FINITE LANGUAGES AND AUTOMATA THEORY

[THEORY OF COMPUTATION]

CSPC 41 → SUBJECT A

ASSIGNMENT - 1

Units covered :

1. Finite Automata
2. Regular Expression (RE)

Submitted TO :

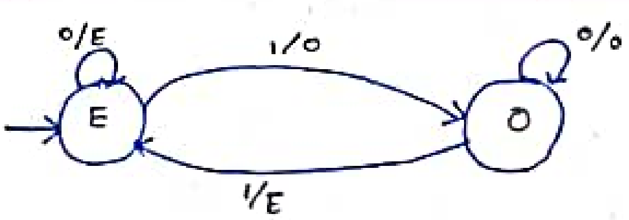
Ms. Usha kiruthiga,
FLAT Teacher,
semester - 4 ,
CSE - B

Done By :

PRAJWAL SUNDAR
CSE - B

ROLL NO 106121092

1. Construct a Mealy Machine which can output **E** (even) or **O** (odd), according to the number of 1's in the input stream.



E : even no. of 1's
O : odd no. of 1's

Initially, an empty string has 0 no. of 1's, so E is the starting state.

when @ state E :

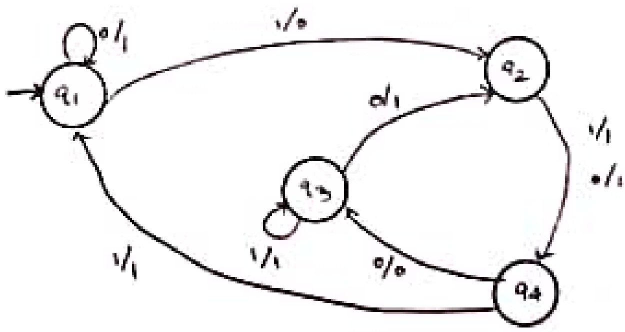
- on input 0, no. of 1's remain even.
- on input 1, no. of 1's become odd.

when @ state O :

- on input 0, no. of 1's remain odd.
- on input 1, no. of 1's become even.

∴ Mealy machine is successfully constructed.

2. Convert the given **Mealy** Machine into its equivalent **Moore** Machine.



Analysing incoming transitions for each state.

- q_1 : $q_1 \xrightarrow{0/1} q_1$, $q_4 \xrightarrow{1/1} q_1$

  In both cases output is 1.

  In both cases output is 1.
  So q_1 can be associated with output 1. $q_1/1$

- q_2 : $q_1 \xrightarrow{1/0} q_2$, $q_3 \xrightarrow{0/1} q_2$

  Both 0 and 1 are outputs.

So q_2 is divided into two states, $q_2/0$ and $q_2/1$, each being associated with a particular output.

* q_3 : $q_3 \xrightarrow{1/1} q_3$, $q_4 \xrightarrow{0/0} q_3$

Both 0 and 1 are outputs.

So q_3 is divided into two states, $q_3/0$ and $q_3/1$, each being associate with a particular output.

