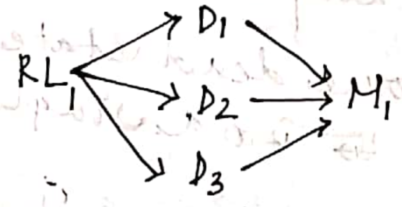
TOPIC: MINIMIZATION OF DFA.

Why it is important to minimize a DFA?

Because,
When we are dealing with only designing unit of DFA, We are concentrating on the fact that whatever logic we are performing whether it is correct or not. So, consistency is there. But in engineering after designing a consistent System. 2nd step is to make it efficient.
i.e. Same operation must be performed by the m/c but using minimum no. of state.

* if we say we take a m/c and then we ask to minimize then it does not mean the language acceptance capability of the m/c changes. The performance of the m/c should improve but the capability of the m/c should be same.

* A DFA should accept one language but a L may be accepted by more than one DFA.



(i) We 12 improve the efficiency or performance
(ii) Minimum DFA removes all the ambiguity and confusion as it is unique.

States
- Productive State
- Non-Productive State
  - Dead States
  - Unreachable Sta[te]
  - Equal state

Productive State:

Presence or absence of that State changes the L accepting capability of the m/c.

Non-Productive State:

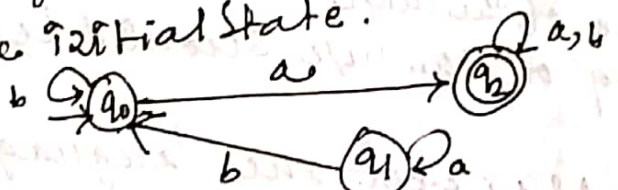
When the presence or absense of the that state in the m/c donot change the L accepting capability of the m/c.

In DFA minimization we target non-productive state in order to ~~avoid any~~ remove it.

Dead State: Not possible to reach FS.

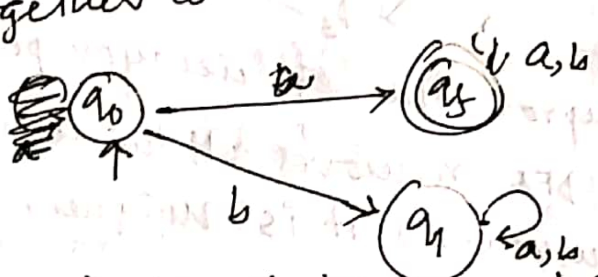
Unreachable State:

If ∄ no path possible to reach that state from the initial state.



* A m/c can have more than one dead State but practically it is not required.

* More than one dead state can merge ~~them~~ together ~~to~~ into a single state.



* If unreachable state present in the DFA then remove them directly.

Equal States:

Two states are said to equal if,

$\delta^*(q_i, w) \longrightarrow FS$

$\delta^*(q_j, w) \longrightarrow FS$ $\quad w \in \Sigma^*$

FS must not be same.

&

$\delta^*(q_i, w) \longrightarrow NFS$

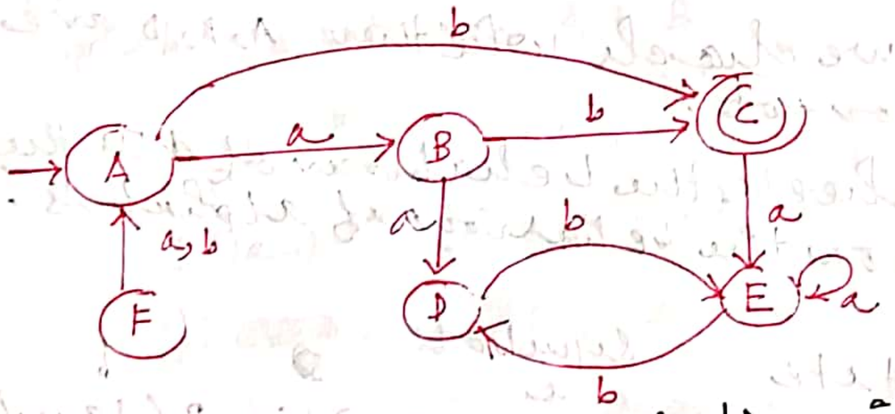
$\delta^*(q_j, w) \longrightarrow NFS.$

NFS must not be same (Nature of the State should be equal)

Q: How to minimize of a DFA?

