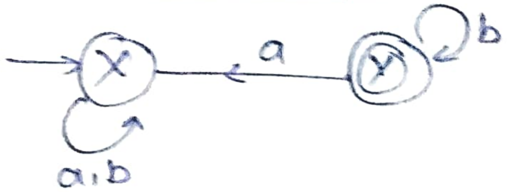
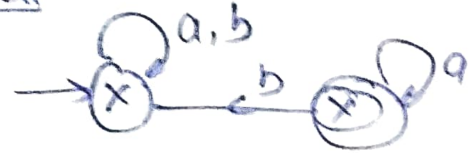
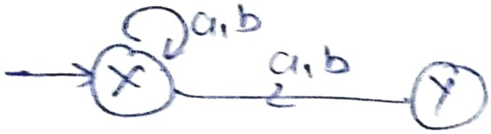
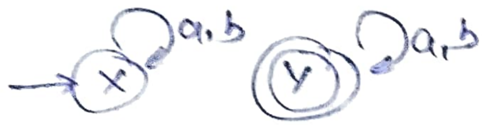
4 transaction with final state

① 

② 

③ 

④ 

No. of 2 state DFA with a designated initial state accepting the universal language over {a, b}.

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

All the possible string with a,b of all the length.

①

	a	b
→*X	X/Y	X/Y
*Y	X/Y	X/Y

] 16

X must be both initial and final due epsilon.

②

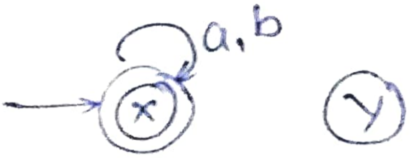
	a	b
→*X	X	X
Y	X/Y	X/Y

] 4

but Y may or may not be final state.

① 

→ Every transaction will get accepted.

② 

] Y can't accept anything bcoz its not a final state.

Total possible = 16 + 4 = 20

No. of 3 state DFA with a designated initial state over $\Sigma = \{a, b\}$.

	a	b
→X	X/Y	X/Y
Y	X/Y	X/Y
Z	X/Y	X/Y

] 6^{6}
3^{6}

There can by multiple final state as –

X Y Z (1)   X Y Z (1)

X Y Z (0)   X Y Z (1)

X Y Z (2)   X Y Z (2)

X Y Z (2)   X Y Z (3)

8

0 → no final state

1 → one ______

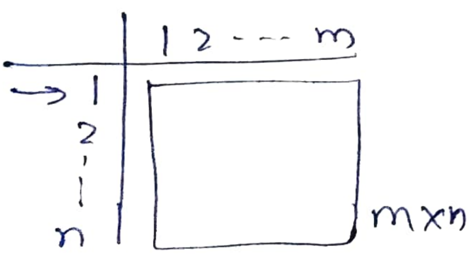
2 → two ______

3 → three ______

Total possibities = $3^6 \times 8$

Containing n state and m symbols with designated initial state

$|Q| = n \quad |\Sigma| = m$



No. of final State –

1 → 0 final state

n → 1 final State

So, no. of Structure possible

$S_n = ({}^nC_0 + {}^nC_1 + {}^nC_2 + \dots + {}^nC_n)$

Total of entries in table $= m \times n$

no. of ways a entry can filled $= n$

$= (n)^{m \times n}$

So to no of DFA possible

$= (S_n)(n)^{n \times m}$

12

In binomial theorem expansion — (1)

$(1+1)^n = {}^nC_0 1^0 1^n + {}^nC_1 1^1 1^{n-1} + {}^nC_2 1^2 1^{n-2} \cdots {}^nC_n$

$= {}^nC_0 + {}^nC_1 + {}^nC_2 \cdots + {}^nC_n$

So, Total no. of DFA possible with n state and m symbols

$= \boxed{2^n \times n^{mn}}$

→ Import for Gate

Important Points regarding DFA

→ from a string of length 'n' no. of substring can be formed equals to $\boxed{\frac{n(n+1)}{2} + 1}$

Substring → selection of continuous elements from string

→ If R_1 and R_2 are regular sets then, $R_1 \cap R_2$ and $R_1 \cup R_2$ are not regular but $\boxed{R_1 + R_2 = \Sigma^*}$. so Σ^* is regular, but R_1^* and R_2^* is not regular.

→ In regular expression
$(a+b^*)^* = (a+b)^*$

② NFA

Introduction to NFA

→ Non-deterministic finite Automata

DFA	NFA
$q_1 \xrightarrow{a} q_2$	$q_1 \xrightarrow{a} q_2$, $q_1 \xrightarrow{a} q_3$, $q_1 \xrightarrow{a} q_4$

On same input it can reached multiple states.

Non-determintistic states or machines are not reals but we can use them in case of back-tracking and exhaustive search.

DFA में हम सारे transitions fix कर देते हैं तो एक DFA केवल 1 Sol^n देता है but NFA multiple Transition show करता है तो हम उससे Multiple Solutions प्राप्त कर सकते हैं।

Quintuple $(Q, \Sigma, \delta, q_0, F)$

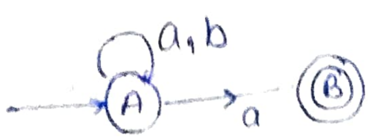
- Q → Set of State (finite)
- Σ → Input symbols
- q_0 → Initial state
- F → A set of final states.

Here : $\boxed{\delta \mid Q \times \Sigma \rightarrow 2^Q}$

Ex

$L = \{$every string ends with 'a'$\}$

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



$A \xrightarrow{a} A$, $A \xrightarrow{a} B$ } Non-determinism

take string ab

a b

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

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

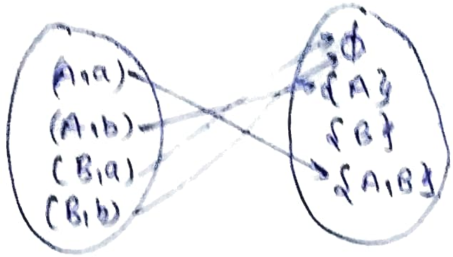
since A is not a final state so, it can be rejected here.