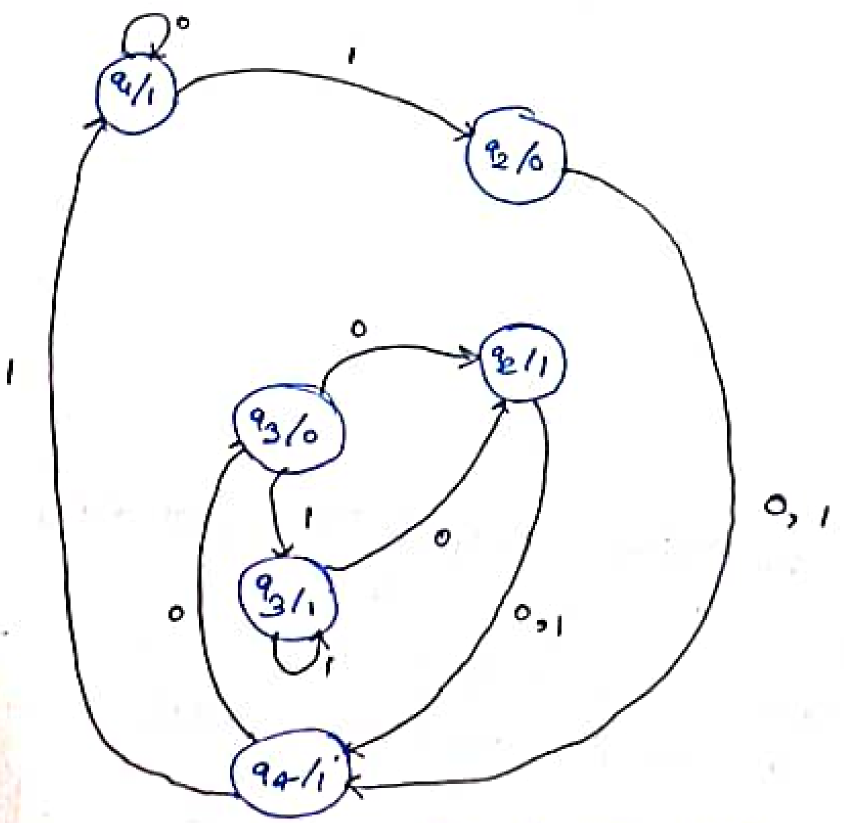
* q_4 : $q_4 \xrightarrow{0/1} q_4$, $q_2 \xrightarrow{1/1} q_4$

In both cases output is 1.

So q_4 can be associated with output 1.

$q_4/1$

Moore Machine :

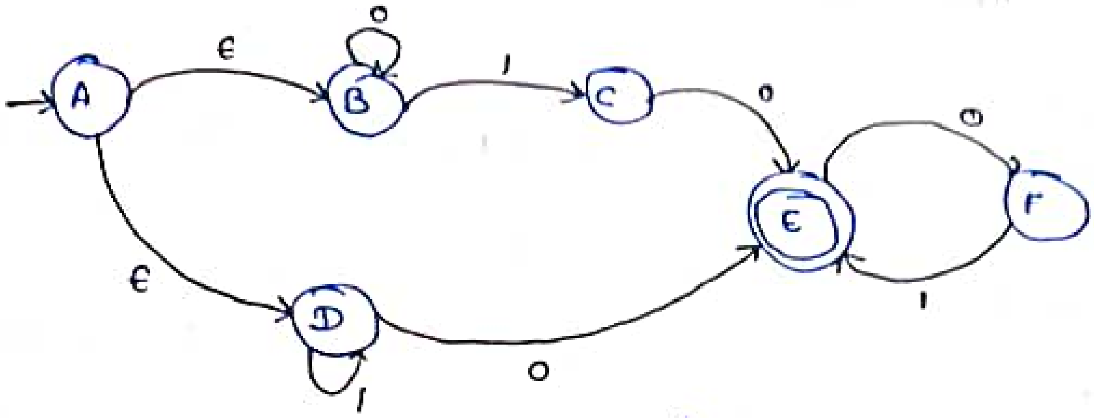


∴ Moore machine is successfully constructed

3(a). Construct a **minimal DFA** for the regular expression $(0^*10 + 1^*0)(01)^*$.

Expression : $(0^* 1 + 1^*) 0 (01)^*$

Drawing an ε-NFA for the RE :



Constructing transition Table :-

	ε CLOSE	0	1
→A	A, B, D	ϕ	ϕ
B	B	B	C
C	C	E	ϕ
D	D	E	D
★E	E	F	ϕ
F	F	ϕ	E

Converting the above ε-NFA into a DFA :-

	0	1		0	1
→ABD	BE	CD	D	E	D
★BE	BF	C	B	B	C
CD	E	D	★CE	EF	X
★F	B	CE	F	X	E
C	E	X	★EF	F	E
★E	F	X	X	X	X

Minimizing the DFA using concept of equivalence

0 equivalence

(ABD, CD, BF, C, D, B, F, X) (BE, E, CE, EF)

