

## PART-A

1) Positive closure: (+ closure)

$$\text{Let } S = \{a\}$$

$$\text{Positive closure } L^+ = \sum_{i=1}^n L^i$$

$$L^+ = \{a, aa, aaa, \dots\}$$

Positive closure is a set of one or more occurrence of a string.

Kleen closure: (\* closure)

$$\text{Let } S = \{a\}$$

$$\text{Kleen closure } L^* = \sum_{i=0}^n L^i$$

$$L^* = \{ \epsilon, a, aa, aaa, \dots \}$$

Kleen closure is a set of zero or more occurrence of a string.

2) Non-Deterministic Finite Automata with E-transition:-

NFA with E-transition is defined

$$as \quad M = (\Omega, \Sigma, \delta, q_0, \{F\}).$$

$\Omega \rightarrow$  finite set of states

$\Sigma \rightarrow$  set of input string

$q_0 \rightarrow$  initial state

$F \rightarrow$  set of final state.

$\delta \rightarrow$  is transition function mapping from  $\Omega \times (\Sigma \cup \{\epsilon\}) \rightarrow \Omega$ .

(q, a), q is the set of all states of P, where a is  $\epsilon$  or any symbol.

3) Regular Expression:

\*  $\emptyset$  is a regular expression denote the language  $\{\}$  (empty set)

\*  $\epsilon$  is a regular expression denote the language  $\{\epsilon\}$

\* For any a in  $\Sigma$ , a is a regular expression denote the language  $\{a\}$ .

\* If r and s are regular expression denote the language R and S, then  $r+s$ ,  $rs$  and  $r^*$  are regular expressions denote the language  $R \cup S$ ,  $RS$  and  $R^*$ .

4) For language accepting the string, it must start with 1 and ends with 0,

The regular expression is

$$r = 1(a+)^* 0.$$

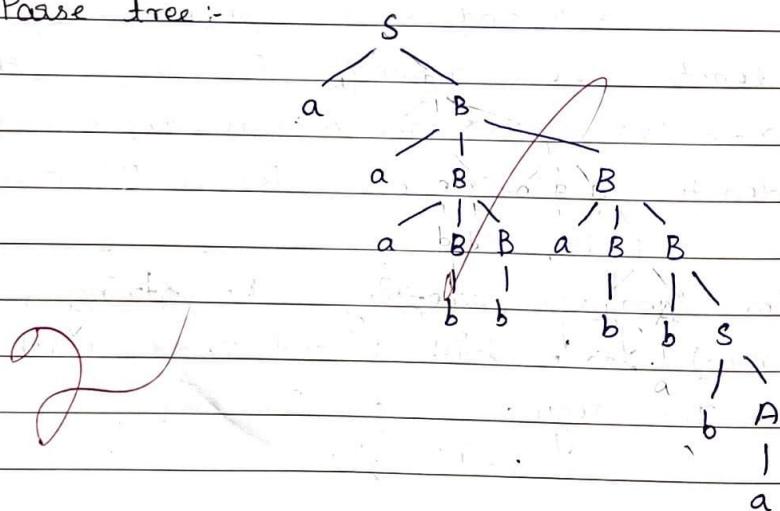
5)

$$\begin{aligned} S &\rightarrow aB \mid bA \\ A &\rightarrow a \mid as \mid bAA \\ B &\rightarrow b \mid bs \mid abb. \end{aligned}$$

$$\begin{aligned} S &\Rightarrow aB \quad :: B \rightarrow aBB \\ &\Rightarrow aaBB \quad :: B \rightarrow aBB \\ &\Rightarrow aaaBBB \quad :: B \rightarrow b \\ &\Rightarrow aaabBB \quad :: B \rightarrow b \\ &\Rightarrow aaabbB \quad :: B \rightarrow aBB \\ &\Rightarrow aaabbabb \quad :: B \rightarrow b \\ &\Rightarrow aaabbabbB \quad :: B \rightarrow bs \\ &\Rightarrow aaabbabbs \quad :: S \rightarrow bA \\ &\Rightarrow aaabbabbA \quad :: A \rightarrow a \end{aligned}$$

$S \Rightarrow aaabbabbA$

Parse tree:-



6) Context Free Grammar:-

Context free Grammar  $G_7$  is denoted as  $M = (V, \Sigma, F, S, q_0, X_0, P)$ .  
 $G_7 = (V, T, P, S)$ .

where,  $V$  = vset of non terminals & variable.

