \Rightarrow (q_f, \lambda)$~~

$\delta(q_1, a, Z_0) \Rightarrow (q_1, aZ_0)$

$\delta(q_1, b, a) \Rightarrow (q_f, \lambda)$



1) $\delta(q_0, b, Z_0) \Rightarrow (q_0, bZ_0)$

2) $\delta(q_0, b, b) \Rightarrow (q_0, bb)$

3) $\delta(q_0, a, b) \Rightarrow (q_1, \lambda)$

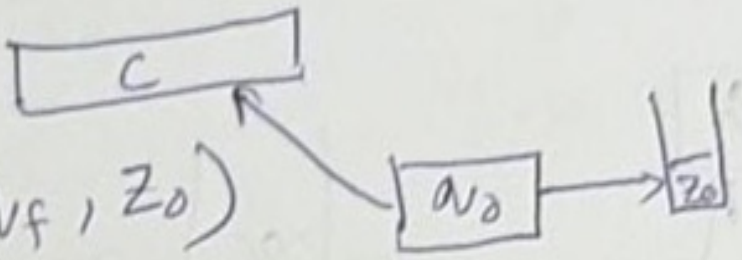
4) $\delta(q_1, a, b) \Rightarrow (q_f, \lambda)$

5) $\delta(q_f, \lambda, Z_0) \Rightarrow (q_f, \lambda)$

$L(a) = W c W^T$ $\quad$ $wt\,(a,b)$ $\quad$ $w = ab$

$W^T = ba$

If null $= c$



1. $\delta(q_0, c, Z_0) \Rightarrow (q_f, Z_0)$

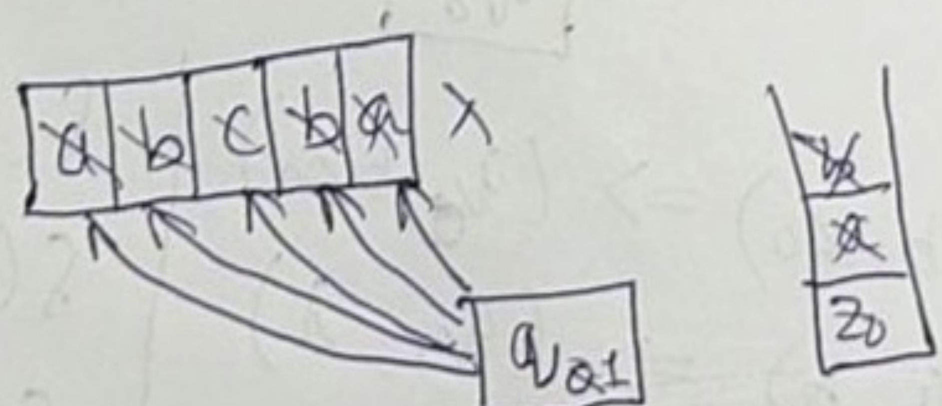
a	c	a

2. $\delta(q_0, a, Z_0) \Rightarrow (q_0, aZ_0)$
3. $\delta(q_0, c, a) \Rightarrow (q_1, a)$
4. $\delta(q_1, a, a) \Rightarrow (q_f, \lambda)$
5. $\delta(q_f, \lambda, Z_0) \Rightarrow (q_f, Z_0)$

b	c	b

6. $\delta(q_0, b, Z_0) \Rightarrow (q_0, bZ_0)$
7. $\delta(q_0, c, b) \Rightarrow (q_1, b)$
8. $\delta(q_1, b, b) \Rightarrow (q_f, \lambda)$
9. $\delta(q_f, \lambda, Z_0) \Rightarrow (q_f, Z_0)$