1 equivalence

(ABD, CD, C, D) (BF, F) (B) (X) (BE, E) (CE)

2 equivalence

(ABD, CD, D) (C) (BF) (F) (B) (X) (BE) (E) (CE) (

3 equivalence

(ABD) (CD, D) (C) (BF) (F) (B) (X) (BE) (E) (CE)

4 equivalence

(ABD) (CD, D) (C) (BF) (F) (B) (X) (BE) (E) (CE) (

3 equivalence = 4 equivalence obtained

clearly CD = D is the only reduction pos

12 ⟶ 11 states ⇒ in minimized

Renaming :

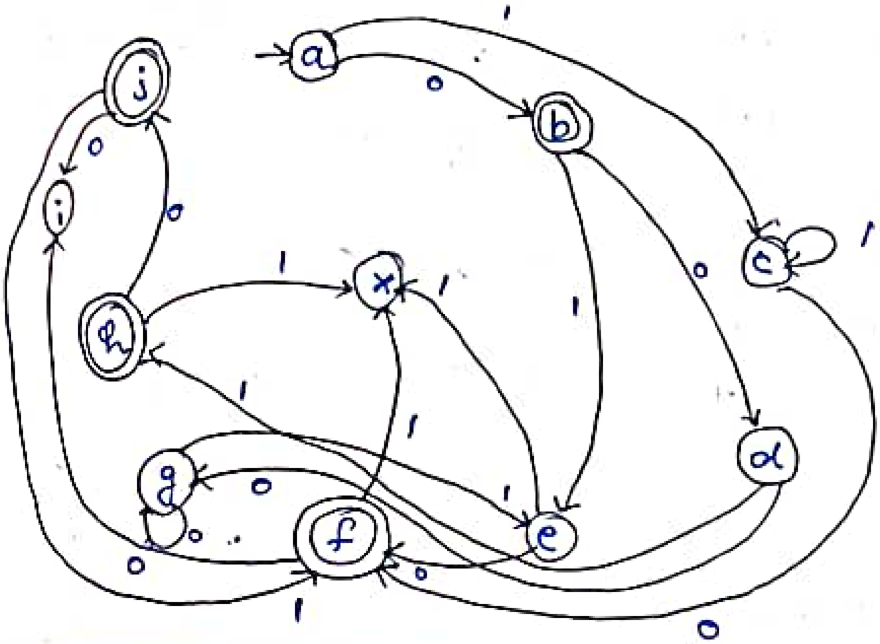
ABD → a, BE → b, CD → c (= D), BF → d, C → e, E -

