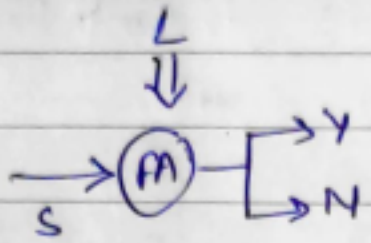
Unit - 1

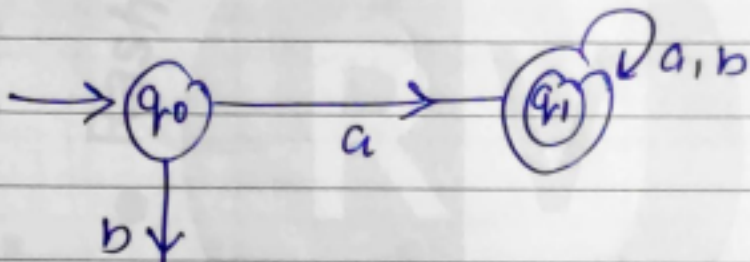
JANUARY 2017 SUN - 1, MON - 2, & TUE - 3

FA (Finite Automata)



Q① L_1 = set of all strings which starts with a.

= {a, aa, ab, aaa - - - }



String: aab

$q_0 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1$ final state

accepted.

String : bba

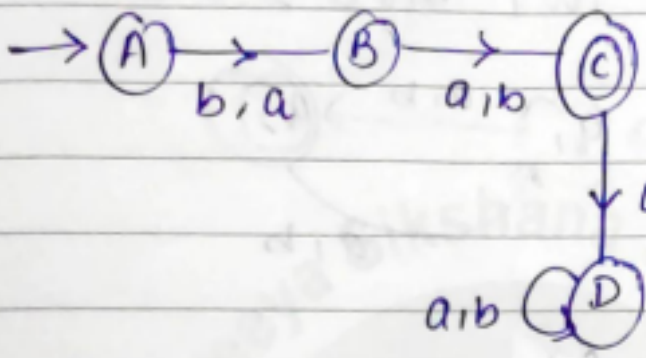
$q_0 \rightarrow q_2 \rightarrow q_2 \rightarrow q_2$ non final state

not accepted.

Q② Construct a DFA which accepts set of all strings over $\Sigma = \{a, b\}$ of length 2.

Note:- A FA is said to accept a language if all the strings in the language are accepted and all the strings not in the language are rejected

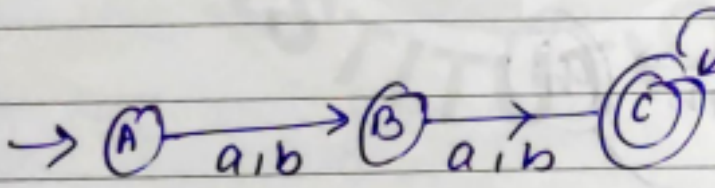
$L = \{aa, ab, ba, bb\}$



Once we have reached D, there is no way to come back to C.

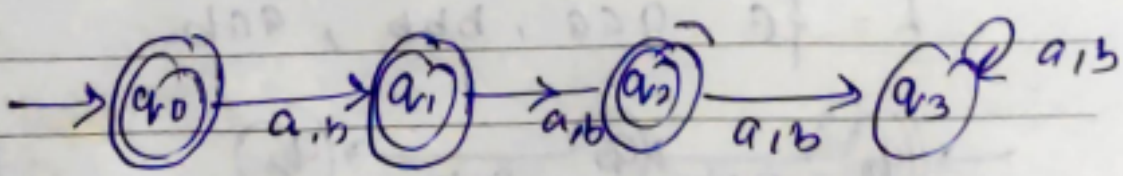
Q. (3) $L = \{w \in \{a,b\} : |w| \geq 2\}$

$L = \{aa, ab, ba, bb, aaa, bbb \ldots\}$

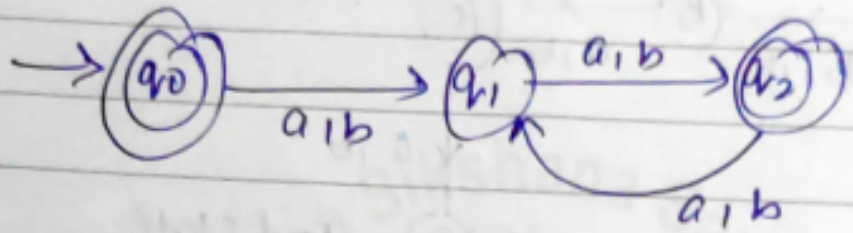


Q. (4) DFA: $w \in \{a,b\}$ $|w| \leq 2$

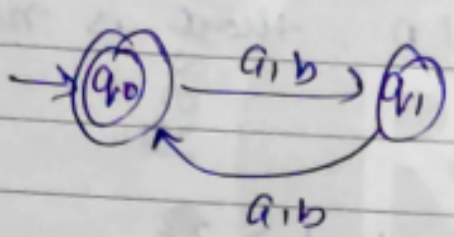
$L = \{\epsilon, a, b, aa, bb, ab, ba\}$



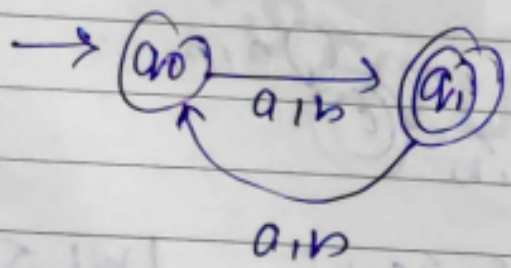
Q. (5) Set of all strings over $\Sigma = \{a, b\}$, where $|w| \bmod 2 = 0$



Or

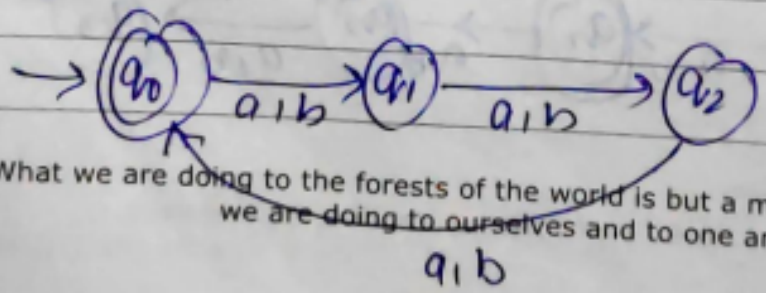


Q. (6) $|w| \bmod 2 = 1$ String length is odd

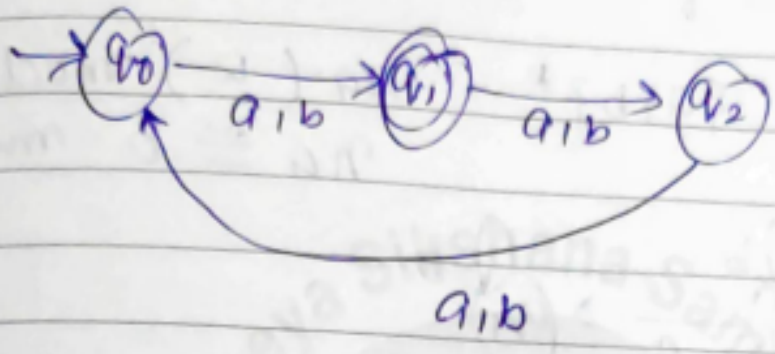


Q. (7) $|w| \bmod 3 = 0$

$L = \{\epsilon, aaa, bbb, aab \ldots\}$



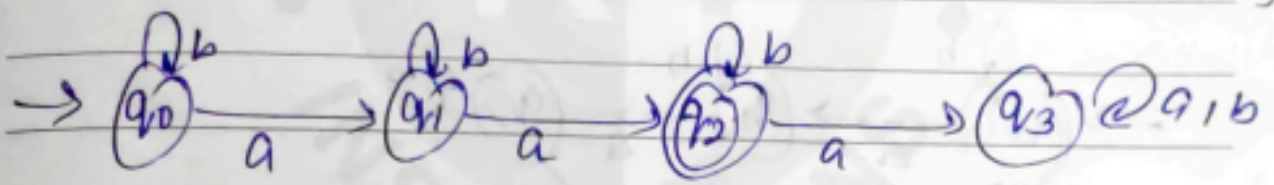
Q.(8) $|w| = 1 \bmod 3$ or $|w| \bmod 3 = 1$



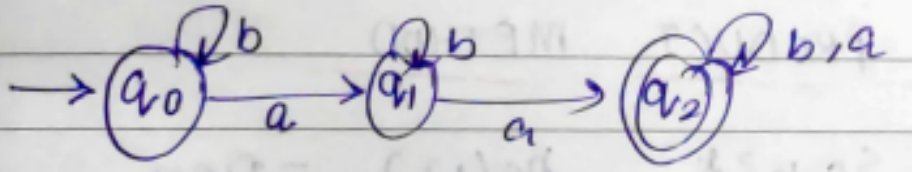
Q.(9) $w \in \{a,b\}^*$ No. of a's = 2

$n_a(w) = 2$

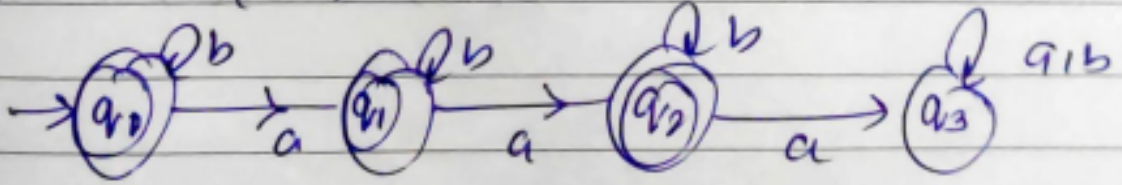
$L = \{aa, baa, aba, aab, bbaa \cdots\}$



Q.(10) $n_a(w) \geq 2$



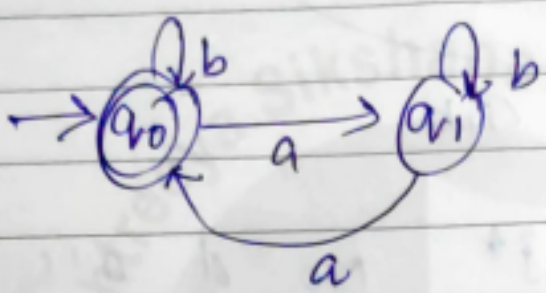
Q.(11) $n_a(w) \leq 2$



Q. (12) no. of a's is even

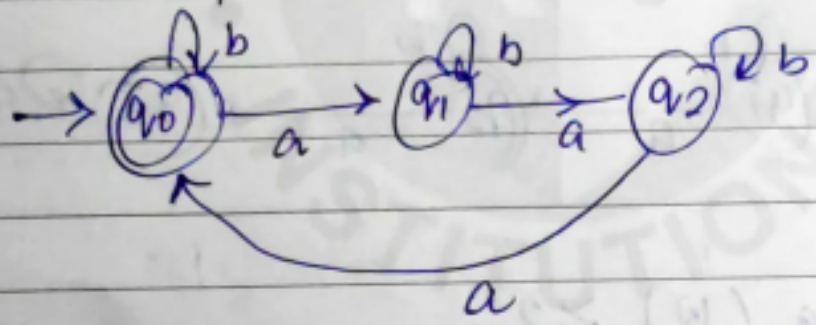
$w \in \{a,b\}^*$ $n_a(w) \bmod 2 = 0$

$n_a \cong 0 \bmod 2$



Q. (13) $n_a(w) \bmod 3 = 0$, $|w| \in \{a,b\}^*$

$L = \{\epsilon, aaa, aaab, baaa, bbaaa, \ldots\}$

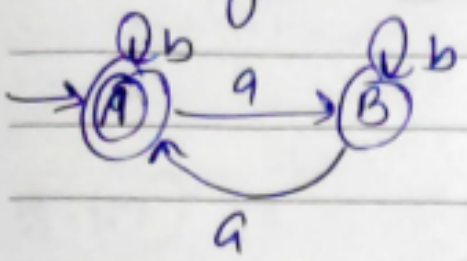


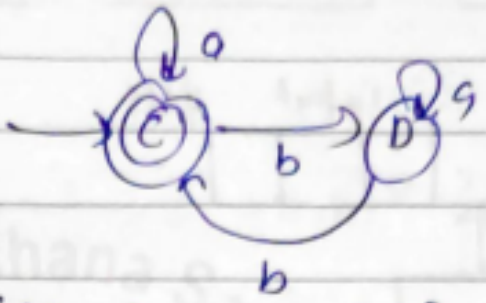
CROSS PRODUCT METHOD

$\Sigma = \{a,b\}^*$ $n_a(w)$ = Even

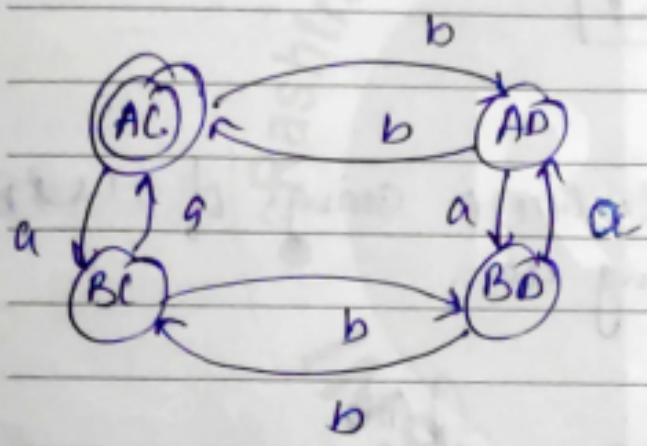
$n_b(w)$ = Even.