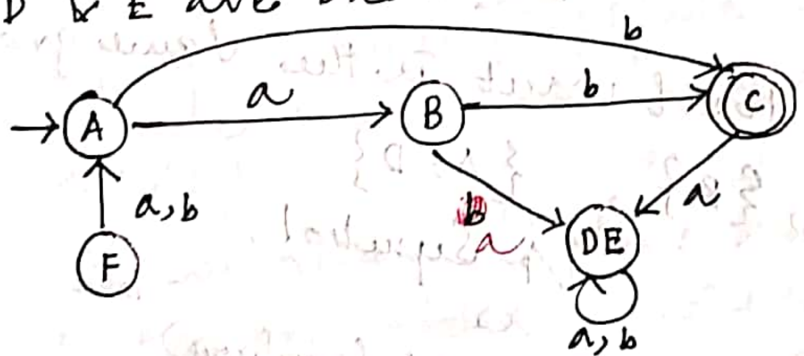
A:
(i) Merge all the dead state into one dead state.
(ii) Delete all the unreachable states.
(iii) Merge all the equal state into one ~~f.~~ state.

Prob: 1



Step 1: Check for dead state & if there is are many then merge them into a single state.

D & E are dead state.

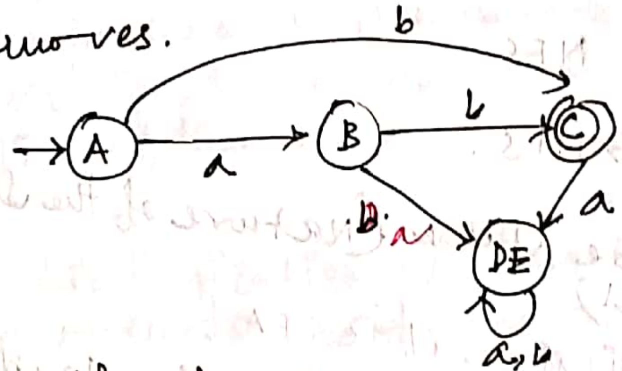


Step 2: If ∃ any unreachable state then delete it.

F is a unreachable state.

(Because from a we can not reach F)

Hence, F is a unreachable state. So, F must be removes.



Step 3: Check equal state.

We make 2 groups of states

1. FS → {C}
2. NFS → {A, B, D}

Now, we check whether A, B, D are equal or not.

We check the behaviour of all the states based on the behaviour of alphabets.

State	Symbol a	b
A	B (Same group)	B (Same Group)
B	D (")	C ~~(")~~ (Not Same grp.)
D	D (")	D (Same Group).

i.e. B donot want in the same group.

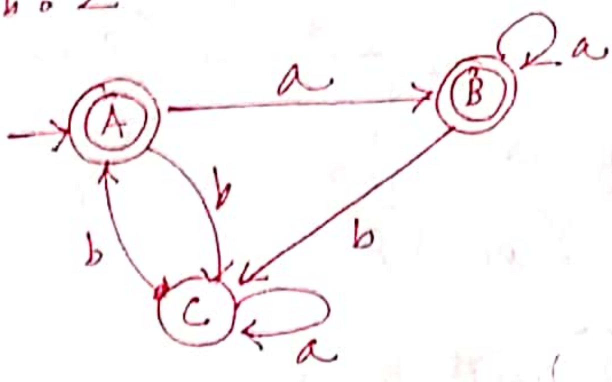
{C}, {B}, {A, D}

State	I/P Symbol a	b
A	B (Not in Same Group)	
D	D (Same Group)	

{C}, {B}, {A}, {D}

i.e. all State behaviour are different i.e. row of the two states are equal.

Prob: 2



Step 1: Make 2 groups NFS & FS.

NFS : {C}

FS : {A, B}

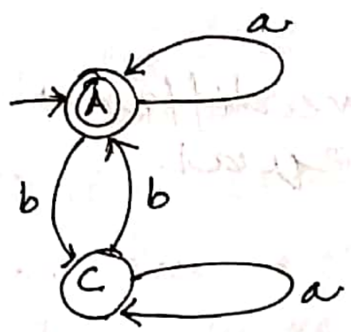
Step 2: let us check the behaviour of both alphabet a & b. on A & B.