b, b ⇒ pop bom value



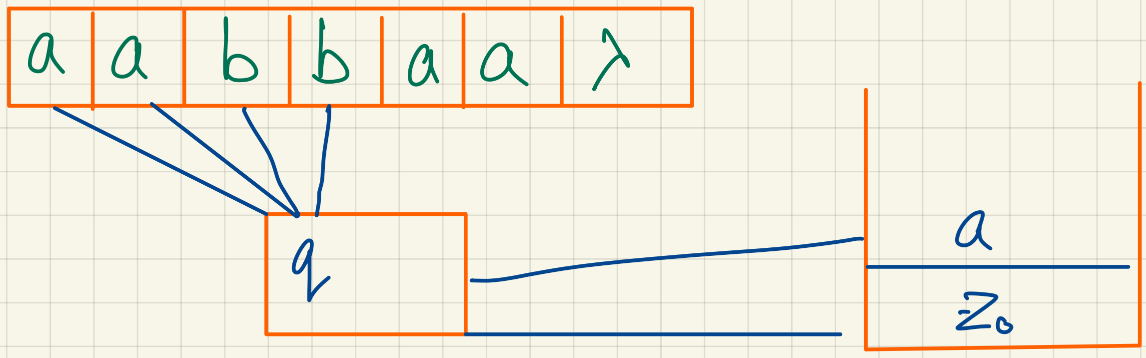
$L(n)$

Design the PDA for $= a^n b^n c^n$, $n \geq 1$

Pushdown automata designing :-

31/March/2023

$L(G) = a^n b^m a^n \quad m, n \geq 1$



$\delta(q_0, a, Z_0) \Rightarrow (q_0, aZ_0)$

$\delta(q_0, a, a) \Rightarrow (q_0, a, a)$

$\delta(q_0, b, a) \Rightarrow (q_f, a)$

$\delta(q_0, b, a) \Rightarrow (q_1, a)$

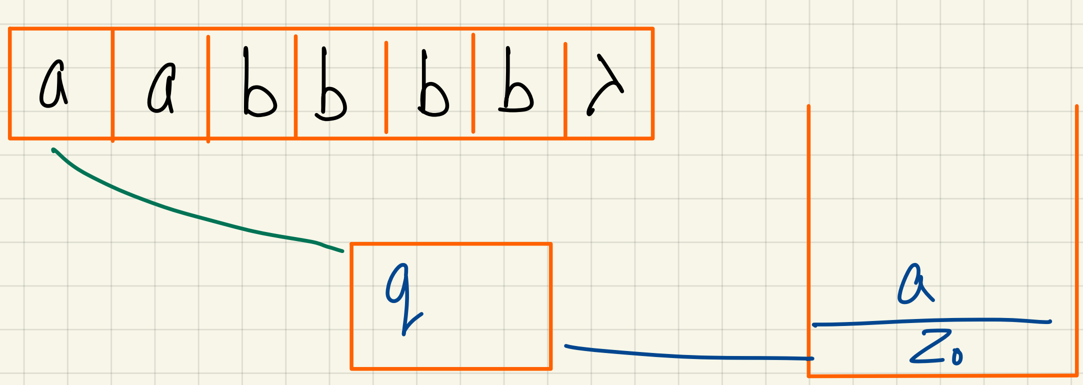
$\delta(q_1, a, a) \Rightarrow (q_1, \lambda)$

$\delta(q_1, \lambda, Z_0) \Rightarrow \delta(q_f, Z_0)$

$L(G) = a^n b^{2n} \quad n \geq 1$

$= a^n b^n b^n$

$= aa\ bb\ bb$



$\delta(q_0, a, Z_0) \Rightarrow (q_0, aZ_0)$

$\delta(q_0, a, a) \Rightarrow (q_0, aa)$

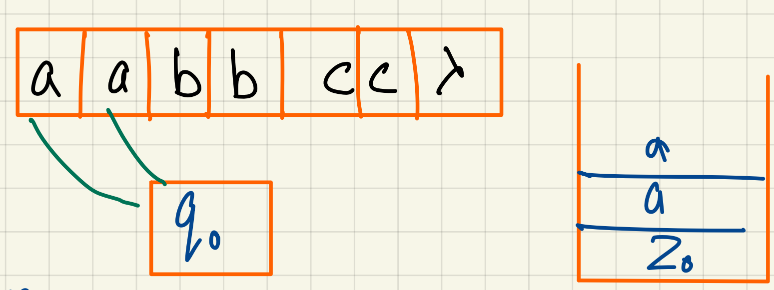
$\delta(q_0, a, b) \Rightarrow (q_1, b)$

$\delta(q_1, b, b) \Rightarrow (q_1, \lambda)$

$\delta(q_1, b, a) \Rightarrow (q_1, b)$

$\delta(q_1, \lambda, Z_0) = (q_f, Z_0)$

$L(G) = a^n b^m c^n \quad m, n \geq 1$



a^n, c^n

$\delta(q_0, a, Z_0) \Rightarrow (q_0, aZ_0)$

$\delta(q_0, a, a) \Rightarrow (q_0, aa)$

$\delta(q_0, a, b) \Rightarrow (q_1, a)$

$\delta(q_1, b, a) \Rightarrow (q_1, a)$

$\delta(q_1, c, a) \Rightarrow (q_1, \lambda)$

$\delta(q_1, \lambda, Z_0) \Rightarrow (q_f, Z_0)$

12/April/2023

- If L is a Context Free Language then we can the PDA accepting L by empty stare.