$Q = \{A, B\}$ $\Sigma = \{a, b\}$

$Q \times \Sigma = (A, a), (A, b), (B, a), (B, b)$

$2^Q = \{\phi, \{A\}, \{B\}, \{A, B\}\}$

$(Q \times \Sigma)$ (2^Q)



We can say that every DFA is NFA.

Examples of NFA $\Sigma = \{a, b\}$

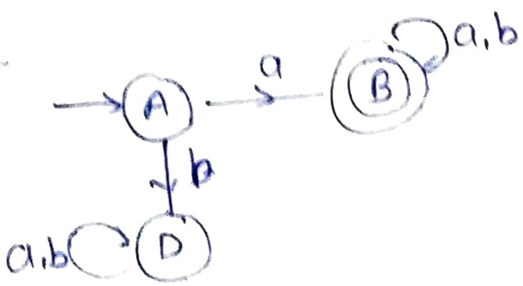
$L_1 = \{$ string starts with a $\}$

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

$A \xrightarrow{b} \phi$ (Dead configuration)

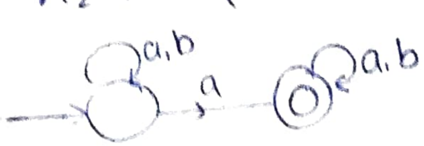
In DFA we have dead state instead of NFA

DFA

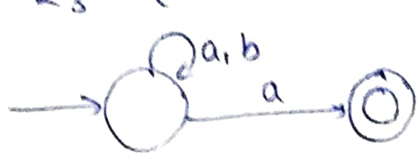


In NFA we have no need to put Dead states.

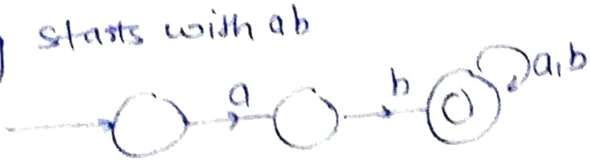
$L_2 = \{$ containing a $\}$



$L_3 = \{$ ends with a $\}$

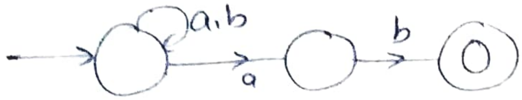


$L_4 = \{$ string starts with ab



L_5 = {Contains ab}

L_6 = {Ends with ab}



NFA to DFA Conversion

NFA is easier and both NFA and DFA are equally powerful.

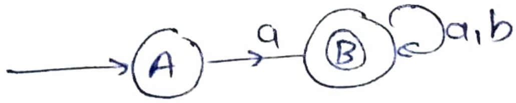
Both are equivalent in power if

i) we can convert NFA to DFA

ii) we can convert DFA to NFA

Since every DFA is NFA so (ii) is already True and we are going to verify (i) using examples.

Ex- L_1 = { Starts with 'a' } $\Sigma = \{a, b\}$

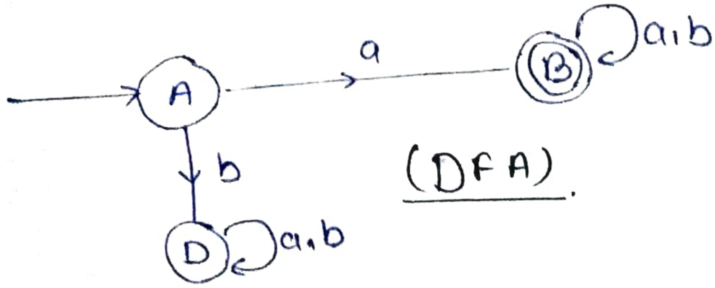
NFA 

$\phi \rightarrow$ Dead configuration in NFA

Conversion using Subset Construction

	a	b
→A	B	ϕ
*B	B	B

	a	b
→A	B	D
*B	B	B
D	D	D



(DFA)

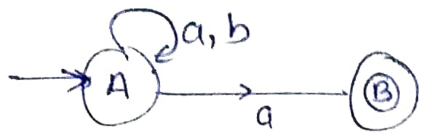
Note -

Convert dead configuration to a new dead state in DFA and also write the transition over alphabets from Dead state in transition table.

Initial & Final state will remain same in both NFA & DFA.

L_2 = { all strings end with 'a' }

NFA-

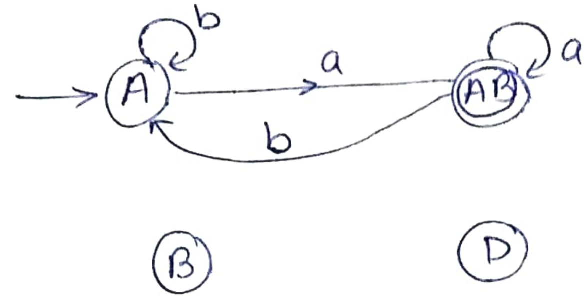


	a	b
→ A	A/B	A
* B	ϕ	ϕ

	a	b
→ A	[AB]	[A]
AB	[AB]	[A]
* B	D	D
D	D	D

①

②

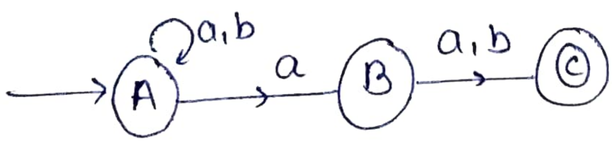


AB can be the final state bcoz AB containg B.

from row ① and ② in transition diagram of DFA the state is not reachable so we have no need to draw it or write it. It means final state of NFA is not same as final state of DFA. 'So DFA conversion is ~~not~~ possible but it doesn't mean that DFA ~~doesn't exist~~ have same states as NFA'.

