★ Automata theory is the study of computing devices or "machines".

17/7/13

Finite State Machine

- Some concepts like finite automata and certain kinds of formal grammars, are used in the design and construction of important kinds of software.
- Other concepts, like the Turing machine, help us understand what we can expect from s/w.

Concepts of Automata Theory:

★ Alphabets: An alphabet is a finite, non empty set of symbols. Symbol: Σ

Egs: $\Sigma = \{0, 1\}$, the binary alphabet

$\Sigma = \{a, b, \ldots\ldots z\}$, the set of all lower case letters.

★ Strings: A string (or word) is a finite sequence of symbols chosen from some alphabet.

eg: 'computer' is a string from alphabet $\Sigma = \{a, b \ldots z\}$

'theory' is another string from same alphabet.

0 1 0 1

- empty string
- length of string
- power of alphabet
- Concatenation of strings

0101

$\Sigma = \{0,1\}$ L = set of all strings of length 2
0101 = $\{00, 01, 10, 11\}$
$\Sigma^* = \{\epsilon,$
L = { 0,1, }

* Language: A language is defined as a set of strings of symbols from one alphabet Σ.
strings of symbols over a fixed alphabet Σ
eg: Set of all strings over a fixed alphabet Σ is a language denoted by Σ^*.

Types
① regular lang
② non-reg lang

Let $\Sigma = \{0,1\}$ then
$\Sigma^* = \{\epsilon, 00, 11, 01, 10, 0, 1, 111, 000, \ldots\}$
ϕ (null set) and the sets consisting of empty string i.e $\{\epsilon\}$ are also the languages.
From above language if we exclude e, the empty word, the remaining set is lang denoted by Σ^+
$\Sigma^+ = \{0, 00, 11, 1, 111, 000, \ldots\}$

⟹ We begin our study of theory of computing by first looking at the most simple computing machines known as automata.

Automata:
An automaton is an abstract model of a digital computer.
An automaton has a mechanism to read i/p, which is a string over a given alphabet. This i/p is written on an "input file", which can be read by the automaton but cannot change it.

Vending Machine

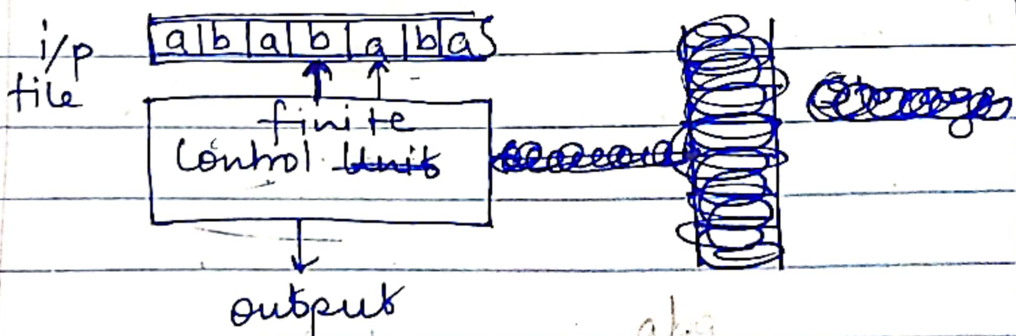


fig. Automaton.

Automata is basically lang acceptors or lang ~~rejectors~~ recognizers

Theoretical computer science

Different models of computation

Appl^n:-

1. Building of compiler
2. translator
3. searching

Symbol: It is basic nondivisible automic unit of TOC

FSM

Appl^n

1. Automata is the base for the formal lang and formal lang is used in programming lang
2. Plays imp role in compiler design.
3. design and analysis of digital circuits
4. Text search for patterns. (lexical analyzer)

Defination of FA.

FA is a collection of 5-tuple' $\boxed{(Q, \Sigma, \delta, q_0, F)}$
where $Q \rightarrow$ finite set of states, which is non empty
$\Sigma \rightarrow$ input alphabet, input set
$q_0 \rightarrow$ initial state
$F \rightarrow$ set of final states
$\delta \rightarrow$ transition function or mapping function

Notations used in transition diagram

q_0 representing the state

$\longrightarrow$ representing transition from one state to another

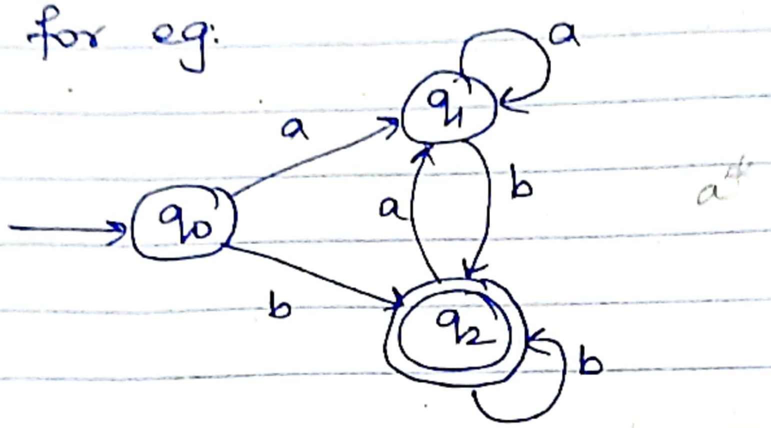
$\rightarrow q_0$ start state

q_n final state

Deterministic Finite Automata (DFA)

★ if there is only one path for a specific i/p from current state to next state.

for eg:



1 0 1 0 1 0

Example 1: Design a FA which ~~decides~~ checks whether the given binary number is even.

Step 1: $FA = (Q, \Sigma, q_0, F, \delta)$ | 1 | 0 | 0 | 1 | = 9

Step 2: $Q = \{q_0, q_1, q_2\}$

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

q_0 = start state $\{q_0\}$

$F = \{q_2\}$

δ = transition table

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

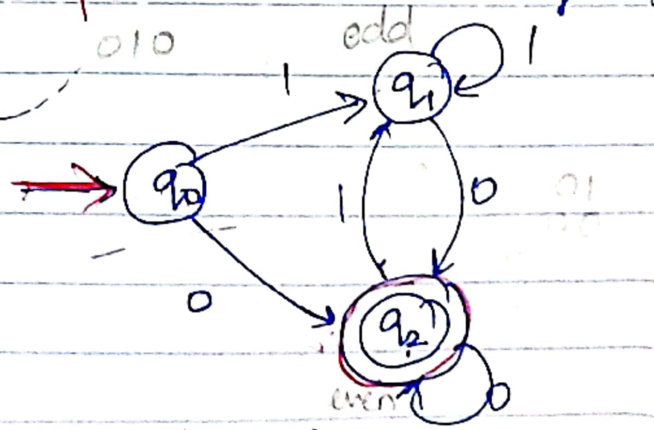
8 4 2 1   1 1 = 3   1 0 1 = 5

1 0 0 = 4

Step 3:

states \ i/p	0	1
→ q_0 start	q_2	q_1
q_1	q_2	q_1
(q_2)*	q_2	q_1

Step 4 transition diagram



01
10

8 4 2 1
1 1 1 0

0 0 0 0

Step 5:

String – 1011010

$(q_0, 1011010)$

$\vdash (q_1, 011010)$

$\vdash (q_2, 11010)$

$\vdash (q_1, 1010)$

$\vdash (q_1, 010)$

$\vdash (q_2, 10)$

$\vdash (q_1, 0)$

$\vdash (q_2, \epsilon)$

Accepted → number is even

Ex. 2 Design FSM, which has odd number of 0's and any number of 1's

odd 0's, even 0's

000

001

00 01

odd-0's
any-1's

{2, 4, 6, 8, 0} i/p {3, 1, 5, 7, 9} i/p

S \ i	0	1
$\rightarrow q_0$	q_2^*	q_1
even 0's q_1	q_2	q_1
odd 0's q_2^*	q_1	q_2

String – 0011

$(q_0, 0011)$

$\vdash (q_2, 011)$

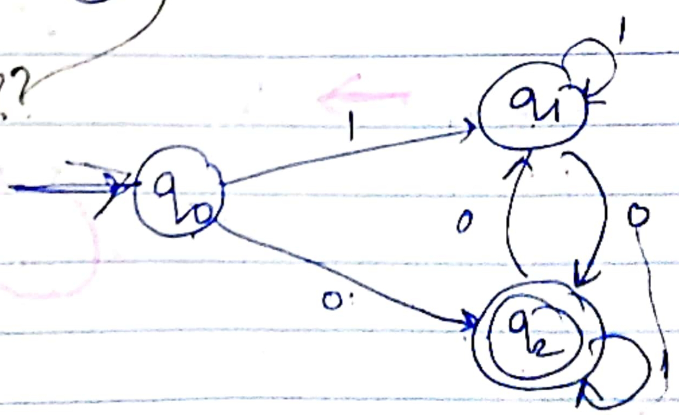
$\vdash (q_1, 11)$

$\vdash (q_1, 1)$

$\vdash (q_1, \epsilon)$

not accepted.

??



i/p 0 1 1 1

i/p 1

i/p 1 1 1

what is q0?

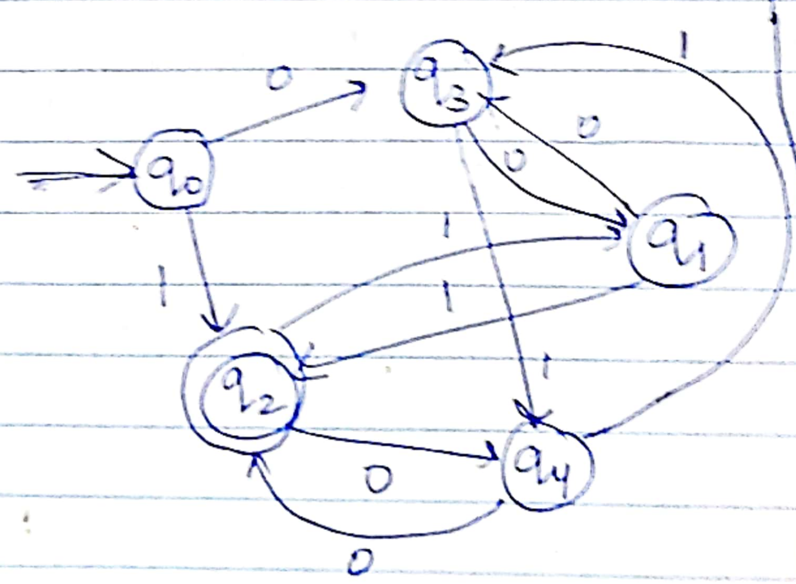
13/7/23

Ex 3 Design FSM which accepts a string if it contains even number of 0's and odd number of 1's

q_1	even - 0	even - 1		ε ε
q_2	even - 0	odd - 1		ε 0
q_3	odd - 0	even - 1		0 ε
q_4	odd - 0	odd - 1		0 0

δ :-

	δ	0	1
	→ q_0	q_3	q_2
ε0 ε1	q_1	q_3	q_2
ε0 d1	q_2	q_4	q_1
d0 ε1	q_3	q_1	q_4
d0 d1	q_4	q_2	q_3



String - 00110

$(q_0, 0011011)$
$\vdash (q_3, 011011)$
$\vdash (q_1, 11011)$
$\vdash (q_2, 1011)$
$\vdash (q_1, 011)$
$\vdash (q_3, 11)$
$\vdash (q_4, 1)$
(q_3, ϵ)
not accepted

$(q_0, 001)$
$\vdash (q_3, 01)$
$\vdash (q_1, 1)$
$\vdash (q_2, \epsilon)$

accepted//

" ababbaababb "

abaabba

a q_1

ab q_2

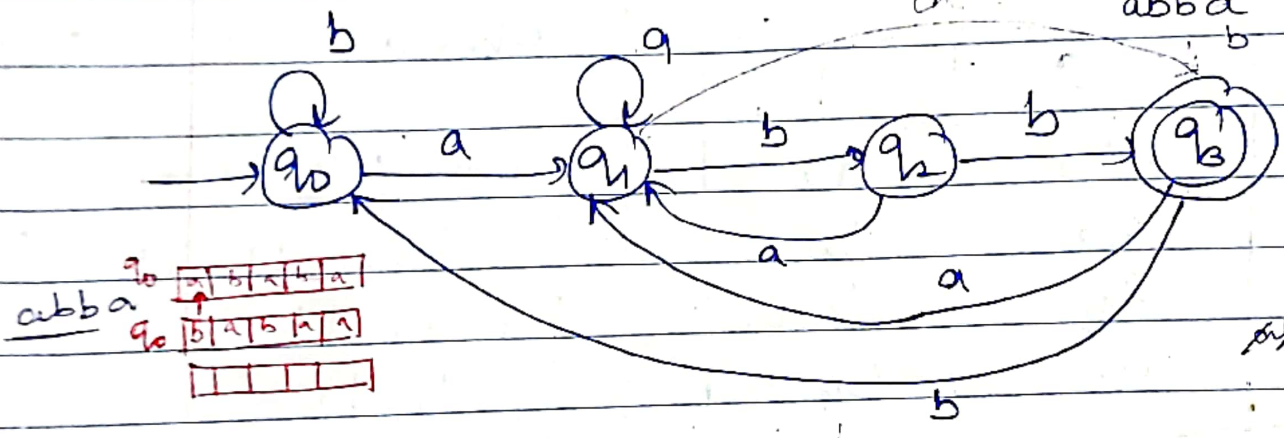
contain abb ending abb.

Ex. 4 Design FSM that accepts set of all strings ending with 'abb'.

S \ i/p		a	b
nothing →	q_0	q_1	q_0
a	q_1	q_1	q_2
ab	q_2	q_1	q_3
abb	((q_3))	q_1	q_0
		q_3	q_3

aa ab abb ab aa

abb abba abbb abba abbb aba abb



I/P string - babbabb

$(q_0, babbabb)$

$\vdash (q_0, abbabb)$

$\vdash (q_1, bbabb)$

$\vdash (q_2, babb)$

$\vdash (q_3, abb)$

$\vdash (q_1, bb)$

$\vdash (q_2, b)$

$\vdash (q_3, \epsilon)$

accepted

string - ababaaba

$(q_0, ababaaba)$

$\vdash (q_1, bababa)$

$\vdash (q_2, ababa)$

$\vdash (q_1, baba)$

$\vdash (q_2, aba)$

$\vdash (q_1, ba)$

$\vdash (q_2, a)$

$\vdash (q_1, \epsilon)$

not accepted

0, 1, 2, 3 q_s $3\overline{)96}$ = 32, -9, $q_0 \rightarrow 06$, -6, 0 $\qquad$ $3\overline{)97}$ = 32, -9, 07, -6, 1

Ex. 5 Design FA which checks whether the given unary number is divisible by 3.

Soln: unary number represented as 3 – 111

unary no divisible by 3 are 111 111111 111111111

S \ i	1
$\rightarrow q_0$	q_1
q_1	q_2
q_2	q_3
q_3	q_1

String – 111

$(q_0, 111)$

$\vdash (q_1, 11)$ $\qquad$ 111111

$\vdash (q_2, 1)$

$\vdash (q_3, \epsilon)$

accepted//

Exp. 6 Find FA check whether given decimal number is divisible by 3 or not

$\Sigma = \{0, \ldots 9\}$

S \ i	{0,3,6,9}	{1,4,7}	{2,5,8}
$\rightarrow q_s$	q_0	q_1	q_2
0 q_0	q_0	q_1	q_2
1 q_1	q_1	q_2	q_0
2 q_2	q_2	q_0	q_1

15 $\quad$ 14 $\quad$ 16

Q \ Σ	0	1
$\rightarrow q_0$	q_0	q_1
q_0	q_0	q_0
q_1	q_2	q_0
q_2	q_1	q_2

Transition diagram:

String $(q_s, 5738)$

$\vdash (q_2, 738)$

$\vdash (q_0, 38)$

$\vdash (q_0, 8)$

$\vdash (q_2, \epsilon)$

$3\overline{)73}$ = 24, -6, 13, -12, 1 $\qquad$ 14, -12, 2 $\qquad$ $3\overline{)80}$ = 27, -6, 23, -21

10 $\quad$ 11 $\quad$ 100 $\quad$ 101

Scanned with OKEN Scanner

Total Marks of Question no. | Moderator

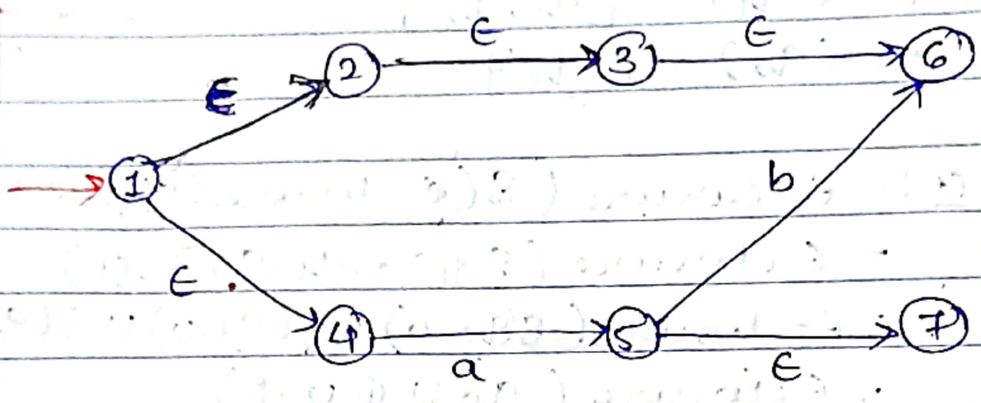
6/8/13

NFA.

Finite Automata with Epsilon Transitions.

→ Introduce another extension of FA as ε-transition
→ The new feature is that we allow a transition on ε, the empty string.
→ Finding every state that can be reached from q along any path whose arcs are all labelled ε.

Ullman Pg. 75



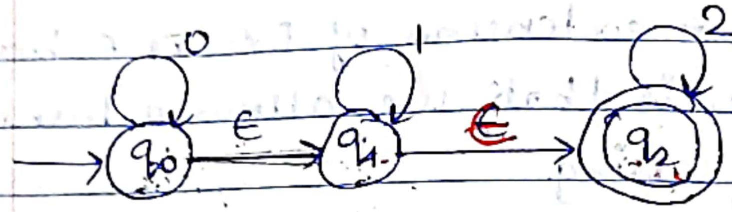
ε-closure(1) = {1, 2, 3, 4, 6}
$\hat{\delta}(1, \epsilon)$ = ε-closure (1). That is, if the label of the path is ε, then we can follow only ε-labeled arcs extending from state 1.

ε-closure (1) = {1, 2, 3, 6, 4}.

1

RE → NFA w ε → NFA w/o ε → DFA → minimized DFA

Convert the given NFA with ε to NFA without ε.