State	I/p symbol a	b
A	B (same)	C (not in same)
B	B (same)	C (not same)
	(same behaviour)	(same behaviour).

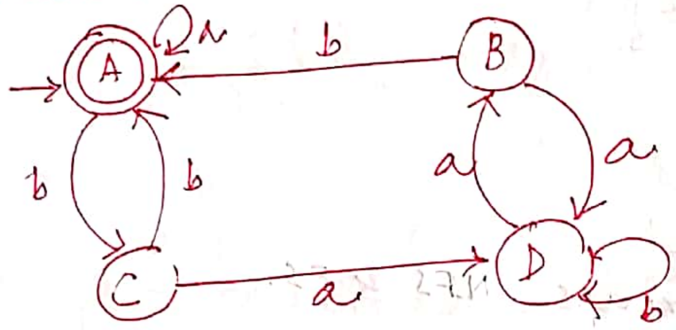
∴ i.e. A & B are equal state.

Step 3: We must remove either A or B. As A is the initial state so, we delete B because it can not be deleted.

Note: If we remove a state all the outgoing Transaction will automatically removed. & whatever incoming edge it must return back.



Prob: 3



① Check NFS & FS.

ⓐ NFS : {B, C, D}

ⓑ FS : {A}

② Let us check the behaviour of ~~both~~ B, C, D on both the i/p alphabet a & b.

State	i/p Symbol a	b
B	D (same)	A (not same)
C	D (u)	A (u)
D	B (u)	D (same)

i.e. behaviour of D is different than B & C.
So, separate it.

{A}, {D}, {B, C}

Now, check whether B & C are same.

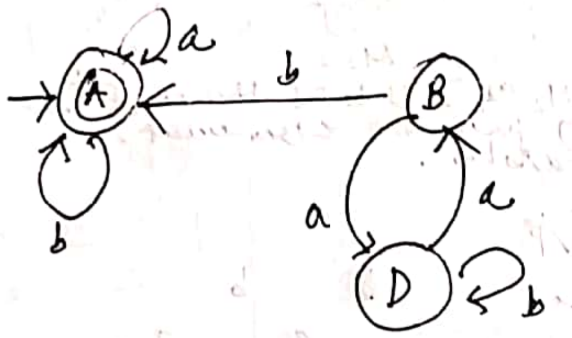
State	i/p Symbol a	b
B	D (Not Same)	A (Not Same)
C	D (Not Same)	A (Not Same)

(i.e. B & C behaviour is same)

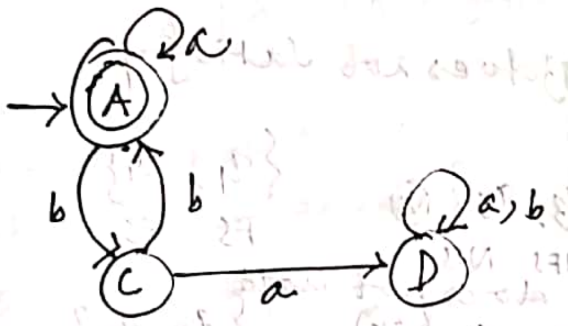
Therefore. B & C are equal State.

③ Now, let us remove C.

All the outgoing transition from C will be removed.
" " incoming " on C must be rollback



if we remove B then,



TOPIC: EQUIVALANCE OF 2 FA

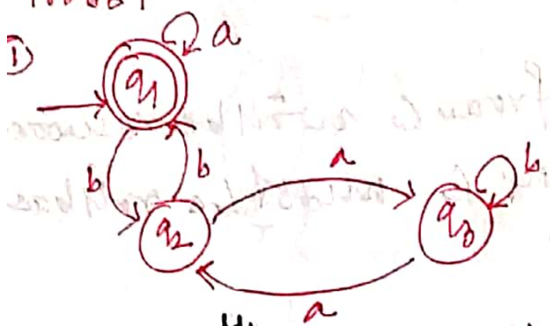
① If is of 1st FA is FS, then is of the 2nd FA must also be FS.

② For any pair of state $\{q_i, q_j\}$ the transition for i/p Σ is defined by $\{q_a, q_b\}$ where, $\delta(q_i, a) = q_a$ & $\delta(q_j, b) = q_b$.

③ The 2 automata are not equivalent if any pair $\{q_a, q_b\}$ one is final & other is NFS.