~~$T = vset of terminals$~~

~~$P = vset of Production in the form A \rightarrow x$~~   
~~where  $A \rightarrow$  variable~~

$\alpha \rightarrow$  string of terminals & non terminals

$S$  = special symbol, called start symbol.

Eg:-

$G_7 = (\{E\}, \{+, *\}, \{(\), \), id\}, P, E)$ .

where  $P$  consists of  $E \rightarrow E + F$

$E \rightarrow E * F$

$E \rightarrow (E)$

$E \rightarrow id$ .

7) Context Free Grammar  $G_7$  which has more than one parse tree is said to be ambiguous.

The equivalent ambiguous is defined as the derivation contain one or more

left most derivation and rightmost derivation.

$$\text{Eg: } E \rightarrow E+E \mid E * E \mid (E) \mid \text{id.}$$

$$F \Rightarrow E+E$$

$$\Rightarrow E+E+E$$

$$\Rightarrow \text{id} + E+E$$

$$\Rightarrow \text{id} + \text{id} + E$$

$$\Rightarrow \text{id} + \text{id} + \text{id}.$$

$$E \Rightarrow E+E$$

$$\Rightarrow E+E+E$$

$$\Rightarrow \text{id} + E+E$$

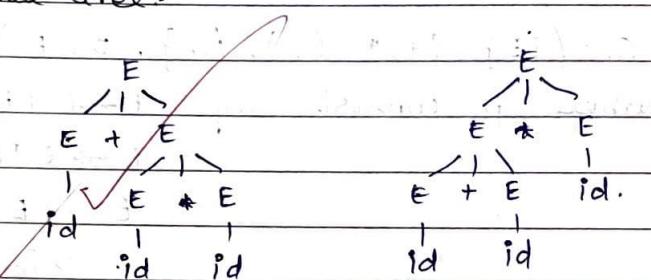
$$\Rightarrow \text{id} + \text{id} + E$$

$$\Rightarrow \text{id} + \text{id} + \text{id}.$$

The string  $\text{id} + \text{id} + \text{id}$  is derived in

different derivation.

Parse tree:



Given example obtain a different

parse tree.

The grammar is ambiguous.

$$8) S \rightarrow 0B11A$$

$$A \rightarrow 01S1AA$$

$$B \rightarrow 111S1DBB.$$

Right Most Derivation:-

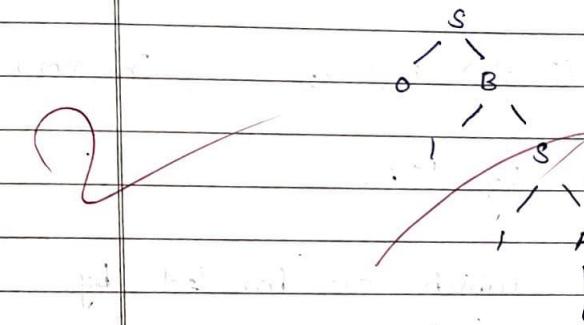
$$S \Rightarrow 0B$$

$$\Rightarrow 01S$$

$$\Rightarrow 011A$$

$$[S \Rightarrow 0110]$$

Parse tree:-



9) Two types of moves in PDA:-

\* In first type of move, input string is needed. Depending on the input string, state of finite control, stack top of a stack symbol, which

have  
to choose the first control input string.

- \* Each choice has the finite control which has to move next
- \* Input head is incremented.

$$(q, a, z) = (P_1, \gamma_1), (P_2, \gamma_2), (P_3, \gamma_3), \dots, (P_m, \gamma_m)$$

\* In second type of move is called e-move, in this input string is not required and input head need not to be increment.

$$(q, \epsilon, z) = (P_1, \gamma_1), (P_2, \gamma_2), (P_3, \gamma_3), \dots, (P_m, \gamma_m)$$

10) Languages handled by PDA:-

i) Languages which are handled by PDA by Empty stack

$$N(M) = \{ w \mid (q_0, w, z_0) \xrightarrow{*} (P, \epsilon, \epsilon') \}$$

ii) Languages which are handled by PDA by final final state

$$L(M) = \{ w \mid (q_0, w, z_0) \xrightarrow{*} (P, \epsilon, \epsilon'), \text{ for } P \in F \}$$