Step 1:

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

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

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

$\epsilon\{\delta(\epsilon(q_0, 0)$ ② ①

Step 2:

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

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

$\epsilon(q_1)$

$$\hat{\delta}(q_0, 2) = \epsilon(q_2) = \{q_2\}$$

2/6/23, 9:30 AM

20221222_091444.jpg

Total Marks of Question no		Examiner	30 14 44
		Moderator	

$\epsilon(\delta(\epsilon$

$\hat{\delta}(q_0, 0) = \phi$

$\hat{\delta}(q_1, 1) = \{q_1, q_2\}$

$\hat{\delta}(q_2, 0) = \phi$

$\hat{\delta}(q_2, 1) = \phi$

$\hat{\delta}(q_0, 2) = \{q_2\}$

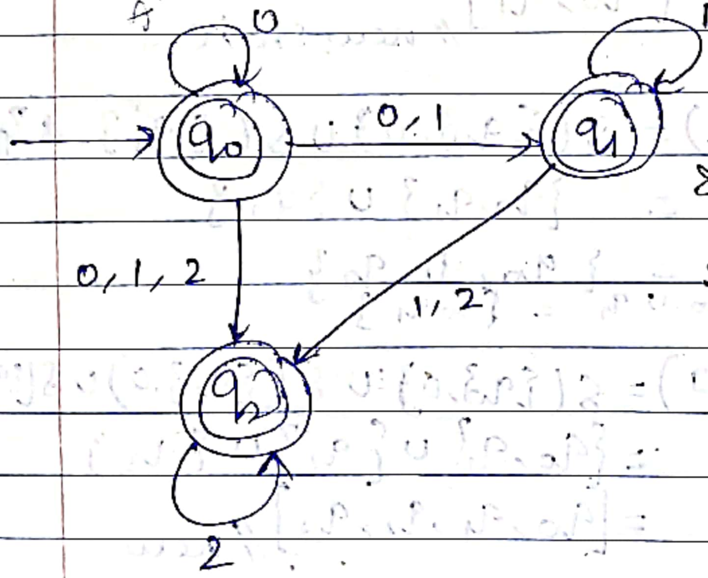
$\hat{\delta}(q_0, 2) = \{q_2\}$

$\hat{\delta}(q_2, 2) = \{q_2\}$ $\quad \epsilon(q_2) = q_2$

elinated ϵ

S \ i/p	0	1	2
→ q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$

2 FA 3/0



$\hat{\delta}(q_0, 1) = q_1, q_2$

$\hat{\delta}(q_0, 2) = q_2$

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

$\hat{\delta}(q_1, 1) = q_1, q_2$

$\delta(q_1, 2) = q_2$

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

$\delta(q_2, 1) = \phi$

$\delta(q_2, 2) = q_2$

leave space

(3)

Convert the given NFA to DFA.

i/P \ S	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
q_1	q_2	q_1
q_2	q_3	q_3
q_3^*	ϕ	q_2

A

DFA = $(Q', \Sigma', \delta', q_0', F')$

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

$\hat{\delta}(\{q_0\}, 0) = [q_0, q_1]$ // new state

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

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

$\hat{\delta}(\{q_0, q_1, q_2\}, 0) = \delta(\{q_0\}, 0) \cup \delta(\{q_1\}, 0) \cup \delta(\{q_2\}, 0)$
$= \{q_0, q_1\} \cup \{q_2\} \cup \{q_3\}$
$= [q_0, q_1, q_2, q_3]$ // new

$\hat{\delta}(\{q_0, q_1, q_2\}, 1) = \delta(\{q_0\}, 1) \cup \delta(\{q_1\}, 1) \cup \delta(\{q_2\}, 1)$
$= \{q_0\} \cup \{q_1\} \cup \{q_3\}$
$= \{q_0, q_1, q_3\}$ // new

Total Marks of Question no		Examiner
		Moderator

30
14
44

$\hat{\delta}(\{q_0, q_1, q_3\}, 0) = \delta(\{q_0\}, 0) \cup \delta(\{q_1\}, 0) \cup \delta(\{q_3\}, 0)$
$= \{q_0, q_1\} \cup \{q_2\} \cup \{0\}$
$= [q_0, q_1, q_2]$

$\hat{\delta}(\{q_0, q_1, q_3\}, 1) = \delta(\{q_0\}, 1) \cup \delta(\{q_1\}, 1) \cup \delta(\{q_3\}, 1)$
$= q_0 \cup q_1 \cup q_3$
$= [q_0, q_1, q_3]$

$\hat{\delta}(\{q_0, q_1, q_2, q_3\}, 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_2, 0)$
$\cup \delta(q_3, 0)$
$= \{q_0, q_1\} \cup q_2 \cup q_3 \quad 0$
$= [q_0, q_1, q_2, q_3]$

$\hat{\delta}(\{q_0, q_1, q_2, q_3\}, 1) = \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)$
$\cup \delta(q_3, 1)$
$= q_0 \cup q_1 \cup q_3 \cup q_2$
$= [q_0, q_1, q_2, q_3]$

	δ i/p S	0	1
A	$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
B	q_1	q_2	q_1
C	q_2	q_3	q_3
D	q_3^*	ϕ	q_2
E	q_0, q_1	$q_0 q_1 q_2$	$q_0 q_1$
F	$q_0 q_1 q_2$	$q_0 q_1 q_2 q_3$	$q_0 q_1 q_3$
G	$q_0 q_1 q_3^*$	$q_0 q_1 q_2$	$q_0 q_1 q_3$
H	$q_0 q_1 q_2 q_3^*$	$q_0 q_1 q_2 q_3$	$q_0 q_1 q_2 q_3$

	0	1
$\rightarrow$ A q_0	E	A
B q_1	C	B
C q_2	D	D
D q_3^*	$\{\phi\}$	C
E	F	E
F	H	G
G*	F	G
H*	H	H

3

Convert the given NFA to DFA.

S \ i/P	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
q_1	q_2	q_1
q_2	q_3	q_3
q_3^*	ϕ	q_2

A

$DFA = (Q, \Sigma, \delta, q_0, F)$

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

$\hat{\delta}(\{q_0\}, 0) = [q_0, q_1]$ // new state

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

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

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

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

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

$= \{q_0, q_1\} \cup \{q_2\} \cup \{q_3\}$

$= [q_0, q_1, q_2, q_3]$ // new

$\hat{\delta}(\{q_0, q_1, q_2\}, 1) = \delta(\{q_0\}, 1\} \cup \delta(\{q_1\}, 1\} \cup \delta(\{q_2\}, 1\}$

$= \{q_0\} \cup \{q_1\} \cup \{q_3\}$

$= \{q_0, q_1, q_3\}$ // new

$\hat{\delta}(\{q_0, q_1, q_3\}, 0) = \delta(\{q_0\}, 0) \cup \delta(\{q_1\}, 0) \cup \delta(\{q_3\}, 0)$
$= \{q_0, q_1\} \cup \{q_2\} \cup \{0\}$
$= [q_0, q_1, q_2]$

$\hat{\delta}(\{q_0, q_1, q_3\}, 1) = \delta(\{q_0\}, 1) \cup \delta(\{q_1\}, 1) \cup \delta(\{q_3\}, 1)$
$= q_0 \cup q_1 \cup q_3$
$= [q_0, q_1, q_3]$

$\hat{\delta}(\{q_0, q_1, q_2, q_3\}, 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_2, 0) \cup \delta(q_3, 0)$
$= \{q_0, q_1\} \cup q_2 \cup q_3 \quad 0$
$= [q_0, q_1, q_2, q_3]$

$\hat{\delta}(\{q_0, q_1, q_2, q_3\}, 1) = \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1) \cup \delta(q_3, 1)$
$= q_0 \cup q_1 \cup q_3 \cup q_2$
$= [q_0, q_1, q_2, q_3]$

	S \ i/p	0	1
A	$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
B	q_1	q_2	q_1
C	q_2	q_3	q_3
D	q_3^*	ϕ	q_2
E	q_0, q_1	$q_0 q_1 q_2$	$q_0 q_1$
F	$q_0 q_1 q_2$	$q_0 q_1 q_2 q_3$	$q_0 q_1 q_3$
G	$q_0 q_1 q_3^*$	$q_0 q_1 q_2$	$q_0 q_1 q_3$
H	$q_0 q_1 q_2 q_3^*$	$q_0 q_1 q_2 q_3$	$q_0 q_1 q_2 q_3$

	0	1
$\rightarrow q_0$ (A)	E	A
B q_1	C	B
C q_2	D	D
D q_3^*	$\{\phi\}$	C
E	F	E
F	H	G
G^*	F	G
H^*	H	H

Scanned with OKEN Scanner

Convert NFA to DFA

	0	1
→p	{p, q}	{p}
q	{r, s}	{t}
r	{p, r}	{t}
* s	ϕ	ϕ
* t	ϕ	ϕ

① $\delta[(p,q),0] = \delta(p,0) \cup \delta(q,0)$
$= \{p,q\} \cup \{r,s\}$
$= \{p,q,r,s\}$ // ①

$\delta[(p,q),1] = \delta(p,1) \cup \delta(q,1)$
$= \{p\} \cup \{t\}$
$= \{p,t\}$ // ②

$\delta[(r,s),0] = \delta(r,0) \cup \delta(s,0)$
$= \{p,r\} \cup \{\phi\}$
$= \{p,r\}$ //

$\delta[(r,s),1] = \delta(r,1) \cup \delta(s,1)$
$= \{t\} \cup \{\phi\}$
$= \{t\}$

Total Marks of		Examiner	
		Moderator	

14/44

$$\delta[(p,r),0] = \delta(p,0) \cup \delta(r,0)$$
$$= \{p,q\} \cup \{p,r\}$$
$$= \{p,q,r\}$$

$$\delta[(p,r),1] = \delta(p,1) \cup \delta(r,1)$$
$$= \{p\} \cup \{t\}$$
$$= \{p,t\}$$

(3) $\delta[(p,t),0] = \delta(p,0) \cup \delta(t,0)$
$$= \{p,q\} \cup \phi$$
$$= \{p,q\}$$

$$\delta[(p,t),1] = \delta(p,1) \cup \delta(t,1)$$
$$= \{p\}$$

(2) $\delta[(p,q,r,s),0] = \delta\ \{p,q\}\ \{r,s\}\ \{p,r\}\ [0]$
$$= \{p,q,r,s\}$$

$$\delta[(p,q,r,s),1] = \{p,t\}$$

$$\delta[(p,q,r),0] = \{p,q,r,s\}$$

$$\delta[(p,q,r),1] = \{p,t\}$$

$Q \backslash \Sigma$	0	1
$\rightarrow p$	$\{p,q\}$	$\{p\}$
$\{p,q\}$	$\{p,q,r,s\}$	$\{p,t\}$
$\{p,q,r,s\}^*$	$\{p,q,r,s\}$	$\{p,t\}$
$\{p,t\}^*$	$\{p,q\}$	$\{p\}$

F

E

	S \ i/p	0	1
A	→ p	B	A
B	(p,q)	F	E
~~C~~	~~(r,s)~~		
~~D~~	~~(p,r)~~		
E	(p,t)*	B	A
F	(p,q,r,s)*	F	E
~~G~~	~~(p,q,r)~~		

club

cannot club//

** Steps :-

1. Start from start state and show transition on each symbol.
2. For every ~~stat~~ new start evaluate further on each input symbol (i.e show transitions on new symbols for all possible inputs)
3. Repeat step 2, until ~~we~~ ~~get~~ ~~wont~~ no new state is produced.
4. Final state : where-ever final state appears make that state as final state
5. DFA minimization: minimize DFA if possible,

** if final state and non-final state has same transition it cannot be clubbed

** only final with final or non-final with non-final can be clubbed together.

Total Marks of		Examiner	
		Moderator	

30
14
44

Regular Expression.

The language accepted by finite automata are easily described by simple expressions called regular expressions.

r (regular expⁿ)	L(r) - regular language
a	{a}
b	{b}
a|b	{a, b}
ab	{ab}
a	{a}
a^*	{ε, a, aa, aaa, aaaa, ...}
a^+	{a, aa, aaa, aaaa, ...}
$(a+b)^* = (a\|b)^*$	{ε, a, ab, b, ba, bb, ...}
$(ab)^*$	{ε, ab, abab, ababab, ...}

Operations:

1. Union: L = {001, 10, 111} & M = {ε, 001}

   L ∪ M = {ε, 10, 001, 111}

2. Concatenation: L = {001, 10, 111} & M = {ε, 001}

   LM = {001, 10, 111, 001001, ~~[illegible]~~ 10001, 111001}

3. Closure

Specify the language using regular expression in which the strings are

(i) valid if they contain even no. of a's over $\Sigma = \{a\}$

$\Rightarrow$ $r = (aa)^*$

$L(r) = \{aa, \epsilon, aaaa, \ldots\}$

(ii) if they contain odd no of 0's over $\Sigma = \{0, 1\}$

$\Rightarrow$ $r = \{00\}^* 0$

$L(r) = \{\epsilon, 0, 000, 00000, \ldots\}$

(iii) if they contain any combination of 0's and 1's over $\Sigma = \{0, 1\}$

$\Rightarrow$ $r = (0+1)^*$ or $(0|1)^*$

$L(r) = \{\epsilon, 0, 1, 01, 10, 00, 11, \ldots\}$

(iv) if string starts with a and end with b over $\Sigma = \{a, b\}$

$r = a(a+b)^* b = a(a^* b^*)^* b$

$= a(a|b)^* b$

$= a(a^*|b^*)^* b$

(v) starts and ends with b over $\Sigma = \{a, b\}$

$r = b(a+b)^* b + b$

Total Marks of		Examiner	
		Moderator	

30
14
44

(vi) if starts and ends with different letter over $\Sigma = \{a, b\}$

$r = a(a+b)^* b + b(a+b)^* a$

(vii) if starts and ends with same letter over $\Sigma = \{a, b\}$

$r = a(a+b)^* a + b(a+b)^* b + a + b$

(viii) if starts with 10 and ends with 01 over $\Sigma = \{a, b\}$

$r = 10(0+1)^* 01 + 101$

(ix) if starts with aab and ends with baa over $\Sigma = \{a, b\}$

$r = aab(a+b)^* baa + aabaa$

$L(r) = \{aabaa, aabbaa, \ldots\}$

(x) if starts with abb and ends with bba over $\Sigma = \{a, b\}$

$r = abb(a+b)^* bba + abbba + abba$

(xi) if it contains exactly one a over the $\Sigma = \{a, b\}$

$r = ab^* + b^* a + b^* a b^* = b^* a b^*$

(xii) if the third symbol from beginning is 'a' over $\Sigma = \{a, b\}$

$r = (a+b)(a+b)a(a+b)^*$

(xiii) if they contain (i) exactly one 0

(ii) atmost two 0's

(iii) atleast three 0's over $\Sigma = \{0,1\}$

(i) $r = 1^* 0 1^*$

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

or

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

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

(xiv) if string end in double letter

$r = (a+b)^* (aa+bb)$

(xv) if string starts & end in double letter

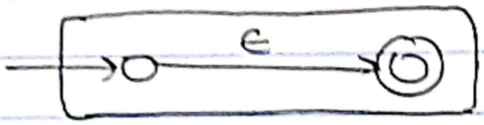
$r = (00+11)(0+1)^*(00+11) + 00 + 11 + 000 + 111$

Regular Expression to NFA.

Divide the given regular expression into smaller subexpressions and construct NFA for each using rules

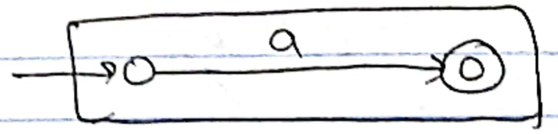
① Basic construction of an automata from a regular expression