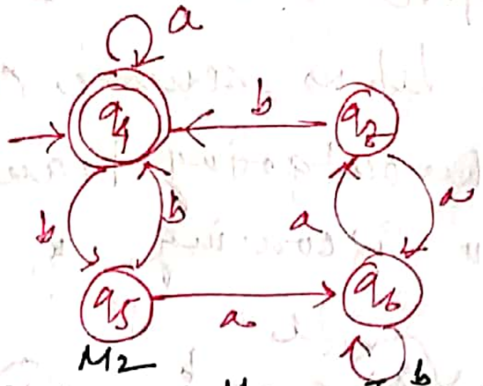
Prob: 1

(i)



M1

(ii)



M2

① As in M1 q_1 is IS & FS & in M2 q_4 is IS & FS. Hence ∃ a possibility that (i) & (ii) may be equivalent.

② Make equivalence table.

Pair State	I/P a	b
$\{q_1, q_4\}$	$\{q_1, q_4\}$ FS FS	$\{q_2, q_5\}$ (NS) NFS NFS.

Condition 3 does not satisfy.

$\{q_2, q_5\}$	$\{q_3, q_6\}$ (NS) NFS NFS	$\{q_1, q_4\}$ FS FS

Condition 3 does not hold.

$\{q_3, q_6\}$	$\{q_2, q_7\}$ (NS) NFS NFS	$\{q_3, q_6\}$ NFS NFS

Condition 3 does not hold

$\{q_2, q_7\}$	$\{q_3, q_6\}$ NFS NFS	$\{q_1, q_4\}$ FS FS

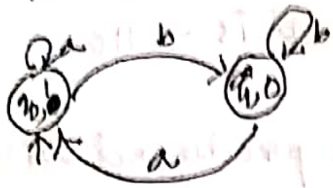
Condition 3 does not hold.

Hence (i) & (ii) are equivalent.

* TOPIC: FINITE AUTOMATA WITH O/P

FA WITH O/P

Moore M/c



$\lambda : Q \longrightarrow \Delta$

CAPITAL SIGMA, CAPITAL DELTA, CAPITAL LAMDA

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

$Q \rightarrow$ Finite set of states

$\Sigma \rightarrow$ i/p alphabet

$\delta \rightarrow$ Transition Function

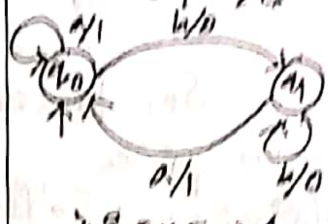
$\delta : Q \times \Sigma \longrightarrow Q$

$q_0 \rightarrow$ Initial State

$\Delta \rightarrow$ O/p alphabet

$\lambda \rightarrow$ O/P function.

Mealy M/c



$\lambda : Q \times \Sigma \rightarrow \Delta$

$q_0, q_1 \longrightarrow Q$ (states)

$\Sigma \longrightarrow \{a, b\}$ (i/p alphabet)

$\delta : Q \times \Sigma \longrightarrow Q$

$q_0 \times a \longrightarrow q_0$
$q_1 \times b \longrightarrow q_1$ } (Transition Function).

$q_0 \longrightarrow q_0$ (IS)

$\Delta \longrightarrow \{0, 1\}$ (output)

$\lambda \longrightarrow \lambda : Q \longrightarrow \Delta$

$q_0 \longrightarrow 1$
$q_1 \longrightarrow 0$ } (O/P function)

(For every state an o/p is associated)

$q_0, q_1 \longrightarrow Q$ (States)

$\Sigma \longrightarrow \{a, b\}$ (i/p alphabet)

$\delta : Q \times \Sigma \longrightarrow Q$

$q_0 \times a \longrightarrow q_0$
$q_0 \times b \longrightarrow q_1$
$q_1 \times a \longrightarrow q_0$
$q_1 \times b \longrightarrow q_1$ } (Transition Function).

$q_0 \longrightarrow q_0$ (IS)

$\Delta \longrightarrow \{0, 1\}$ (output alphabet)

$\lambda \longrightarrow \lambda : Q \times \Sigma \rightarrow \Delta$

$q_0 \times a \longrightarrow 1$
$q_0 \times b \longrightarrow 0$
$q_1 \times a \longrightarrow 1$
$q_1 \times b \longrightarrow 0$

(For every transition an o/p is associated)

How to remember?