Counting a's

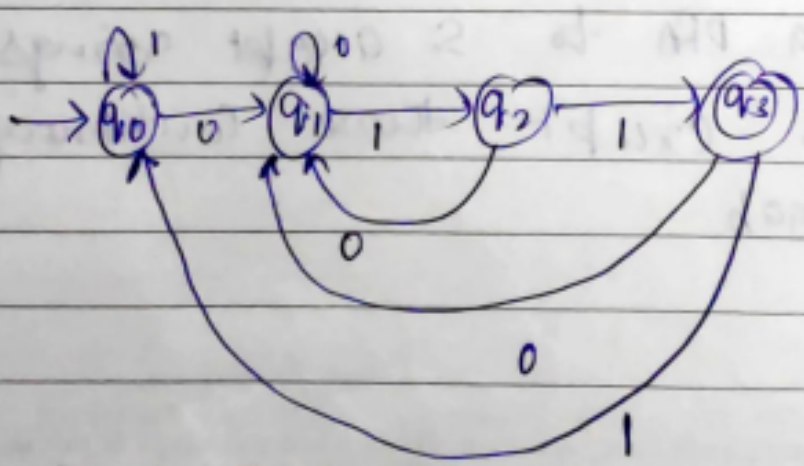




$\{A \times B\} \times \{C \times D\} \Rightarrow \{AC, AD, BC, BD\}$



Q(14) Draw a DFA to accept string of 0's & 1's ending with the string 011.

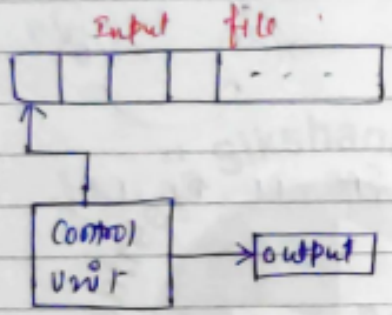


→ 0 1 0 1 1

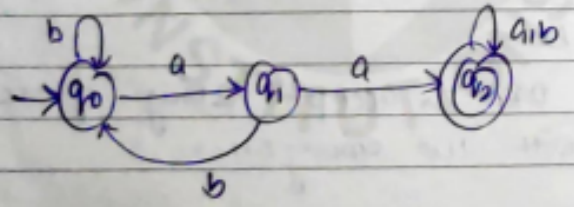
→ 0 1 1 1 1 1 1 0 1 1

→ 0 1 1 0 0 0 0 1 1

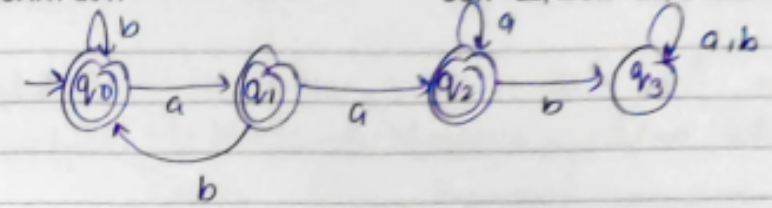
What is Finite Automata: →



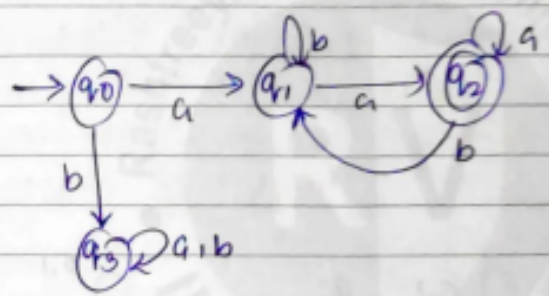
Q. (15) Obtain a DFA to accept strings of a's & b's having a substring aa.



Q. (16) Obtain a DFA to accept strings of a's & b's except those containing the substring aab.

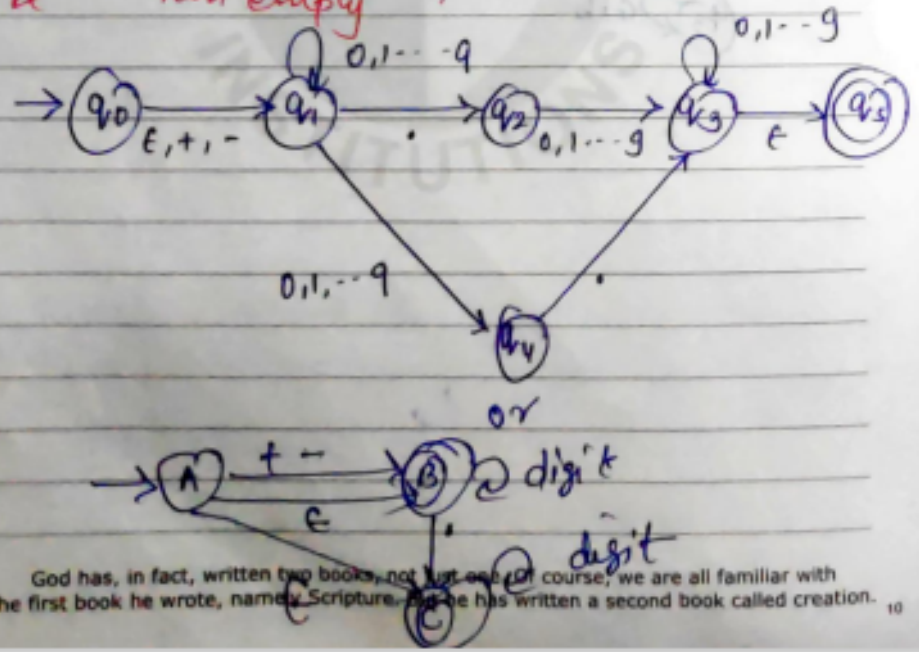


Q. (17) $L = \{awa \mid w \in \{a,b\}^*\}$



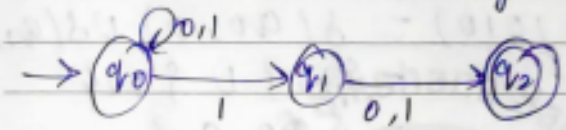
In fig 2.18 is an ε-NFA that accepts decimal numbers consisting of

1. An optional + or - sign
2. A string of digits.
3. A decimal point, and
4. Another string of digits, Either this string of digits, or the string can be empty, but at least one of the two strings of digits must be non empty.

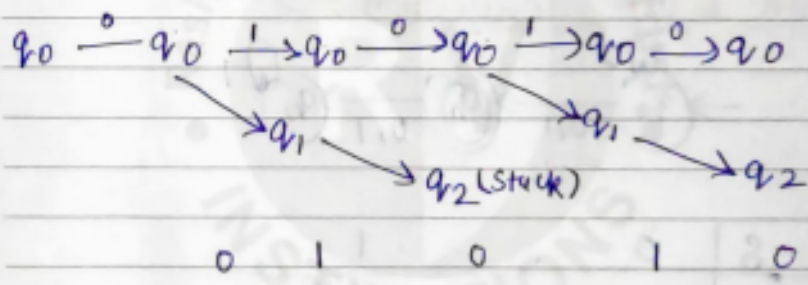


NFA

following FA shows, it accepts '1' as second last symbol in string.



Consider example string 01010



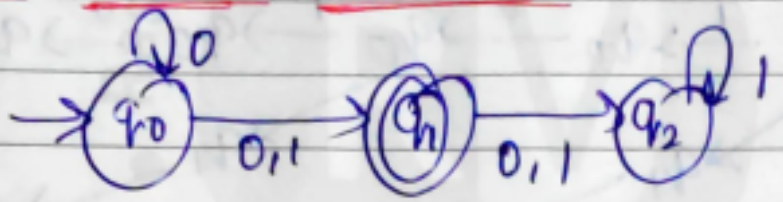
Extended transition function of 01010

1. $\hat{\delta}(q_0, \epsilon) = \{q_0\}$
2. $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0\}$
3. $\hat{\delta}(q_0, 01) = \delta(q_0, 1) = \{q_0, q_1\}$
3. $\hat{\delta}(q_0, 010) = \delta(q_0, 0) \cup \delta(q_1, 0)$
   $= \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

5. $\hat{\delta}(q_0, 0101) = \delta(q_0, 1) \cup \delta(q_2, 1)$
$= \{q_0, q_1\} \cup \phi = \{q_0, q_1\}$

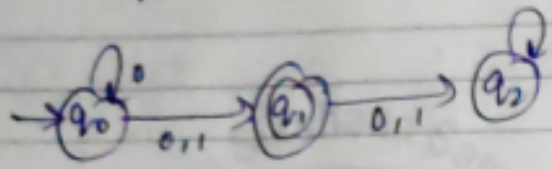
6. $\hat{\delta}(q_0, 01010) = \delta(q_0, 0) \cup \delta(q_1, 0)$
$= \{q_0\} \cup \{q_2\}$
$= \{q_0, q_2\}$

NFA to DFA Conversion :→

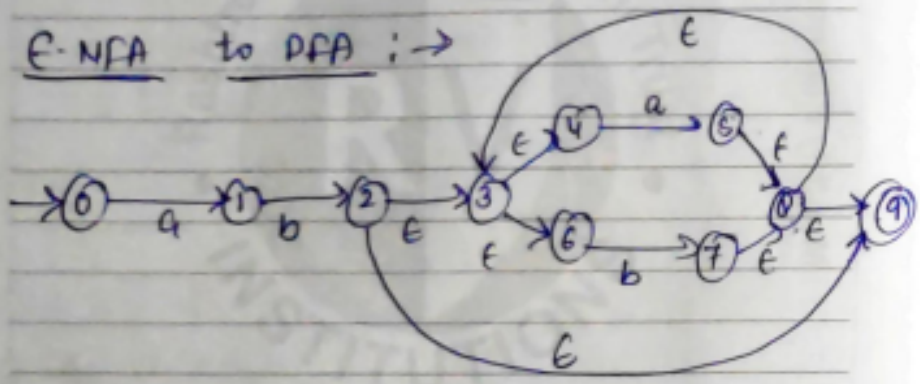


	δ	0	1
→	$[q_0]$	$[q_0, q_1]$	$[q_1]$
*	$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_1, q_2]$
*	$[q_1]$	$[q_2]$	$[q_2]$
*	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$	$[q_1, q_2]$
*	$[q_1, q_2]$	$[q_2]$	$[q_2]$
	$[q_2]$	ϕ	$[q_2]$

Q. Convert the following NFA into an equivalent DFA



ε-NFA to DFA :→



Soln Start State of DFA q_0 D = ε-closure (0)

= {0} – A

δ (0,a) = ε-closure (δN (0,a))
= ε-closure (1)
= {1} – B

δ (0,b) = ε-closure (δN (0,b))
= ε-closure (ε)
= ϕ

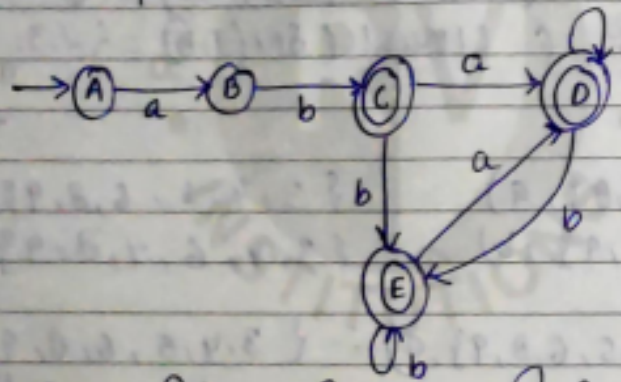
δ (1,a) = ε-closure (δN (1,a)) = ϕ
δ (1,b) = ε-closure (δN (1,b)) = {2,3,6,4,9} – C

δ ({2,3,4,6,9}, a) = {3,4,5,6,8,9} – D
δ ({2,3,4,6,9}, b) = {3,4,6,7,8,9} – E

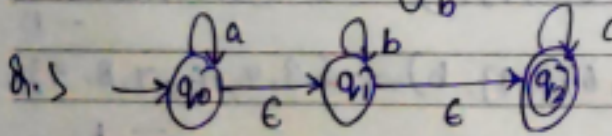
δ ({3,4,5,6,8,9}, a) = {3,4,5,6,8,9} – D
δ ({3,4,5,6,8,9}, b) = {3,4,6,7,8,9} – E

δ ({3,4,6,7,8,9}, a) = {3,4,5,6,8,9} – D
δ ({3,4,6,7,8,9}, b) = {3,4,6,7,8,9} – E

δ	a	b
A	b	-
B	-	C
* C	D	E
* D	D	E
* E	D	E



8.)