(i) $r = \epsilon$



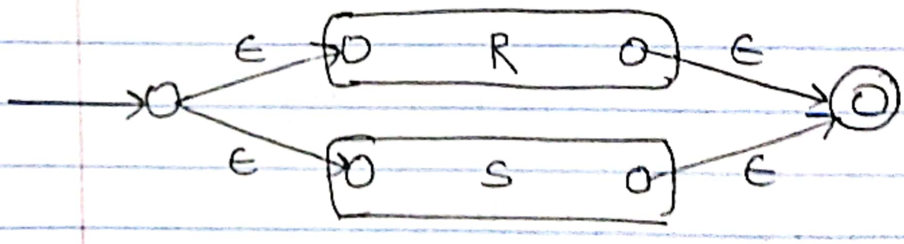
(ii) $r = \phi$

(iii) $r = a$

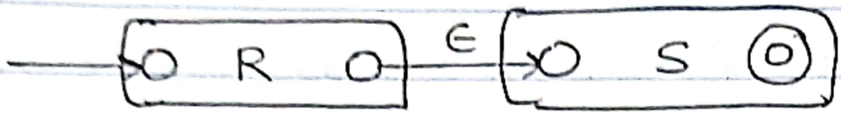


② Steps in regular exp to ε-NFA construction

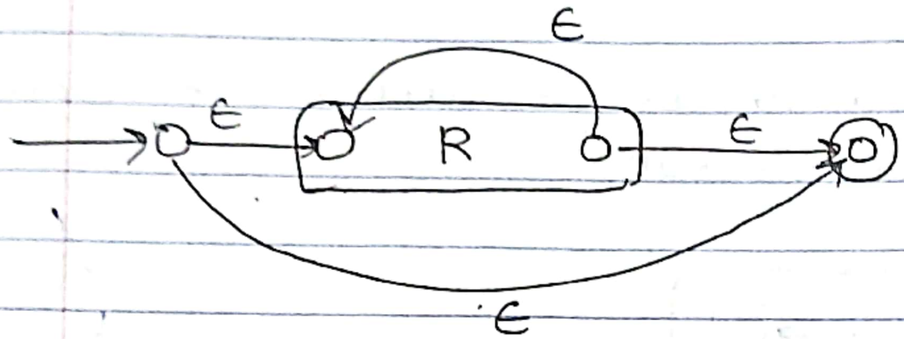
(i) $r = R|S$ or $r = R+S$



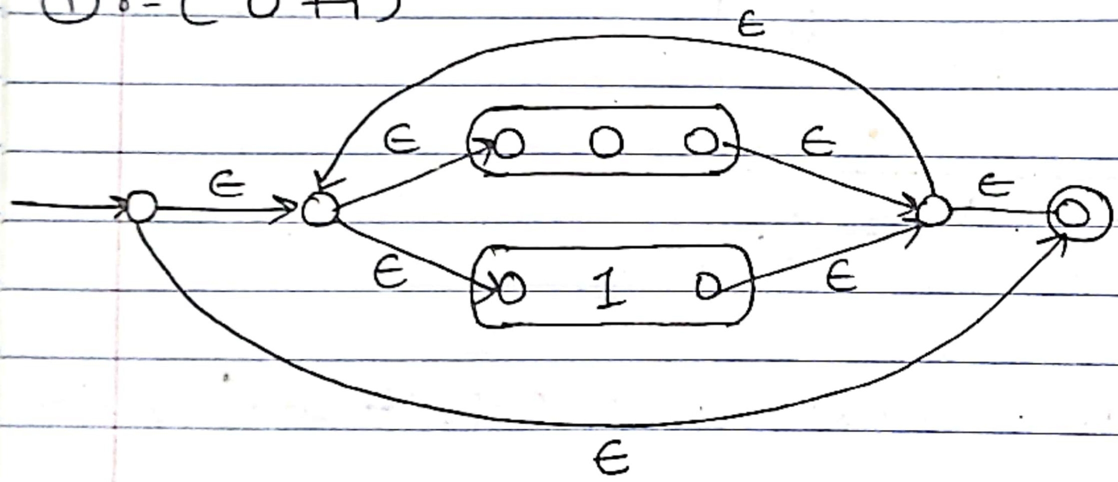
(ii) $r = R \cdot S$



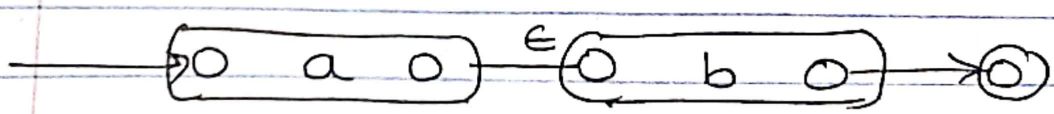
(iii) $r = R^*$

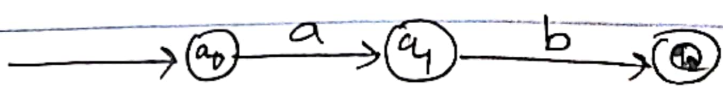


① $r = (0+1)^*$



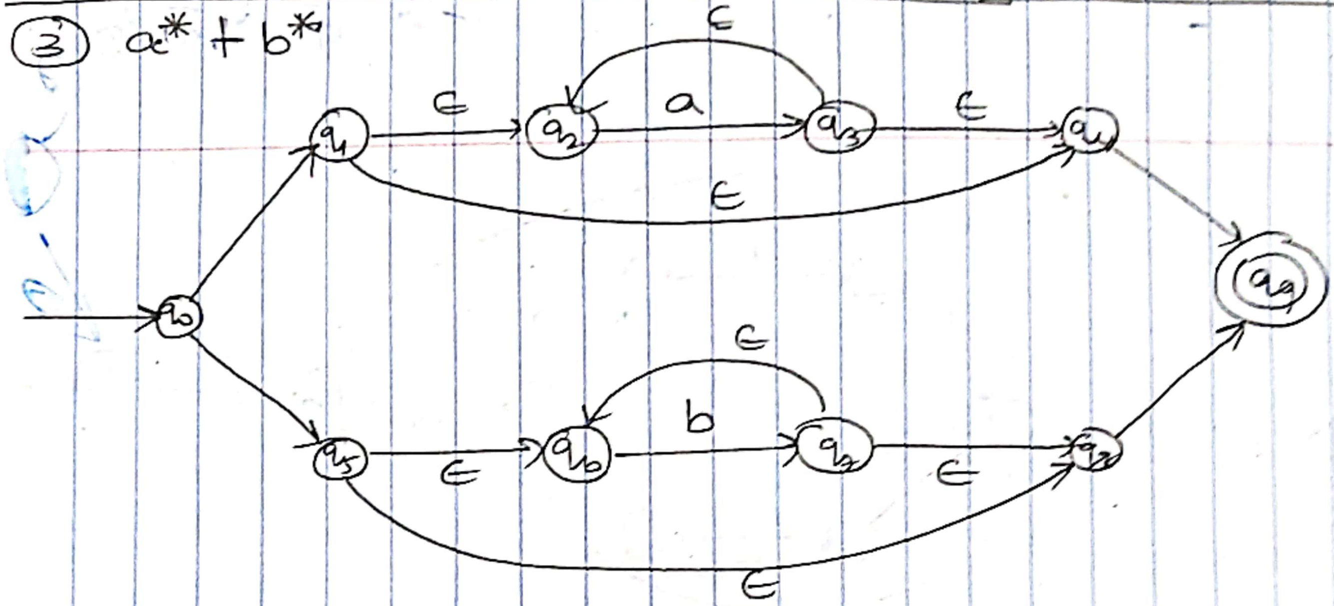
② $r = ab$



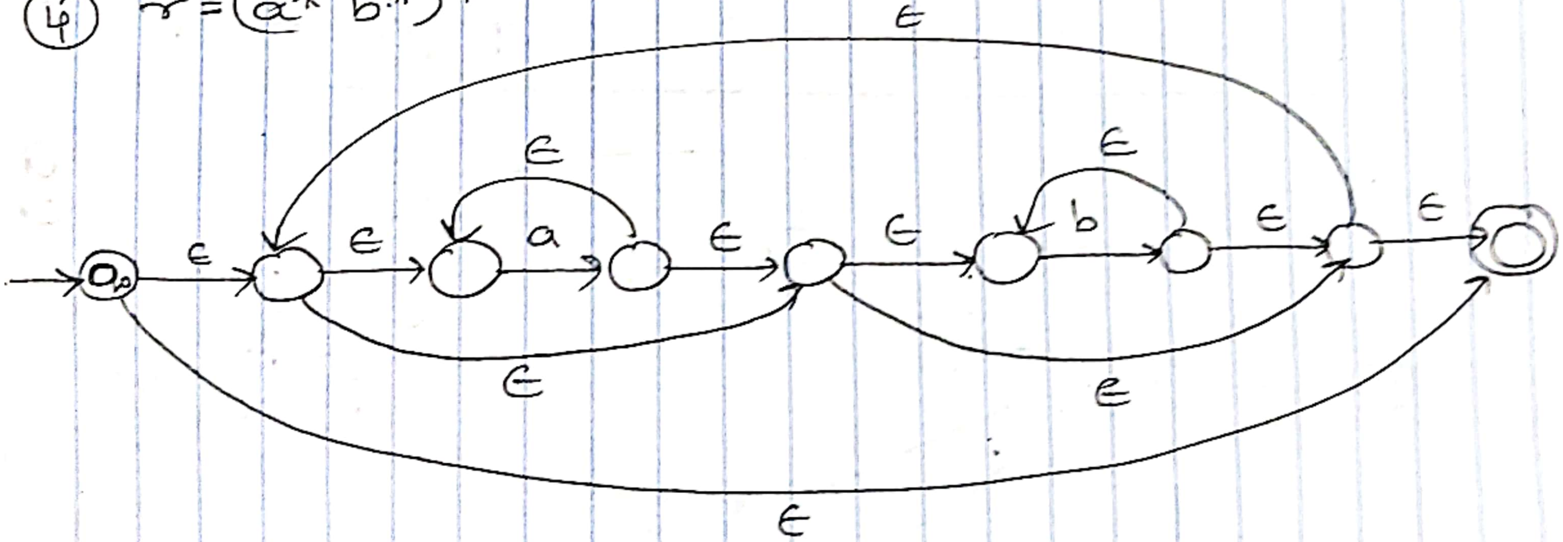


~~(0+1)*~~ ~~(0+1)~~ ① $(0+1)^* \mid (0+1)$

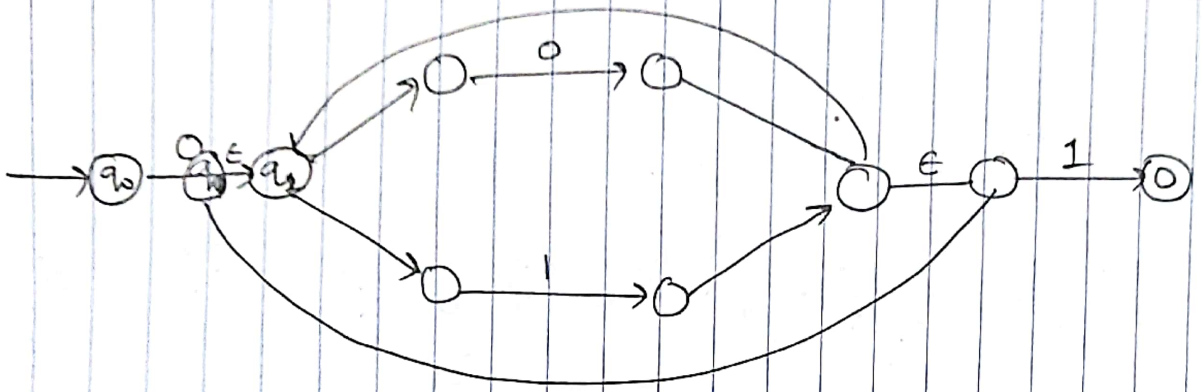
③ $a^* + b^*$



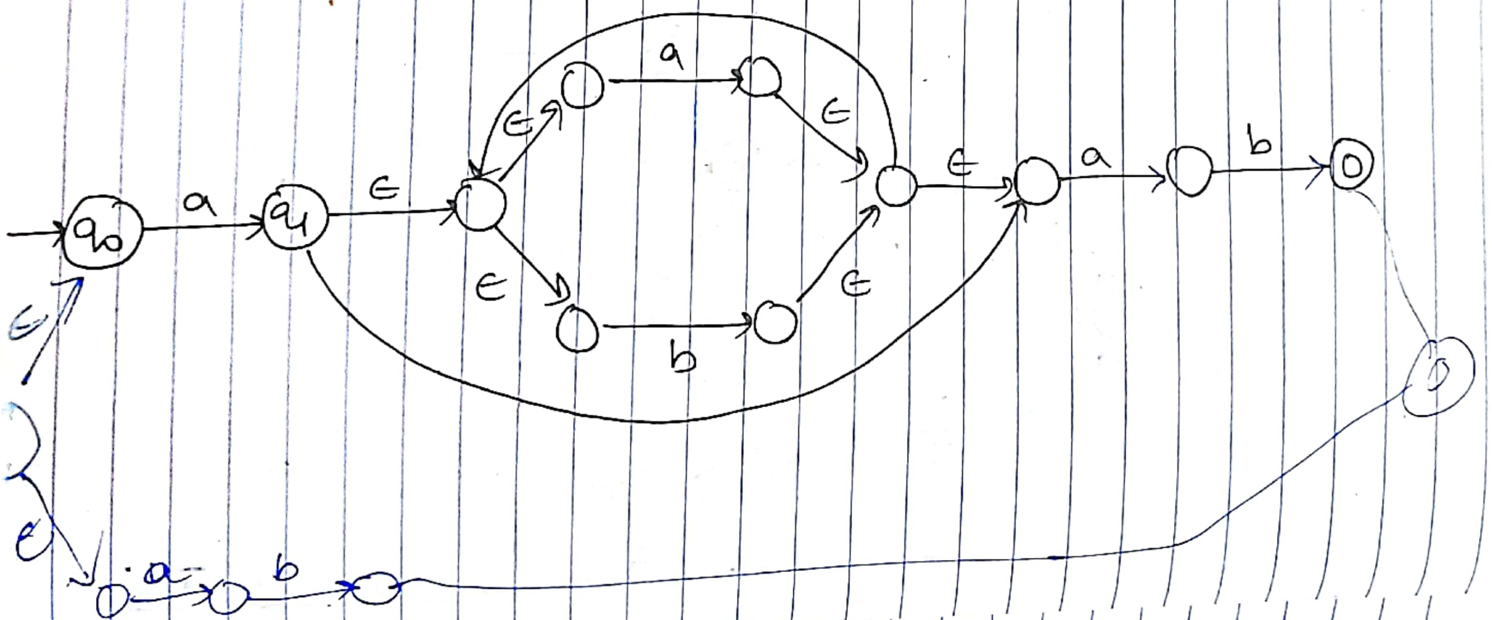
④ $r = (a^* b^*)^*$



⑤ $0(0+1)^*1$

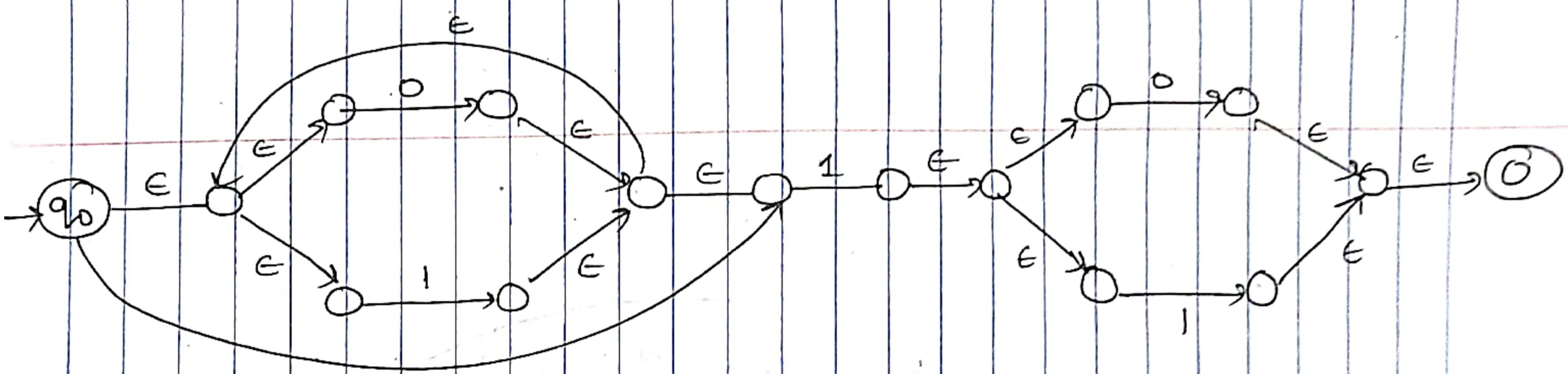


⑥ $a(a+b)^*ab$

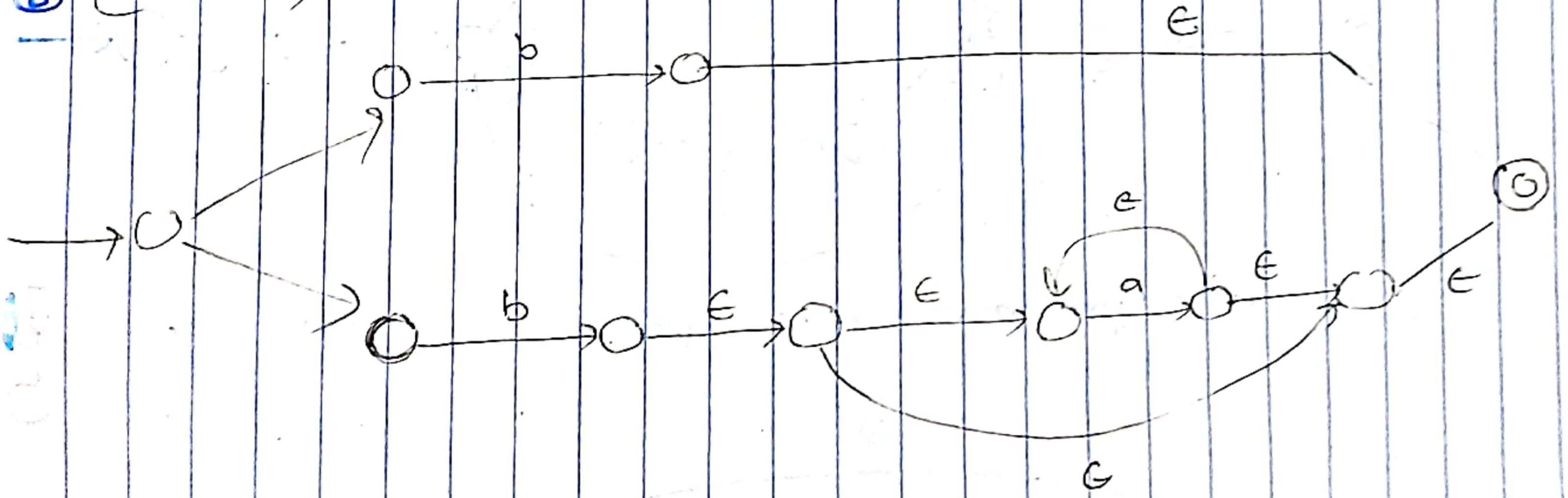


⑦ $(0+1)^*1(0+1)$

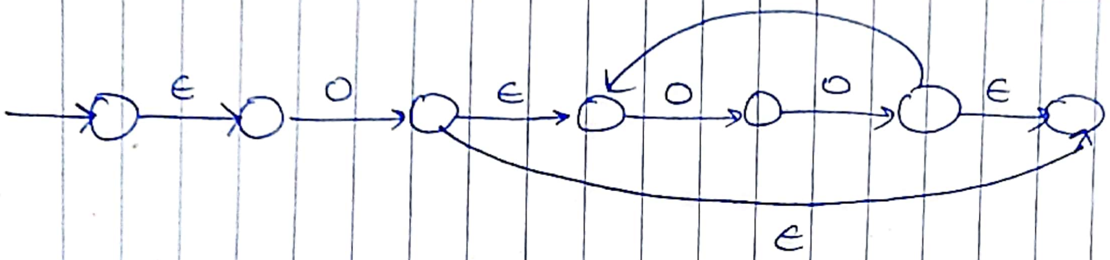
7) $(0+1)^* 1 (0+1)$



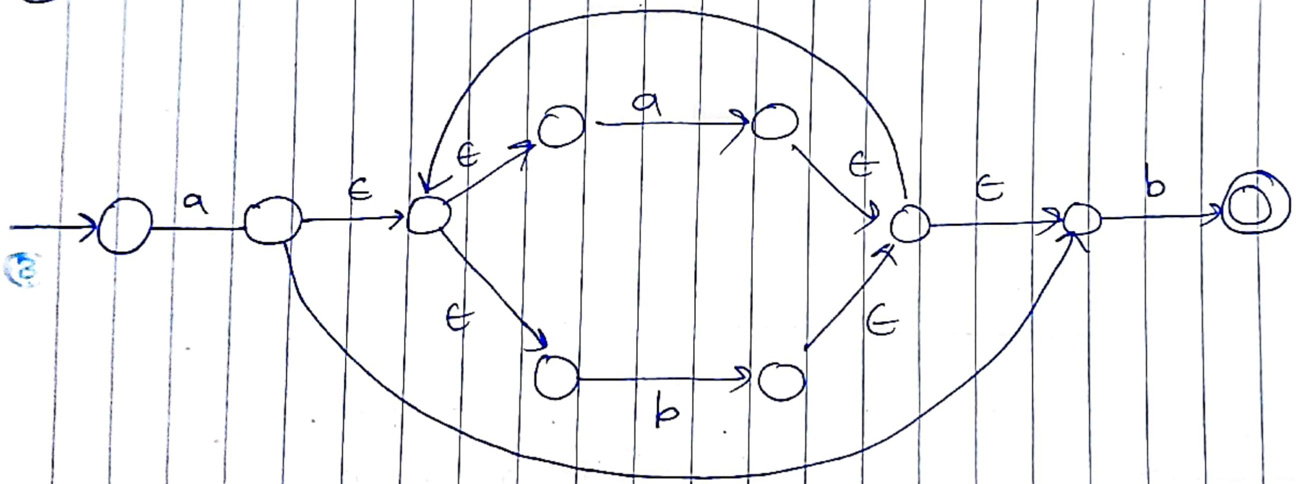
8) $(b+ba^*)$



9) $0(00)^*$ odd number of 0's

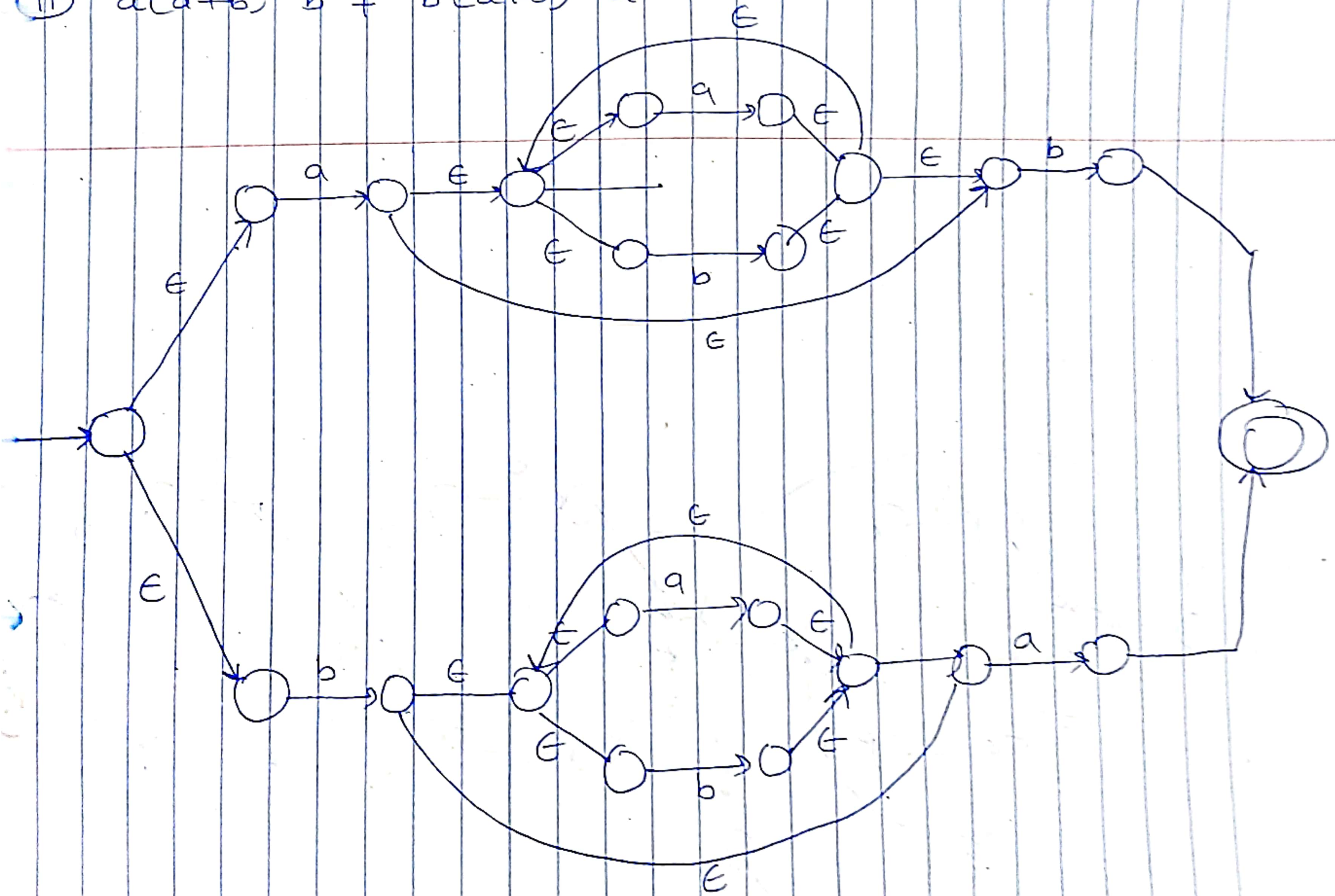


10) $r = a(a+b)^*b$ (start and end with a and b)



11) $a(a+b)^*b + b(a+b)^*a$

(11) $a(a+b)^*b + b(a+b)^*a$



Total Marks of

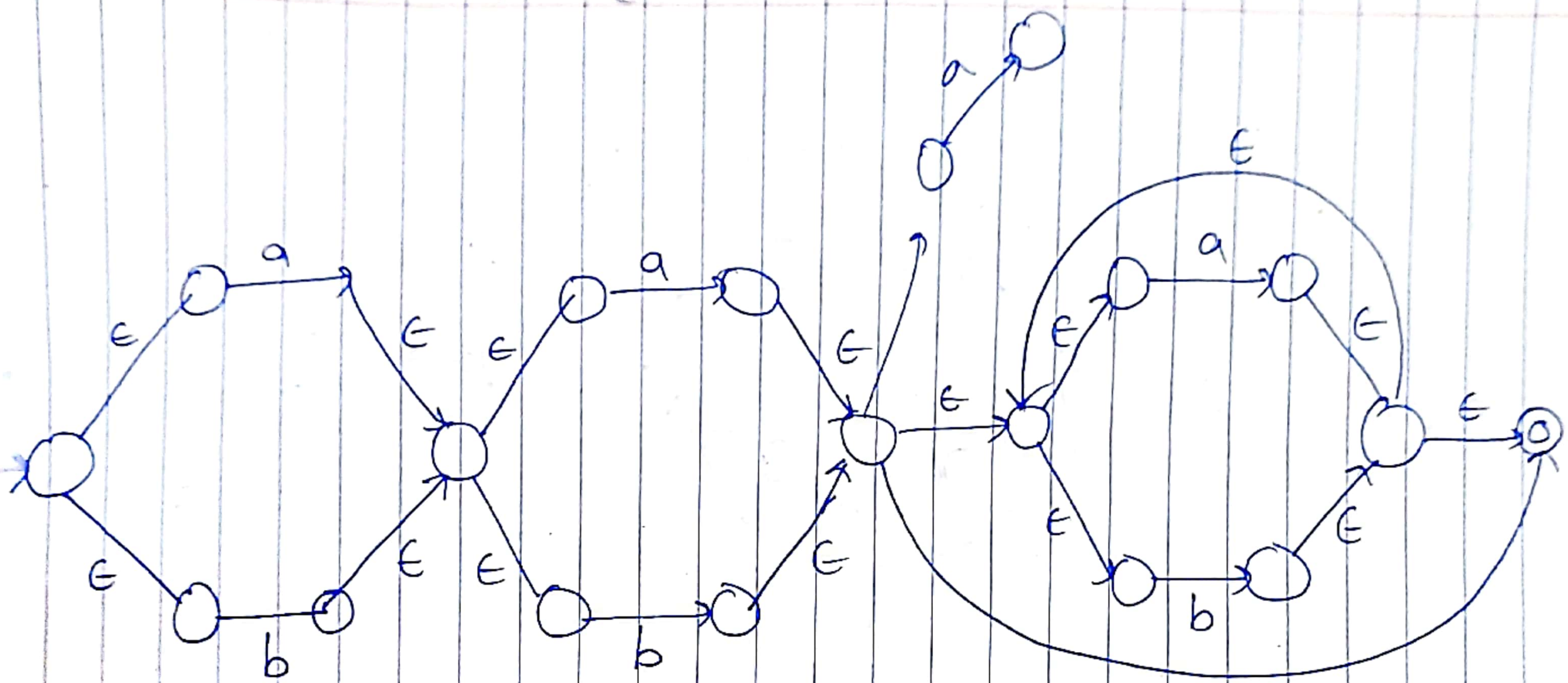
Examiner

Moderator

30
14
44

20221222_091444.jpg

⑫ $(a+b)(a+b)a(a+b)^*$



| Moderator |

R.E → NFA w ε → NFA w/o ε → DFA → minimized DFA

** R.E to DFA

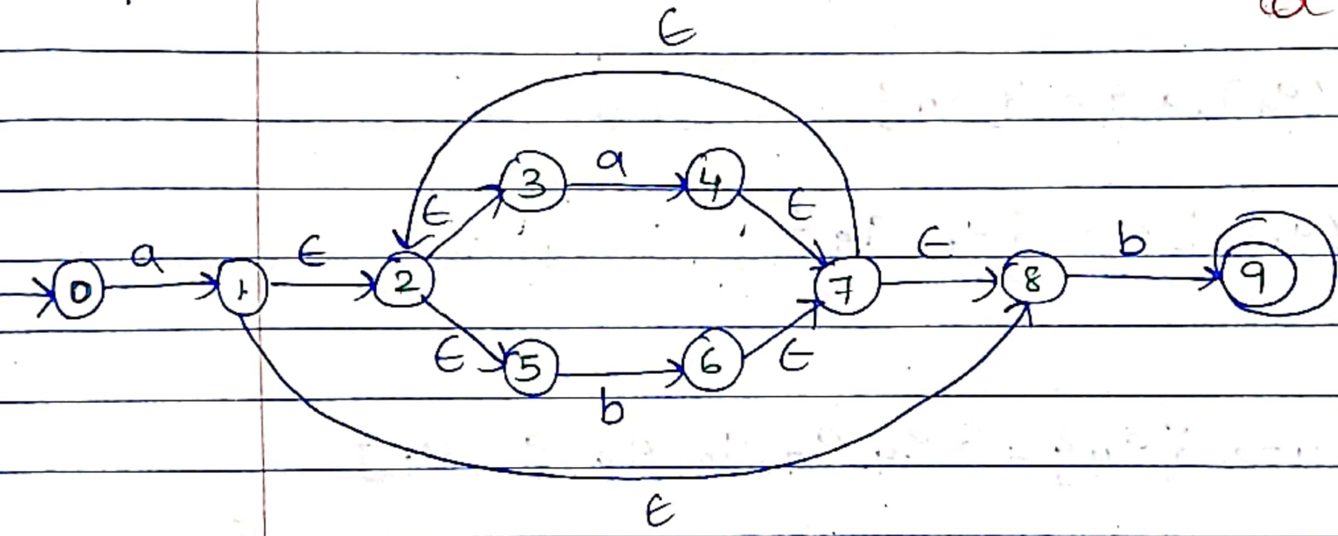
① Construct minimized DFA to recognize strings starting with 'a' and ending with 'b', $\Sigma = \{a, b\}$

Step 1: Construct reg exp

$r = a(a+b)^* b$

Step 2: Convert RE to NFA

a a



Step 3: NFA to DFA

	x	y = εclosure of x	$\delta(\epsilon(x), a)$ $\delta(y, a)$	$\delta(\epsilon(x), b)$ $\delta(y, b)$
A	{0}	{0}	{1} B	{∅} E
B	{1}	{1, 2, 3, 5, 8}	{4} C	{6, 9}
C	{4}	{4, 7, 2, 8, 3, 5}	{4}	{6, 9}
D	{6, 9}	{6, 7, 8, 2, 3, 5, 9}	[illegible]	[illegible]