L = { all strings in which second symbol from RHS is 'a' }

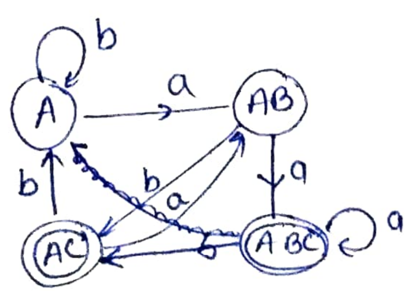
NFA L = { ab, aa, aaa, aab, ... }



	a	b
→ A	{A, B}	A
~~AB~~ B	~~AB~~ {C}	{C}
* C	{ }	{ }

for DFA

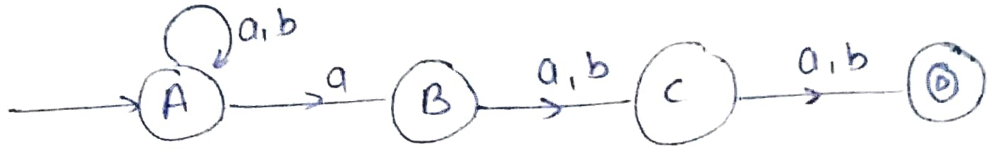
	a	b
→ A	[AB]	[A]
AB	ABC	AC
ABC	ABC	AC
AC	AB	A



Every state that contain C will be final state.

$L = \{$ 3rd symbol from RHS is 'a' $\}$

NFA

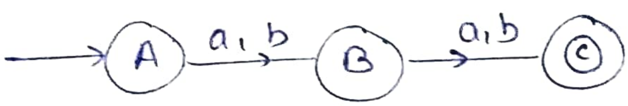


	a	b
→ A	{A,B}	A
B	C	C
C	D	D
* D	∅	∅

for DFA

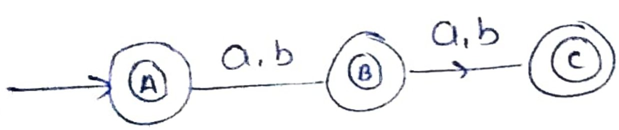
	a	b
→ A	AB	A
AB	ABC	AC
AC	ABD	AD
ABC	ABCD	ACD
* AD	AB	A
* ABD	ABC	AC
* ACD	ABD	AD
* ABCD	ABCD	ACD

$L = \{$ string of length exactly 2 $\}$.

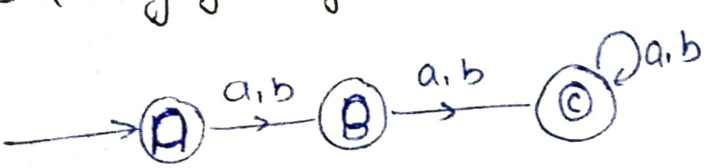


	a	b
→ A	B	B
B	C	C
C	∅	∅

$L = \{$ String of Length atmost 2 $\}$



$L = \{$ string of Length atleast 2 $\}$



Note → for a Language no. of states in DFA is always greater than or equals to no. of state in NFA.

* In exam if given about finite Automata Consider it as NFA.

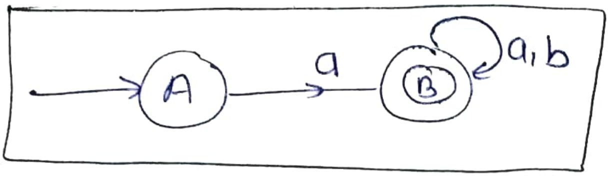
* If we are accepting the string of length 'n' in NFA then no. of States required are 'n+1', different from DFA.

Complementation of NFA

In DFA on taking complement Language was going to be complement but in NFA it's not true.

Ex → $\Sigma = \{a, b\}$

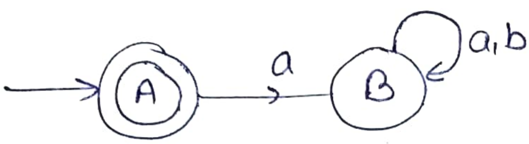
$L_1 = \{\text{Starts with } a\} = \{a, aa, ab, a\ldots\}$



→ M

$\overline{L_1} = \{\varepsilon, b, bb, b\ldots\}$

$\overline{M} =$



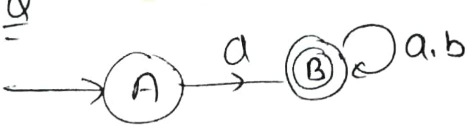
$L_2 = \{\varepsilon\}$ → doesn't contain string starts with 'b'.

Here $\overline{L_1} \neq L_2$

Complementing the NFA is different from complementing a Language.

In Complement a DFA its language will also get complement but complementing the NFA doesn't Complement the language.

Q

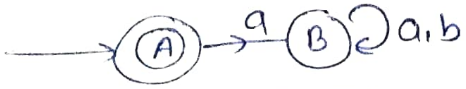


for the given NFA, what is the complement of language accepted by NFA.

Soln $L = \{a, aa, ab, aaa, \ldots\}$

$\overline{L} = \{\varepsilon, b, bb, ba, bbb, \ldots\}$

what is the language accepted by complement of NFA?



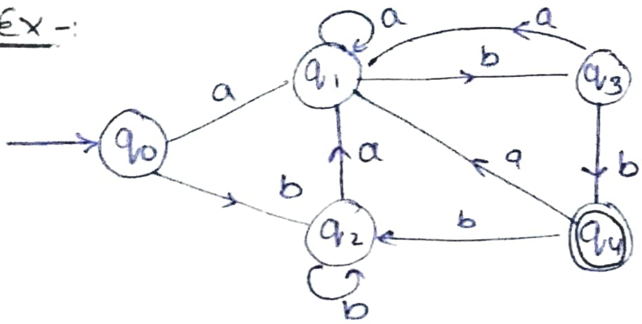
| NFA's complement

$\boxed{L = \{\varepsilon\}}$

But we get the same languages for DFA in both the questions ask above.

Minimisation of DFA

Ex -:



Only two equivalent states can be combine two minimise the DFA.

Two states (p, q) can be combine if

$$\boxed{\delta(p, w) \in F \Rightarrow \delta(q, w) \in F}$$

It means a state 'p' after seeing any string 'w', if it reaches any final state then state 'q' on seeing the same string must be reaches to any final state.

or

$$\boxed{\delta(p, w) \notin F \Rightarrow \delta(q, w) \notin F}$$

If $|w| = 0$ then (p, q) $\longrightarrow$ zero equivalent
— $|w| = 1$ ——— $\longrightarrow$ one equivalent
— $|w| = 2$ ——— $\longrightarrow$ Two ———
⋮
— $|w| = n$ ——— $\longrightarrow$ n equivalent

Note → We can minimize DFA not NFA.

Steps.