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

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

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

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

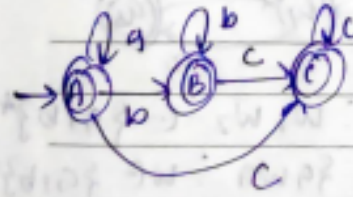
$\delta(\{q_1, q_2\}, a) = \phi$

$\delta(\{q_1, q_2\}, b) = \epsilon\text{-closure}(\{q_1\})$
$= \{q_1, q_2\}$ — B

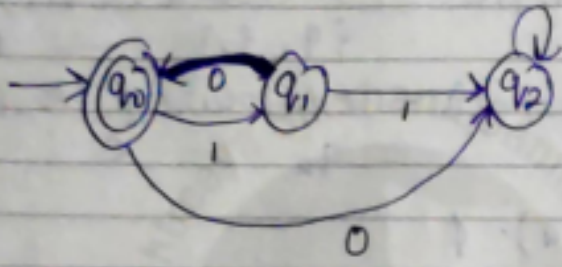
$\delta(\{q_1, q_2\}, c) = \epsilon\text{-closure}(\{q_2\})$
$= q_2$ — C

$\delta(q_2, a) = \phi$
$\delta(q_2, b) = \phi$
$\delta(q_2, c) = q_2$

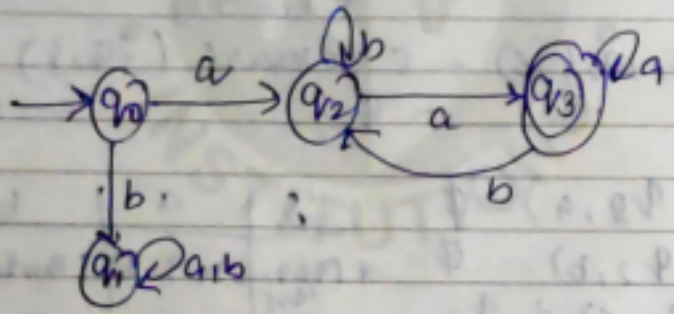
δ	a	b	c
* $[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$	$[q_1, q_2]$	$[q_2]$
* $[q_1, q_2]$	ϕ	$[q_1, q_2]$	$[q_2]$
* q_2	ϕ	ϕ	$[q_2]$



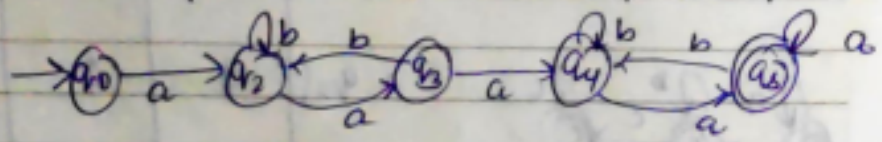
8. Construct DFA for language
$L = \{(10)^n : n \geq 0\}$



8. $L = \{aWa : W \in \{a,b\}^*\}$

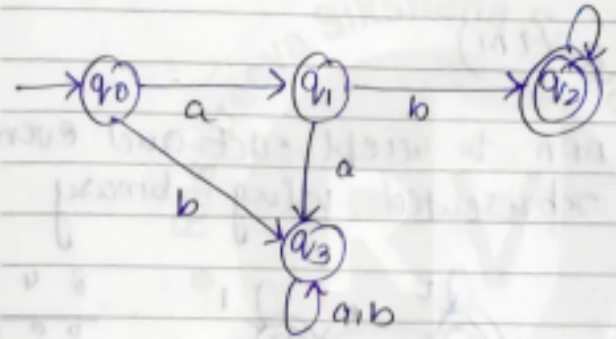


8. $L = \{aW_1aaW_2a : W_1, W_2 \in \{a,b\}^*\}$



$L^2 = \{aW_1aaW_2a : W_1W_2 \in \{a,b\}^*\}$
to $L = \{aWa : W \in \{a,b\}^*\}$

8. Find a deterministic finite automata that recognises the set of all strings on $\Sigma = \{a,b\}$ starting with the prefix ab.



Languages and DFA :→

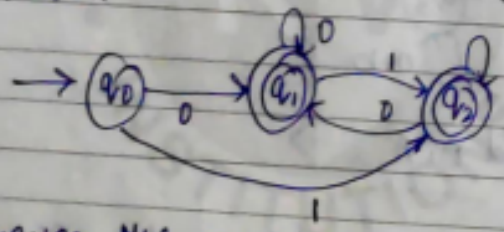
The lang accepted by a dfa $M = (Q, \Sigma, \delta, q_0, F)$ is the set of all languages on Σ accepted by M. In formal notation.

$$L(M) = \{W \in \Sigma^* : \hat{\delta}(q_0, W) \in F\}$$

→ " A language L is called regular if and only if there exists some deterministic finite accepter M such that

$L = L(M)$

Q. Obtain a DFA to accept odd and even numbers represented using binary notation

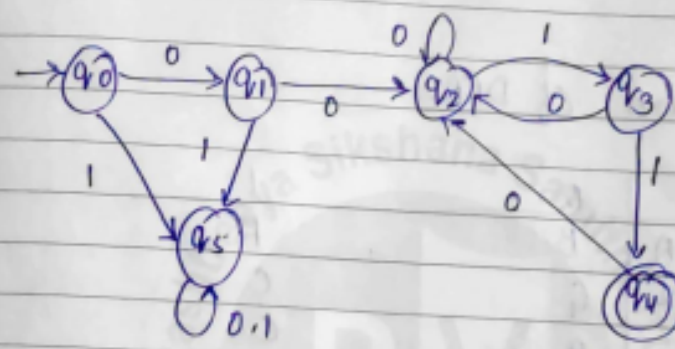


8	4	2	1
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1

Whenever the machine is in state q_1, even numbers are accepted and whenever the machine is in q_2, odd numbers are accepted.

Q. Obtain a DFA to accept strings of 0's and 1's starting with at least

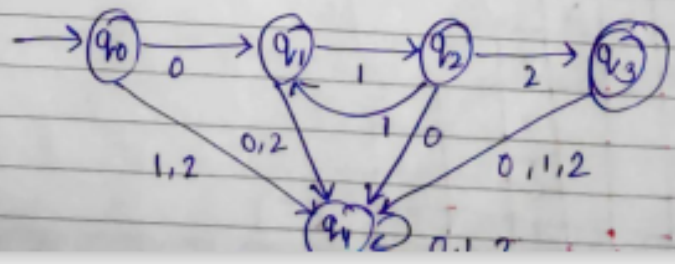
two 0's and ending with at least two 1's.



00110 11 ✓
00101 ✗

$L = \{ w \mid w \in 00(0+1)^*11 \}$

Q. Obtain a DFA to accept strings of 0's, 1's and 2's beginning with a '0' followed by odd number of 1's and ending with a '2'.

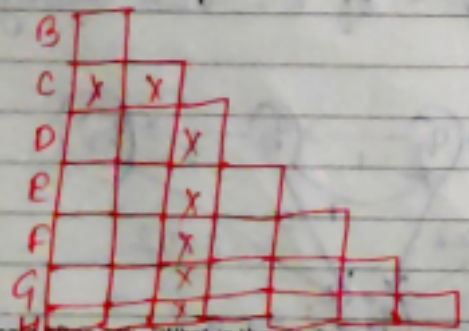


$\{w \mid w \in 0(11)^* 12\}$

Minimisation of DFA:-

8.

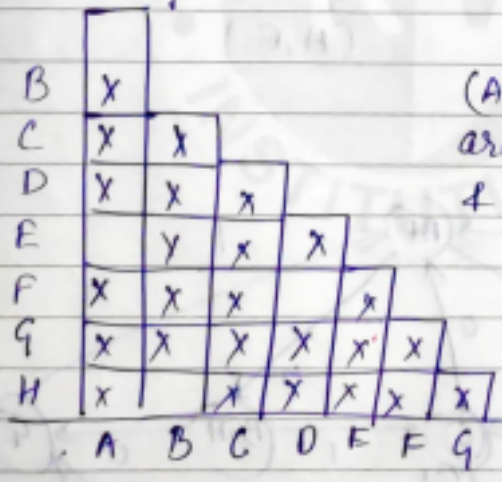
	0	1
→A	B	F
B	G	C
C	A	C
D	C	G
E	H	F
F	C	G
G	G	E
H	G	C



Σ

8.

δ	0	1
→A	B	F
B	G	C
* C	A	C
D	C	G
E	H	F
F	C	G
G	G	E
H	G	C



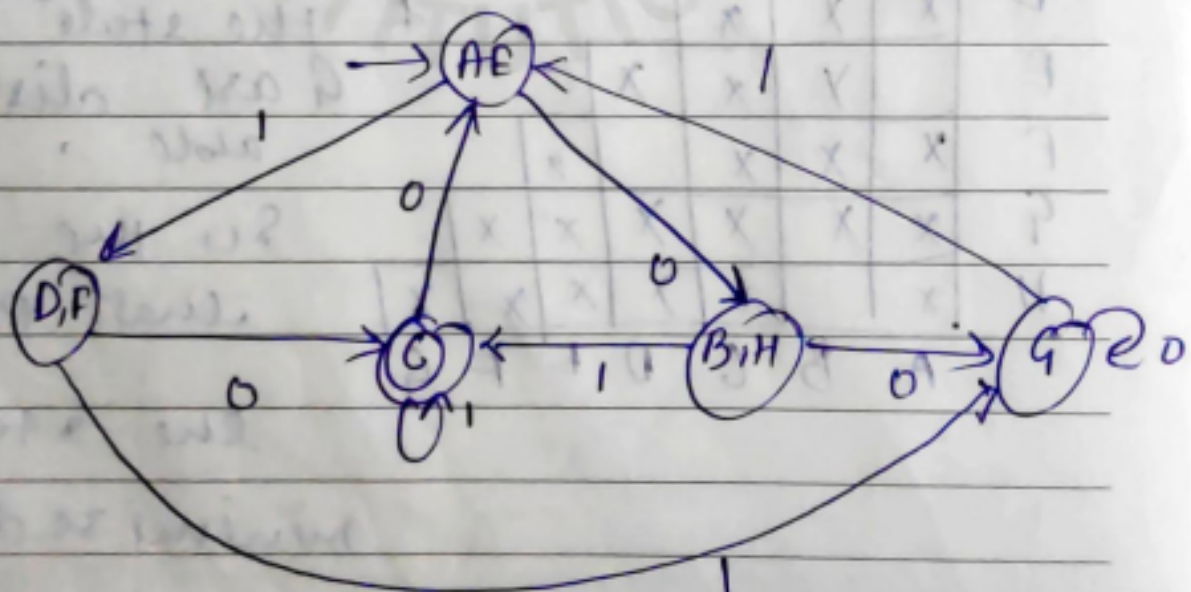
The Unmarked pairs (A,E) (B,H) & (D,F) are indistinguishable & the state C & G are distinguishable.

So the groups that present the states of minimized DFA are:

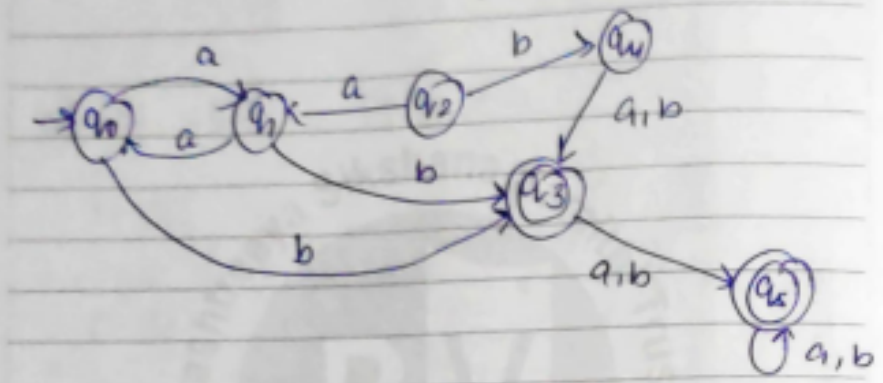
(A,E) , (B,H) , C , (D,F) , G

Minimized Transition table :-

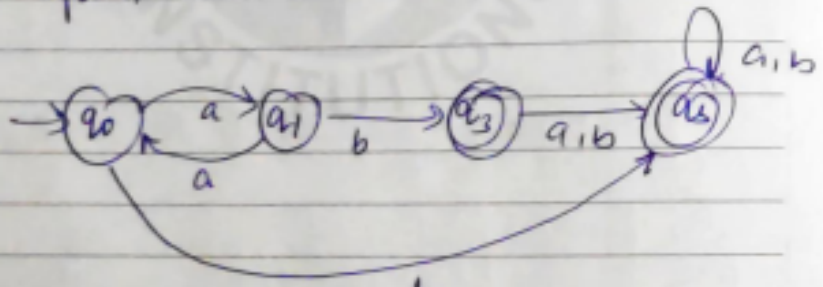
States	0	1
→ (A,E)	(B,H)	(D,F)
(B,H)	G	C
* C	(A,E)	C
(D,F)	C	G
G	G	(A,E)



8. Minimize the following DFA.



Sol:→ q_2 & q_4 are not reachable from start state.



	q_0	q_1	q_3
q_1			
q_3	X	X	
q_5	X	X	

(q_0, q_1) and (q_3, q_5) are indistinguishable.