2/6/23, 9:30 AM   20221222_091444.jpg

Total Marks of		Examiner	$\frac{30}{14}{44}$

R.E $\rightarrow$ NFA w ϵ $\rightarrow$ NFA w/o ϵ $\rightarrow$ DFA $\rightarrow$ minimized DFA

** R.E to DFA

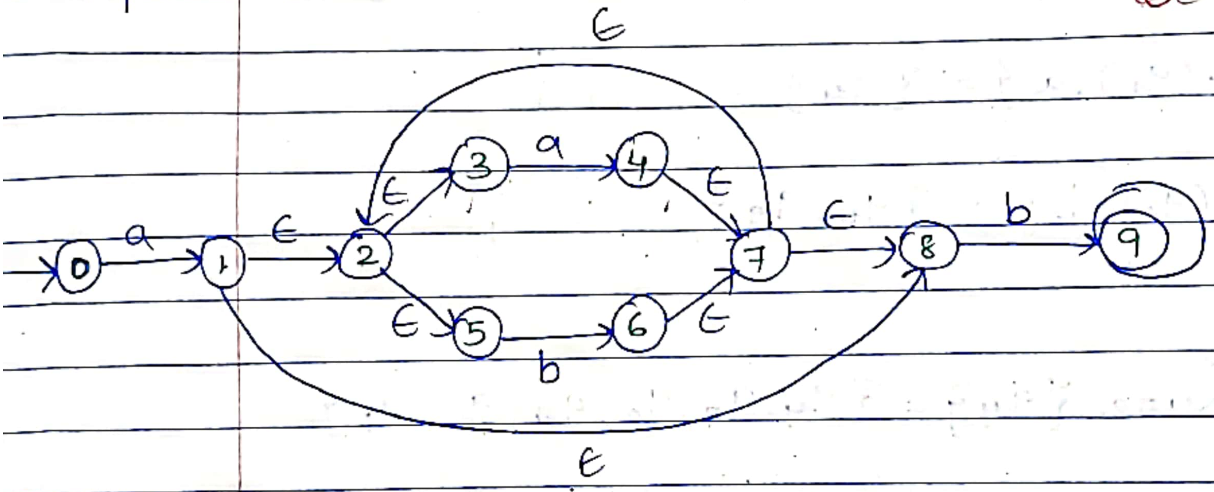
(1) Construct minimized DFA to recognize strings starting with 'a' and ending with 'b', $\Sigma = \{a, b\}$

Step 1: Construct reg exp

$r = a(a+b)^* b$

Step 2: Convert RE to NFA

a a



Step 3: NFA to DFA

x	y = Eclosure of x	$\delta(E(x), a)$ $\delta(y, a)$	$\delta(E(x), b)$ $\delta(y, b)$
A $\{0\}$	$\{0\}$	$\{1\}$ B	$\{\emptyset\}$ E
B $\{1\}$	$\{1, 2, 3, 5, 8\}$	$\{4\}$ C	$\{6, 9\}$
C $\{4\}$	$\{4, 7, 2, 8, 3, 5\}$	$\{4\}$	$\{6, 9\}$
D $\{6, 9\}$	$\{6, 7, 8, 2, 3, 5, 9\}$	[illegible]	[illegible]

$\rightarrow \{q_0\}$ A

ϵ-closure $\{q_0\} = \{q_0\}$

$\hat{\delta}\{\epsilon(q_0), a\} = \{q_1\}'$ new state

$\hat{\delta}\{\epsilon(q_0), b\} = \{\ \}$

$\rightarrow \{q_1\}$ B

ϵ-closure $\{q_1\} = \{q_1, q_2, q_3, q_5, q_8\}$

$\hat{\delta}\{\epsilon(q_1), a\} = \{q_4\}$ new state

$\hat{\delta}\{\epsilon(q_1), b\} = \{q_6, q_9\}$ new state

$\rightarrow \{q_4\}$ C

ϵ-closure $\{q_4\} = \{q_4, q_7, q_2, q_8, q_3, q_5\}$

$\hat{\delta}\{\epsilon(q_4), a\} = \{q_4\}$

$\hat{\delta}\{\epsilon(q_4), b\} = \{q_6, q_9\}$

$\rightarrow \{q_6, q_9\}$ D

ϵ-closure $\{q_6, q_9\} = \{q_6, q_7, q_8, q_2, q_3, q_5, q_9\}$

$\hat{\delta}\{\epsilon(q_6, q_9), a\} = \{q_4\}$

$\hat{\delta}\{\epsilon(q_6, q_9), b\} = \{q_6, q_9\}$

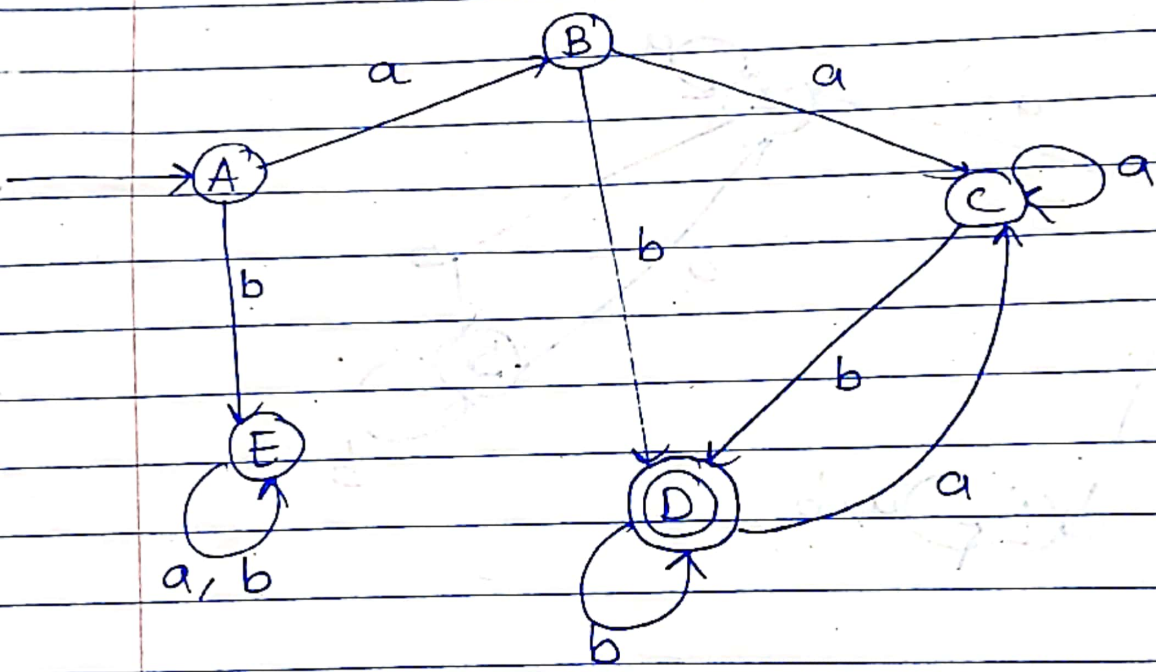
E

Total Marks of		Examiner	
		Moderator	

	s \ i/p	a	b
$\{q_0\}$ →	A	B	E
$\{q_1\}$	B	C	D
$\{q_4\}$ ε-closure	C	C	D
$\{q_6, q_7\}$	D*	C	D
$\{\}$	E	E	E

club ∵ both are non-final states

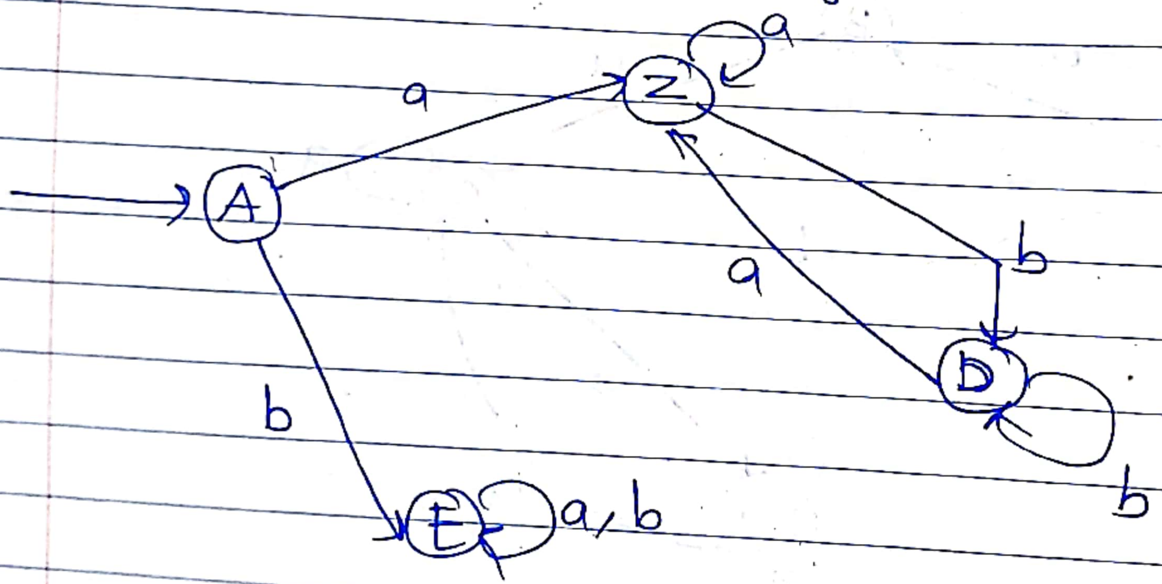
state transition diagram



Step 4: Minimized DFA

S \ i/p	a	b
→ A	Z	E
Z	Z	D
D*	Z	D
E	E	E

state transition diagram.