10) a) G1 be a grammar

$$S \rightarrow O B \quad | A$$

$$A \rightarrow O \quad | S \quad | A A$$

$$B \rightarrow 1 \quad | \quad S \quad | \quad D \quad B B$$

string - 0010101

Left most derivation:-

$$S \Rightarrow O B \quad :: B \rightarrow O B B$$

$$\Rightarrow O O B \quad B \quad :: B \rightarrow I$$

$$\Rightarrow O O I \quad B \quad | \quad S \quad :: B \rightarrow I S$$

$$\Rightarrow O O I \quad I \quad S \quad :: S \rightarrow O B$$

$$\Rightarrow O O I \quad I \quad O \quad B \quad :: B \rightarrow I S$$

$$\Rightarrow O O I \quad I \quad O \quad I \quad S \quad :: S \rightarrow O B$$

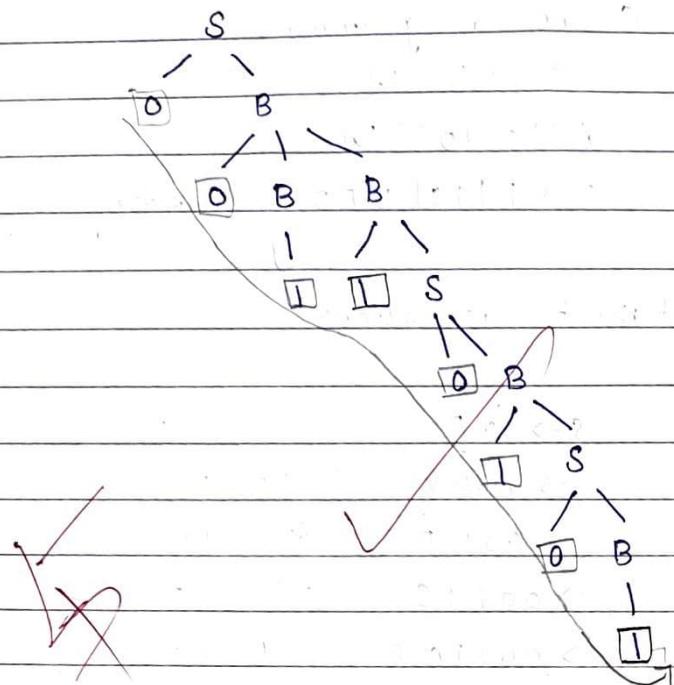
$$\Rightarrow O O I \quad I \quad O \quad I \quad O \quad B \quad :: B \rightarrow I$$

$$\Rightarrow O O I \quad I \quad O \quad I \quad O \quad I$$

$$S \Rightarrow O O I \quad I \quad O \quad I$$

By, left most derivation, the string 00110101 is obtained.

derivation tree :-



b) ~~G<sub>1</sub> is the grammar.~~  
 $S \rightarrow S b S \mid a.$

① One derivation

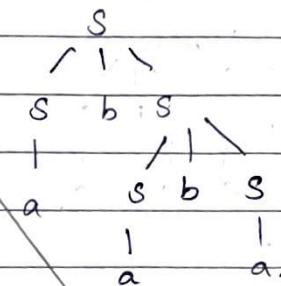
$$\begin{aligned} S &\Rightarrow S b S \\ &\Rightarrow S b S b S \\ &\Rightarrow a b S b S \\ &\Rightarrow a b a b S \\ S &\Rightarrow a b a b a \end{aligned}$$

② Another derivation.

$$\begin{aligned} S &\Rightarrow S b S \\ &\Rightarrow S b S b S \\ &\Rightarrow a b S b S \\ &\Rightarrow a b a b S \\ S &\Rightarrow a b a b a \end{aligned}$$

Derivation tree:-

①



② Another derivation:-

$$\begin{aligned} S &\Rightarrow S b S \\ &\Rightarrow a b S \\ &\Rightarrow a b S b S \\ &\Rightarrow a b \underline{a} b S \\ &\Rightarrow a b a b \underline{S} b S \\ &\Rightarrow a b a b \underline{a} b S \\ S &\Rightarrow a b a b a \end{aligned}$$

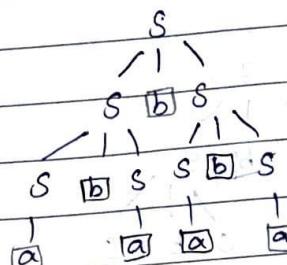
16) b)

~~G<sub>1</sub> is the grammar.~~  
 $S \rightarrow S b S \mid a.$

First derivation:-

$$\begin{aligned} S &\Rightarrow \underline{S} b S \\ &\Rightarrow \underline{S} b S b S \\ &\Rightarrow \underline{a} b S b S \\ &\Rightarrow a b \underline{\tilde{a}} b S \\ &\Rightarrow a b a b \underline{\tilde{S}} b S \\ &\Rightarrow a b a b \underline{\tilde{a}} b S \\ S &\Rightarrow a b a b a \end{aligned}$$