B → g, CE → h, F → i, EF → j, X → x

Transition Table for reduced DFA :-

	0	1		0	1
→a	b	c	g	g	e
☆b	d	e	☆h	j	x
c	f	c	i	x	f
d	g	h	☆j	i	f
e	f	x	x	x	x
☆f	i	x			

Representing the DFA graphically :-

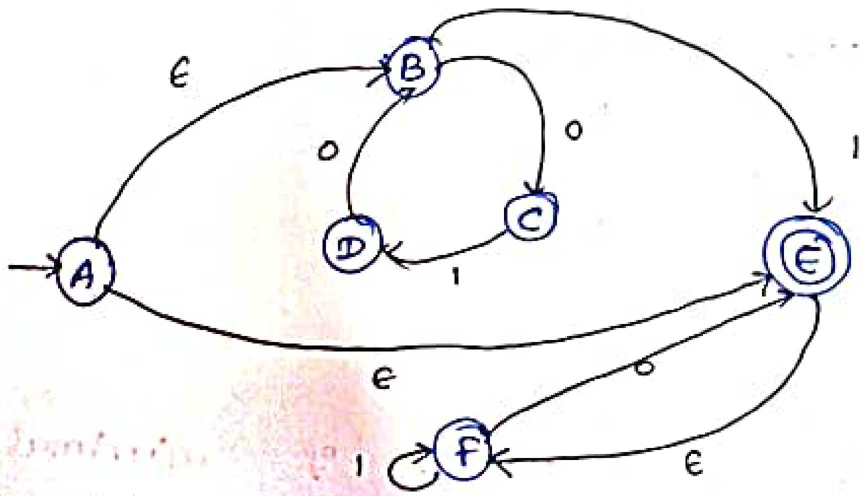


∴ The minimal DFA with 11 states was constructed successfully.

3(b). Construct a **minimal DFA** for the regular expression (010)*1 + (1*0)*.

Expression : $(010)^*1 + (1^*0)^*$

ε- NFA :-



Transition Table :-

	ECLOSE	0	1
A	A, B, E, F	Φ	Φ
B	B	C	E
C	C	Φ	D
D	D	B	Φ
E	E, F	Φ	Φ
F	F	E	F

converting ε-NFA into DFA :-

	0	1
→* ABEF	CEF	EF
* CEF	EF	DF
* EF	EF	F
DF	BEF	F
F	EF	F
* BEF	CEF	EF

state Reductio Diagra

CEF	×				
EF	×	×			
DF	×	×	×		
F	×	×	×	×	
BEF	✓	×	×	×	×
	ABEF	CEF	EF	DF	F

using Myhill Nerode theorem,

states ABEF = BEF , all others are destin

6 states $\xrightarrow{-1}$ 5 states

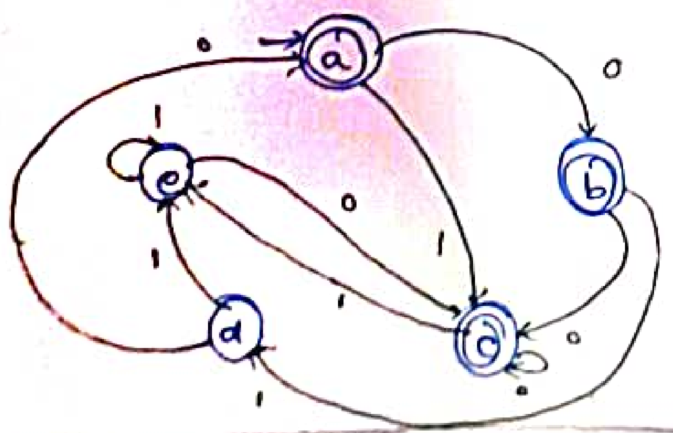
Renaming

ABEF → a, CEF → b, EF → c, DF → d, F →
= BEF

Transition Table :

	0	1
→* a	b	c
* b	c	d
* c	c	e
d	a	e
e	c	c

Graphical representation of minimal DFA

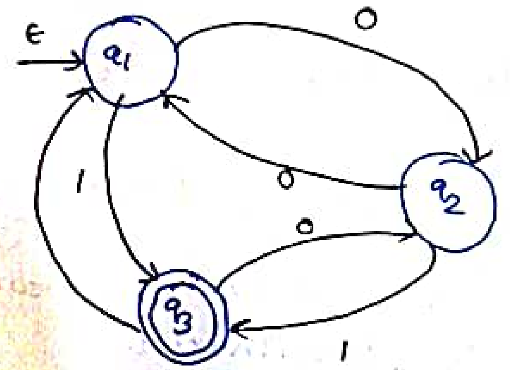


∴ The minimal
with 5 states
constructed
successfull

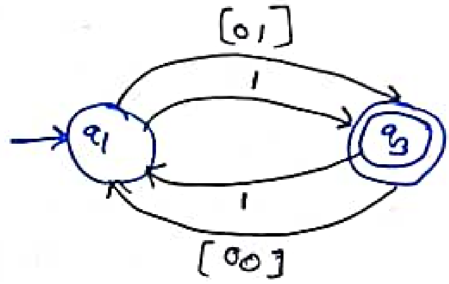
4. Convert the given **DFA** into its corresponding **regular expression** (either using Arden's Theorem or State Elimination Method).