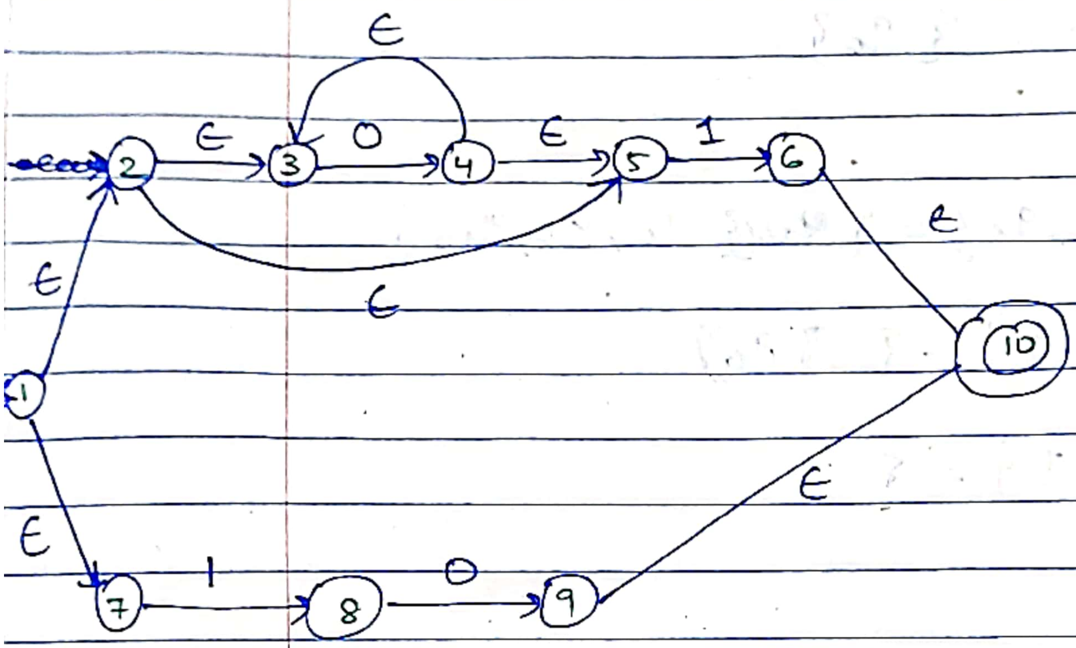


Total Marks of		Examiner	30 14 44
		Moderator	

(2)

Step 1 : $r = 0^* 1 + 10$

Step 2 :



Step 3 :

A $\{q_1\}$

$\epsilon\text{-closure} = \{q_1, q_2, q_3, q_5, q_7\}$

$\hat{\delta}\{\epsilon(q_1), 0\} = \{q_4\}$ new state

$\hat{\delta}\{\epsilon(q_1), 1\} = \{q_6, q_8\}$ new state

B $\{q_4\}$

ϵ-closure $\{q_4\} = \{q_4, q_5, q_3\}$

$\hat{\delta}\{\epsilon(q_4), 0\} = \{q_4\}$

$\hat{\delta}\{\epsilon(q_4), 1\} = \{q_6\}$ //new state

C $\{q_6, q_8\}$

ϵ-closure $\{q_6, q_8\} = \{\cancel{\quad} q_6, q_8, q_{10}\}$

$\hat{\delta}\{\epsilon(q_6, q_8), 0\} = \{\ \cancel{\}}\ q_9\}$

$\hat{\delta}\{\epsilon(q_6, q_8), 1\} = \{\ \}$

D $\{q_6\}$

ϵ-closure $\{q_6\} = \{\cancel{\quad} q_6, q_{10}\}$

$\hat{\delta}\{\epsilon(q_6), 0\} = \{\}$

$\hat{\delta}\{\epsilon(q_6), 1\} = \{\ \}$

E $\{\ \}$ $\{\ \}$ $\{\}$ $\{\ \}$

$\{q_9\}$

F ϵ-closure $\{q_9\} = \{q_9, q_{10}\}$

$\hat{\delta}(\epsilon(q_9), 0) = \{\}$

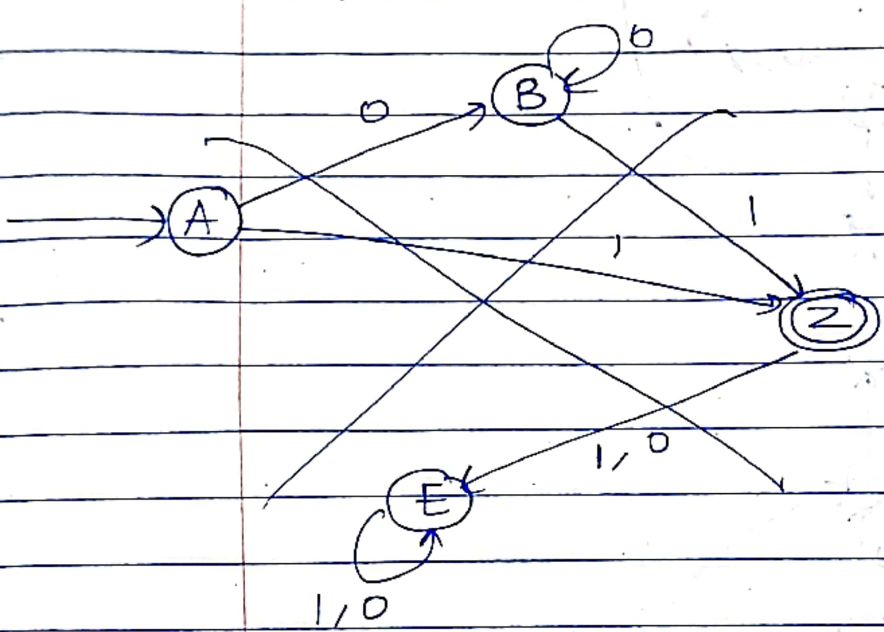
$\hat{\delta}(\epsilon(q_9, 1) = \{\}$

20221222_091444.jpg

Total Marks of | Examiner | 30 / 14 / 44

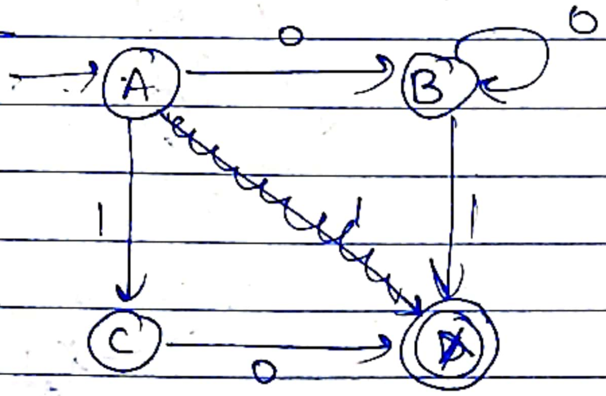
Moderator

S \ i/p	0	1
→ A	B	C
B	B	D
C*	~~E~~ F	E
D*	E	E
E	E	E
F*	E	E



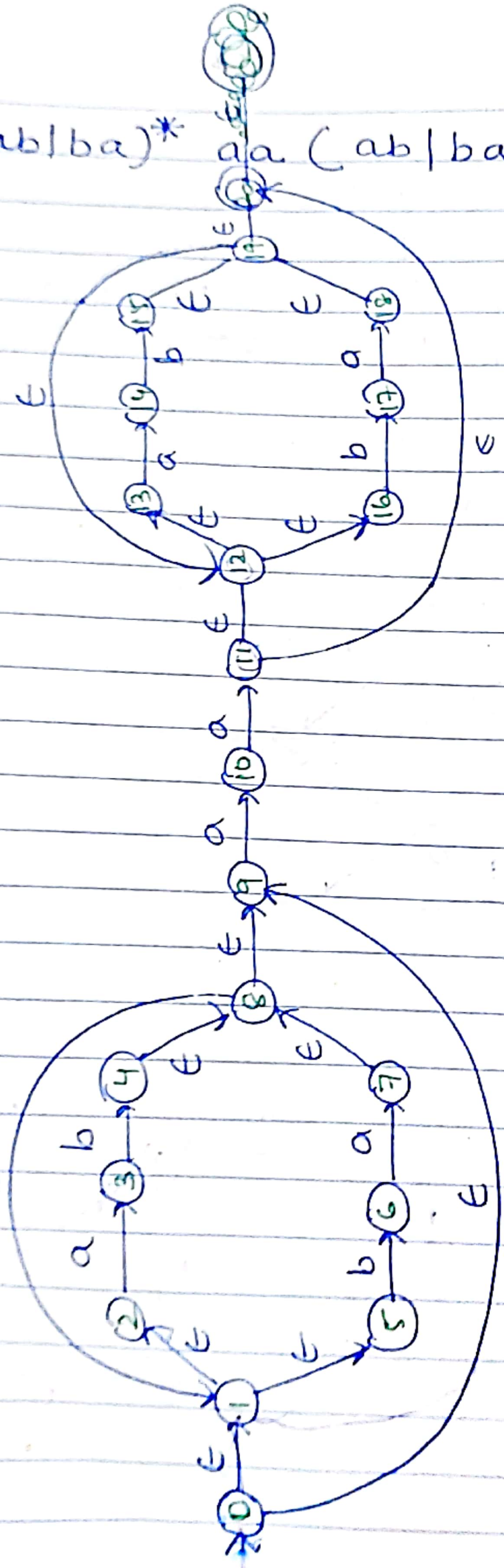
0001

S \ i/p	0	1
→ A	B	C
B	B	~~D~~
C*	~~F~~	E
~~X~~*	E	E
E	E	E



Total Marks of

③ $r = (ab|ba)^* \, aa \, (ab|ba)^*$



abab aaba

x / a
{0.}
{3,10}
{6}
{11}
{4}
{7}
{14}
{17
{15
{18
{

q	y = ε-closure	δ(y, a)	δ(y, b)
{0}	{0,1,2,5,8 }	{3,10}	{6}
{3,10}	{3,10}	{11}	{4}
{6}	{6}	{7}	{ }
{11}	{11,12,13,16,20}	{14}	{17}
{4}	{4,8,1,9,2,5}	{3,10}	{6}
{7}	{7,8,9,1,2,5}	{10,3}	{6}
{14}	{14}	{ }	{15}
{17}	{17}	{18}	{ }
{15}	{15,19,20,12,13,16}	{14}	{17}
{18}	{18,19,20,12,13,16}	{14}	{17}
{ }	{ }	{ }	{ }

DFA
FSM.

b
baa
baa *
a
ab*

Finite Automata with o/p

* Moore Machine

→ Moore m/c is a FA that produces the o/p sequence for the given i/p sequence

→ It is a FA with no final state

→ New symbol, Δ - o/p alphabet and λ - o/p mapping

→ There is a symbol (op) associated with each state

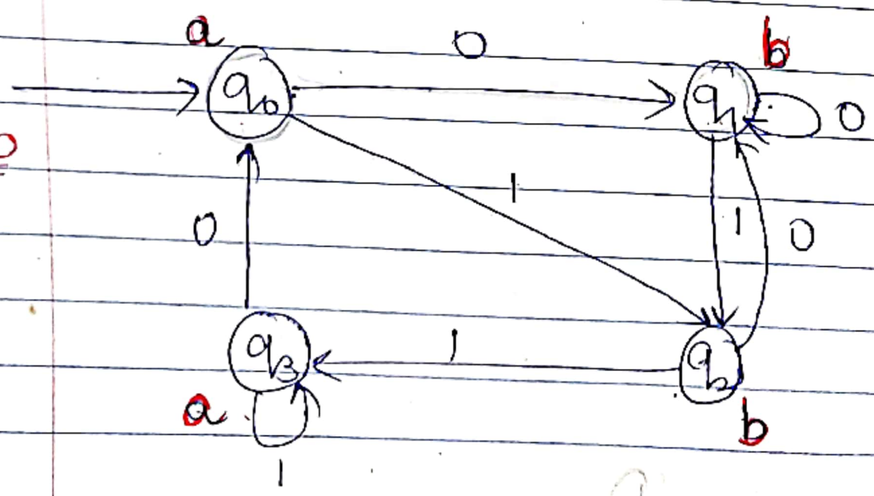
$$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$$

where, 6-tuples

Q = Finite set of states
Σ = i/p alphabet
Δ = o/p alphabet
δ = transition function
λ = o/p mapping
q_0 = start state

abbaa
n+1
0011

i/p ~~abba~~ 0110
abbaa



0110
n

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

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

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

q_0 = start state

$Q \setminus \Sigma$	0	1	λ
$\rightarrow q_0$	q_1	q_2	a
q_1	q_1	q_2	b
q_2	q_1	q_3	b
q_3	q_0	q_3	a

$(q_0, \overset{n}{1100})$ $\underbrace{\text{abaab}}_{n+1}$

$\vdash (q_2, 100)$

$\vdash (q_3, 00)$

$\vdash (q_0, 0)$

$\vdash (q_1, \epsilon)$

O/P

Note: In moore m/c, if the length of i/p sequence is n then the length of o/p sequence is n+1.

Mealy machine :-

→ It is a FA with no final state and it produces the o/p sequence for the given i/p sequence.

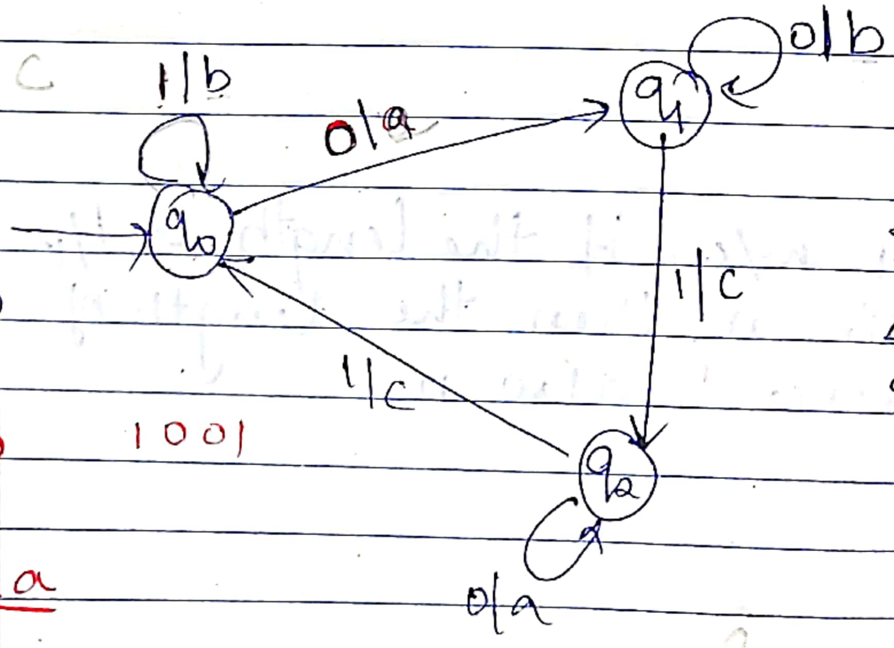
→ In mealy m/c, there is a symbol associated with each transition. Such a symbol is called a o/p symbol.

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

Q = finite set of states
Σ = i/p alphabet
Δ = o/p alphabet
δ = transition function
λ = o/p mapping
q_0 = start state

0 0 1 1
n

a b c c
n



$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{1, 0\}$
$\Delta = \{a, b, c\}$
q_0 = Start state

1010
baca

0110
n

1001

acca
n

δ :-

Q \ Σ	0	1	
→ q_0	q_1	q_0	
q_1	q_1	q_2	
q_2	q_2	q_0	

λ :-

Q \ Σ	0	1
→ q_0	a	b
q_1	b	c
q_2	a	c

o/p's

$(q_0, 11\overset{n}{0}0)$

$\vdash (q_0, 100)$

$\vdash (q_0, 00)$

$\vdash (q_1, 0)$

$\vdash (q_1, \epsilon)$

n
bbab

Note:- In mealy m/c, if the length of i/p sequence in n then the length of o/p sequence is also n.