Derivation tree:-



Second derivation:-

$$S \Rightarrow S b S$$

$$\Rightarrow a b S$$

$$\Rightarrow a b S b S$$

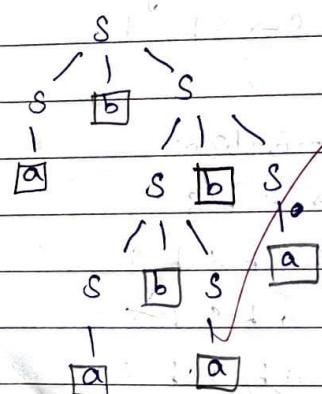
$$\Rightarrow a b S b S b S$$

$$\Rightarrow a b a b S b S$$

$$\Rightarrow a b a b a b S$$

$$S \Rightarrow a b a b a b a$$

Derivation tree:-



Grammar G<sub>1</sub> has a string abababa which is derived from two different derivation.

Grammar G<sub>1</sub> has a different parse tree.  
Explainer needed: G<sub>1</sub> is ambiguous.

PART-B.

14) a) Greibach Normal Form:-

$$x_1 \rightarrow x_2 x_3$$

$$x_2 \rightarrow x_3 x_1 \mid b$$

$$x_3 \rightarrow x_1 x_2 \mid a$$

First we has to check the given form is in CNF, if not convert it into CNF.

Then convert CNF into GNF.

For GNF, the left side is less than or equal to the right side.

$$x_1 \rightarrow x_2 x_3 \quad (\text{Acceptable})$$

$$x_2 \rightarrow x_3 x_1 \quad (\text{Acceptable})$$

Step 1:-

$$x_3 \rightarrow x_1 x_2 \quad (\text{Not accepted})$$

So replace  $x_1$ .

$$x_3 \rightarrow x_2 x_3 x_2 \quad (\text{Not accepted})$$

Again replace it by  $x_2$ .

$$x_3 \rightarrow x_2 x_1 x_3 x_2 \mid b x_3 x_2$$

Result:-

$$x_1 \rightarrow x_2 x_3$$

$$x_2 \rightarrow x_3 x_1 \mid b$$

$$x_3 \rightarrow x_3 x_1 x_3 x_2 \mid b x_3 x_2 \mid a.$$

Step 2:-

Consider which has L.H.S = R.H.S

(Same production)

$$x_3 \rightarrow x_3 x_1 x_3 x_2 \mid b x_3 x_2 \mid a.$$

$$x_3 \rightarrow b x_3 x_2 \mid a.$$

Now apply lemma 2,

a new non-terminal  $B_3$  is added.

$$x_3 \rightarrow b x_3 x_2 B_3 \mid a B_3$$

$$B_3 \rightarrow x_1 x_3 x_2$$

$$B_3 \rightarrow x_1 x_3 x_2 B_3$$

Step 2:-

Replace  $x_2$  in  $x_2 \rightarrow x_3 x_1$ 

Resultant:-

$$x_1 \rightarrow x_2 x_3$$

$$x_2 \rightarrow x_3 x_1 \mid b$$

$$x_3 \rightarrow b x_3 x_2 \mid a B_3 \mid b x_3 x_2 B_3 \mid a B_3$$

$$B_3 \rightarrow x_1 x_3 x_2 \mid x_1 x_3 x_2 B_3$$

Step 3:-

Replace  $x_3$  in  $x_2 \rightarrow x_3 x_1 \mid b$ .

~~$$x_2 \rightarrow b x_3 x_2 x_1 \mid a x_1 \mid b x_3 x_2 B_3 x_1 \mid a B_3 x_1 \mid b$$~~

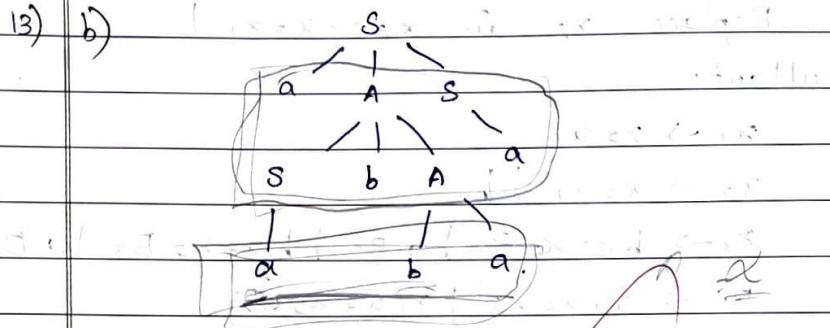
Replace  $x_2$  in  $x_1 \rightarrow x_2 x_3$ .