1) $\boxed{\text{Delete all the non-reachable states in DFA.}}$

★★★ Never forget this step because it reduces many of the states in given DFA to minimise the time.

2) Draw the state transition table.

	a	b
→ q_0	q_1	q_2
q_1	q_1	q_3
q_2	q_1	q_2
q_3	q_1	* q_4
* q_4	q_1	q_2

3) Find equivalent sets start from zero equivalent sets.

zero-equivalent sets → separate final and and non-final state

$\boxed{[q_0\, q_1\, q_2\, q_3]\ [q_4]}$

1 Equivalent

$(q_0, q_1) \xrightarrow{a} q_1$
$(q_0, q_1) \xrightarrow{b} (q_2, q_3)$

$\boxed{[q_0, q_1, q_2]\, [q_3]\, [q_4]}$

Take anyone from q_0 and q_1 with q_2

$(q_0, q_2) \xrightarrow{a} q_1$
$(q_0, q_2) \xrightarrow{b} q_2$
] Both are in same set of 0 equivalent

Take anyone from q_0, q_1, q_2 with q_3

$(q_2, q_3) \xrightarrow{a} q_1$
$(q_2, q_3) \xrightarrow{b} (q_2, q_4)$
} Both are in different sets of 0 equivalent

Compare RHS of transition with sets of 0 equivalent if all the states are in same set then they are equivalent otherwise make them as a separate set.

2 equivalent

$(q_0, q_1) \xrightarrow{a} q_1$
$(q_0, q_1) \xrightarrow{b} q_2, q_3$
] q_3 and q_2 are in different sets of 1 equivalent

$\boxed{[q_0, q_2]\ \ [q_1]\, [q_3]\, [q_4]}$

$(q_0, q_2) \xrightarrow{a} q_1$
$(q_0, q_2) \xrightarrow{b} q_2$
] Equivalent

$(q_2, q_1) \xrightarrow{a} q_1$
$(q_2, q_1) \xrightarrow{b} q_2\, q_3$
] Not Equivalent

We check for q_2 with both q_0 and q_1 bcoz q_0 and q_1 are not equivalent, they can make an equivalent state with q_2.

3 equivalent - Check for q_0, q_2 only

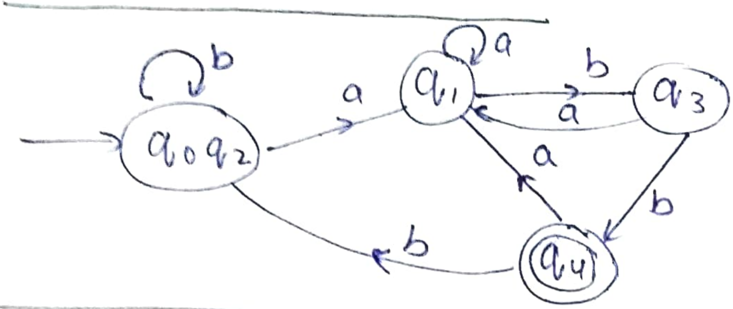
$(q_0, q_2) \xrightarrow{a} q_1$
$(q_0, q_2) \xrightarrow{b} q_2$
] Both are in same set of 2 equivalent

So, final set of states are -

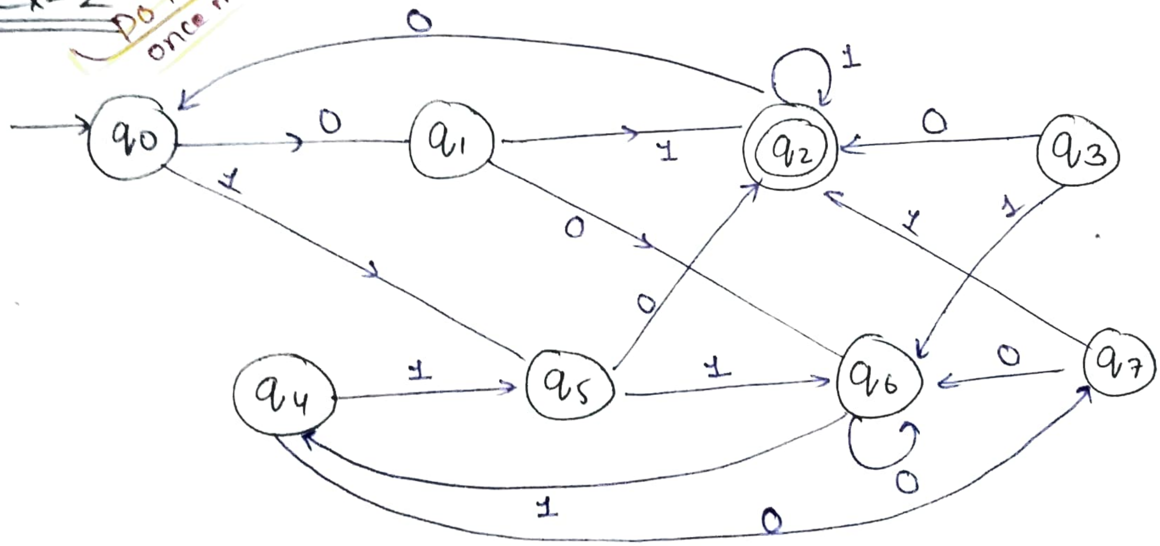
$[q_0, q_2]\, [q_1]\, [q_3]\, [q_4]$

Here 2 and 3 equivalent gives the same output.
So, we can say that we need to continue the procedure till we get same output in two consecutive equivalents.

Final minimized DFA



Ex-2 Do it once more



Step-1 ~~No need to do because there is no unreachable state.~~ State (q_3) is not reachable from q_0 so remove it.

Step-2 Transition Diagram

	a	b
→ q_0	q_1	q_5
q_1	q_6	*q_2
* q_2	q_0	*q_2
~~q_3~~	~~*q_2~~	~~q_6~~ → Trash it.
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

Step-3

0 equivalent

[q_0 q_1 q_4 q_5 q_6 q_7] [q_2]

1 equivalent

[q_0 q_4 q_6] [q_5] [q_1 q_7] [q_2]

→ same transition on both 0 and 1.