M O O R E

M O → State

O R E → O/P.

Q: How moore m/c work?

Present State	i/p Symbol.	NS	O/p
$\rightarrow q_0$	—	—	1
$\rightarrow q_0$	a	q_0	1
q_0	b	q_1	0

So, for string 'ab' O/p printed is → 110

if we give i/p string of length n then the o/p produced will be, $n+1$

Reason,

Even without seeing anything q_0 is producing o/p.

Q: How Mealy m/c work?

Present State (PS)	i/p Symbol	NS	O/p
$\rightarrow q_0$	a	q_0	1
q_0	b	q_1	0
q_1	b	0	q_1
q_1	a	1	q_0

* O/p is associated with i/p.

* Therefore, if we give an i/p string of length n then the o/p produced will be, n.

Prob: 1

Construct a moore m/c that takes set of all strings over $\Sigma = \{a, b\}$ as i/p & prints '1' as o/p for every occurrance of 'ab' as a substring.

Answer:

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

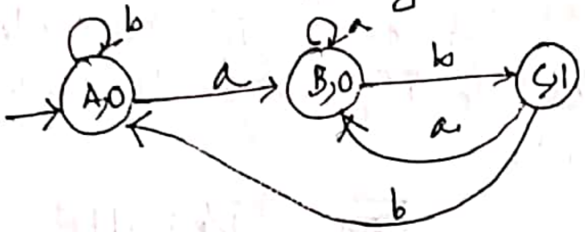
$\Delta = \{0, 1\}$

if I give 'ab' it will print '1'

" I " 'abab' " " '11'

" I " 'abbbbabb' " " '11'

① We need to design a deterministic m/c which that accepts strings ~~start~~ with ab ending.



ab
aba
abab
ababb
ababba
ababbab.

Try:

$A \xrightarrow{a} B \xrightarrow{b} C$

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

$0 \quad 0 \quad 1$

$A \xrightarrow{a} B \xrightarrow{b} C \xrightarrow{a} B \xrightarrow{b} C$

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

$0 \quad 0 \quad 1 \quad 0 \quad 1$

Prob: 2

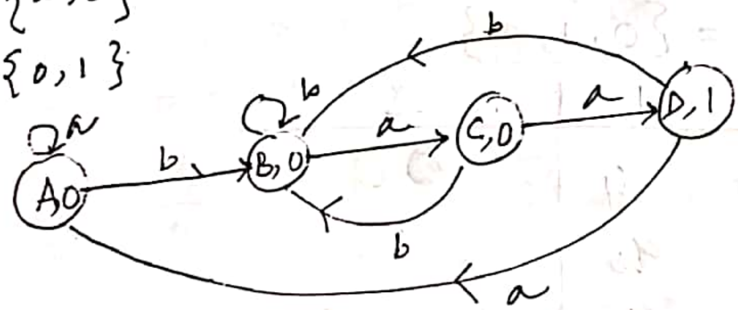
Construct a moore m/c that takes set of all strings over {a, b} & counts no of occurrences of substring 'baa'.

baa
baabaa

Answer:

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

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



Try:

$A \xrightarrow{b} B \xrightarrow{a} C \xrightarrow{a} D$

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

$0 \quad 0 \quad 0 \quad 1$

$A \xrightarrow{b} B \xrightarrow{a} C \xrightarrow{a} D \xrightarrow{b} B \xrightarrow{a} C \xrightarrow{a} D$

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

$0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1$

Prob: 3

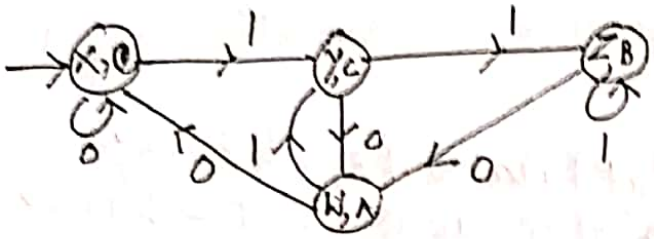
Construct a moore m/c that takes set of all strings over {0,1} & produces 'A' as o/p if i/p ends with '10' or produces 'B' as o/p if i/p ends with '11' otherwise produces 'C'.