	0	1
→q1	q2	q3
q2	q1	q3
*q3	q2	q1

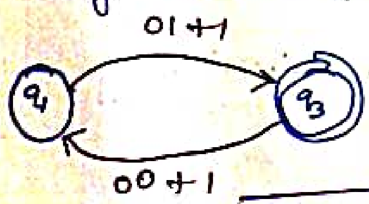
Drawing DFA :-



Eliminating q2



Eliminating extra lines



writing RE :

$$(01+1)[(00+1)(01+1)]^*$$

simplifying :

$$\boxed{(0+\epsilon)\cdot 1\,[(00+1)(0+\epsilon)\cdot 1]^*}$$

5. **Minimize the following DFA.**

	a	b
→q0	q0	q3
q1	q2	q5
q2	q3	q4
q3	q0	q5
q4	q0	q6
q5	q1	q4
*q6	q1	q3

Myhill Nerode theorem

q_1	X					
q_2	X	X				
q_3	X	X	X			
q_4	X	X	X	X		
q_5	X	X	X	X	X	
q_6	X	X	X	X	X	X
	q_0	q_1	q_2	q_3	q_4	q_5

Equivalence concept

0 equivalence

$(q_0, q_1, q_2, q_3, q_4, q_5)$ (q_6)

1 equivalence

$(q_0, q_1, q_2, q_3, q_5)$ (q_4) (q_6)

2 equivalence

(q_0, q_1, q_3) (q_2, q_5) (q_4) (q_6)

3 equivalence

(q_0) (q_1) (q_3) (q_2, q_5) (q_4) (q_6)

4 equivalence

(q_0) (q_1) (q_2) (q_3) (q_4) (q_5) (q_6)

5 equivalence

(q_0) (q_1) (q_2) (q_3) (q_4) (q_5) (q_6)

4 equivalence row = 5 equivalence row.
so process is terminated.

clearly all states are distinct.

$\therefore$ $\boxed{\text{DFA cannot be minimized}}$

6(a) Prove that 0^{2^n} is not a regular language using pumping lemma.

let $L = \{ 0^{2^n} \; \forall \; n \geq 0 \}$ be a regular language

Taking $n = 2$ (pumping length) as an example to analyse.

$0^{2^2} = 0^4 \Rightarrow$ string $S = \underbrace{0}_{x}\underbrace{0\,0}_{y}\underbrace{0}_{z}$

spliting s into $S = xyz$, such that

L① $|xz| \leq P \rightarrow 2 \leq 2$ ✓

L② $|y| > 0 \rightarrow 2 > 0$ ✓

✗③ $w = xy^iz \in L \Rightarrow$ let $i = 2$

$w = 0(0^2)^2 0 = 0^6 \notin L$ as $2^n \neq 6$ for any integer

condition 3 is violated.

$\therefore$ Pumping lemma Failed $\rightarrow$ L is not a regular language

6(b) Prove that 0^{n^2} is **not a regular language** using pumping lemma.

let $L = \{ 0^{n^2} \; \forall \; n \geq 0 \}$ be a regular language.

Taking $n = 3$ as an example to analyse.
(pumping example length)

$0^{3^2} = 0^9 \Rightarrow$ string $S = \underbrace{0}_{x}\underbrace{0000000}_{y}\underbrace{0}_{z}$

splitting S into $S = xyz$, such that

① $|xz| \leq p \rightarrow 3 \leq 3$ ✓

② $|y| > 0 \rightarrow 6 > 0$

③ $w = xy^iz \in L \Rightarrow$ Let $i = 2$

$w = 0^1(0^6)^2 0^2 = 0^{15} \notin L$ as $n^2 \neq 15$ for any integer n.

condition 3 is violated

∴ Pumping Lemma failed → L is not a regular language