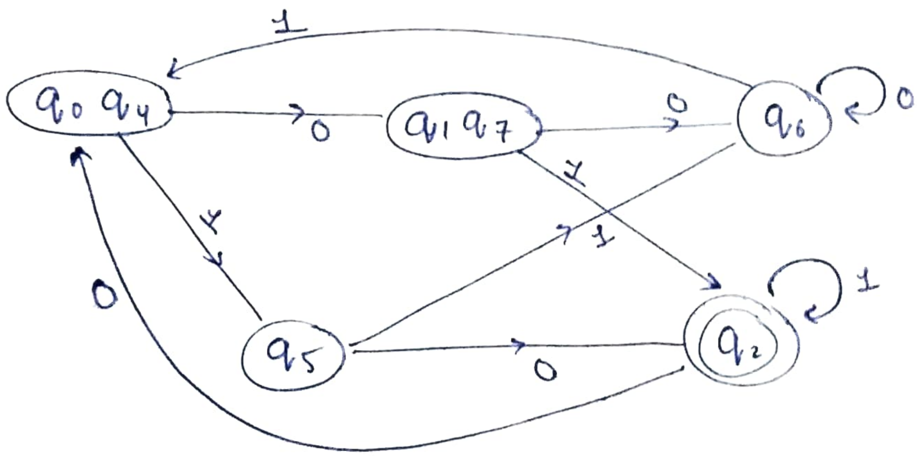
2 equivalent

[q_5] [q_2] [q_0 q_4] [q_6] [q_1 q_7]

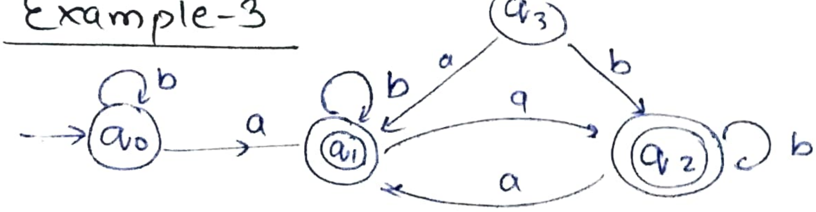
3 equivalent

[q_5] [q_2] [q_6] [q_0 q_4] [q_1 q_7]

so we can draw the minimise DFA.



Example-3



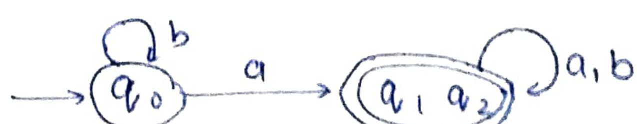
① q_3 is not reachable, so delete it.

②

	a	b
→ q_0	* q_1	q_0
* q_1	* q_2	* q_1
* q_2	* q_1	* q_2

[q_0] [q_1 q_2] → 0 equivalent

[q_0] [q_1 q_2] → 1 equivalent

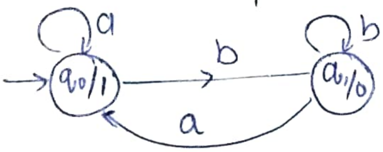


Ans

Moore and Mealy

Introduction → Finite Automata with output.

Moore m/c



$\boxed{\lambda : Q \rightarrow \Delta}$

$q_0 \rightarrow 1$

$q_1 \rightarrow 0$

$\boxed{(Q, \Sigma, \delta, q_0, \Delta, \lambda)}$

$Q \rightarrow$ Set of State

$\Sigma \rightarrow$ I/p Alphabet

$\delta \rightarrow$ Transition fn

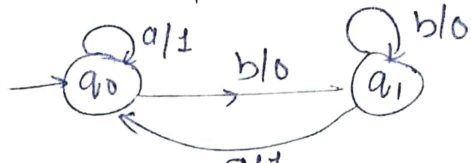
$Q \times \Sigma \rightarrow Q$

$q_0 \rightarrow$ Initial State

$\Delta \rightarrow$ o/p alphabet

$\lambda \rightarrow$ o/p function

Mealy m/c



$\boxed{\lambda : Q \times \Sigma \rightarrow \Delta}$

$(q_0, a) \rightarrow 1$

$(q_0, b) \rightarrow 0$

$(q_1, a) \rightarrow 0$

$(q_1, b) \rightarrow 1$

In a moore machine if we pass the string 'a b' then o/p will be –

State	$q_0 \xrightarrow{a} q_0 \xrightarrow{b} q_1$
o/p	1 1 0

$\boxed{\text{o/p } 110}$

After accessing the input string of size 'n' we get an o/p string of size 'n+1'

On passing same string on mealy m/c then

$q_0 \xrightarrow[1]{a} q_0 \xrightarrow[0]{b} q_1$

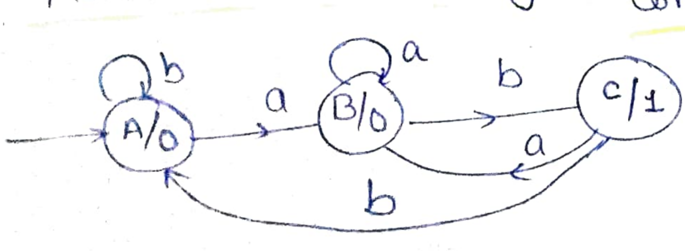
we get 'n' bit output on giving 'n' bit input.

Ex → Moore m/c counting the occurence of substring 'ab', and print 1 on each occurence

$\Sigma = \{a, b\}$ $\Delta = \{0, 1\}$

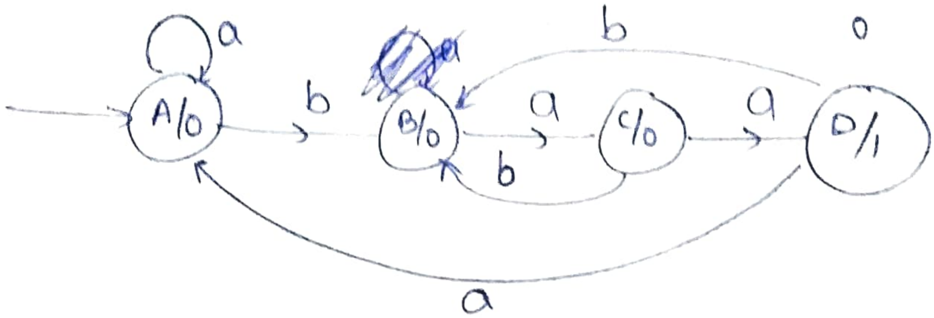
Ex – $a\underset{1}{b}a\underset{1}{b}$, $a\underset{1}{b}bba\underset{1}{b}$ .... and many more

Make a DFA ending with ab because we want to consider all ab substring in given string.