Answer:

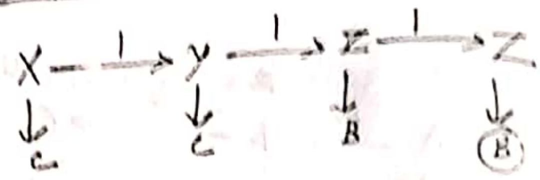
$\Sigma = \{0, 1\}$

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

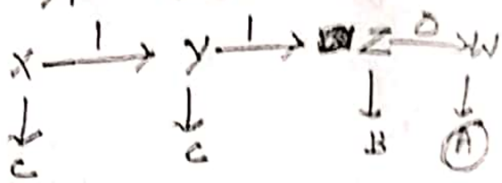
10 ⟶ A
11 ⟶ B
00 ⟶ C
01 ⟶ C



Try: I/p : 111



I/p : 110

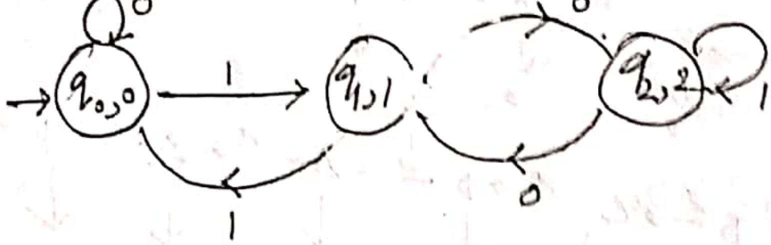


Prob: 4

Construct a moore m/c that takes binary nos as i/p & produces residue modulo '3' as o/p.

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

	0	1	Δ
$\rightarrow q_0$	q_0	q_1	0
q_1	q_2	q_0	1
q_2	q_1	q_2	2



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

$\Delta = \{0, 1, 2, 3, 4\}$

	0	1	2	3	Δ
q_0	q_0	q_1	q_2	q_3	0
q_1	q_4	q_0	q_1	q_2	1
q_2	q_3	q_4	q_0	q_1	2
q_3	q_2	q_3	q_4	q_0	3
q_4	q_1	q_2	q_3	q_4	4

Moore m/c that takes base 4 nos as i/p & produces residue modulo 5 as o/p.

Prob: 1

Construct a mealy m/c that takes binary no. as i/p & produces 2's complement of that no. as o/p. Assume the string is read LSB to MSB & end carry is discarded.

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

no. 1 0 1 1

$q_0 \to q_0 \to q_0 \to q_0 \to q_0$

0 1 0 0

Wrong.

1011 1's comp

0100

M L

1 1 0 0

1's 0 0 1 1

+ 1

2's 0 1 0 0

1 1 1 0 1 1 0 0

1's comp 0 0 0 1 0 0 1 1

1

2's comp. 0 0 0 1 0 1 0 0

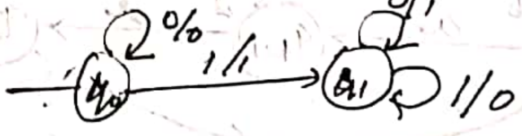
whenever I see 0's, it remains 0.

" I see 1st 1 " " 1

" 1 " 2nd 1 " " 1's comp.

1 1 0 0 0 to.

0 0 1 0 0 2's



1 1 0 0

0 1 0 0

1 0 1 1

0 1 0 0

1

0 1 0 1

0 0 1 1

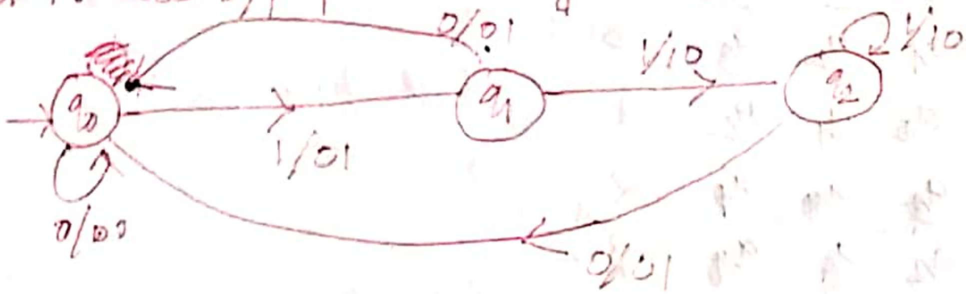
$q_0 \to q_0 \to q_0 \to q_1 \to q_1$

0 0 1 0

Scanned with CamScanner

Prob: 2

What is the O/p produced by the following state m/c



× a) 11 → 01 (For every single symbol the m/c produce two symbols)

× b) 10 → 00

✓ c) Sum of present & previous state

× d) none of these.