Minimization of Moore m/c

Q. Design Moore m/c to o/p 'A' if ends in 101
'B' if ends in 110
'C' otherwise over {0,1}

Moore

Σ	a	b	o/p (λ)
$\rightarrow q_0$	q_0	q_1	C
~~q_1~~	~~q_0~~	~~q_2~~	C
q_1	q_2	q_4	C
q_2	q_0	q_3	C
q_3	q_2	q_4	A
q_4	q_5	q_4	C
q_5	q_0	q_3	B

Mealy

Σ	0	1
$\rightarrow q_0$	C	C
q_1	C	C
q_2	C	C
q_3	C	A
q_4	C	C
q_5	B	C
q_6	C	C

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$

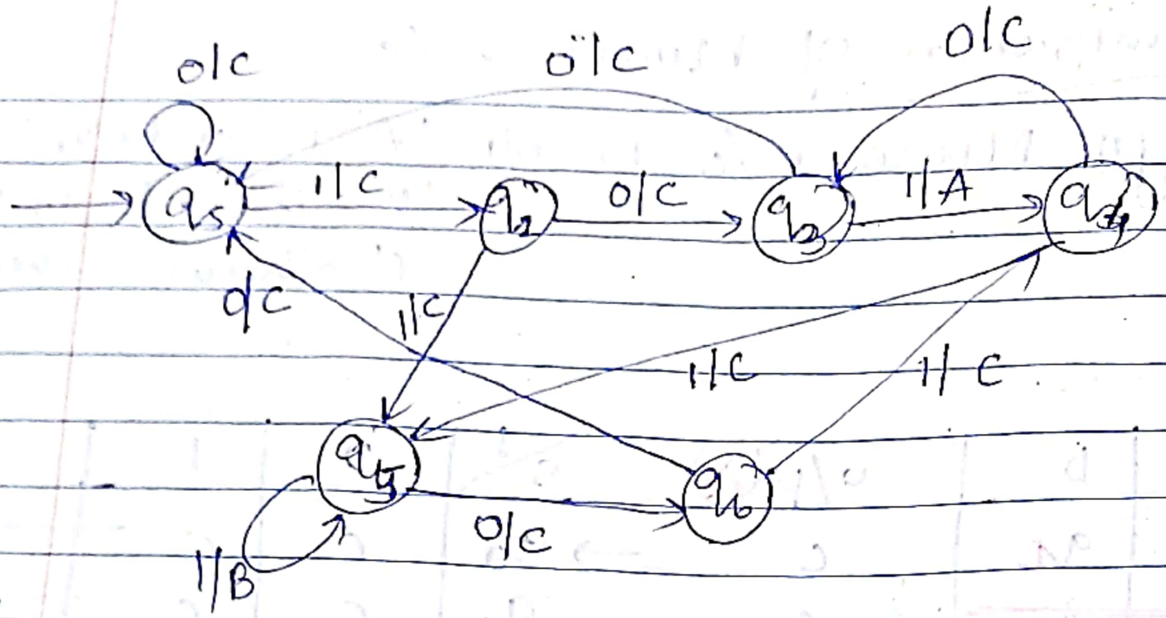
$\Sigma = \{0, 1\}$

$\Delta = \{A, B, C\}$

q_0: start state

Minimization rule:- States can be merged if
1. All the states are the same transition and
2. o/p symbols associated with the states are same.

$\rightarrow q_0$ | | | | |



Example

$(q_0, 1100)$ $(q_0, 0101)$

$\vdash (q_2, 100)$

$\vdash (q_3, 00)$

$\vdash (q_0, 0)$

$\vdash (q_4, \epsilon)$

CCCCC

1015 20

10 5 100 101

5
10
15
20

150

(Q) Design Moore m/c to o/p the reminder when binary number is divided by 3.

2 cond"

Q \ Σ	0	1	o/p
q_s → q_0	q_1	q_2	0
0 q_1	q_1	q_2	0
1 q_2	q_3	q_1	1
2 q_3	q_2	q_3	2

0 1

00 10 11
01

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

$Q = \{ \cancel{q_0}, q_s, q_2, q_3 \}$

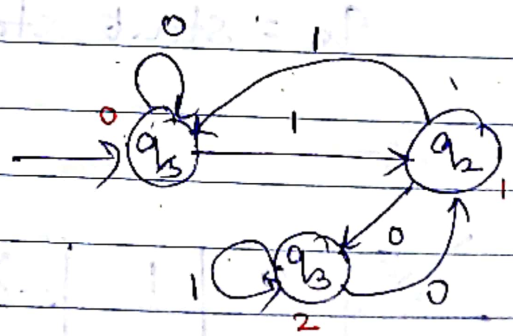
$\Sigma = \{0, 1\}$

$\Delta = \{0, 1, 2\}$

q_s = { start state

δ:-

Q \ Σ	0	1	λ
→ q_s	q_s	q_2	0
q_2	q_3	q_s	1
q_3	q_2	q_3	2



$(q_s, \overset{n}{1100})$

$\vdash (q_2, 100)$

$\vdash (q_s, 00)$

$\vdash (q_s, 0)$

$\vdash (q_s, \epsilon)$

$\underset{n+1}{01000}$

(5) Design Moore m/c to change each occurrence of (1000) to (1001)

$Q \backslash \Sigma$		0	1	λ	
$\rightarrow$	q_0	q_1	q_2	0	
0	q_1	q_1	q_2	0	
1	q_2	q_3	q_2	1	1
10	q_3	q_4	q_2	0	0
100	q_4	q_5	q_2	0	1
1000	q_5	q_1	q_2	1	0

q_s { q_0, q_1

1000
$\Downarrow$
1010

1001
1001

1000
1010

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

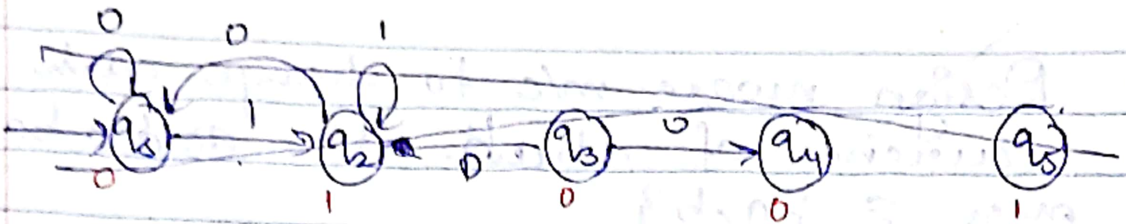
$Q = \{q_s, q_2, q_3, q_4, q_5\}$
$\Sigma = \{0, 1\}$
$\Delta = \{0, 1\}$
q_s = start state

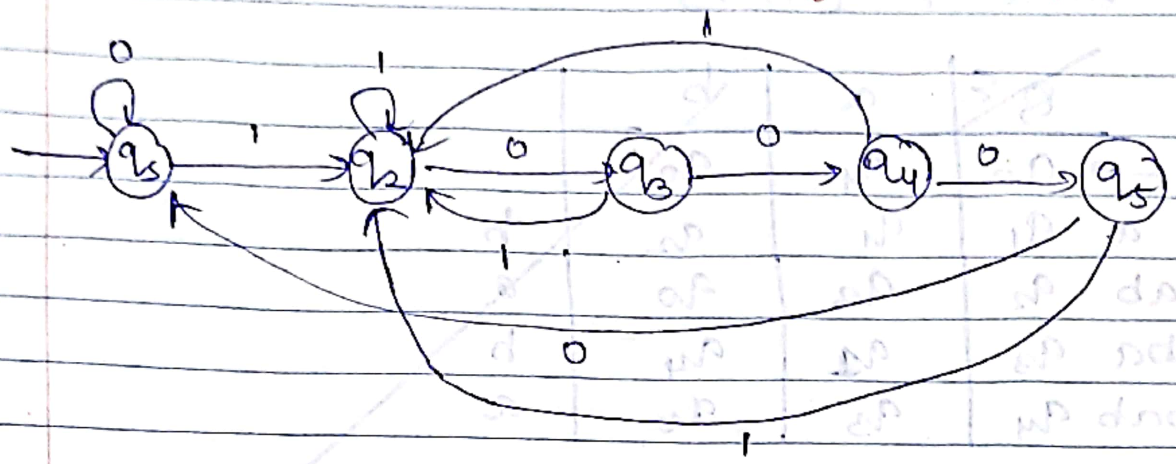
δ:-

$Q \backslash \Sigma$	0	1	λ
$\rightarrow q_s$	q_s	q_2	0
q_2	q_3	q_2	1
q_3	q_4	q_2	0
q_4	q_5	q_2	0
q_5	q_s	q_2	1

Scanned with OKEN Scanner

abb aba ab





$(q_s, 1000)$

$\vdash (q_2, 000)$

$\vdash (q_3, 00)$

$\vdash (q_4, 0)$

$\vdash (q_5, \epsilon)$

aba q_3	q_4	q_4	b
abab q_4	q_3	q_6	a

4) Design moore m/c to o/p 'x' if i/p does not contain a double letter, 'y' otherwise over $\Sigma = \{0, 1\}$

$Q \backslash \Sigma$	0	1	
$\rightarrow q_0$	q_1	q_2	x
0 q_1	q_3	q_2	x
1 q_2	q_1	q_4	x
00 q_3	q_3	q_3	y
11 q_4	q_4	q_4	y

q_d

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

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

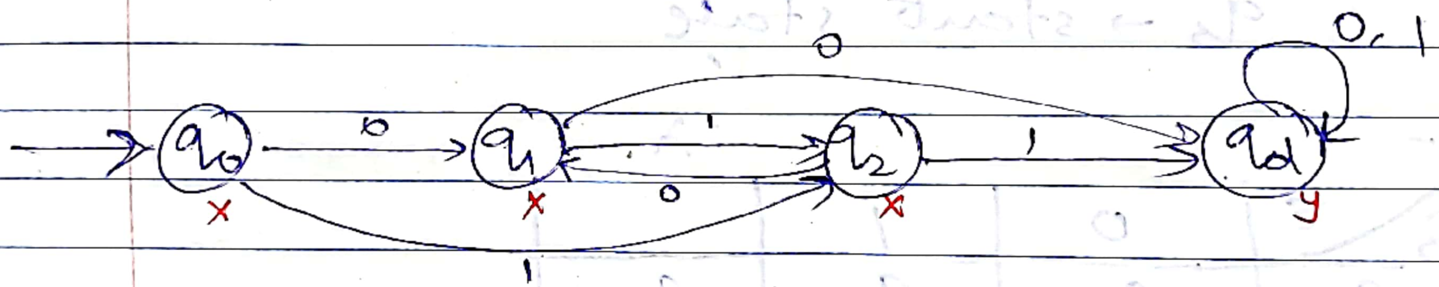
$\Sigma = \{0, 1\}$

$\Delta = \{x, y\}$

q_0 = start state

δ :-

Q \ Σ	0	1	λ
→ q_0	q_1	q_2	x
q_1	q_d	q_2	x
q_2	q_1	q_d	x
q_d	q_d	q_d	y



$(q_0, 1100)$

$\vdash (q_2, 100)$

$\vdash (q_d, 00)$

$\vdash (q_d, 0)$

$\vdash (q_d, \epsilon)$

x x y y y

(5) Design Moore m/c to o/p "ODD" and "EVEN" depending on the number of 1's over Σ={0,1}

Q \ Σ	0	1	λ
→ q_0	q_1	q_2	EVEN
even q_1	q_1	q_2	EVEN
odd q_2	q_2	q_1	ODD

$q_s \{$

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

$Q = \{q_s, q_2\}$

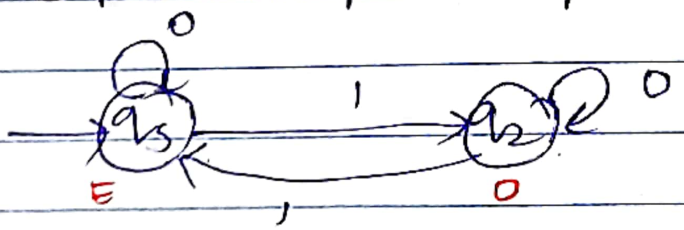
$\Sigma = \{0, 1\}$

$\Delta = \{EVEN, ODD\}$

$q_s \rightarrow$ start state

δ:-

Q \ Σ	0	1	λ
→ q_s	q_s	q_2	Even
q_2	q_2	q_s	ODD



$(q_s, 01)$ even even odd

$\vdash (q_s, 1)$

$\vdash (q_2, \epsilon)$

$\Rightarrow$

Minimization of Mealy m/c

Steps:-
States can be minimized if
(i) all the states have same transitions
(ii) output symbols associated with the transitions are same

① Design Mealy m/c to o/p 'A' if it end in 101
'B' otherwise over $\Sigma = \{0, 1\}$.

	Q \ Σ	0	1
	$\rightarrow q_0$	q_1	q_2
0	q_1	q_1	q_2
1	q_2	q_3	q_2
10	q_3	q_1	q_4
101	q_4	q_3	q_2

q_a

q_b

Q \ Σ	0	1
$\rightarrow q_0$	B	B
q_1	B	B
q_2	B	B
q_3	B	A
q_4	B	B

$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

$Q = \{q_a, q_b, q_3\}$

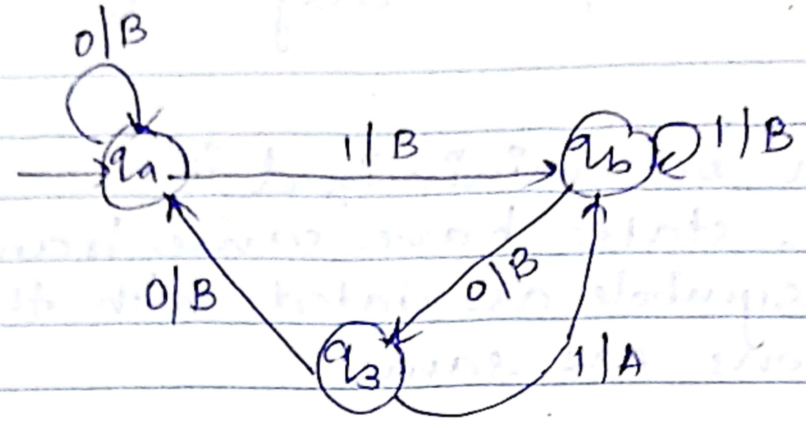
$\Sigma = \{0, 1\}$

$\Delta = \{A, B\}$

q_a = start state

Q \ Σ	0	1
$\rightarrow q_a$	q_a	q_b
q_b	q_3	q_b
q_3	q_a	q_b

Q \ Σ	0	1
$\rightarrow q_a$	B	B
q_b	B	B
q_3	B	A



Example:

$(q_a, 101)$ BBA

$\vdash (q_b, 01)$

$\vdash (q_3, 1)$

$\vdash (q_b, \epsilon)$

② Design mealy m/c to o/p the remainder whe[n] binary no. ~~is~~ divided by 4.

$\Rightarrow$

	S \ I	0	1
	$\rightarrow q_0$	q_1	q_2
0	q_1	q_1	q_2
1	q_2	q_3	q_4
2	q_3	q_1	q_2
3	q_4	q_3	q_4

S \ I	0	1
$\rightarrow q_0$	0	1
q_1	0	1
q_2	2	3
q_3	0	1
q_4	2	3

q_a q_b

100
101
110
111

$M = \{Q, \Sigma, \Delta, \delta, \lambda, q_0\}$

$Q = \{q_a, q_b\}$ $q_a \rightarrow$ start state

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

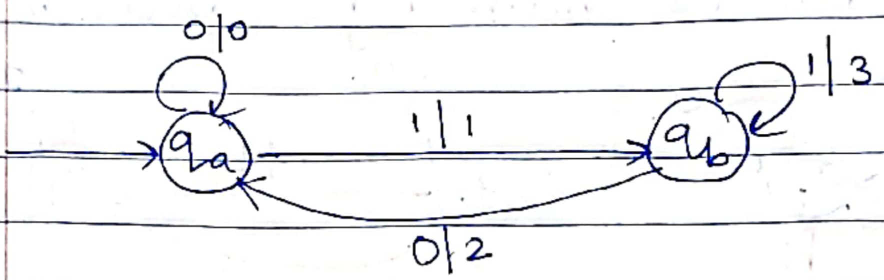
$\Delta = \{0, 1, 2, 3\}$

δ:-

Q \ Σ	0	1
→ q_a	q_a	q_b
q_b	q_a	q_b

λ:-

Q \ Σ	0	1
→ q_a	0	1
q_b	2	3



Example:-

$(q_a, 1100)$ 1320 λ

$\vdash (q_b, 100)$

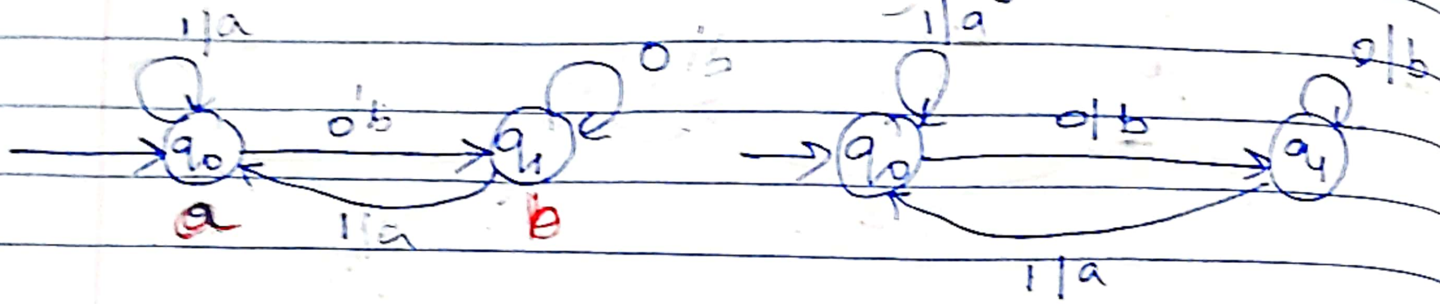
$\vdash (q_b, 00)$

$\vdash (q_a, 0)$

$\vdash (q_a, \epsilon)$

Moore m/c to Mealy

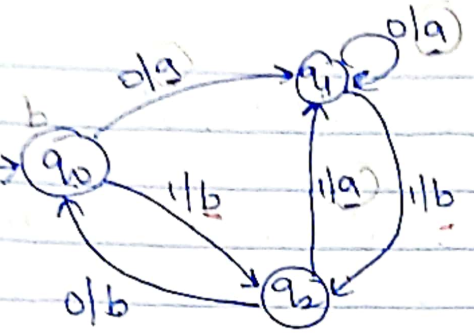
* Assign the o/p symbol associated with the state to all its incoming transition

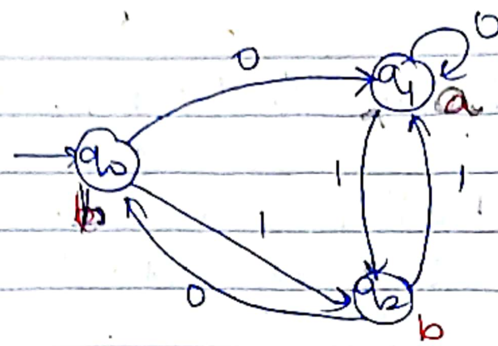


Mealy m/c to Moore m/c

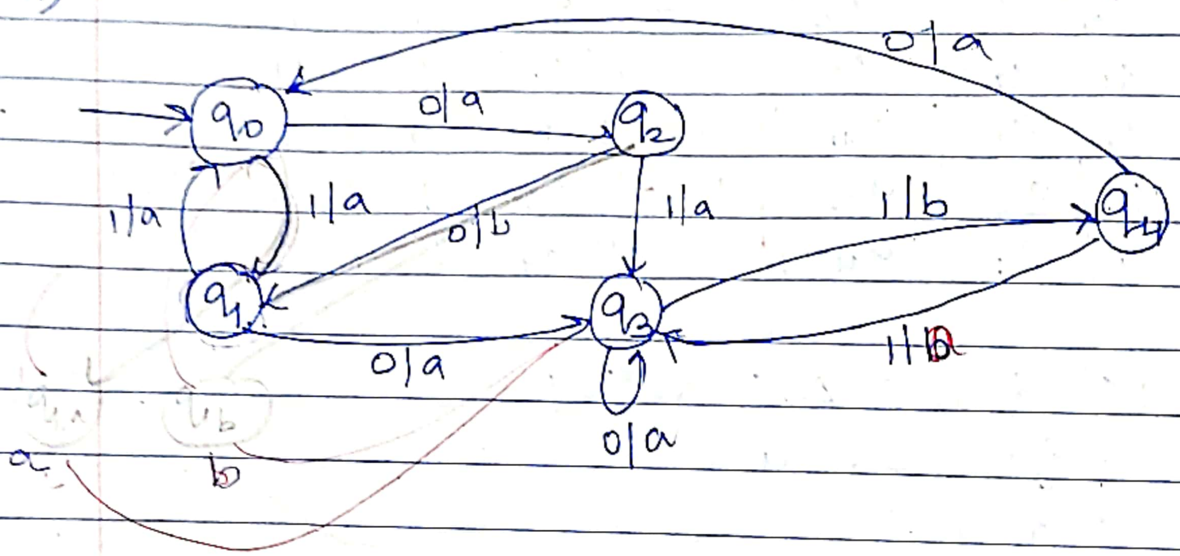
* If the o/p symbols associated with the incoming transitions to a state are same, then assign that symbol to that state
* If the o/p symbols associated with the incoming transitions to a state are not same, then split that state as many times as the o/p symbols with each state producing a diff o/p symbol.
* If ~~there~~ there are no incoming transition to a state, then any o/p symbol can be assigned to that state