Counting the occurence of substring 'baa'.

$\Sigma = \{a, b\}$ $\Delta = \{0, 1\}$



Note → If there is a question regarding counting no. of substring then design the DFA ending with that string.

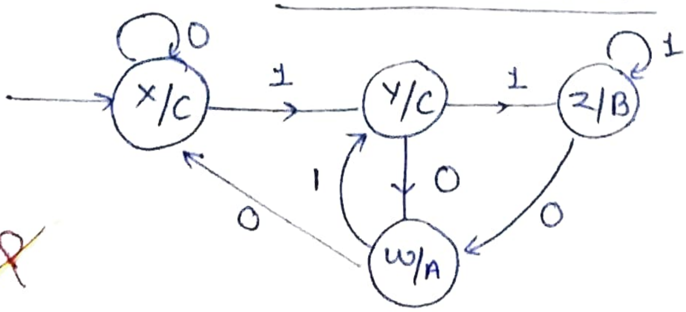
Q $\Sigma = \{0, 1\}$ $\Delta = \{A, B, C\}$

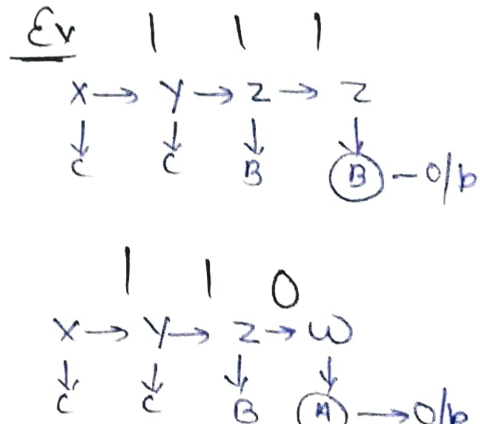
If i/p ends with 10 → o/p A

11 → o/p B

otherwise → o/p C

$Q = (X, Y, Z, W)$



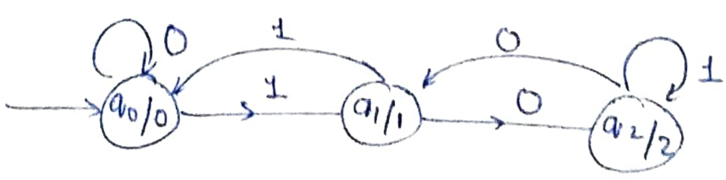


imp

Q Moore m/c accepts binary no and produces residue modulo '3' as o/p.

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

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



In the same problem replace
binary by base 4 numbers
residue modulo 3 by residue modulo 5.

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

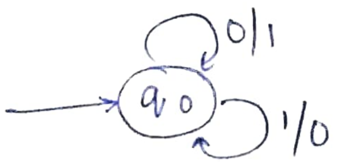
	0	1	2	3	Δ
$\rightarrow q_0$	q_0	q_1	q_2	q_3	0
q_1	q_4	q_0	q_1	q_2	1
q_2	q_3	q_4	q_0	q_1	2
q_3	q_2	q_3	q_4	q_0	3
q_4	q_1	q_2	q_3	q_4	4

Draw the DFA for given transition table. we can frame more questions as similar to this.

imp
Ex → Mealy machine gives 2's complement of a binary number.
(Assume that string is read LSB to MSB and carry is discarded)

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

1's complement



$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0 \xrightarrow{1} q_0$
0 1 0 0

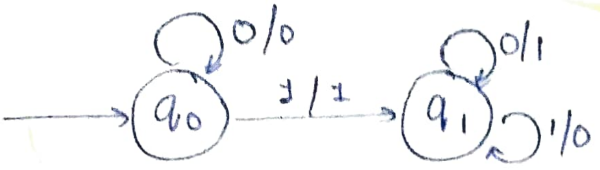
for 2's complement = 1's complement + 1

	1 1 0 0
1's	0 0 1 1
	+ 1
	0 1 0 0

or

1 1 0 1 1 0 0
0 0 1 0 0 1 1
+ 1
0 0 1 0 1 0 0

Steps-
1. Leave starting zero as 0
2. Leave first one as one
3. Complement all the other zero's and 1's



	1	1	0	0
	0	0	1	1
			+	1
	0	1	0	0

0 0 1 1 [Read from LSB]

$q_0 \longrightarrow q_0 \longrightarrow q_0 \longrightarrow q_1 \longrightarrow q_1$

0 0 1 0

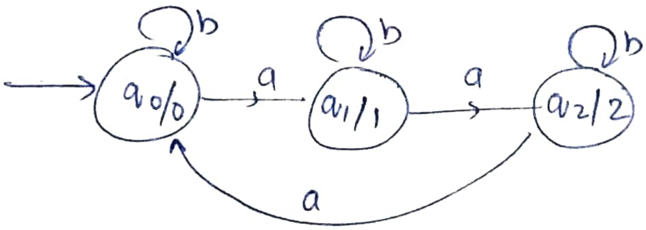
Q Conversion of Moore machine to Mealy machine.

FA with o/p → Moore, Mealy

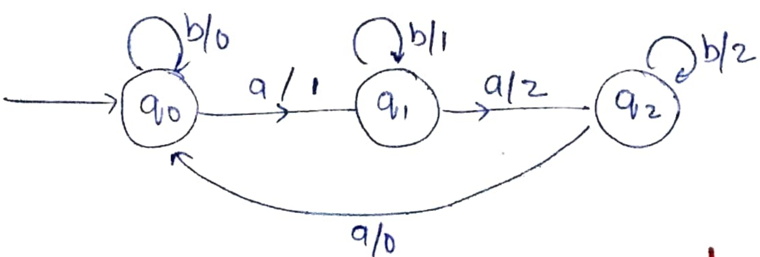
*** $\boxed{\text{Both are equivalent in power.}}$ So we can convert one into another.