$PDA \rightarrow (Q, \Sigma, \Gamma, \delta, Z_0, q_0, F)$

$S \rightarrow 0SA$ and $A \rightarrow \alpha$

① $\delta(q, \lambda, A) \Rightarrow (q, \alpha)$

$\delta(q, \lambda, S) \Rightarrow (q, 0SA)$

② $\delta(q, a, a) \Rightarrow (q, \lambda)$

$\delta(q, 0, 0) \Rightarrow (q, \lambda)$

$\delta(q, 1, 1) \Rightarrow (q, \lambda)$

Q. Construct PDA, 'A' equivalent to the following CFG.

$S \rightarrow 0BB \quad B \rightarrow 0S \mid 1S \mid 0$

Soln: According Rule ①,

$\delta(q, \lambda, S) \Rightarrow (q, 0BB)$ — ①

$\delta(q, \lambda, B) \Rightarrow (q, 0S)$ — ②

$\delta(q, \lambda, B) \Rightarrow (q, 1S)$ — ③

$\delta(q, \lambda, B) \Rightarrow (q, 0)$ — ④

According Rule ②:-

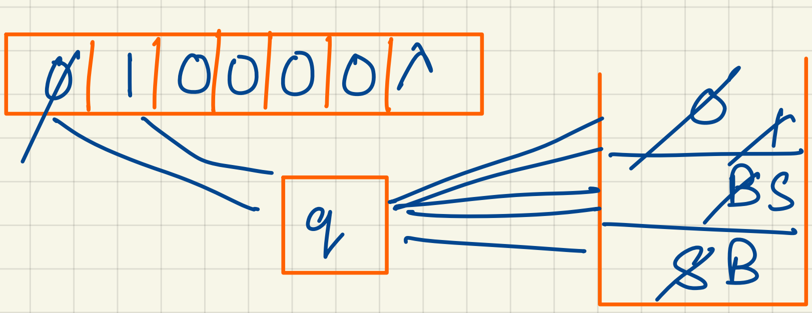
$\delta(q, 0, 0) \Rightarrow (q, \lambda)$ — ⑤

$\delta(q, 1, 1) \Rightarrow (q, \lambda)$ — ⑥

Now check whether the string 010^4 is accepted by the system or not.

$010^4 =$ | 0 | 1 | 0 | 0 | 0 | 0 | $\wedge$ |

1) $\delta(q, 010^4, S) \Rightarrow (q, 010^4, 0BB)$ By rule ①



2) $\delta(q, \cancel{0}10^4, \cancel{0}BB) \Rightarrow (q, 10^4, BB)$ By rule ⑤

3) $\delta(q, 10000, BB) \Rightarrow (q, 10^4, 1SB)$ By rule ③

4) $\delta(q, \cancel{1}0000, \cancel{1}SB) \Rightarrow (q, 0000, SB)$ By rule ⑤

5) $\delta(q, 0000, SB) \Rightarrow (q, 0000, 0BBB)$ By rule ①

6) $\delta(q, \cancel{0}000, \cancel{0}BBB) \Rightarrow (q, 000, BBB)$ By rule 5

7) $\delta(q, 000, BBB) \Rightarrow (q, 000, 0BB)$ By rule 4

8) $\delta(q, \cancel{0}00, \cancel{0}BB) \Rightarrow (q, 00, BB)$ By rule 5

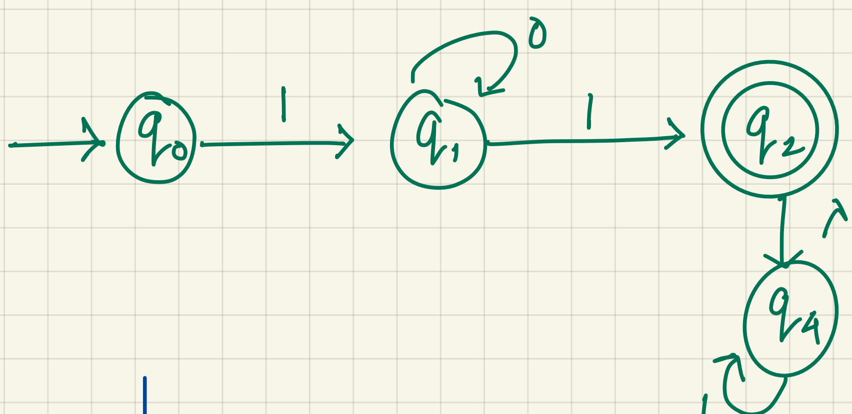
9) $\delta(q, 00, BB) \Rightarrow (q, 00, 0B)$ By rule 4

10) $\delta(q, \cancel{0}0, \cancel{0}B) \Rightarrow (q, 0, B)$ By rule 5

11) $\delta(q, 0, B) \Rightarrow (q, 0, 0)$ By rule 4

12) $\delta(q, 0, 0) \Rightarrow (q, \lambda, \lambda)$ ∴ So, the string is accepted by the system.