∈

States	a	b
→(q_0,q_1)	(q_0,q_1)	(q_3,q_5)
*(q_3,q_5)	(q_3,q_5)	(q_3,q_5)

	q_0	q_1	q_3
q_1	—		X
q_3	X	X	X
q_5	X	X	X

Unit 2

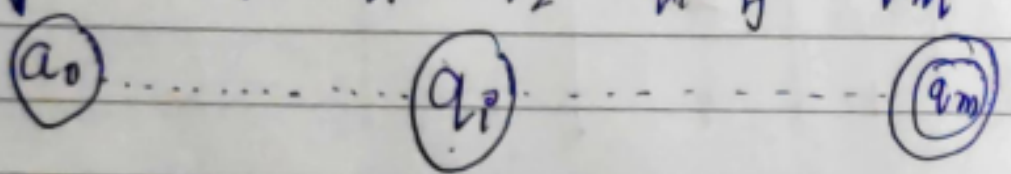
Pumping lemma for Non Regular languages

Proof :-

Let L be a regular set, then there is a constant n such that if z is any word in L, and $|z| > n$, we may write $z = uvw$ in such a way that $|uv| \leq n$, $|v| \geq 1$, and for all $i \geq 0$, $uv^i w$ is in L. Furthermore, n is no greater than the number of states of the smallest DFS accepting L.

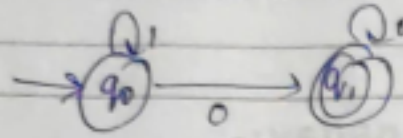
$z = a_1 a_2 \cdots a_m \quad m > n$

$q_0 \; q_1 \; q_2 \; q_i \; q_j \; q_m$

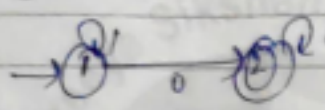


Suppose DFA has n states.

Q. Obtain a regular expression for the FA shown below.



Sol:-



Basis: $K=0$

$R_{11}^{(0)} = \epsilon + 1$

$R_{12}^{(0)} = 0$

$R_{21}^{(0)} = \phi$

$R_{22}^{(0)} = \epsilon + 0 + 1$

Induction:-

$R_{ij}^{(K)} = R_{ij}^{(K-1)} + R_{ik}^{(K-1)} \left[R_{kk}^{(K-1)}\right]^* R_{kj}^{(K-1)}$

$K=1$

$R_{11}^{(1)} = R_{11}^{(0)} + R_{11}^{(0)} \left[R_{11}^{(0)}\right]^* R_{11}^{(0)}$

$= (\epsilon+1) + (\epsilon+1)(\epsilon+1)^* (\epsilon+1)$

$= (\epsilon+1) + (\epsilon+1)1^* (\epsilon+1)$

$= (\epsilon+1) + 1^* = 1^*$

$R_{12}^{(1)} = R_{12}^{(0)} + R_{11}^{(0)} \left[R_{11}^{(0)}\right]^* R_{12}^{(0)}$

$= 0 + (\epsilon+1)(\epsilon+1)^* 0 = 0 + 1^* 0$

$= 1^* 0$

$R_{21}^{(1)} = R_{21}^{(0)} + R_{21}^{(0)} \left[R_{11}^{(0)}\right]^* R_{11}^{(0)}$

$= \phi + \phi(\epsilon+1)^*(\epsilon+1)$

$= \phi$

$R_{22}^{(1)} = R_{22}^{(0)} + R_{21}^{(0)} \left[R_{11}^{(0)}\right]^* R_{12}^{(0)}$

$= (\epsilon+0+1) + \phi(\epsilon+1)^* 0$

$= (\epsilon+0+1)$

$K=2$ $R_{11}^{(2)} = R_{11}^{(1)} + R_{12}^{(1)} \left[R_{22}^{(1)}\right]^* R_{21}^{(1)}$

$= 1^* + 1^* 0 (\epsilon+0+1)^* \phi$

$= 1^*$

$R_{12}^{(2)} = R_{12}^{(1)} + R_{12}^{(1)} \left[R_{22}^{(1)}\right]^* R_{22}^{(1)}$

$= 1^*0 + 1^*0 (\epsilon+0+1)^* (\epsilon+0+1)$

$= 1^*0 + 1^*0 (0+1)^*$

$= 1^*0 (0+1)^*$

$R_{21}^{(2)} = R_{21}^{(1)} + R_{22}^{(1)} \left[R_{22}^{(1)}\right]^* R_{21}^{(1)}$

$= \phi + (\epsilon + 0 + 1)(\epsilon + 0 + 1)^* \phi$

$= \phi$

$R_{22}^{(2)} = R_{22}^{(1)} + R_{22}^{(1)} [R_{22}^{(1)}]^* R_{22}^{(1)}$

$= (\epsilon + 0 + 1) + (\epsilon + 0 + 1)(\epsilon + 0 + 1)^* (\epsilon + 0 + 1)$

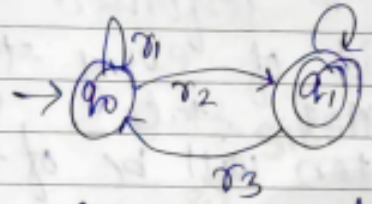
$= (0 + 1)^*$

Note: - Since the total no. of states in the given machine is 2, maximum value of K should be 2

Since the start state is 1 & final state is 2, the regular expression is given by

$$\boxed{R_{12}^{2} = 1^*0(0+1)^*}$$

Obtain a RE from FA (by eliminating states) :→



Diag 1: Generalized transition graph.

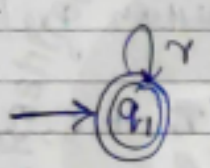
$$r = r_1^* r_2 (r_4 + r_3 r_1^* r_2)^* \quad \text{— (1)}$$

Note:-

1. for each final state q, apply the reduction procedure & bring the graph to the form shown in the diag 1

2. If q is not the start state reduce graph obtained will be of the form shown in diag 1. & use eqn (1) to obtain the regular expression.

3. If the start state is also an accepting state, the state elimination process has to be performed so that we would get rid of every state except the start state. The final automaton will be of the form



4. The final regular expression is the sum of all the regular expressions obtained from the reduced automata for each accepting state.

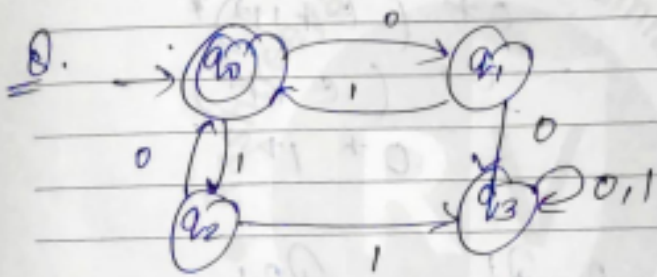
for eg:- If r_3 is not there in diag 1. the regular expression

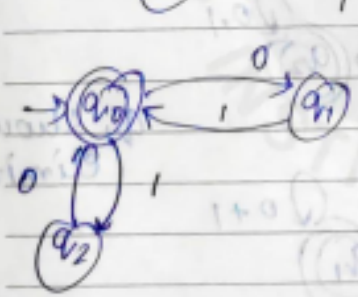
$$r = r_1^* r_2 r_4^*$$

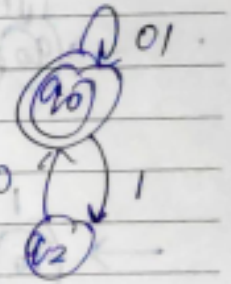
If q_0 & q_1 are the final states, then the regular expression can be of the form

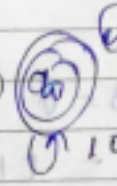
$$r = r_1^* + r_1^* r_2 r_4^*$$

Q.

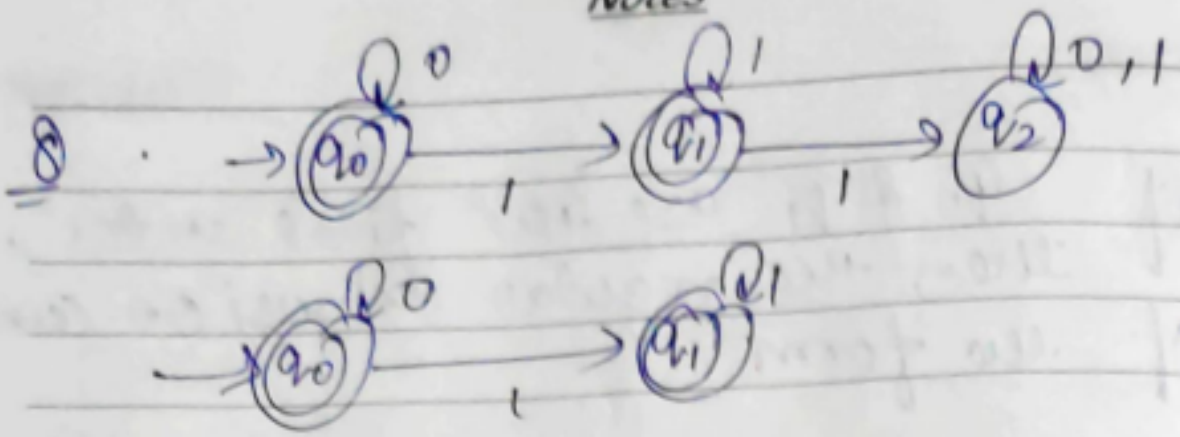








$= (01+10)^*$

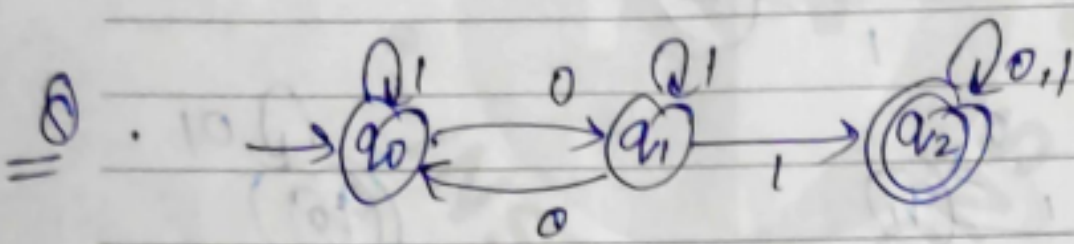


R.E. $0^* + 0^*11^*$

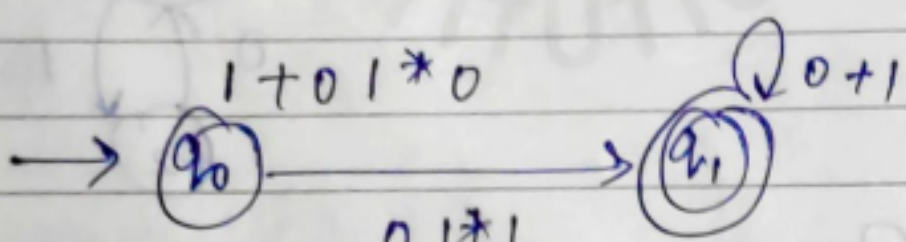
$= 0^*(\epsilon + 11^*)$

$= 0^*(\epsilon + 1^+)$

$= 0^*1^+$



q_1 should be eliminated



$r_1 = 1 + 01^*0$

$r_2 = 01^*1$

$r_3 = \phi$

$r_4 = 0+1$

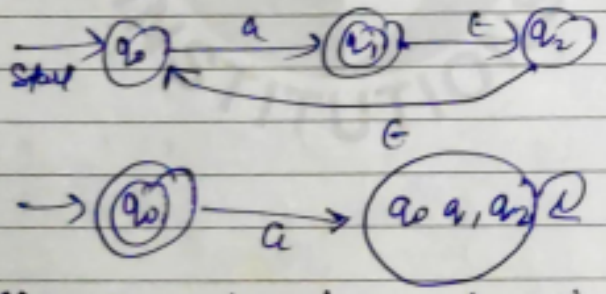
$r = (1+01*0)^* 01^* 1 [(0+1) + \phi$

$(1+01*0)^* 01^* 1]^*$

$= (1+01*0)^* 01^*1[(0+1)+\phi]^*$

$= (1+01*0)^* 01^*1(0+1)^*$

Q. Let $\Sigma = \{a\}$ what is the complement of the language accepted by the NFA shown below?



Q. Give the transition function for DFA & ε-NFA

DFA $\delta: Q \times \Sigma \to Q$

ε-NFA $\delta: Q \times \Sigma \cup \{\epsilon\} \to 2^Q$

Q. Find a string of minimum length in $\{a,b\}^*$ NOT in the language $a^*b^*(ba)^*a^*$

Sol bab

Q. Obtain a regular expression for the following:

i) $L = \{a^{2n} b^{2m} \mid n \geq 0, m > 0\}$

ii) To accept strings of a's & b's ending with b & has no substring aa.

iii) To accept set of all strings that do not end with 01, over $\{0,1\}^*$

Sol:- i) $(aa)^*(bb)^*$

ii) $(ab+b)^*(b+ab)$

iii) $((0+1)^*(0+11)) + 11 \epsilon$

Unit-3

Simplification of CFG

FRI - 7, SAT - 8, & SUN - 9

APRIL 2017

Eliminating useless Symbols :-

Rule :- a) Remove non-generating symbols
b) Remove non-reachable symbol

Eg. (1) $S \to AB/a$
$A \to b$

Sol :- B is non generating so $S \to AB$ will be eliminated

$S \to a$
$A \to b$

A is not reachable from S. So $A \to b$ will be eliminated.