Here we convert moore → mealy.

Ex-1 counts no. of a's % 3.



→ Moore Machine.



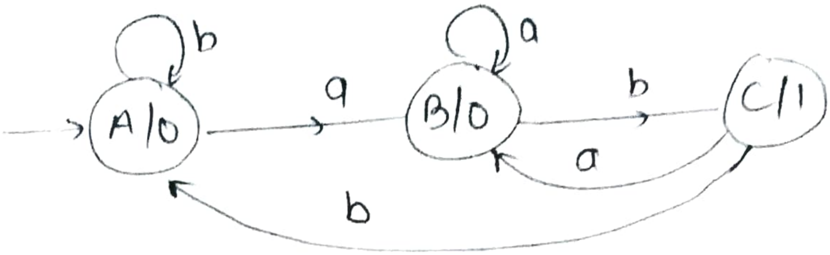
State पर जो output है उसे हर transition के साथ associate कर देना है।

इसे direct करो जब तक Que transition table की demand न कर रहा हो।

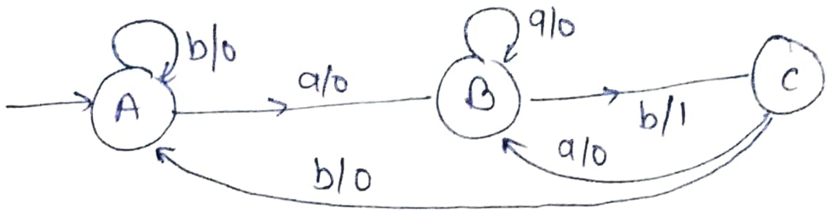
Moore (δ)

	a	b	Δ
→q_0	$(q_1,1)$	$(q_0,0)$	0
q_1	$(q_2,2)$	$(q_1,1)$	1
q_2	$(q_0,0)$	$(q_2,2)$	2

Ex-2



→ Moore m/c



→ Mealy m/c

Output on string a b a b

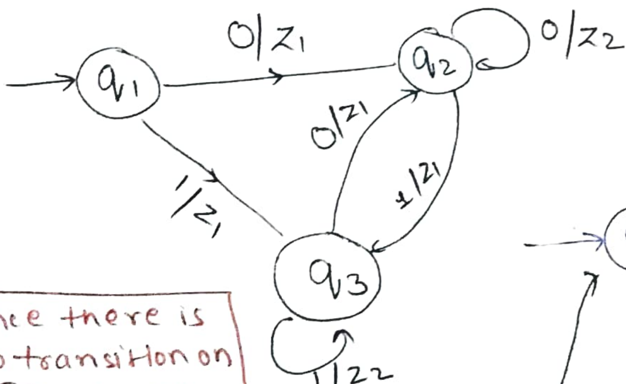
$A \xrightarrow[0]{a} B \xrightarrow[1]{b} C \xrightarrow[0]{a} B \xrightarrow[1]{b} C$

1+1

It means there are 2 ← ab substring.

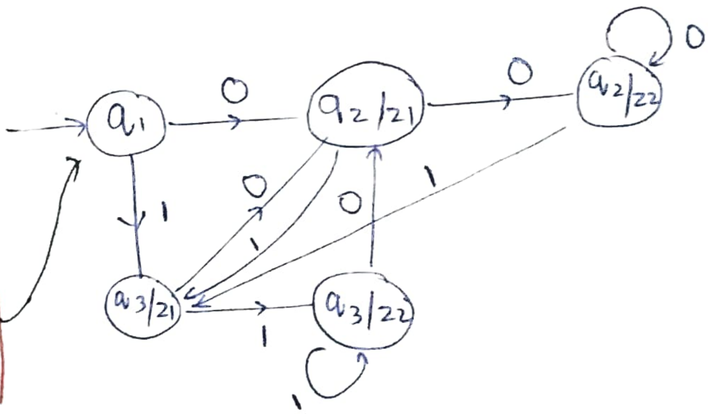
Imp

Conversion of Mealy to moore machine



→ Mealy machine

Since there is no transition on → q_1 so we can draw it as either q_1/z_1 or q_1/z_2 → both are right

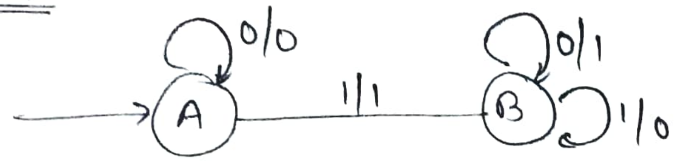


⇒ In conversion of Moore → Mealy no. of states remain same in both, but in conversion of Mealy → Moore no. of state may get increase in Moore as if Mealy has 'N' states and 'M' output then no. of states in new Moore m/c can be 'N×M'.

↓

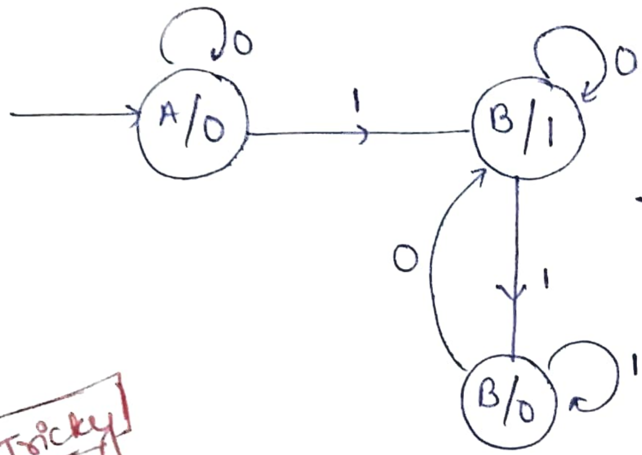
not necessarily but at max it can be.