1 1 0 1

$q_0 \xrightarrow{} q_1 \rightarrow q_2 \longrightarrow q_0 \rightarrow q_1$

01 10 01 01

Let us assume initially previous bit is 0.

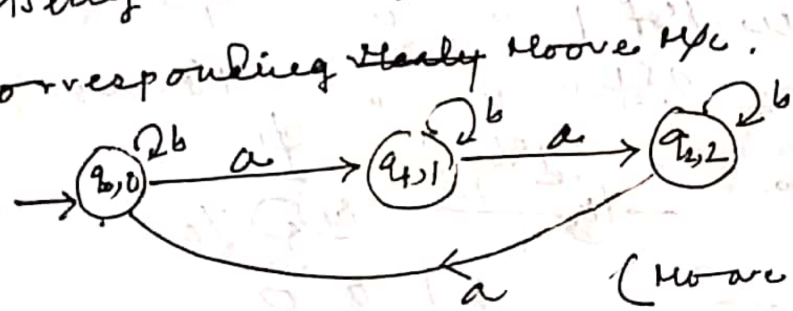
So, $\begin{array}{r} 1 \\ 0 \\ \hline 01 \end{array}$ So, sum of present + previous bit.

Now, $\begin{array}{r} 1 \\ 1 \\ \hline 10 \end{array}$ $\begin{array}{r} 1 \\ 0 \\ \hline 01 \end{array}$ $\begin{array}{r} 0 \\ 1 \\ \hline 01 \end{array}$

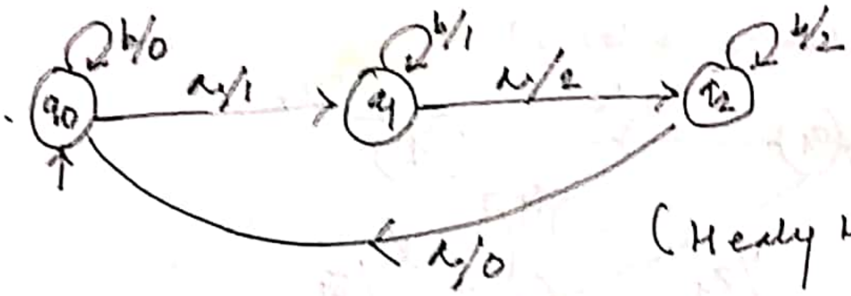
Conversion of Moore to mealy m/c:

* Mealy & Moore M/c's power is same.
* Using Moore m/c we'll count no of a's % 3.
* Corresponding ~~Mealy~~ Moore M/c.

Prob: 1



(Moore M/c)



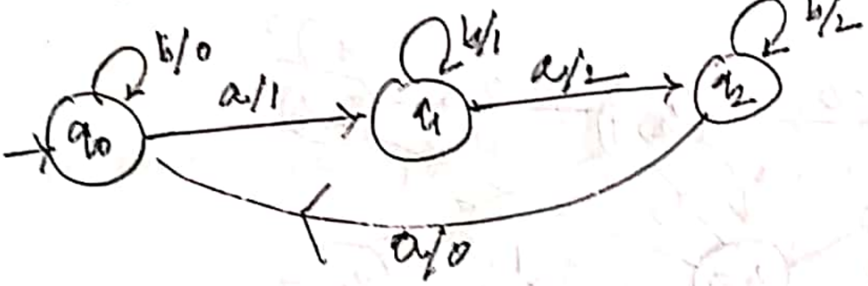
(Mealy M/c)