$S \to a$

APRIL 2017

MON - 10, TUE - 11, & WED - 12

Removal of ϵ - Production

Eg :- $S \to aA$
$A \to b/\epsilon$

Rule :-

If $A \to \epsilon$ is a production to be eliminated then we look for all productions, whose right side contains A, and replace each occurrence of A in each of these productions to obtain the ϵ - productions. Now these resultant non ϵ - production must be added to the grammar to keep the language generated, the same.

Sol :- $S \to aA/a$
$A \to b$

Eg (2)

$S \to ABAC$
$A \to aA/\epsilon$
$B \to bB/\epsilon$
$C \to c$

Sol:-

$S \to ABAC / BAC / ABC / BC / AAC / AC / C$
$A \to aA/a$
$B \to bB/b$
$C \to c$

Eliminating Unit Productions

$A \to B$

Where $A, B \in V$ is a Unit-Production

Eg (1)

$A \to B$
$B \to GB/b$

In this $A \to B$ is Unit Production

Since B is generated from A, whatever is generated by B, the same things can be generated from A also

So we can have

$A \to aB$
$A \to b$

$A \to aB/b$
$B \to aB/b$ } Ans

Eg (3)

$S \to AB$
$A \to a$
$B \to C/b$
$C \to D$
$D \to E/bC$
$E \to d/Ab$

S → AB
A → a
B → b/d / Ab/bC
C → bC / d/Ab
D → bC / d/Ab
E → d/ Ab

CNF

A → BC
A → a.

Where A, B & C ∈ V and a ∈ T.

Q (1) S → a/b/CSS
= (2) S → abSb / a / aAb
A → bS / aAAb

Useless Symbol

(1) S → aB / bX

A → BAd / bSX / a
B → aSB / bBX
X → SBD / aBX / ad

(2) A → xyz / Xyzz
X → Xz / x Yz
Y → y Yy / Xz
Z → Zy / z

GNF

S → AA/0
A → SS/1

Sol:- $A_1 \to 0A_1A_2 / 1A_2 / 0A_1ZA_2 / 1ZA_2 / 0$
$A_2 \to 0A_1 / 1 / 0A_1Z / 1Z$
$Z \to 0A_1A_1 / 1A_1 / 0A_1ZA_1 / 1ZA_1$
$Z \to 0A_1A_1Z / 1A_1Z / 0A_1ZA_1Z / 1ZA_1Z$

1) Write a CFG for lang $L = \{a^{2n} b^m / n>0, m \geq 0\}$

$S \rightarrow CaAB$
$A \rightarrow aaA/\epsilon$
$B \rightarrow bB/\epsilon$

2) Write a CFG for

$L = \{a^n b^n c^m d^m / n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n / n \geq 1, m \geq 1\}$

P_1: $S_1 \rightarrow AB$
$A \rightarrow aAb / ab$
$B \rightarrow cBd / cd$

P_2: $S_2 \rightarrow aDd$
$D \rightarrow aDd / E$
$E \rightarrow bEc / bc$

$\therefore P$: $S \rightarrow S_1 / S_2$

$S_1 \rightarrow AB$
$A \rightarrow aAb/ab$
$B \rightarrow cBd/cd$
$S_2 \rightarrow DE$
$D \rightarrow aDd/E$
$E \rightarrow bEc/bc$

3) Design a CFG for the language

$L = \{(0^n 1^n / n>0) \cup (1^n 0^n / n>0)\}$

$L_1 = \{0^n 1^n / n \geq 0\}$

P_1: $S_1 \rightarrow 0S_1 1 / \epsilon$

$L_2 = \{1^n 0^n / n \geq 0\}$
P_2: $S_2 \rightarrow 1S_2 0 / \epsilon$

$\therefore$ Production of G
$S \rightarrow S_1 / S_2$
$S_1 \rightarrow 0S_1 1 / \epsilon$
$S_2 \rightarrow 1S_2 0 / \epsilon$

4.) Design a CFG for the language

$L(G) = \{ab(bbaa)^n bba(ba)^n : n \geq 0\}$

P: $S \to abX$
$X \to bbYa$
$Y \to aaXb$
$Y \to \epsilon$

OR

P: $S \to abX$
$X \to bbaaXba$
$X \to bba$.

5.) Let $L = \{a^n b^n : n \geq 0\}$
L^2 is Context-free.

$L = \{a^n b^n \mid n \geq 0\}$
P: $S \to aSb \mid \epsilon$

$S_1 \rightarrow SS$

$S \rightarrow aSb \mid \epsilon$

6.) Write a CFG for the language

$L = \{a^n b^m c^{2m} \mid n, m \geq 0\}$

$S \rightarrow AB$

$A \rightarrow aA \mid \epsilon$

$B \rightarrow bBcc \mid \epsilon$

7.) Find the CFG for the following language.

$L = \{a^n b^{2n} c^m \mid n, m \geq 0\}$

$S \rightarrow AB$

$A \rightarrow aAbb \mid \epsilon$

$B \rightarrow cB \mid \epsilon$

Unit-4

Push down Automata

$\delta(q_0, a, z_0) = (q_0, az_0)$

$\delta(q_0, b, z_0) = (q_0, bz_0)$

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

$\delta(q_0, b, a) = (q_0, ba)$

$\delta(q_0, b, b) = (q_0, bb)$

$\delta(q_0, c, z_0) = (q_1, z_0)$

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

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

$\delta(q_1, a, a) = (q_1, \epsilon)$

$\delta(q_1, b, b) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, z_0) = (q_2, z_0)$

$\delta: Q \times (\Sigma \cup \epsilon) \times \Gamma \longrightarrow Q \times \Gamma^*$

$M = (Q \times \Sigma, \Gamma, \delta, q_0, z_0, F)$

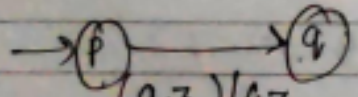
Σ = set of i/p alphabet
Γ = set of stack alphabets.
δ = transition from

$Q \times (\Sigma \cup \epsilon) \times \Gamma$ to $Q \times \Gamma^*$

δ is called the transition function of M.
$q_0 \in Q$ is the start state of machine.
$Z_0 \in \Gamma$ is the initial symbol on the stack.
$F \subseteq Q$ is set of final states.

Graphical Representation of PDA

$\delta(p, a, Z) = (q, aZ)$



Instantaneous description:-

Current configuration of PDA at any given instant can be described by an instantaneous description (ID). ID gives the current state of PDA, the remaining string to be processed & the entire contents of the stack.

An ID is defined as 3-tuple or a triple.

(q, w, α) ← q: current state; w: string to be processed; α: current contents of stack.

If the current configuration of PDA be $(q, aw, Z\alpha)$

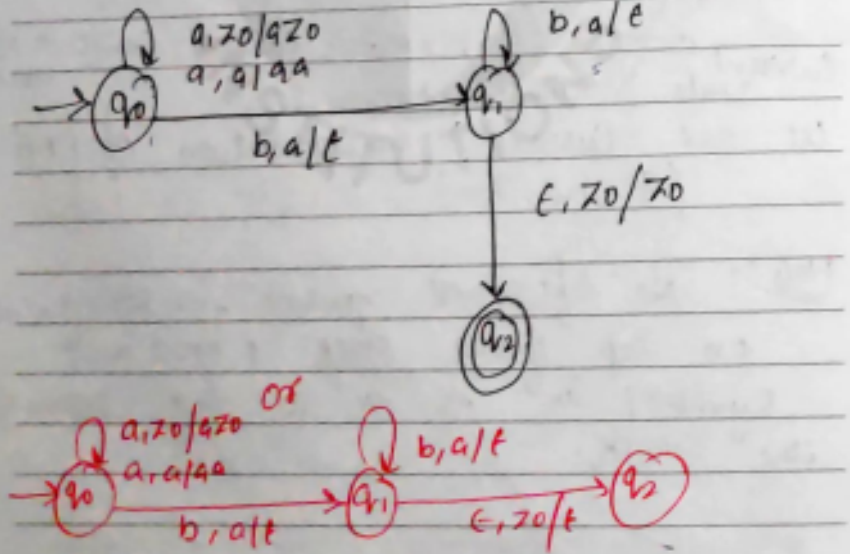
Note:- The left most symbol in string α is on top of the stack & right most symbol in α is at the bottom of the stack.

Acceptance of a lang by PDA: -

→ Get the final state from the start state

→ Get an empty stack from the start state

1.) Obtain a PDA to accept the language $L = \{a^n b^n \mid n \geq 1\}$ by a final state



Initial ID

$(q_0, aaabbb, Z_0) \vdash (q_0, aabbb, qZ_0)$
$\vdash (q_0, abbb, qqZ_0)$
$\vdash (q_0, bbb, qqqZ_0)$
$\vdash (q_1, bb, qZ_0)$
$\vdash (q_1, b, Z_0)$
$\vdash (q_2, \epsilon, Z_0)$
(final configuration)

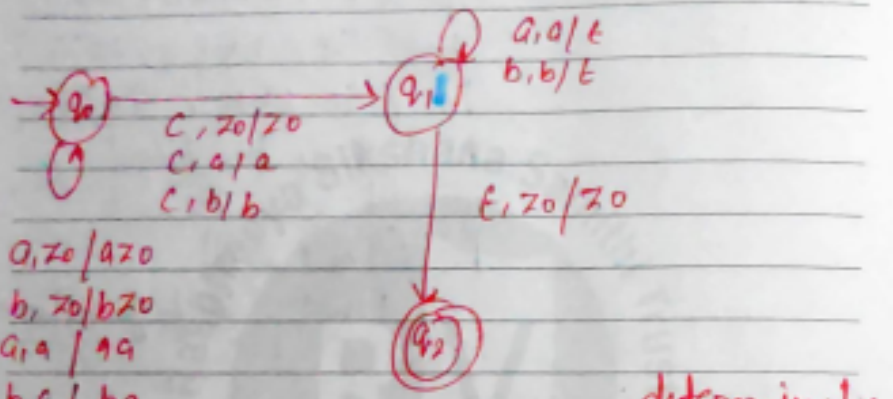
To reject the string

Initial ID

$(q_0, aabbb, Z_0) \vdash (q_0, abbb, qZ_0)$
$\vdash (q_0, bbb, qqZ_0)$
$\vdash (q_1, bb, qZ_0)$
$\vdash (q_1, b, Z_0)$
(final configuration)

Since the transition $\delta(q_1, b, Z_0)$ is not defined, the string aaabbb is rejected by the PDA.

2.) $L(M) = \{wcw^R \mid w \in (a+b)^*\}$



deterministic

Initial ID

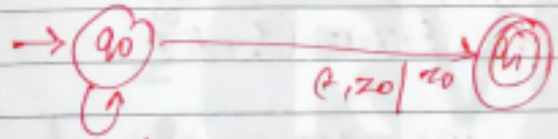
$(q_0, aabcbaa, z_0) \vdash (q_0, abcbaa, az_0)$
$\vdash (q_0, bcbaa, aaz_0)$
$\vdash (q_0, cbaa, baaz_0)$
$\vdash (q_1, baa, baaz_0)$
$\vdash (q_1, aa, aaz_0)$
$\vdash (q_1, a, az_0)$
$\vdash (q_1, \epsilon, z_0)$

Speechless is not even a good enough word to describe what I feel when I see the pictures of how we have transformed the world from the good to bad, from natural to artificial, physical appearance to daily makeup. I think I want to go with the word ENRAGED, DISGUSTED, or better yet INSULTED.

$\vdash (q_1, \epsilon, z_0)$

final configuration

3.) $L(M) = \{w \mid w \in (a+b)^* \text{ and } n_a(w) = n_b(w)\}$



non deterministic

4.) $L = \{a^n b^{2n} \mid n \geq 1\}$

Man can destroy a mountain within a day, but a Man cannot build a mountain in a single day or the rest of his days

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

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

$\Gamma = \{a, Z_0\}$

δ:

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

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

$\delta(q_0, b, a) = (q_1, \epsilon)$

$\delta(q_1, b, a) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, Z_0) = (q_2, \epsilon)$

$q_0 \in Q$ is the start state of machine

$Z_0 \in \Gamma$ is the initial symbol on the stack.