19/April/2023



	0	1
$\rightarrow q_0$	ϕ	q_1
q_1	q_1	q_2, q_4
q_2	ϕ	q_4
q_4	ϕ	q_4

q_0

ϵ^*	0	ϵ^*
q_0	—	—

q_1

ϵ^*	0	ϵ^*
q_1	q_1	q_1

q_0

ϵ^*	1	ϵ^*
q_0 —	q_1 —	q_1

q_1

ϵ^*	1	ϵ^*
q_1	q_2	q_2, q_4

q_2

ϵ^*	0	ϵ^*
q_2 —	ϕ —	ϕ
q_4 —	ϕ —	ϕ

q_2

ϵ^*	1	ϵ^*
q_2 —	ϕ —	ϕ
q_4 —	q_4 —	q_4

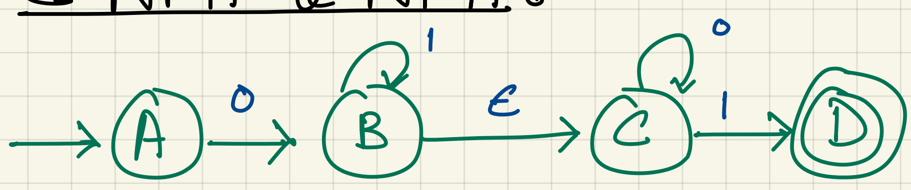
q_4

ϵ^*	0	ϵ^*
q_4 —	ϕ —	ϕ

q_4

ϵ^*	1	ϵ^*
q_4 —	q_4 —	q_4

ϵ NFA to NFA:

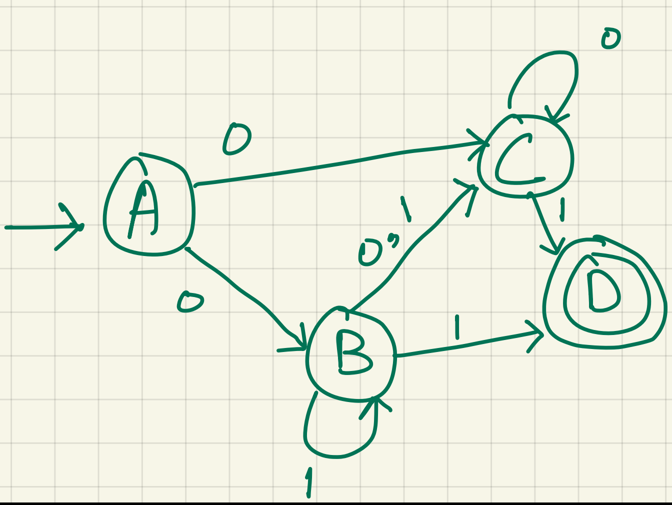


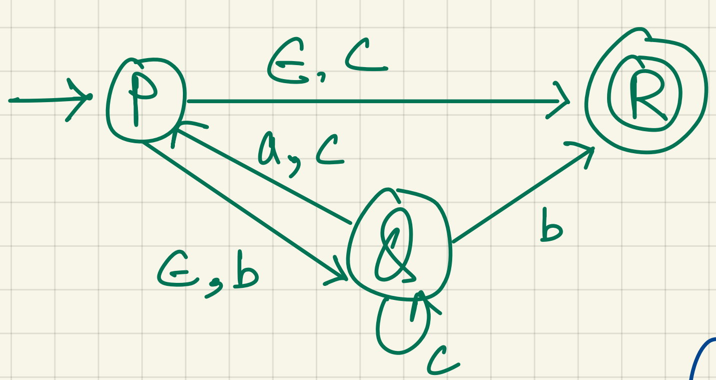
	ϵ^*	0	ϵ^*
A	A	B	B, C
B	B	ϕ	ϕ
	C	C	C
C	C	C	C
D	D	ϕ	ϕ

	ϵ^*	1	ϵ^*
A	A	ϕ	ϕ
B	B	B	B, C
	C	D	D
C	C	D	D
D	D	ϕ	ϕ

NFA

	0	1
A	B, C	ϕ
B	C	B, C, D
C	C	D
D	ϕ	ϕ





	ϵ^*	a	ϵ^*	ϵ^*	b	ϵ^*	ϵ^*	c	ϵ^*
P	P	ϕ	ϕ	P	Q	Q	P	R	R
	R	ϕ	ϕ	Q	R	R	Q	P	P, Q, R
	Q	P	P, Q, R	R	ϕ	ϕ	R	ϕ	ϕ
Q	Q	P	P, Q, R	Q,	R,	R	Q	P	P, Q, R
R	R	ϕ	ϕ	R	ϕ	ϕ	R	ϕ	ϕ

	a	b	c
P	P, Q, R	Q, R	P, Q, R
Q	P, Q, R	R	P, Q, R
R	ϕ	ϕ	ϕ