	a	b	Δ
→ q_0	q_1	q_0	0
q_1	q_2	q_1	1
q_2	q_0	q_2	2

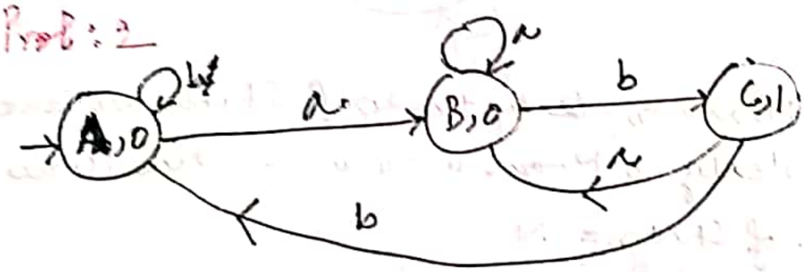
Moore M/c

	a	b
→ q_0	q_1, 1	q_0, 0
q_1	q_2, 2	q_1, 1
q_2	q_0, 0	q_2, 2

Mealy M/c



Prob: 2

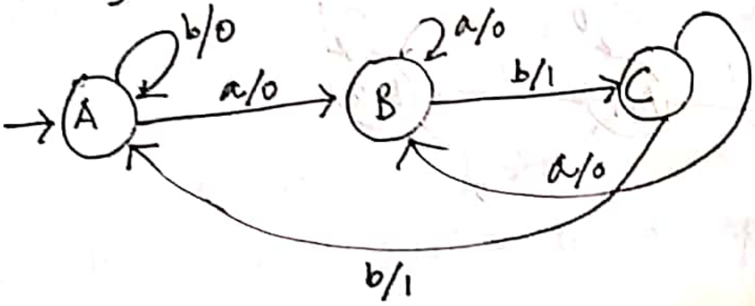


Transition Table of Moore M/c

	a	b	Δ
→ A	B	A	0
B	B	C	0
C	B	A	1

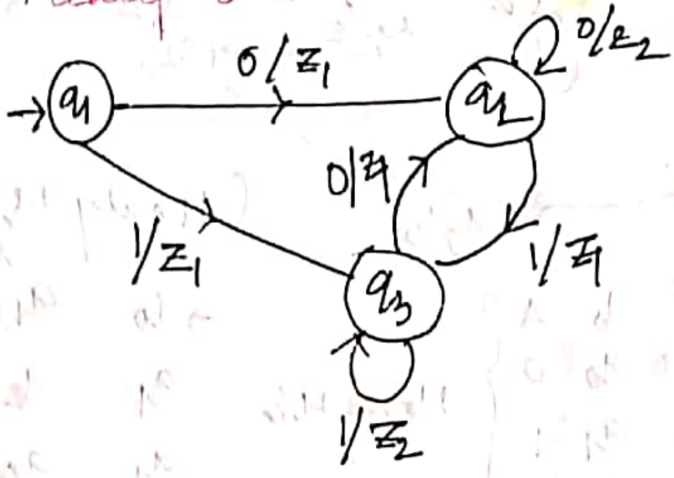
Transition Table of Mealy M/c

	a	b
→ A	B, 0	A, 0
B	B, 0	C, 1
C	B, 0	A, 1

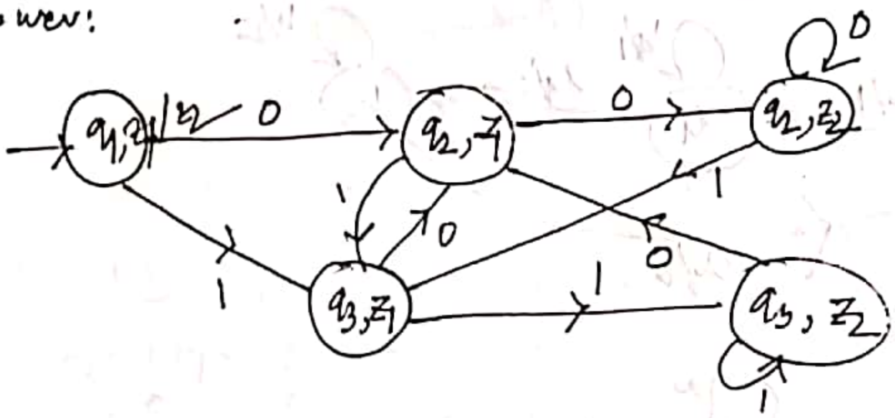


Scanned with CamScanner

Conversion of Mealy to Moore:



Answer:



In case of Conversion of Moore to Mealy the no of States are same.
" " " " " Mealy to Moore " " " " are More.

If in a Mealy M/c no. of state = N
" " " " " " " O/ps = M

Then after converting that m/c to Moore no. of States = M×N.

Prob : 2