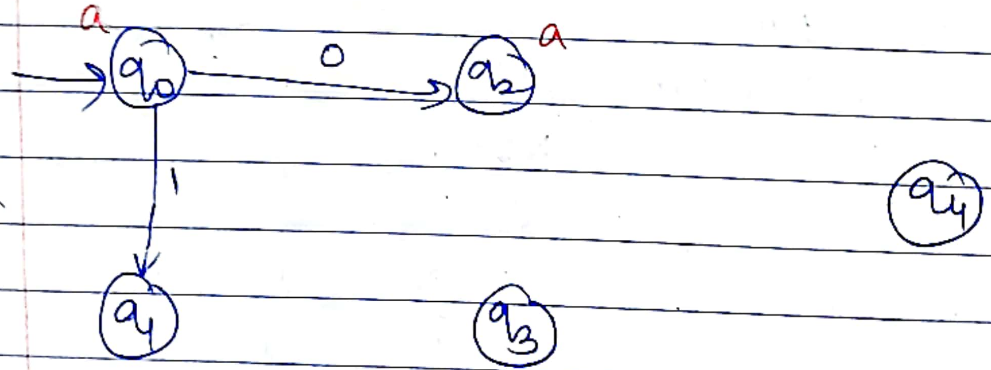
i)

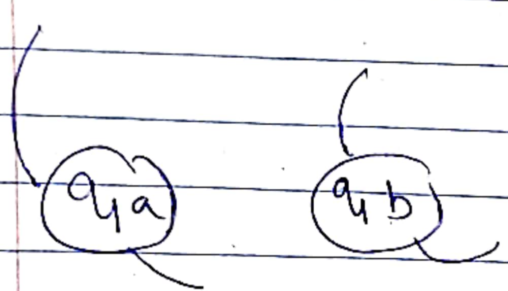

q0
q1
q2
0|a
1|b
1|a
0|b
b


q0
q1
q2
0
1
a
b

ii)


q0
q1
q2
q3
q4
0|a
1|a
0|b
1|b


q0
q1
q2
q3
q4
a
0
1


q4a
q4b

Design Mealy m/c for $r = (0+1)^*(00+11)$

OR

Design a mealy m/c to o/p 'Y' if i/p ends in double letter and 'N' otherwise over $\Sigma = \{0,1\}$

	S \ I	0	1
	$\rightarrow q_0$	q_1	q_2
q_a	0 q_1	q_3	q_2
q_a	1 q_2	q_1	q_4
q_a	00 q_3	q_3	q_2
q_b	11 q_4	q_1	q_4

	S \ I	0	1
	$\rightarrow q_0$	N	N
q_a	0 q_1	Y	N
q_a	1 q_2	N	Y
q_a	00 q_3	Y	N
q_b	11 q_4	N	Y

$M = \{Q, \Sigma, \Delta, \delta, \lambda, q_0\}$

$Q = \{q_0, q_a, q_b\}$

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

$\Delta = \{N, Y\}$

q_0 = Start state

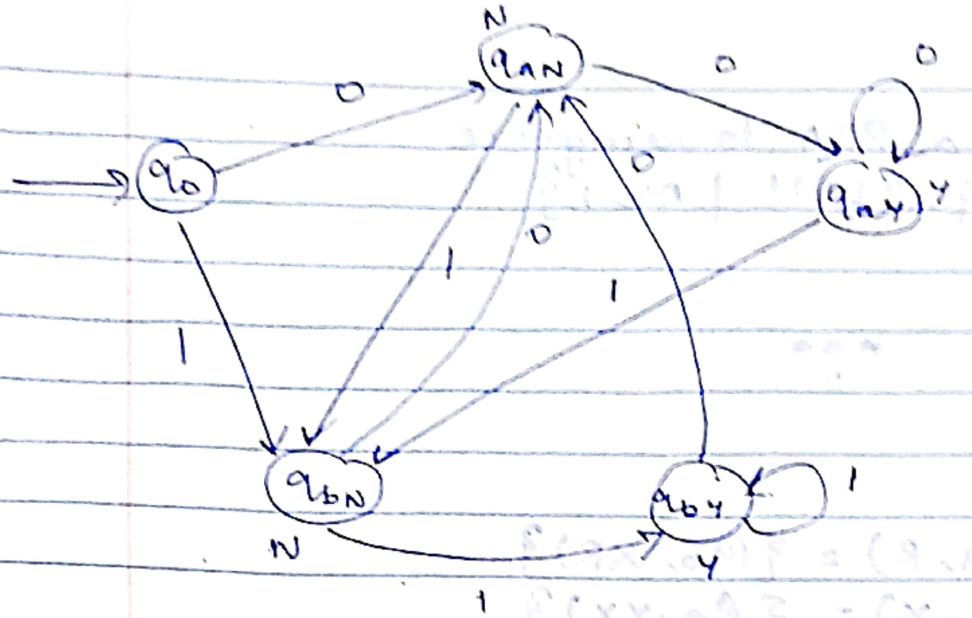
δ :-

Q \ Σ	0	1
$\rightarrow q_0$	q_a	q_b
q_a	q_a	q_b
q_b	q_a	q_b

λ :-

Q \ Σ	0	1
$\rightarrow q_0$	N	N
q_a	Y	N
q_b	N	Y

transition diagram & example



q0
qaN
qaY
qbN
qbY
N
Y
0
1

→ CFG is a simple recursive method of specifying grammar rules by which strings in a lang can be generated.

2/8/7/13 Grammars

* Context-free grammars have played a central role in compiler technology, turned the implementation of parser (functⁿ that discovers the structure of prgm)
* Grammar define language give syntax.
* CFG is represented by four components: tuples

$G = (V, T, P, S)$

T = finite set of symbols that form the string of the lang being defined; we call that alphabet as terminals. eg $\Sigma = \{0, 1\}$

V = finite set of variables / non terminals / syntactic categories. Each variable represents a lang

S = start variable

P = There is a finite set of productions or rules that represent the recursive defⁿ of lang.

$\alpha \rightarrow \beta$

Where α and β are in some sentential form (any combination of V & T)

V = cap
T = small
S = start
P = prodⁿ

$L(G) = \{id, id * id, id + id, \ldots\}$

$G = \{\{E\}, \{+, *, id\}, P, E\}$

P:- $E \rightarrow E * E$ $E \rightarrow id$

$E \rightarrow E * E$ $E \rightarrow E + E$

$E \rightarrow id$ $E \rightarrow id + id$

or $E \rightarrow E + E / E * E / id$

Derivation of a sentence:- "id+id"

$E \Rightarrow E+E$ using $E \to E+E$
$\Rightarrow id+E$ using $E \to id$
$\Rightarrow id+id$ using $E \to id$

* Context Free Grammar CFL : CFG
A grammar is said to be CFG if all the productions are of the form

$$A \to \alpha$$

Where A is a variable
α is in some sentential form

eg: $G = (\{S, A\}, \{a, b\}, P, S\}$

P:-

$$S \to aSb \mid aA$$
$$A \to aA \mid Ab \mid \epsilon$$

Derivation:
It is a process to determine whether the given G can generate / derive the given sentence using any combination of production Rules starting from start variable production.

$3 \times 3 \leftarrow 3$

$3+3 \leftarrow 3$

$4i \leftarrow 3$

$bi \backslash 3+3 \backslash 3+3 \leftarrow 3$

Two types:-

1. Left most Derivation (LMD):
   In LMD of a sentence, at every step select and replace the leftmost variables by its production rules

2. Right most Derivation: (RMD)
   In RMD of a sentence, at every step select and replace the leftmost variable by its production rules.

~~eg~~

Examples on LMD & RMD.

1. P:-

$$S \rightarrow aAS \mid a$$
$$A \rightarrow SbA \mid SS \mid ba$$

Derive "aabbaa" using LMD & RMD

LMD :-

$S \Rightarrow aAS$	using $S \rightarrow aAS$
$\Rightarrow aSbAS$	using $A \rightarrow SbA$
$\Rightarrow aabAS$	using $S \rightarrow a$
$\Rightarrow aabbaS$	using $A \rightarrow ba$
$\Rightarrow aabbaa$	using $S \rightarrow a$

** string-only terminals aabbaa

RMD:-

$S \Rightarrow aAS$ using $S \to aAS$

$\Rightarrow aAa$ using $S \to a$ **

$\Rightarrow aSbAa$ using $A \to SbA$

$\Rightarrow aSbbaa$ using $A \to ba$

$\Rightarrow aabbaa$ using $S \to a$

(2) P:-

$S \to AS \mid \epsilon$

$A \to aa \mid ab \mid ba \mid bb$

Derive:- aabbba, baabab, aaabbb using LMD & RMD

LMD- $S \to AS$
$\to aaS$
$\to aaAS$
$\to aabbS$
$\to aabbAS$
$\to aabbba\epsilon$

$S \to AS$
$\to baS$
$\to baAS$
$\to baabS$
$\to baabAS$
$\to baabab$

$S \to AS$
$\to aaS$
$\to aaAS$
$\to aaabS$
$\to aaabAS$
$\to aaabbb$

RMD- $S \to AS$ string S
$\to AAS$
$\to AAAS$
$\to aabbba$

$S \to AS$
$\to AAS$
$\to AAAS$
$\to baabab$

$S \to AS$
$\to AAS$
$\to AAAS$
$\to aaabbb$

(3) P:- $S \to aB \mid bA$
$A \to a \mid aS \mid bAA$
$B \to b \mid bS \mid aBB$

aabbabab

$S \to aB$
$\to aaBB$ Add ao
$\to aabSB$
$\to aabbAB$ dao
$\to aabbaSB$
$\to aabbabAB$ ao
$\to aabbabaB$
$\to aabbabab$

$S \to aB$
$\to aaBB$
$\to aaBbS$
$\to aaBbaB$
$\to aaBbab$
$\to aabSbab$
$\to aabbAbab$
$\to aabbabab$

Parse Tree Generation.

25/7/13

** Simplification of CFG's

- Various langs can effectively be represented by context free grammar.
- All the grammars are not always optimized
- i.e grammar may consist of some extra symbols (non terminals).
- having unnecessary symbols increases the length of grammar.
- Hence simplification is necessary, symbols that are not necessary are removed.

Properties:

- Each variable and each terminal of G appears in the derivation of some word in L
- There should not be any production as $X \rightarrow Y$ where X and Y are non terminals.
- If ϵ is not in the lang L then there need not be the production $X \rightarrow \epsilon$.

Simplification is done by

1. Useless variables
2. unit production
3. Elimination of ϵ production

Normal form

① Elimination of useless variable:-

Defn of useful variable

A variable 'X' is said to be useful if

$$S \Rightarrow \alpha X \beta \Rightarrow w$$

Where S – start variable

α, β – sentential form

w – sentence

Otherwise "X" is said to be useless.

Steps of Elimination:-

① Initialize P' to P

② Find useless variables

* Variable is said to be useless if it cannot derive any sentence
* Variable is useless if it is not reachable from S.

③ Delete all the productions which involve useless variables.

① Eliminate useless variables:-

$S \to aSb \mid ba \mid aA$

⟹ Given:-

$G = (\{S, A\}, \{a, b\}, P, S)$

P :-

$S \to aSb \mid ba \mid aA$

~~Define:-~~

~~$G' = (V', T', P', S)$~~

Now, in above CFG, the variables are S, A

$S \to aSb$

$\to a\cancel{ba}b$

Thus we can reach to certain string after following these rules

But if $S \to aA$ then there is no further rule as a defn of A. no point in rule $S \to aA$

So we can declare it as useless symbol.

We can remove and after removal, CFG is

$G = (V, T, P, S)$

where $V = \{S\}$

$T = \{a, b\}$.

$P = S \to aSb \mid ba\}$

$S \to$ start symbol.

② Consider CFG $G = (V, T, P, S)$ reachable useless

$V = \{S, A, B\}$ $T = \{0, 1\}$

$P = \{ S \to A1 \mid \cancel{B} \mid 11A \mid \cancel{1B} \mid 11$

$A \to 0$

$\cancel{B \to BB}$ $\}$

no terminals

01

$S \to B$

$\cancel{1B}$

$S \to \cancel{B}$

$A \mid$

$0\ 1$

③

$S \to aSa \mid b \mid aAB$

$B \to bB \mid d$

Since A cannot derive any sentence A is useless

$S \to aSa \mid b$

$B \to bB \mid d$

Since B is not reachable from S B is useless

$P :-$ $S \to aSa \mid b$

④ $S \to \cancel{A}B \mid CA$ $\cancel{B \to BC \mid AB}$ — x

$A \to a$ $C \to aB \mid b$

⑤ $S \to aA \mid \cancel{bB}$

$A \to aA \mid a$

$\cancel{B \to bB}$

$D \to ab \mid Ea$

$E \to a\cancel{C} \mid d$

⑥ $S \to aS \mid A \mid \cancel{C}$

$\cancel{A} \to a$

$\cancel{B \to aa}$

$\cancel{C \to aCb}$

⑦ $S \to aA \mid a \mid Bb \mid \cancel{cC}$

$A \to aB$

$B \to a \mid Aa$

$C \to cCD$

$D \to ddd$

3, 9:30 AM

★★ 1st check for useless variables
2nd eliminate unit production
3rd again check for useless variables

29|7|13

(II) Elimination of Unit production.

Def^n of unit production: are the productions in which one non-terminal gives another non-terminal

A production of the form

$A \to B$'

where A and B are variables is called as unit production.

Def^n of chain of Unit Production

$A \to X_1 \to X_2 \to \ldots\ldots X_n \to B$

Elimination: Find unit Production, say 'B → C' is the unit production of B, then add to B, the non-unit productions of C & then delete all unit production

eg: Eliminate unit productions

$S \to aSb \mid a \mid A$

$A \to aA \mid Ab \mid a \mid \cancel{B}$

$S \to aSb$

$S \to a$

$S \to \cancel{A}$

$S \to aA$

$S \to Ab$

$S \to a$

Sol^n:

$\cancel{S \to a}$ $A \to a$

$A \to aA$

$A \to Ab$

$S \to A$ add $S \to aA$, $S \to Ab$, $\cancel{S \to a}$

$S \to a$

$S \to aSb$

final gr

$S \to aSb \mid a \mid aA \mid Ab$

$A \to Ab \mid aA \mid a$

Q3: $S \to A \mid bb$

$A \to B \mid a$

$B \to S \mid b$

Sol^n:

$B \to b$

$B \to S$ add $B \to bb$

$B \to S \to A$ add $B \to a$

$B \to S - A - B$

$A \to a$

$A \to B$ add $A \to b$

$A \to B \to S$ add $A \to bb$

$A \to B \to S \to A$

$S \to bb$

$S \to A$ add $S \to a$

$S \to A \to B$ add $S \to b$

$S \to A \to B \to S$ –

$S \to \cancel{A} \, bb \mid a \mid b$

$A \to a \mid b \mid bb$

$B \to b \mid a \mid bb$

Eliminating useless variables

Since A & B are not reachable they are discarded.

$S \to a \mid b \mid bb$

(2) $S \to 0A \mid 1B \mid C$

$A \to 0S \mid 00$

$B \to 1 \mid A$

$C \to 01$

Sol^n: Clearly, $S \to C$ is unit production, But while removing C we have to consider rules of C, so add rule to S

$S \to 0A \mid 1B \mid 01$

|||ly, $B \to A$ is also unit production, so

$B \to 1 \mid 0S \mid 00$

Finally, CFG without unit production as

$S \to 0A \mid 1B \mid 01$

$A \to 0S \mid 00$

$B \to 1 \mid 0S \mid 00$

$C \to 01$

(4) $S \to AB$

$A \to a$

$B \to C \mid b$

$C \to D$

$D \to E \mid bC$

$E \to d \mid Ab$

(1) useless symbols
(2) reachable
(3) unit productions

$D \to d \mid Ab \mid bC$

$E \to d \mid Ab$

$C \to d \mid Ab \mid bC$

$B \to d \mid Ab \mid bC \mid b$

$A \to a$

$S \to AB$

(5) $S \to AB$
$A \to a$
$B \to C \mid b$
$C \to D$
$D \to E \mid bC$
$E \to d \mid Ab$

(III) Elimination of ϵ-production from Grammar.