~~$$x_1 \rightarrow b x_3 x_2 x_1 x_3 \mid a x_1 x_3 \mid b x_3 x_2 B_3 x_1 x_3 \mid$$~~

~~$$a B_3 x_1 x_3 \mid b x_3$$~~

Replace  $x_1$  in  $B_3 \rightarrow x_1 x_3 x_2$

$$\begin{array}{l} B_3 \rightarrow bx_3x_2x_1x_3x_3x_2 \mid ax_1x_3x_3x_2 \\ bx_3x_2B_3x_1x_3x_3x_2 \mid aB_3x_1x_3x_3x_2 \\ \vdots \\ bx_3x_3x_2. \end{array}$$



Derivation and Language:-

Let  $A \rightarrow B$  be a production, where  $\alpha$  and  $\beta$  be any string of symbols.

$$\alpha A \beta \rightarrow \alpha B \beta$$

In the production  $A \rightarrow B$ , the string  $\alpha A \beta$  is derived to obtain  $\alpha B \beta$ .

Leftmost derivation:-

In the derivation, the production of the string is derived from the left most non-terminal is called leftmost derivation.



b28/3/12

Rightmost derivation:-

In the derivation, the production of the string is derived from the right-most non-terminal is called right-most derivation.

Derivation tree:-

- \* Derivation tree is also known as Parse tree.

- \* The Derivation is displayed as a Derivation tree.

- \* In the Derivation tree, the internal nodes are non-terminals.

- \* The leaf nodes are contain only non-terminals.

- \* The non-terminals in the leaf nodes combine to form a string.

~~Left most Derivation:~~

$$S \Rightarrow a A S$$

$$\Rightarrow a S b A S$$

$$\Rightarrow a a b A S$$

$$\Rightarrow a a b b a S$$

$$\Rightarrow a a b b a a$$

The left most derivation for the given parse tree is aaabbba.

~~Right most Derivation:~~

$$S \Rightarrow a A S$$

$$\Rightarrow a A a$$

$$\Rightarrow a S b A a$$

$$\Rightarrow a S b b a a$$

$$S \Rightarrow a a b b a a$$

The right most Derivation for the given parse tree is aaabbba.

i) a) Convert NFA to DFA:-

$$M = (\emptyset, \{0, 1\}, \delta, q_0, \{F\})$$

$$\delta(q_0, 0) = \{q_0, q_1\}$$

$$\delta(q_0, 1) = \{q_1\}$$

$$\delta(q_1, 0) = \{q_1\}$$

$$\delta(q_1, 1) = \{q_0, q_1\}$$

$$\delta = \begin{array}{c|cc} & 0 & 1 \\ \hline q_0 & \{q_0, q_1\} & \{q_1\} \\ * q_1 & \emptyset & \{q_0, q_1\} \end{array}$$

~~Now construct DNF:-~~

$$\delta = \begin{array}{c|cc} & 0 & 1 \\ \hline [q_0] & [q_0, q_1] & [q_1] \end{array}$$