$F = \{q_2\}$ is the final state.

4.) $L = \{w \mid w \in (a,b)^* \text{ and } n_a(w) > n_b(w)\}$

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

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

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

$\delta(q_0, b, b) = (q_0, bb)$

$\delta(q_0, a, b) = (q_0, \epsilon)$

$\delta(q_0, b, a) = (q_0, \epsilon)$

$\delta(q_0, \epsilon, Z_0) = (q_1, a)$

Deterministic & Non-deterministic PDA :-

$M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA

The PDA is deterministic if

1. $\delta(q, a, Z)$ has only one element
2. If $\delta(q, \epsilon, Z)$ is not empty, then $\delta(q, a, Z)$ should be empty

Both the conditions should be satisfied for the PDA to be deterministic. If one of the condition fails, the PDA is non-deterministic.

5) PDA to CFG

1) The I/P symbols of PDA will be the terminals of CFG.

2) If the PDA moves from state q_i to q_j on consuming the i/p $a \in \Sigma$ where Z is the top of the stack, then the non terminals of CFG are the triplets of the form $(q_i Z q_j)$

3) If q_0 is the start state & q_f is the final state then $(q_0 Z q_f)$ is the start symbol of CFG.

4) The productions of CFG can be obtained from the transitions of PDA as shown below:

a) $\delta(q_i, a, Z) = (q_j, AB)$

introduce the productions of the form

$(q_i Z q_k) \rightarrow a(q_j A q_l)(q_0 B q_k)$

where q_k & q_l will take all possible values from Q.

b) $\delta(q_i, a, Z) = (q_j, \epsilon)$

introduce the production

$(q_i Z q_j) \rightarrow a$

Show that the following grammar is ambiguous by taking the string aab & also obtain the equivalent unambiguous grammar.

$S \to aS \mid aSbS \mid \epsilon$

I $S \Rightarrow aSbS$ by using $S \to aSbS$
$\Rightarrow aaSbS$ $S \to aS$
$\Rightarrow aabS$ $S \to \epsilon$
$\Rightarrow aab$ $S \to \epsilon$

II $S \Rightarrow aS$ $S \to aS$
$\Rightarrow aaSbS$ $S \to aSbS$
$\Rightarrow aabS$ $S \to \epsilon$
$\Rightarrow aab$ $S \to \epsilon$

$S \to aS \mid aAbS \mid \epsilon$
$A \to aAbA \mid \epsilon$

} Unambiguous grammars can be obtained by introducing a non-terminal A.

Eg 1: Obtain a CFG that generates the language accepted by PDA $M = (\{q_0, q_1\}, \{a, b\}, \{A, Z\}, \delta, q_0, Z, \{q_1\})$ with the transitions.

$\delta(q_0, a, Z) = (q_0, AZ)$
$\delta(q_0, b, A) = (q_0, AA)$
$\delta(q_0, a, A) = (q_1, \epsilon)$

Sol:-

For δ of the form	Resulting Production
$\delta(q_i, a, Z) = (q_j, \epsilon)$	$(q_i Z q_j) \to a$
$\delta(q_0, a, A) = (q_1, \epsilon)$	$(q_0 A q_1) \to a$

Now the transitions
$\delta(q_0, a, Z) = (q_0, AZ)$
$\delta(q_0, b, A) = (q_0, AA)$

For δ of the form	Resulting Productions
$\delta(q_i, a, Z) = (q_j, AB)$	$(q_i Z q_k) \to a(q_i A q_0)(q_0 B q_k)$
$\delta(q_0, a, Z) = (q_0, AZ)$	$(q_0 Z q_0) \to a(q_0 A q_0)(q_0 Z q_0)$ / $a(q_0 A q_1)(q_1 Z q_0)$
	$(q_0 Z q_1) \to a(q_0 A q_0)(q_0 Z q_1)$ / $a(q_0 A q_1)(q_1 Z q_1)$
$\delta(q_0, b, A) = (q_0, AA)$	$(q_0 A q_0) \to b(q_0 A q_0)(q_0 A q_0)$ / $b(q_0 A q_1)(q_1 A q_0)$
	$(q_0 A q_1) \to b(q_0 A q_0)(q_0 A q_1)$ / $b(q_0 A q_1)(q_1 A q_1)$

The start symbol of the grammar will be $q_0 Z q_1$.

2-> $\delta(q_0, a, Z) = (q_0, AZ)$ 1
$\delta(q_0, a, A) = (q_0, AA)$ 2
$\delta(q_0, b, A) = (q_1, \epsilon)$ 3
$\delta(q_1, \epsilon, Z) = (q_2, \epsilon)$ 4

Transition 2 can be rewritten as

$\delta(q_0, a, A) = (q_3, \epsilon)$
$\delta(q_3, \epsilon, Z) = (q_0, AZ)$

CFG to PDA :-

1. For each variable A
$\delta(q, \epsilon, A) = \{(q, \beta) \mid A \to \beta$ is a production of P$\}$

2. For each terminal a, $\delta(q, a, a) = \{(q, \epsilon)\}$

1.> $I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
$E \to I \mid E*E \mid E+E \mid (E)$

Sol:-
Set of terminals for the PDA is
$\{a, b, 0, 1, (,), +, *\}$
And the symbols I & E form the stack alphabet
$\delta(q_0, \epsilon, Z) = (q_1, SZ)$

a.) $\delta(q_1, \epsilon, I) = \{(q_1, a), (q_1, b), (q_1, Ia), (q_1, Ib), (q_1, I0), (q_1, I1)\}$

b.) $\delta(q_1, \epsilon, E) = \{(q_1, I), (q_1, E+E), (q_1, E*E), (q_1, (E))\}$

c.) $\delta(q_1, a, a) = \{(q_1, \epsilon)\}$; $\delta(q_1, b, b) = \{(q_1, \epsilon)\}$;
$\delta(q_1, 0, 0) = \{(q_1, \epsilon)\}$; $\delta(q_1, 1, 1) = \{(q_1, \epsilon)\}$;
$\delta(q_1, (, () = \{(q_1, \epsilon)\}$; $\delta(q_1,),)) = \{(q_1, \epsilon)\}$;
$\delta(q_1, +, +) = \{(q_1, \epsilon)\}$; $\delta(q_1, *, *) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, Z) = (q_2, Z)$

JUNE 2017

TUE - 13, WED - 14, & THU - 15

Problems: -

1.) for the grammar

S → aABB / aAA

A → aBB / a

B → bBB / A

C → a

Obtain the corresponding PPA.

JUNE 2017

FRI - 16, SAT - 17, & SUN - 18

To create a fifo file:

↓

mkfifo which is make first in first our

open system call to open a file

read " read

write " write

close " close

Writes process opens a fifo file, writes data into the fifo file & two reades opens the fifo file & reads the data from fifo file. So fifofile is accessible for both reades & writes

Message queues: → one of the interprocess comm^n mechanisms available under linux

fifo are created using the function mkfifo() which takes as arguments.

→ The name of fifo that has to be created.

2. The Permissions for the file.

One the file is created, it needs to be opened using the system call open() & the data can be read & written from the file using read() & write() system calls.

Eg:- If there are two fifos one of the server & the other of the client then the client can send req to the server on the server fifo which the server will read & respond back with the reply on the client fifo.

Message queue:-
- Created or opened with msgget()
- messages are added to the queue with msgsnd()

specifies type, length, and data of msg.

- messages are read with msgrcv() can be fetched based on type

Sending to the queue:-

- Once you have connected to the message queue using msgget(), you are ready to send & receive messages.

Each message is made up of two parts, which are defined in the struct msgbuf as defined in sys/msg.h

```
struct msgbuf {
        long mtype;
        char mtext[1];
};
```

The field mtype is used later when retrieving messages from the queue & can be any positive number.

next is the data, this will be added to the queue.

How do we pass it to message queue? ⟹ use msgsnd();

int msgsnd (int msqid, const void *msgp, size_t msgsz, int msgflg);

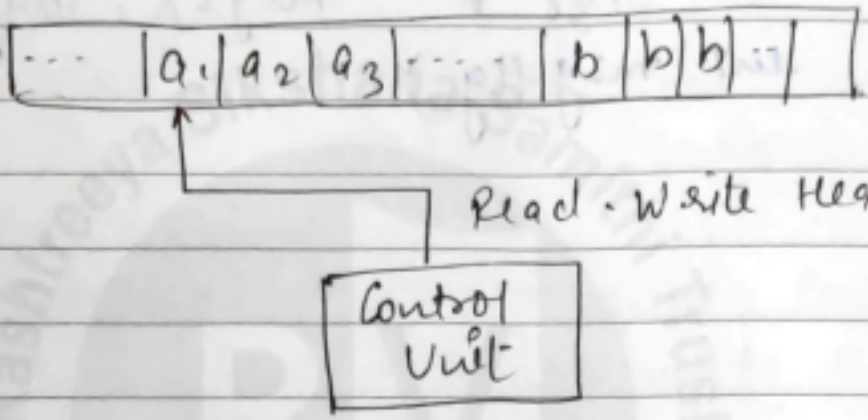
- msqid is the msg queue identifier returned by msgget(). The pointer msgp is a pointer to the data you want to put on the queue. msgsz is the size in bytes of the data to add to the queue. msgflg allows you to set some optional flag parameters.

Receiving from the queue: →

int msgrcv (int msqid, void *msgp, size_t msgsz, long msgtyp, int msgflg);

Unit-V

Turing Machine :-



Tape is used to store the information & is divided into cells. Each cell can store the information of only one symbol. The string to be scanned will be stored from the left most position on the tape. The string to be scanned should end with blanks. The tape is assumed to be infinite both on left & right side of the string.

Read - Write Head – The read-write head can read a symbol from where it is pointing to & it can write into the tape to where it points to

Control unit – Reading from the tape or writing into the tape is determined by the control unit.

* Read - write head can move either towards left or right i.e., movement can be on both the directions.

Various actions performed by machine are:

→ 1.) Change of state from one state to another state.

→ 2.) The symbol pointing to by the read - write head can be replaced by another symbol.

→ 3.) The read-write head may move either towards left or towards right.

$\delta(q_0,a) = (q_1,X,R)$
$\delta(q_0,Y) = (q_3,Y,R)$
$\delta(q_1,a) = (q_1,a,R)$
$\delta(q_1,b) = (q_2,Y,L)$
$\delta(q_1,Y) = (q_1,Y,R)$
$\delta(q_2,a) = (q_2,a,L)$
$\delta(q_2,X) = (q_0,X,R)$
$\delta(q_2,Y) = (q_2,Y,L)$
$\delta(q_3,Y) = (q_3,Y,R)$
$\delta(q_3,B) = (q_4,B,R)$

δ: $Q \times \Gamma$ → $Q \times \Gamma \times \{L \text{ or } R\}$

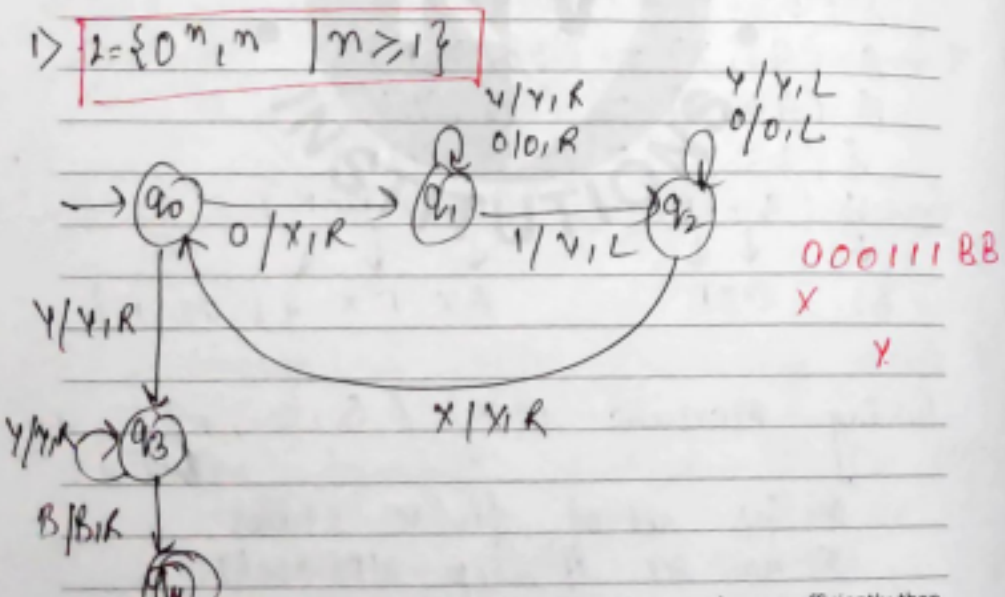
Turing Machine $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

Q is set of finite states
Σ is set of I/p alphabets

Γ is the set of tape symbols
δ is transition function
from Q x Γ to Q x Γ x {L, R}