Ex-2



→ Mealy m/c

→ 2's complement of string reading from LSB

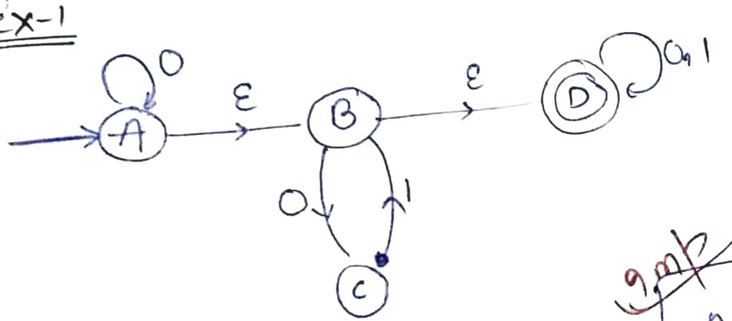


→ Moore machine

Bit Tricky

Epsilon NFA and Conversion of Epsilon NFA to NFA

Ex-1



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

$\delta : Q \times \Sigma \cup \{\varepsilon\} \rightarrow 2^Q$

ε-closure (A) = {B, A, D}

$A \xrightarrow{\varepsilon} A$

$A \xrightarrow{\varepsilon} B \xrightarrow{\varepsilon} D$

Transition table of NFA

	0	1
A	{A, B, C, D}	{D}
B	{C, D}	{D}
C	{φ}	{B, D}
D	{D}	{D}

imp

$A \xrightarrow{\varepsilon^*} A \xrightarrow{0} A \xrightarrow{\varepsilon^*} \{A, B, D\}$

$A \xrightarrow{\varepsilon^*} B \xrightarrow{0} C \xrightarrow{\varepsilon^*} C$

$A \xrightarrow{\varepsilon} D \xrightarrow{0} D \xrightarrow{\varepsilon^*} D$

$A \xrightarrow{\varepsilon^*} A \xrightarrow{1} \cancel{A}$

$A \xrightarrow{\varepsilon^*} B \xrightarrow{1} \cancel{\phi}$

$A \xrightarrow{\varepsilon^*} D \xrightarrow{1} D \xrightarrow{\varepsilon^*} D$

$B \xrightarrow{\varepsilon^*} B \xrightarrow{0} C \xrightarrow{} C$

$B \xrightarrow{\varepsilon} D \xrightarrow{} D \xrightarrow{} D$

$B \xrightarrow{\varepsilon^*} B \xrightarrow{1} \phi$

$B \xrightarrow{\varepsilon^*} D \xrightarrow{1} D \xrightarrow{\varepsilon^*} D$

$C \xrightarrow{\varepsilon^*} C \xrightarrow{0} \phi \xrightarrow{\varepsilon^*}$

$C \xrightarrow{\varepsilon^*} C \xrightarrow{1} B \xrightarrow{\varepsilon^*} B, D$

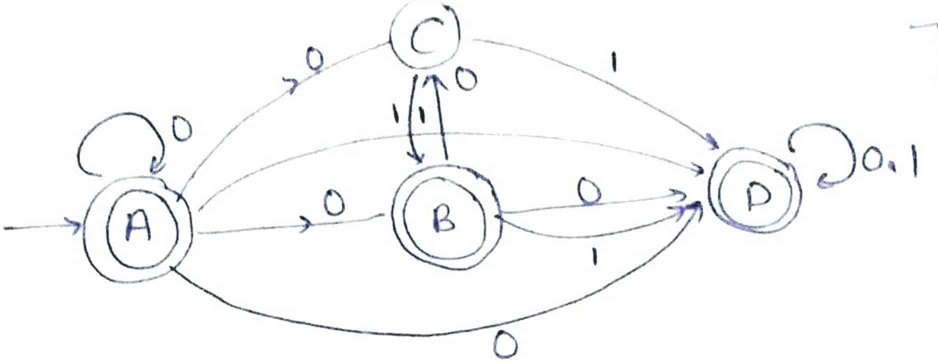
$D \xrightarrow{\varepsilon^*} D \xrightarrow{0} D \xrightarrow{\varepsilon^*} D$

$D \xrightarrow{\varepsilon^*} D \xrightarrow{1} D \xrightarrow{\varepsilon^*} D$

→ No. of states are same in NFA.

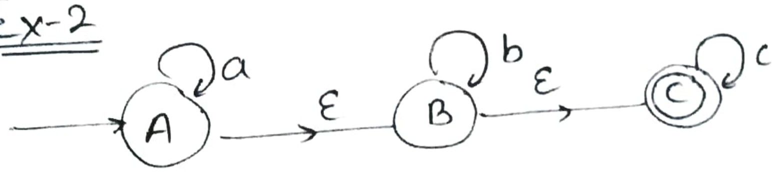
→ final states will remain same.

→ If you can reach to final state from any state by only seeing 'ε' then that state will also become the final state.



NFA for given ε-NFA

Ex-2



	a	b	c
→ A	{A, B, C}	{B, C}	{C}
B	{φ}	{B, C}	{C}
C	{φ}	{φ}	{C}

$A \xrightarrow{\varepsilon^*} A, B, C \xrightarrow{a} A, \phi, \phi \xrightarrow{\varepsilon^*} A, B, C$

$A \xrightarrow{\varepsilon^*} A, B, C \xrightarrow{b} \phi, B, \phi \xrightarrow{\varepsilon^*} B, C$

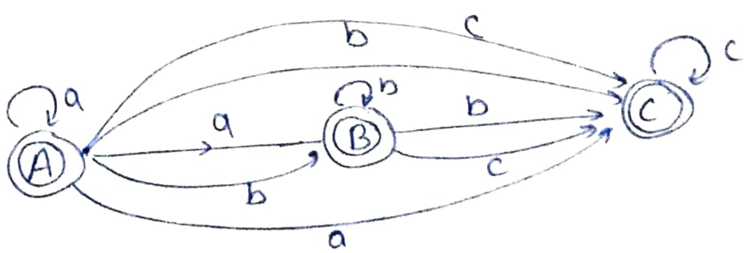
$B \xrightarrow{\varepsilon^*} B, C \xrightarrow{a} \phi, \phi$

$B \xrightarrow{\varepsilon^*} B, C \xrightarrow{b} B, \phi \xrightarrow{\varepsilon^*} C, B$

$C \xrightarrow{\varepsilon^*} C \xrightarrow{a} \phi$

$C \xrightarrow{\varepsilon^*} C \xrightarrow{b} \phi$

$C \xrightarrow{\varepsilon^*} C \xrightarrow{c} C \xrightarrow{\varepsilon^*} C$



NFA