$$[q_1] \quad [\emptyset] \quad [q_0, q_1]$$

$$[q_0, q_1] \quad [q_0, q_1] \quad [q_0, q_1]$$

$$[\emptyset] \quad [\emptyset] \quad [\emptyset]$$



$$\delta([q_0, q_1], 0) = ?$$

$$\delta([q_0, q_1], 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$$

$$= \{ \delta(q_0, q_1), \emptyset \} \cup \{ \emptyset \}$$

$$= [q_0, q_1]$$

$$= [q_0, q_1]$$

$$\delta([q_0, q_1], 1) = ?$$

$$\delta([q_0, q_1], 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$$

$$= \{ q_1 \} \cup \{ q_0, q_1 \}$$

$$= [q_0, q_1]$$

$$= [q_0, q_1]$$

$$M = (\mathbb{Q}, \Sigma, \delta, q_0, F)$$

$$\mathbb{Q} = \{ q_0, q_1 \}$$

$$q_0 = q_0$$

$$\Sigma = \{ 0, 1 \} \quad F = \{ q_1, (q_0, q_1) \}$$

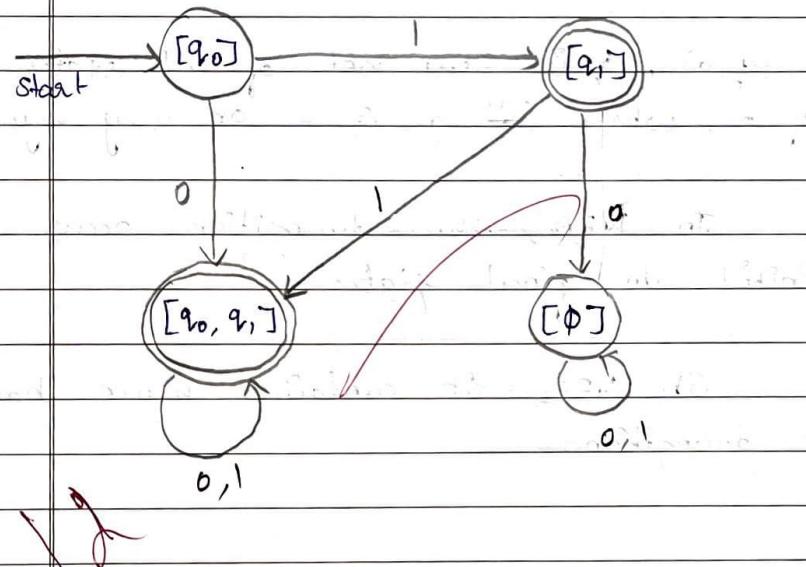
$$\delta = [q_0] \quad [q_0, q_1] \quad [q_1]$$

$$[q_1] \quad [\emptyset] \quad [q_0, q_1]$$

$$[q_0, q_1] \quad [q_0, q_1] \quad [q_0, q_1]$$

$$[\emptyset] \quad [\emptyset] \quad [\emptyset]$$

Transition Diagram:



12) b) Finite Automata with  $\epsilon$  transitions:-

~~Finite Non-Deterministic Finite Automata~~  
is defined with  $\epsilon$ -transition. is defined  
as

$$M = \{Q, \Sigma, \delta, q_0, F\}$$

$Q \Rightarrow$  set of finite state

$\Sigma \Rightarrow$  set of input string

$q_0 \Rightarrow$  initial state

$F \Rightarrow$  set of final state

$\delta \Rightarrow$  transition function mapping  
from  $Q \times (\Sigma \cup \{\epsilon\}) \rightarrow Q$

$(q, a)$ ,  $a$  is the set of all states  
of  $P$ , where  $a$  is  $\epsilon$  or any symbol

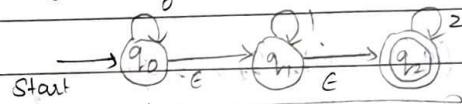
In NFA, the transition occur  
initial to final state.

In NFA, it contains more than  
1 transition.

$\epsilon$ -closure ( $q_0$ ):-

$\epsilon$ -closure ( $q_0$ ) is the set of  
vertices of  $P$  such that there is  
a path from  $q$  to  $p$ .

Eg:-



From the above transition,  
 $\epsilon$ -closure of  $q_0$  is  $\{q_0, q_1, q_2\}$

\* In this  $q_0$  is alone path for  
 $q_0$ , the move is from  $q_0$  to  $q_0$   
Therefore  $\epsilon$ -closure of  $q_0$  is  $\{q_0, q_1, q_2\}$   
as  $q_1$  &  $q_2$  is  $\epsilon$ -moves from  $q_0$  to  $q_1$  &  
 $q_0$  to  $q_2$ .

$q_1$  is in  $\epsilon$ -closure ( $q_0$ )

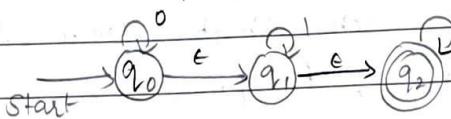
\*  $\epsilon$ -moves from  $q_0$  to  $q_1$  to  $q_2$   
 $q_2$  is in  $\epsilon$ -closure ( $q_0$ )