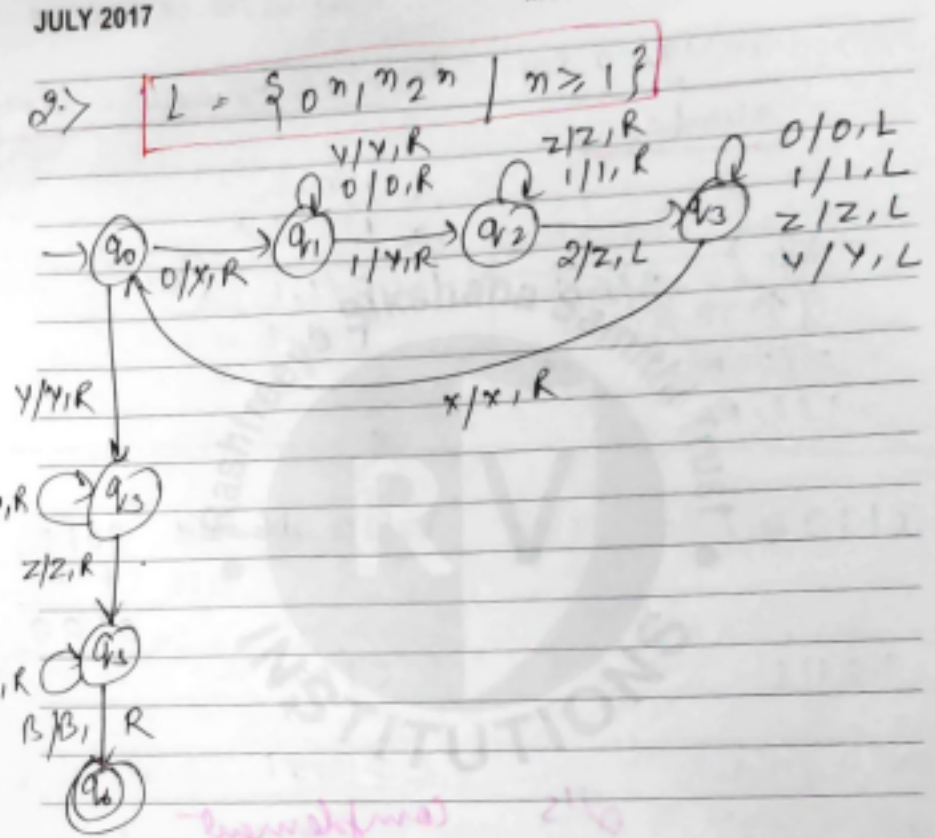
q_0 is the start state
B is special symbol indicating blank character.
$F \subseteq Q$ is set of final state

1) $L = \{0^n 1^n \mid n \geq 1\}$

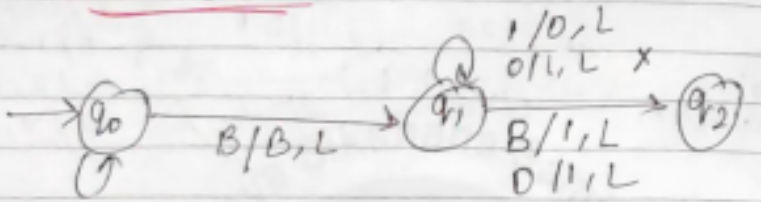


000111 BB
X
Y

2) $L = \{0^{n_1} 1^{n_2} 2^n \mid n \geq 1\}$



3.> Design a TM to increment a binary number.



I/p: B1100B

1's complement: 0011
+ 1
0100

I/p: 10111

01000
1
01001

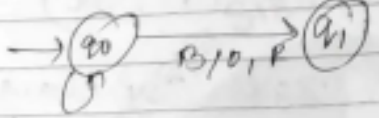
2's complement

B 0011 B
0011 B
1100

01000
10111
11000

I/p: 01100
01101

4.> Design a TM to increment a unary no



I/p: B000B
op: 0000

5.> TM to add two integers

I/p $0^m 1 0^n$
op: 0^{m+n}

1.> In q_0, TM skips 0's until it reads 1, change it to 0 & goes to q_1

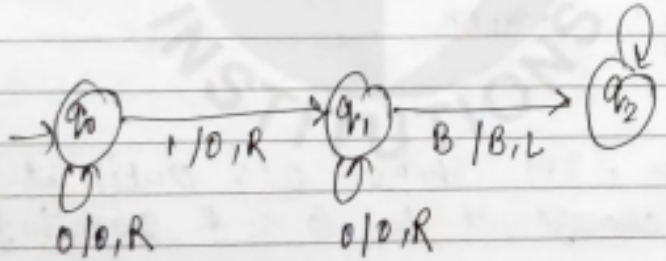
$\delta(q_0,0) = (q_0, 0, R)$
$\delta(q_0,1) = (q_1, 0, R)$

2.) In q_1, TM skips 0's until it reads B, turns left & goes to q_2

$$\delta(q_1, 0) = (q_1, 0, R)$$
$$\delta(q_1, B) = (q_2, B, L)$$

3.) In q_2, TM reads 0 & changes to B & halts.

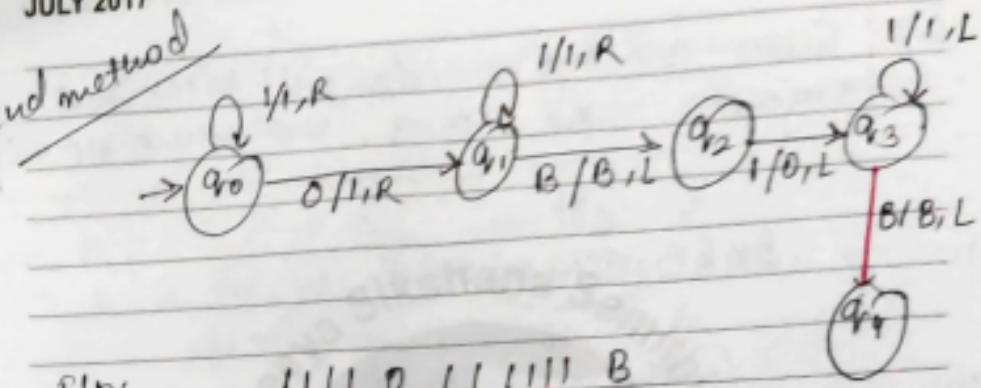
$$\delta(q_2, 0) = (q_2, B, R)$$



I/p: 000 1 000 B
(x) (y)

o/p: 000 0 00B
x y

IInd method



I/p: 1111 0 1111111 B
x 0 y

1111 111111 0
x y

Try seeing, feeling, and tasting the water you swim in the way a land animal might perceive it. You may find the experience fascinating -- and mind-expanding.

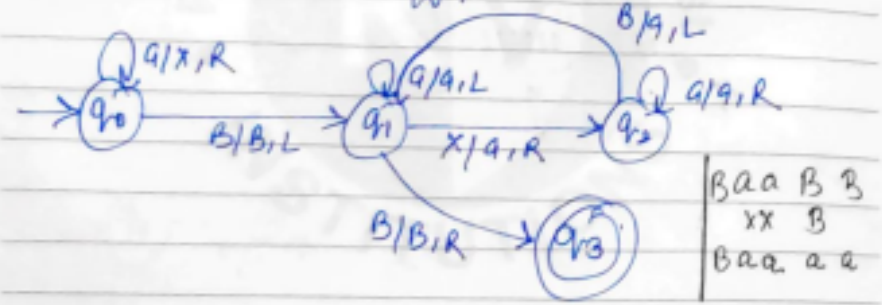
6.) Given a string w, design a TM that generates the string ww where w ∈ {a}+.

I/P: . . . BBBB aaaa BBB . . . .

w

O/P: . . . BBB aaaa aaaa BBB . . .

w w



General procedure to Concatenate the string w with itself

1.) Replace each symbol in w with x.

2.) Find the rightmost x.

3.) Replace rightmost x by symbol a.

4.) Move to the right of rightmost a & replace it by a.

5.) Find the rightmost x.

6.) Repeat through step 3 till we find no more x's.

7.) Obtain a TM to Compute the function - which is called monus or proper subtraction and is defined by $m - n = \max(m-n, 0)$

$m - n = m - n$ if $m \geq n$

And

$m - n = 0$ if $m < n$

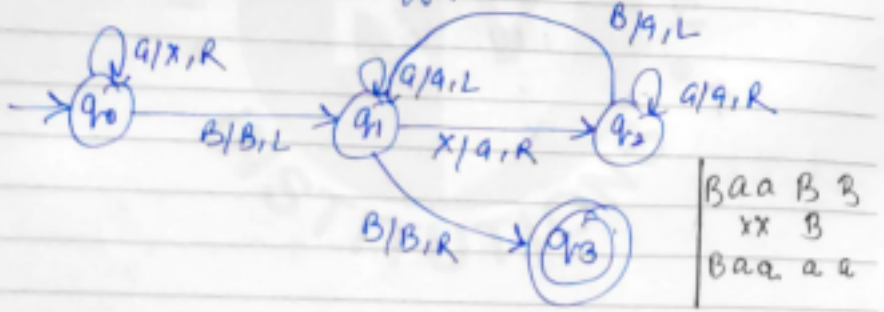
6.) Given a string w, design a TM that generates the string ww where w ∈ {a}*.

i/p: ... BBBB aaaa BBB ....

w

o/p: ... BBB aaaa aaaa BBB ...

w w



General procedure to concatenate the string w with itself

1.) Replace each symbol in w with x.

2.) Find the rightmost x.

3.) Replace rightmost x by symbol a.

4.) Move to the right of rightmost a & replace it by a.

5.) Find the rightmost x.

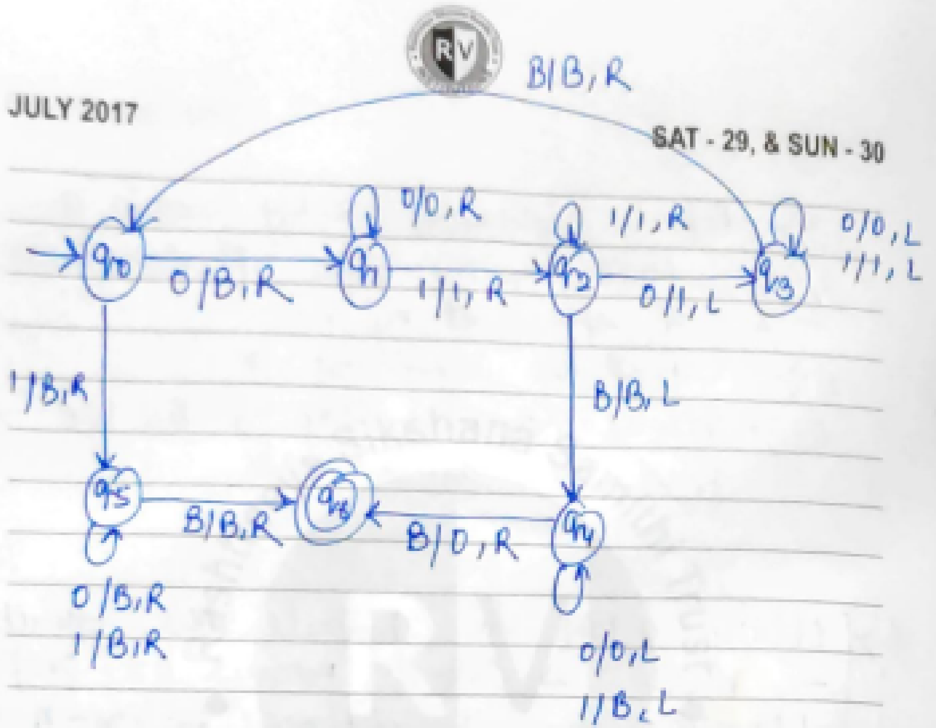
6.) Repeat through step 3 till we find no more x's.