→ In context free grammar, if at all there is ϵ production we can remove it, without changing the meaning of the grammar.

eg. $S \to 0S \mid 1S \mid \epsilon$

$S \to 0S$ — $S \to 0$
$S \to 1S$ — $S \to 1$
$S \to \epsilon$ — $S - \epsilon$

Thus, we can rewrite as
$S \to 0S \mid 1S \mid 0 \mid 1$

YX
$S \to XYX$
ϵ
YX

eg: (1) $S \to XYX$
$X \to 0X \mid \epsilon$
$Y \to 1Y \mid \epsilon$

$S \to XYX$ — $S \to XY,\ S \to YX,\ S \to XX,\ S \to X,\ S \to Y$
$X \to 0X$ — $X \to 0$
$Y \to 1Y$ — $Y \to 1$

$S \to XY \mid YX \mid XX \mid X \mid Y$
$X \to 0X \mid 0$
$Y \to 1Y \mid 1$

(2) $S \to a \mid Ab \mid aBa$
$A \to b \mid \epsilon$
$B \to b \mid A$

$S \to a$
$S \to Ab$ — $S \to b$
$S \to aBa$ — $S \to aa$
$A \to b$ — $A \to \epsilon$
$B \to b$ — $B \to \epsilon$
$B \to A$ — $B \to \epsilon$

$$\left\{\begin{array}{l} S \to a \mid Ab \mid aBa \mid b \mid aa \\ A \to b \\ B \to b \end{array}\right. \Rightarrow \text{Now}$$

(3) $S \to XY$
$X \to Zb$
$Y \to bW$
$Z \to AB$
$W \to Z$
$A \to aA \mid bA \mid \epsilon$
$B \to Ba \mid Bb \mid \epsilon$

~~S → XY~~ - ~~S → X, S → Y~~

~~X → Zb~~ - ~~X → b~~

~~Y → BW~~ - ~~Y → W, Y → P~~

A → a | b

B → a | b

Z → AB putting A, B = ∈

Z = ∈∈ = ∈

X → b

Y → b

S → XY

X → b

Y → b

not reachable A → aA | bA | a | b

B → Ba | Bb | a | b

S → XY

X → b

Y → b

ullman

** pg. 244

eg: $S \rightarrow ABC$

$A \rightarrow BC \mid a$

$B \rightarrow bAC \mid \epsilon$

$C \rightarrow cAB \mid \epsilon$

Sol^n:- B & C are nullable, since $A \rightarrow BC$ is also nullable

$S \rightarrow ABC$, $S \rightarrow BC \mid AC \mid AB \mid C \mid A \mid B$

$A \rightarrow BC$, $A \rightarrow B \mid C$

$B \rightarrow bAC$, $B \rightarrow bA \mid bC \mid b$

$C \rightarrow cAB$, $C \rightarrow cB \mid cA \mid c$

$S \rightarrow ABC \mid AB \mid BC \mid AC \mid A \mid B \mid C$

$A \rightarrow BC \mid B \mid C \mid a$

$B \rightarrow bAC \mid bA \mid bC \mid b$

$C \rightarrow cB \mid cA \mid \cancel{cB}\ cAB \mid c$

If $w \in L(G)$, for some CFG, then w has parse tr[ee]
which tells us the syntactic structure of w

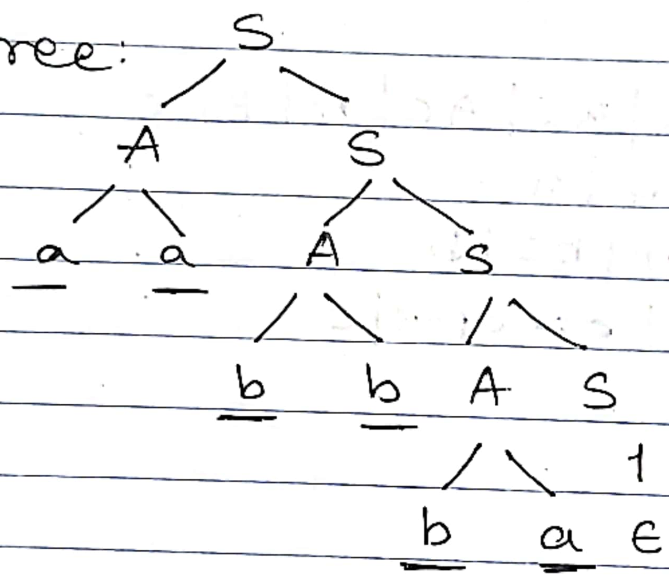
Parse tree

$S \to AS \mid \epsilon$

$A \to aa \mid ab \mid ba \mid bb$

$S \to AS$
$\to aaS$
$\to aaAS$
$\to aabbS$
$\to aabbAS$
$\to aabbba\epsilon$

Parse tree:



aabbba

Each
- → Nodes carry a label that is a symbol
- → Topmost node is called the root, nodes along the bottom are called leaves.
- → All leaves are labelled by terminals, or possibly the empty string ϵ.
- → By concatenating the labels of the leaves

Chomsky Normal Form

We complete our study of grammatical simplification by showing that every nonempty CFL without ϵ has a grammar G in which all productions are in one of two simple forms, either:

1. $A \rightarrow BC$, where A, B, C are variables
2. $A \rightarrow a$, where A is a variable and a is terminal.

Further, G has no useless symbols. Such grammar is said to be in Chomsky Normal form. or CNF.

Steps:

① To put a grammar in CNF, we start with simplifying the grammar ~~one that satisfies the restrictions~~ i.e no useless variables, no unit productions, no ϵ productions

② Every production of such a grammar is either of the form $A \rightarrow a$, which is already in a form allowed by CNF or it has a body of length 2 or more.

** i.e tasks are to:

ⓐ arrange that all bodies of length 2 or more consist only of variables

ⓑ Break bodies of length 3 or more into a

Final grammar,

$E \to EC_1 \mid TC_2 \mid LC_3 \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$

$T \to TC_2 \mid LC_3 \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$

$F \to LC_3 \mid a \mid b \mid IA \mid IB \mid IZ \mid IO$

$I \to a \mid b \mid IA \mid IB \mid IZ \mid IO$

$A \to a$

⋮

$R \to)$

$C_1 \to PT$

$C_2 \to MF$

$C_3 \to ER$

Simplified

a

BC

eg ② $S \to aSab \mid bSb \mid aa \mid a \mid b \mid bb$

$S \to a$

$S \to b$

$S \to aa$

Replace a by C_1 C_1

$S \to C_1C_1$ $C_1 \to a$

$S \to bb$

Replace b by C_2

$S \to C_2C_2$ $C_2 \to b$

$S \to bSb$

$S \to C_2 \underbrace{S C_2}_{C_3}$

C_2C_3 $C_3 \to SC_2$

Since either there can be one terminal or two variables

Replacing SC_2 by C_3

✓ $S \to C_2 C_3$ $\qquad\qquad$ $C_3 \to SC_2$

$S \to aSab$

$S \to C_1 \underbrace{S C_1 C_2}_{C_4}$

Replacing SC_1C_2 by C_4

✓ $S \to C_1 C_4$ $\qquad\qquad$ $C_4 \to S\underbrace{C_1 C_2}_{C_5}$

$C_4 \to S\underbrace{C_1 C_2}_{C_5}$

Replacing C_1C_2 by C_5

✓ $C_4 \to S C_5$ $\qquad\qquad$ $C_5 \to C_1 C_2$

✓ $C_5 \to C_1 C_2$

Final grammar is in CNF that is

$S \to a \mid b \mid C_1 C_1 \mid C_2 C_2 \mid C_2 C_3 \mid C_1 C_4$

$C_1 \to a$

$C_2 \to b$

$C_3 \to SC_2$

$C_4 \to SC_5$

$C_5 \to C_1 C_2$

K

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

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

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

Soln:

$A \to a$

$B \to b$

~~$A \to aS$~~

~~$A \to C_1S$~~ $\qquad$ $C_1 \to a$

$B \to bS$ $\qquad$ $C_1 \to b$

$B \to C_1S$

$B \to aBB$ $\qquad$ $C_2 \to a$

$B \to C_2\underbrace{BB}$

$B \to C_2C_3$ $\qquad$ $C_3 \to BB$

$A \to aS$

$A \to C_2S$

$A \to bAA$

$A \to C_1AA$ $\qquad$ $C_4 \to AA$

$A \to C_1C_4$

$S \to bA$

$S \to C_1A$

$S \to aB$

$S \to C_2B$

$S \to C_1A \mid C_2B$

$A \to a \mid C_2S \mid C_1C_4$

$B \to b \mid C_1S \mid C_2C_3$

$C_1 \to b$, $C_2 \to a$, $C_3 \to BB$, $C_4 \to AA$

$S \to AB\cancel{A}b$
$A \to aA \mid \epsilon$
$B \to bB \mid \epsilon$

Eliminating ϵ,

$S \to ABA \mid BA \mid AB \mid AA \mid A \mid B$
$A \to aA \mid a$
$B \to bB \mid b$

eliminate unit prod:

$S \to ABA \mid BA \mid AB \mid AA \mid aA \mid a \mid bB \mid b$
$A \to aA \mid a$
$B \to bB \mid b$

$S \to ABA$
$S \to aABA \mid aBA$
$S \to BA$
$S \to bBA \mid bA$
$S \to AB$
$S \to aAB \mid aB$
$S \to AA$
$S \to aAA \mid aA$
$S \to aA$
$S \to a$
$S \to bB$
$S \to b$

$S \to aABA \mid aBA \mid bBA \mid bA \mid aAB \mid aB \mid aAA \mid aA \mid a \mid bB \mid b$
$A \to aA \mid a$
$\to bB \mid b$

$a\epsilon$

$V \to bV$

AA

~~Example~~

$S \to AA \mid ABA \mid BA$

$A \to aA \mid a$

$B \to bB \mid b$

Step 1: Simplify the grammar

$A \to aA \mid a$

$B \to bB \mid b$

$S \to AA$

$S \to aAA \mid aA$

$S \to ABA$

$S \to aABA \mid aBA$

$S \to BA$

$S \to bBA \mid bA$

GNF is

$S \to aAA \mid aA \mid aABA \mid aBA \mid bBA \mid bA$

$A \to aA \mid a$

$B \to bB \mid b$

$S \rightarrow AB\cancel{A}b$

$A \rightarrow aA \mid \epsilon$

$B \rightarrow bB \mid \epsilon$

eliminating ϵ,

$S \rightarrow ABA \mid BA \mid AB \mid AA \mid A \mid B$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b.$

eliminate unit prod:

$S \rightarrow ABA \mid BA \mid AB \mid AA \mid aA \mid a \mid bB \mid b$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

1)

$S \rightarrow ABA$

$S \rightarrow aABA \mid aBA$

$S \rightarrow BA$

$S \rightarrow bBA \mid bA$

$S \rightarrow AB$

$S \rightarrow aAB \mid aB$

$S \rightarrow AA$

$S \rightarrow aAA \mid aA$

$S \rightarrow aA$

$S \rightarrow a$

$S \rightarrow bB$

$S \rightarrow b$

$S \rightarrow aABA \mid aBA \mid bBA \mid bA \mid aAB \mid aB \mid aAA \mid aA \mid a \mid bB$

$A \rightarrow aA \mid a$

$\rightarrow bB \mid b$

$A_1 \to A_2 A_3 \mid a$

$A_2 \to A_3 A_1 \mid b$

$A_3 \to A_1 A_2 \mid a$

$A_3 \to A_1 A_2 \mid a$

$A_3 \to A_2 A_3 A_2 \mid a A_2 \mid a$

$A_3 \to A_3 A_1 A_3 A_2 \mid \cancel{A_3 A_1} \; b A_3 A_2 \mid a A_2 \mid a$

$A \to A \quad \alpha_1 \quad\quad \beta_1 \quad \beta_2 \quad \beta_3$

$B \to \alpha_1 \mid \alpha_1 B$

$A \to \beta_i \mid \beta_i B$

$B \to A_1 A_3 A_2 \mid A_1 A_3 A_2 B$

$A_3 \to b A_3 A_2 \mid b A_3 A_2 B \mid a A_2 \mid a A_2 B \mid a \mid a B$

$A_2 \to b A_3 A_2 A_1 \mid b A_3 A_2 B A_1 \mid a A_2 A_1 \mid a A_2 B A_1 \mid$
$a A_1 \mid a B A_1 \mid b$

$A_1 \to b A_3 A_2 A_1 A_3 \mid b A_3 A_2 B A_1 A_3 \mid a A_2 A_1 A_3 \mid$
$a A_2 B A_1 A_3 \mid a A_1 A_3 \mid a B A_1 A_3 \mid b A_3 \mid a$

$B \to b A_3 A_2 A_1 A_3 A_3 A_2 \mid b A_3 A_2 B A_1 A_3 A_3 A_2 \mid$
$a A_2 A_1 A_3 A_3 A_2 \mid a A_2 B A_1 A_3 A_3 A_2 \mid$
$a A_1 A_3 A_3 A_2 \mid a B A_1 A_3 A_3 A_2 \mid \cancel{a B A_1 A_3 A_3 A_2} \mid$
$b A_3 A_3 A_2 \mid b A_3 A_2 A_1 A_3 A_3 A_2 B \mid$
$b A_3 A_2 B A_1 A_3 A_3 A_2 B \mid a A_2 A_1 A_3 A_3 A_2 B \mid$
$a A_2 B A_1 A_3 A_3 A_2 B \mid a A_1 A_3 A_3 A_2 B \mid$
$a B A_1 A_3 A_3 A_2 B \mid \cancel{a B A_1 A_3} A_3 A_2 B \mid$
$b A_3 A_3 A_2 B \mid a A_3 A_2 \mid a A_3 A_2 B$

Scanned with OKEN Scanner

$S \rightarrow ABC$

$A \rightarrow a \mid b$

$B \rightarrow Bb \mid aa$

$C \rightarrow aC \mid cC \mid ba$

$S \rightarrow ABC$

$S \rightarrow aBC \mid bBC$

$C \rightarrow aC \mid cC \mid bG$

$G \rightarrow a$

$G \rightarrow ?$

$B \rightarrow Bb \mid aa$

$A \rightarrow A\,\alpha_1 \quad \beta_1$

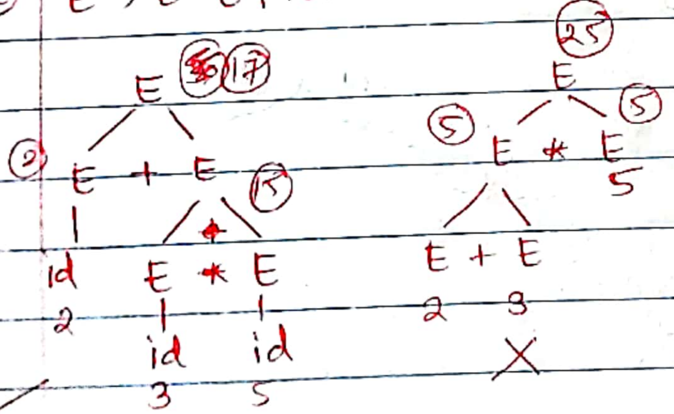
$D \rightarrow b \mid bD$

$B \rightarrow aa\cancel{B} \mid aaD$

$B \rightarrow aG \mid aGD$

① $E \rightarrow E+E \mid E*E \mid id$

② $E \rightarrow E-E \mid id$



+

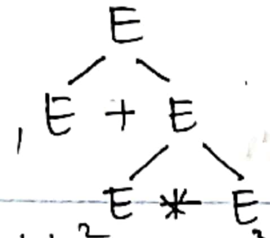
*

id

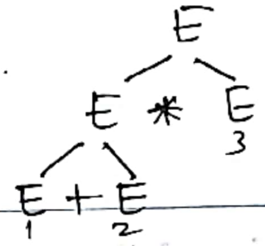
$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow id$



PDA ⊳ DPA



* Ambiguous Grammars

$E \Rightarrow E+E \mid E*E$ "E+E*E"

$E \rightarrow E+\underline{E}$	$E \rightarrow \underline{E}*E$
$E \rightarrow E+E*E$	$E \rightarrow E+E*E$
$1+\underline{2*3}$	$\underline{1+2}*3$
$1+6$	$3*3$
7	9

* Parse tree has to be unique by LMD or RMD.
* There are some CFL's that are "inherently ambiguous" [Def: every grammar for the language puts more than one structure (parse tree) on some strings in the language]
(i.e. more than one derivation for same input)

$S \rightarrow AB \mid C$

$A \rightarrow aAb \mid ab$

$B \rightarrow cBd \mid cd$ aabbccdd

$C \rightarrow aCd \mid aDd$

$D \rightarrow bDc \mid bc$

LMD

$S \rightarrow AB$	$S \rightarrow C$
$\rightarrow aAbB$	$\rightarrow aCd$
$\rightarrow aabbB$	$\rightarrow aaDdd$
$\rightarrow aabbcBd$	$\rightarrow aabDcdd$
$\rightarrow \underline{aabbccdd}$	$\rightarrow \underline{aabbccdd}$

Ambiguous

Dec 2013 → (?)

Grammars.

* Types of Grammars / Classification of Grammars / Chomsky's hierarchy.

As per Chomsky, Grammars can be classified into 4 types:

① Type 0 / Unrestricted Grammar
In this, there are no restriction on production rules $\alpha \to \beta$; $\alpha \neq \epsilon$, $\alpha, \beta \in (V \cup T)^*$

$A \to \alpha$ $A \to aB$ TM & RE
$A \to a$

$A \to AB$
$AB \to BC$
$B \to ACD$
$AC \to abc$
$D \to \epsilon$

② Type 1 / Context Sensitive Grammar.
A production without any restrictions is called a type 0 production.
A production of the form $\phi A \psi \to \phi \alpha \psi$ is called type 1 production where ϕ and ψ are the left context & right context. A can be replaced by α
In this case S does not appear on the RHS of any production.

$S \to abA$ $\alpha \to \beta$
$aA \to bA$
$A \to a$

LBA
~~TM~~ & CFL
$A \to AB$
$AB \to AC$
$AC \to ab$

(3) Type 2 / Context Free Grammar

In this type, all the productions should be of the form

$$A \to \alpha$$

where $A \to$ variable, $\alpha \to$ sentential form, ϵ

$S \to aA$

$A \to aA \mid a$

PDA, CFL

$S \to aSa \mid bSb \mid a \mid b$

(4) Type 3 / Regular Grammar

The restriction in this case S does not appear on the RHS of any production. If variables appear on leftmost position then the grammar is called as LLG.

If variables appear on rightmost position then the grammar is called RLG

RL & FSM

$A \to \alpha B \mid \beta$

$A, B \in V$

$\alpha, \beta \in T^*$

LLG $\quad S \to Ba$

$\quad\quad S \to a$

RLG

(1) Left hand side contains 1 nonterminal

(2) RHS can contain atmost 1 nonterminal, which is allowed to appear at Rightmost or leftmost.

RLG $\quad S \to aB$

$\quad\quad S \to b$

9/C48 RLG

Type	Language (grammars)	Form of prod^n in grammar	Accepting device
3	Regular	$A \to aB$, $A \to a$ $(A, B \in V,\ a \in \Sigma)$	Finite automaton
2	Context free	$A \to \alpha$ $(A \in V,\ \alpha \in (V \cup \Sigma)^*)$	PDA
1	Context-sensitive	$\alpha \to \beta$ $(\alpha, \beta \in (V \cup \Sigma)^*,$ $\lvert\beta\rvert \geq \lvert\alpha\rvert,$ α contains variable	Linear-bounded automaton
0	Recursively enumerable. (unrestricted or phrase-structure)	$\alpha \to \beta$ $(\alpha, \beta \in (V \cup \Sigma)^*,$ α contains a variable)	Turing machine

Regular Grammar;

$G = (V, T, P, S)$

$V = (S, A, B)$

$T = (a, b)$

$S \to aA \mid bB$

$A \to a$

$B \to b$

When a grammar can be represented by some finite automata then it, is a regular grammar.

G can be constructed from DFA such that productions corresponds to transitions.

$P_i \to a P_j$ is a production rule if $\delta(q_i, a) = q_j$
$q_0 \quad a \quad q_1$
where $q_j \notin F$

$P_i \to a P_j$ and $P_i \to a$ are production rules if $\delta(q_i, a) = q_j$ where $q_j \in F$.