F 28/9/22

Example:-

Construct NFA with



$$M = (\{q_0, q_1, q_2\}, \{0, 1, 2\}, \delta, q_0, \{q_2\}).$$

$$\epsilon\text{-closure}(q_0) = \{q_0, q_1, q_2\}$$

$$\epsilon\text{-closure}(q_1) = \{q_1, q_2\}$$

$$\epsilon\text{-closure}(q_2) = \{q_2\}$$

$$\delta(q_0, 0) = \epsilon\text{-closure}(\delta(\delta(q_0, \epsilon), 0))$$

$$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_0), 0))$$

$$= \epsilon\text{-closure}(\delta(\{q_0, q_1, q_2\}, 0))$$

$$= \epsilon\text{-closure}(\delta(q_0, 0) \cup \delta(q_1, 0) \cup$$

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

$$= \epsilon\text{-closure}(\{q_0\} \cup \emptyset \cup \emptyset)$$

$$= \epsilon\text{-closure}(q_0)$$

$$= \{q_0, q_1, q_2\}$$

Reg. No : 950F2D12065

Additional Sheet

Page No : 3

$$\delta(q_0, 1) = \epsilon\text{-closure}(\delta(\delta(q_0, \epsilon), 1))$$

$$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_0), 1))$$

$$= \epsilon\text{-closure}(\delta(\{q_0, q_1, q_2\}, 1))$$

$$= \epsilon\text{-closure}(\delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1))$$

$$= \epsilon\text{-closure}(\emptyset \cup q_1 \cup \emptyset)$$

$$= \epsilon\text{-closure}(q_1)$$

$$= \{q_1, q_2\}$$

$$\delta(q_0, 2) = \epsilon\text{-closure}(\delta(\delta(q_0, \epsilon), 2))$$

$$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_0), 2))$$

$$= \epsilon\text{-closure}(\delta(\{q_0, q_1, q_2\}, 2))$$

$$= \epsilon\text{-closure}(\delta(q_0, 2) \cup \delta(q_1, 2) \cup \delta(q_2, 2))$$

$$= \epsilon\text{-closure}(\emptyset \cup \emptyset \cup q_2)$$

$$= \epsilon\text{-closure}(q_2)$$

$$= \{q_2\}$$

$$\delta(q_1, 0) = \epsilon\text{-closure}(\delta(\delta(q_1, \epsilon), 0))$$

$$= \epsilon\text{-closure}(\delta(\epsilon\text{-closure}(q_1), 0))$$

$$= \epsilon\text{-closure}(\delta(\{q_1, q_2\}, 0))$$

$$= \epsilon\text{-closure}(\delta(q_1, 0) \cup \delta(q_2, 0))$$

$$= \epsilon\text{-closure}(\emptyset \cup \emptyset)$$

$$= \emptyset$$

$$\begin{aligned}
 \delta(q_1, 1) &= \text{E-closure}(\delta(\delta(q_1, \epsilon), 1)) \\
 &= \text{E-closure}(\delta(\text{E-closure}(q_1), 1)) \\
 &= \text{E-closure}(\delta(q_1, q_2), 1) \\
 &= \text{E-closure}(\delta(q_1, 1) \cup \delta(q_2, 1)) \\
 &= \text{E-closure}(q_1 \cup \emptyset) \\
 &= \text{E-closure}(q_1) \\
 &= \{q_1, q_2\}
 \end{aligned}$$

$$\begin{aligned}
 \delta(q_1, 2) &= \text{E-closure}(\delta(\delta(q_1, \epsilon), 2)) \\
 &= \text{E-closure}(\delta(\text{E-closure}(q_1), 2)) \\
 &= \text{E-closure}(\delta(q_1, q_2), 2) \\
 &= \text{E-closure}(\delta(q_1, 2) \cup \delta(q_2, 2)) \\
 &= \text{E-closure}(\emptyset \cup q_2) \\
 &= \text{E-closure}(q_2) \\
 &= \{q_2\}
 \end{aligned}$$

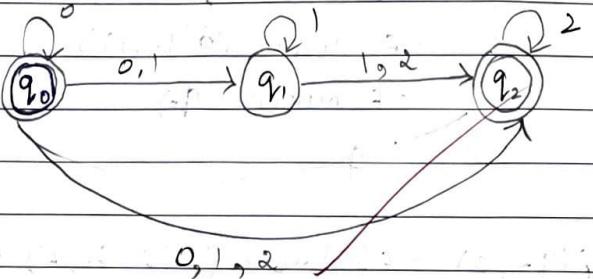
$$\begin{aligned}
 \delta(q_2, 0) &= \text{E-closure}(\delta(\delta(q_2, \epsilon), 0)) \\
 &= \text{E-closure}(\delta(\text{E-closure}(q_2), 0)) \\
 &= \text{E-closure}(\delta(q_2), 0) \\
 &= \text{E-closure}(\delta(q_2, 0)) \\
 &= \text{E-closure}(\emptyset) \\
 &= \emptyset
 \end{aligned}$$

$$\begin{aligned}
 \delta(q_2, 1) &= \text{E-closure}(\delta(\delta(q_2, \epsilon), 1)) \\
 &= \text{E-closure}(\delta(\text{E-closure}(q_2), 1)) \\
 &= \text{E-closure}(\delta(q_2, q_1), 1) \\
 &= \text{E-closure}(\delta(q_2, 1)) \\
 &= \text{E-closure}(q_2) \\
 &= \emptyset
 \end{aligned}$$