7.) Obtain a TM to compute the function - which is called monus or proper subtraction and is defined by $m - n = \max(m - n, 0)$

$m - n = m - n$ if $m \geq n$

And

$m - n = 0$ if $m < n$



$0^m 1 0^n$

I/p: B 00000 1 000 B.... (50's, 30's)
---B B00001100 B---
---B BB0001110 B - - -
-- B BBB001111 B - - -
BBBB01111B - -
O/p: BBB00BBBB (20's)

case 1 m>n

I/p: -- B 000 1 0000 B - - - (30's, 40's)
-- B B00110 B - -
---B BB0111 - - -
-- B BBB111 100B
BBB BBBBBBB

case 2 m<n

O/p: all Blank

UNIT-1

Notes

Equivalence of two finite Automata

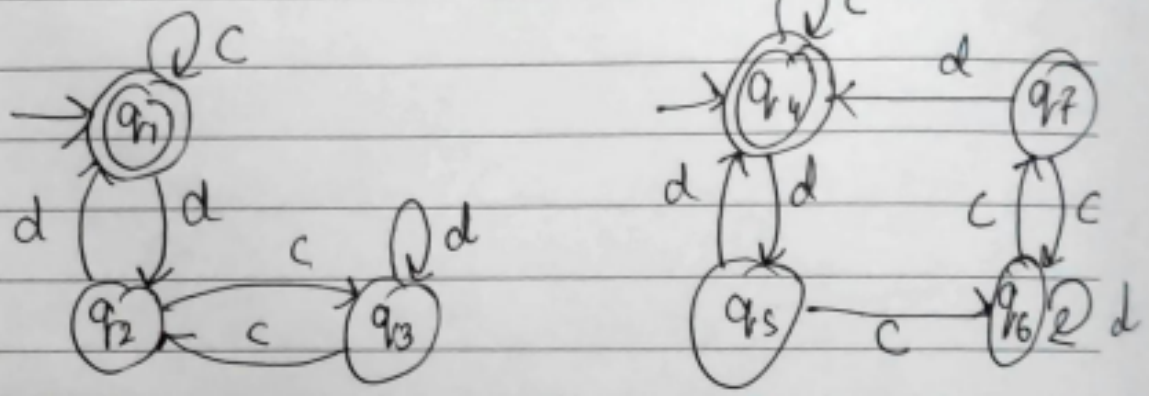
Steps to identify equivalence :→

→ 1) For any pair of states $\{q_i, q_j\}$ the transition for i/p $a \in \Sigma$ is defined by $\{q_a, q_b\}$ where $\delta(q_i, a) = q_a$ and $\delta(q_j, a) = q_b$

The two automata are not equivalent if for a pair $\{q_a, q_b\}$ one is INTERMEDIATE state and the other is FINAL state

→ 2) If initial state is final state of one automata, then in second automata also initial state must be final state for them to be equivalent

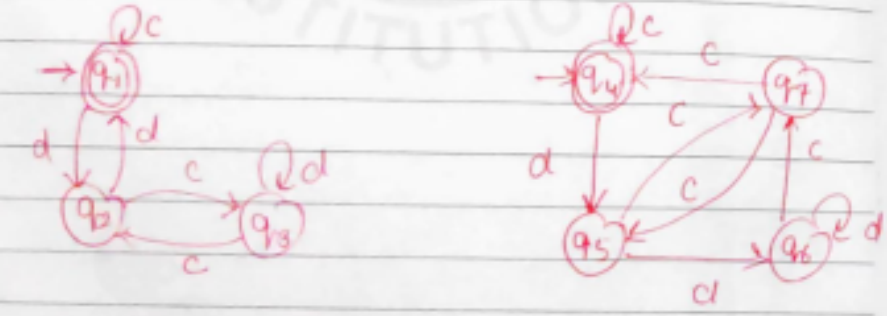
1)



States	c	d
(q_1, q_4)	(q_1, q_4) FS FS	(q_2, q_5) IS IS
(q_2, q_5)	(q_3, q_6) IS IS	(q_1, q_4) FS FS
(q_3, q_6)	(q_2, q_7) IS IS	(q_3, q_6) IS IS
(q_2, q_7)	(q_3, q_6) IS IS	(q_1, q_4) FS FS

So the given automata A & B are equivalent bcoz they are satisfying both the conditions.

Q7



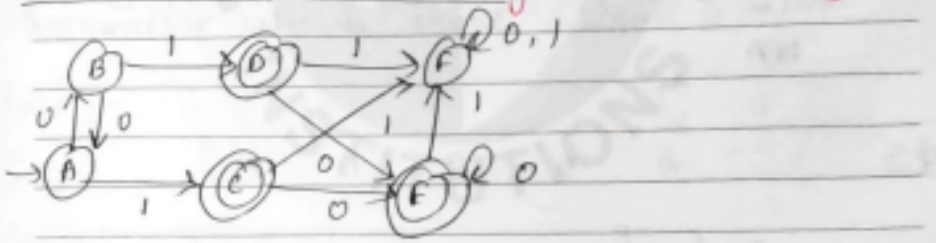
A

B

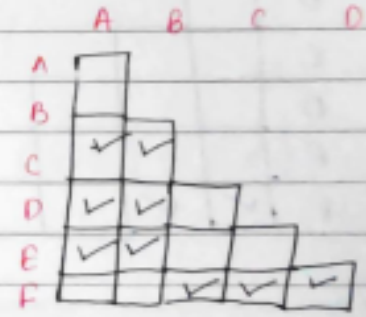
States	C	d
(q_1, q_4)	(q_1, q_4) FS FS	(q_2, q_5) IS IS
(q_2, q_5)	(q_3, q_7) IS IS	(q_1, q_6) FS IS

A & B are not equivalent.

DFA Minimization using table filling method



step 2:



Steps: -

1) Draw a table for all pairs of states (P,Q)
2) Mark all pairs where P∈F and Q∉F
3) If there are any unmarked pairs (P,Q) such that [δ(P,x), δ(Q,x)] is marked, then mark [P,Q] where 'x' is an i/p symbol

Repeat this until no more markings can be made

4) Combine all the unmarked pairs and make them a single state in the minimized DFA.

Step 3.

	A	B	C	D	E
B					
C					
D					
E					
F	✓	✓			

(B,A) - $\delta(B,0) = A$, $\delta(B,1) = D$; $\delta(A,0) = B$, $\delta(A,1) = C$

(D,C) - $\delta(D,0) = E$, $\delta(D,1) = F$; $\delta(C,0) = E$, $\delta(C,1) = F$

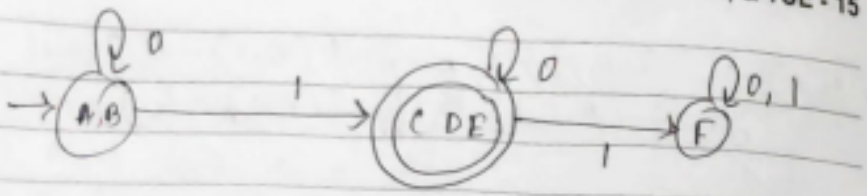
(E,C) - $\delta(E,0) = E$, $\delta(E,1) = F$; $\delta(C,0) = E$, $\delta(C,1) = F$

(E,D) - $\delta(E,0) = E$, $\delta(E,1) = F$; $\delta(D,0) = E$, $\delta(D,1) = F$

(F,A) - $\delta(F,0) = F$, $\delta(F,1) = F$; $\delta(A,0) = B$, $\delta(A,1) = C$

(F,B) - $\delta(F,0) = F$; $\delta(B,0) = A$

Step 4.) (A,B), (E,C) (E,D), (D,C) unmarked pair



GNF

1> $A \to BC$
$B \to CA \mid b$
$C \to AB \mid a$

Sol: $A = A_1, B = A_2, C = A_3$
$A_1 \to A_2A_3$
$A_2 \to A_3A_1 \mid b$
$A_3 \to A_1A_2 \mid a$

first two productions are of the form
$A_i \to A_j\alpha$ for $i<j$

Consider A_3 production

$A_3 \to A_1A_2 \mid a$

$A_3 \to A_3A_1A_3A_2 \mid bA_3A_2 \mid a$
(A_3 → $A_3\alpha_1$ | β_1 | β_2)

A_3 has left Recursion

$A \to A\alpha_1 \mid A\alpha_2 \mid \ldots$ $\quad A \to A\alpha_i \mid \beta_i$

then
$A \to \beta_i A1$
$A1 \to \alpha_i A1 \mid \epsilon$

$\therefore A \to bA_3A_2Z \mid aZ$

$Z \to A_1A_3A_2Z \mid \epsilon$

$\therefore A \to bA_3A_2 \mid a \mid bA_3A_2Z \mid aZ$

$Z \to A_1A_3A_2Z \mid A_1A_3A_2$

Now consider A_2

$A_2 \rightarrow A_3A_1 \mid b$

$A_2 \rightarrow bA_3A_2A_1 \mid qA_1 \mid bA_3A_2ZA_1 \mid qZA_1 \mid b$

Consider A_1

$A_1 \rightarrow A_2A_3 = bA_3A_2A_1A_3 \mid qA_1A_3 \mid bA_3A_2ZA_1A_3 \mid qZA_1A_3 \mid bA_3$

Consider Z

$Z \rightarrow A_1A_3A_2 \mid A_1A_3A_2Z$

$\rightarrow (bA_3A_2A_1A_3 \mid qA_1A_3 \mid bA_3A_2ZA_1A_3 \mid qZA_1A_3 \mid bA_3)A_3A_2Z$

$(bA_3A_2A_1A_3 \mid qA_1A_3 \mid bA_3A_2ZA_1A_3 \mid qZA_1A_3 \mid bA_3)A_3A_2$

Substitution

$A \rightarrow x_1Bx_2$

$B \rightarrow y_1 \mid y_2 \mid \cdots y_n$

$A \rightarrow x_1y_1x_2 \mid x_1y_2x_2 \mid \cdots x_1y_nx_2$

$B \rightarrow y_1 \mid y_2 \mid \cdots y_n$

Eg $A \rightarrow qBq$

$B \rightarrow qb \mid b$

$A \rightarrow qqbq \mid qbq$

$B \rightarrow qb \mid b$

Left Recursion

$A \xrightarrow{+} A\alpha$

Consider the production of the form

$A \rightarrow A\alpha_1 \mid A\alpha_2 \mid A\alpha_3 \cdots A\alpha_m \mid \beta_1 \mid \beta_2 \mid \cdots \beta_m$

where β_i's do not start with A

Then the production A can be replaced by

$A \to B_1A' \mid B_2A' \mid B_3A' \mid \dots \mid B_nA'$
$A' \to \alpha_1A' \mid \alpha_2A' \mid \dots \mid \alpha_nA' \mid \epsilon$

Eg: $S \to Ab \mid a$
$A \to Ab \mid Sa$

$S \to Ab \mid a$
$A \to Ab \mid Aba \mid aa$] after replacement

$S \to Ab \mid a$ not applicable

$A \to Ab \mid Aba \mid aa$ $A = A$ $\therefore A \to aaA'$
$\alpha_1 = b$ $A' \to bA' \mid \epsilon$
$\alpha_2 = ba$
$\beta = aa$

$\therefore S \to Ab \mid a$
$A \to aaA'$
$A' \to bA' \mid baA' \mid \epsilon$

CNF $A \to BC$
$A \to a$

1.> $S \to 0A \mid 1B$
$A \to 0AA \mid 1S \mid 1$
$B \to 1BB \mid 0S \mid 0$

$S \to 0A \mid 1B$, $S \to B_0A \mid B_1B$
$B_0 \to 0$
$B_1 \to 1$

$A \to 0AA \mid 1S$, $A \to B_0AA \mid B_1S$
$B_0 \to 0$
$B_1 \to 1$

$B \to 1BB \mid 0S$, $B \to B_1BB \mid B_0S$
$B_1 \to 1$
$B_0 \to 0$

$A \to B_0AA$, $A \to B_0D_1$
$D_1 \to AA$

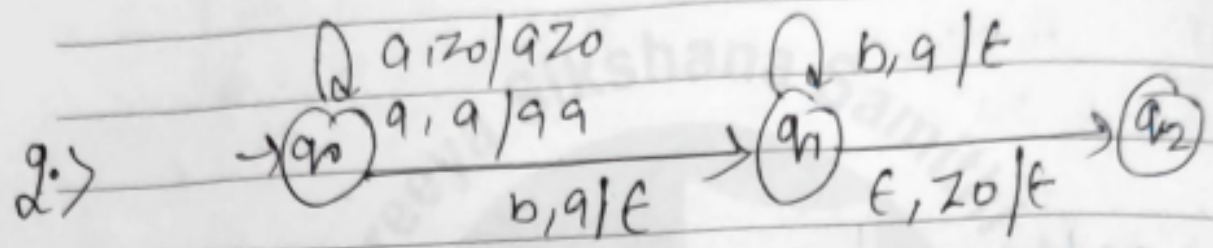
$B \to B_1BB$, $B \to B_1D_2$
$D_2 \to BB$

TUE - 29 & WED - 30

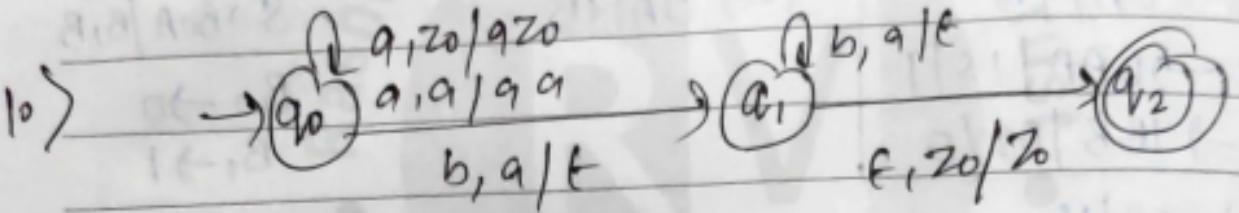
AUGUST 2017

1. $\delta(q_1, \epsilon, Z_0) = (q_2, Z_0)$ accepted by final state
2. $\delta(q_1, \epsilon, Z_0) = (q_2, \epsilon)$ accepted by an empty stack.

2.)



1.)



Recently I keep thinking that this isn't about the survival of a species. It's about why we're never satisfied with what we need, why we always take a bit more.