$$\begin{aligned}
 \delta(q_2, 2) &= \text{E-closure}(\delta(\delta(q_2, \epsilon), 2)) \\
 &\cancel{= \text{E-closure}(\delta(\text{E-closure}(q_2), 2))} \\
 &\cancel{= \text{E-closure}(\delta(q_2), 2)} \\
 &= \text{E-closure}(\delta(q_2, 2)) \\
 &= \text{E-closure}(q_2) \\
 &= \{q_2\}
 \end{aligned}$$

$f =$	0	1	2
$q_0$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
$q_1$	$\emptyset$	$\{q_1, q_2\}$	$\{q_2\}$
$q_2$	$\emptyset$	$\emptyset$	$\{q_2\}$

Transition Diagram:-



Reg. No : 95072012065

Additional Sheet

Page No : 4

15) a) Push Down Automata :-

- It contain input tape, finite control and stack.
- Stack has the string of symbols in any alphabet.

Definition:-

Push Down Automata is defined as

$$\{ M = (\Omega, \Sigma, \Gamma, f, q_0, z_0, F) \}$$

where  $\Omega \Rightarrow$  set of finite states

$\Sigma \Rightarrow$  is a alphabet called Input alphabet

$\Gamma \Rightarrow$  is a alphabet called stack alphabet

$f \Rightarrow$  transition function mapping from  $\Omega \times (\Sigma \cup \{z\}) \times \Gamma^* \rightarrow$  finite subset of  $\Omega \times \Gamma^*$

$q_0 \Rightarrow$  initial state

$z_0 \Rightarrow$  particular symbol, called start symbol.

$F \Rightarrow$  set of final state.

ID :-

- \* ID contains state, input, stack.
- \* ID is defined as 3 tuples.

$$(q, w, U)$$

where  $q \Rightarrow$  state of finite control.

$w \Rightarrow$  string of Input state.

$U \Rightarrow$  string of stack.

These are 2 types of moves in

Push Down Automata:-

\* In first type of move, input string is needed. Depending on the input string, state of finite control, top of a stack symbol, which has to choose input string.

\* Each choice has a finite control which has to move next.

\* Input head is incremented

$$(q, a, z) = (P_1, \gamma_1) (P_2, \gamma_2) (P_3, \gamma_3), \dots (P_m, \gamma_m)$$

\* In second type of move is called e-moves, in this type,

the input string is not needed and the input head need not to be increment.

$$(q, \epsilon, z) = (P_1, \gamma_1) (P_2, \gamma_2), \dots (P_m, \gamma_m)$$

Language Accepted by Push Down Automata:-

There are 2 types of language accepted by Push Down Automata.

\* By empty stack.

language accepted by PDA is to be the set of all strings in which the sequence make a PDA into empty stack.

$$L(M) = \{ w \mid (q_0, w, z_0) \xrightarrow{*} (P, \epsilon, \epsilon) \}$$

\* By final state.

language accepted by PDA is to be the set of all strings which the sequence make a PDA enter into final state.

$$L(M) = \{ w \mid (q_0, w, z_0) \xrightarrow{*} (P, \epsilon, \delta), \text{ for } \delta \in F \}$$



Language accepted by Push Down Automata is context free grammar.

language.

Context free grammar:

Context free grammar  $G_1$  is defined as  $(V, T, P, S)$

where  $V \Rightarrow$  set of non-terminals or

variables

$T \Rightarrow$  set of terminals

$P \Rightarrow$  set of production in

the form of  $A \rightarrow d$ .

where  $A \rightarrow$  is a variable.

$d \Rightarrow$  strings of terminals  
and non-terminals.

$S \Rightarrow$  special symbol called  
start symbol.

Eg:  $G_1 = (S, F, \{, \}, +, *, (, ), id, \{, \}, P, \epsilon)$

where  $P$  consists of:

$$F \rightarrow E + E$$

$$E \rightarrow E * E$$

$$E \rightarrow (E)$$

$$E \rightarrow id$$

Reg. No : 15DFT2012065

Additional Sheet

Page No : 5

Context Free Grammar consist of 1 types of grammar.

In that Type 3 grammar is accepted by push down automata.

The production of the form

$$A \rightarrow d$$

where  $A$  is a variable.

$d$  is string of terminal  
and non-terminal.

✓ Context free language is derived  
from context free grammar.