CLASS : Roll No.: Supplement No.: SBA

(1)

CHAPTER - I

TOC

T - Theory
O - of
C - computation
By this name I
This is some thing about computation

Total :- 100 marks
35 + 35 + 30
↓ ↓ ↓
Test 1 Test 2 CVV.
ESE.

what is computation
↳ computation nothing but any task that can be performed by calculator or computer

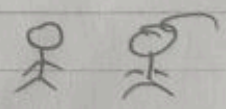
so we are going to mathematical model a computer or any machine in general an then we are going to study the theory about it.

which means what are the ~~ability~~ capability of the machine what are the problems could be solve by this machine and what are the limitations of this machine

Basics of the subject

* Symbol :-
symbol is the basic building block in the subject TOC.

symbol can be any letter or anything also pics.

symbol a, b, 1, 0, ... 

↓
symbol can form alphabet
↓
This is general represented by "Σ" sigma
Alphabet -
↳ Alphabet is nothing but some collection of symbols
it is nothing but alphabet
e.g. $\{a, b\}$

now once if you define the alphabet all the strings

We define the alphabets all the strings

what is string → going to discuss it.

↳ every thing is based on this set {a, b}

predefines set called alphabet.

So you can have any no. of alphabet, any no: of symbols in alphabet.

even have

{a, b, c} {0, 1} {0, 1, --- 9}

It is a finite set.

From the alphabet we are going to discuss string.

String:→

string is nothing but sequence of symbol which is nothing but.

eg. a, b, aa, ab, bc ---

If alphabet is {a, b} small (a) is string over this alphabet which is of length one (1) & small (b) which is also string of ~~alphabet~~ alphabet length is one.

aa, ab bc ---- length is 2.

{a, b} how many ~~length~~ string of length n or possible over this ~~string~~ alphabets.

How many strings of length 2 or possible over this alphabets {a, b} length 2.

ie. aa
ab
ba
bb

which means 4

Now I am asking you how many strings of length n or possible.

— — — — — — n

we have n line spaces & we have 2 options to fill in with

so what are the two options.

ie. ⊙ either a or b

{a, b} {a, b}

We define the alphabets all the strings.

<u>What is string</u> → going to discuss it.

↳ every thing is based on this set {a, b}
predefined set called alphabet.

So you can have any no. of alphabet, any no: of symbols in alphabet.

even have
{a, b, c} {0, 1} {0, 1, --- 9}

It is a finite set.
From the alphabet we are going to discuss string.

<u>String</u>:→
string is nothing but sequence of symbol which is nothing but.
eg. a, b, aa, ab, bc ---

If alphabet is {a, b} small (a) is string over this alphabet which is of <u>length one (1)</u> & small (b) which is also string of ~~alphabet~~ alphabet length is one.

aa, ab bc ----- length is 2.

{a, b} how many ~~length~~ string of length n or possible over this ~~string~~ alphabets.

How many strings of length 2 or possible over this alphabets {a, b} length 2.

ie. aa
ab
ba
bb
which means 4

Now I am asking you how many strings of length n or possible.

- - - - - - - n

we have n line spaces & we have 2 options to fill in with

so what are the two options.

ie. ⊙ either a or b
{a, b} {a, b} - - -

so. $2 \times 2 \times 2 \times \cdots n$

so far 2^n

In fact you can even extend this concept any alphabet.

Language :-

In English Language is nothing but collection of words. (alphabets, words, sentence, Grammar)

In TOC - Language is nothing but collection of strings.

Let's see

$\Sigma = \{a, b\}$ sigma is equal to a, b

Languages ~~can~~ can be define like this

L_1 = set of all strings of length 2

so using this alphabet sigma what are set of all strings of length 2 we can form.

so definitely the set is going to be

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

This set is finite set.

so I can called that L_1 is the finite language.

so let us define one more lang.

L_2 = " " " , 3

let is set L_2 is set of all string of length 3 over the same alphabet sigma ie.

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

The L_2 is going to be

$L_2 = \{aaa, aab, aba, abb, baa, bab, bba, bbb\}$

so there 8 string.

L_2 is the finite

CLASS : Roll No.: Supplement No.: 2

L_3 = set of all strings where each string start with 'a'
= {a, ~~ab~~aa, ab, aaa, aab, aba, abb - - - - - }

so L_3 is infinite language.

& L_1 & L_2 is Finite.

A language which can be form over the sigma which can be form over the alphabets can be
Finite or Infinite.

Powers of Σ

~~let is see~~ ~~we have~~ ~~$\Sigma = \{a, b\}$~~

Let us see we have define sigma case $\Sigma = \{a, b\}$ which means our input alphabet symbols a, b only 2 symbols.

Σ^1 - sigma power one.

Σ^1 = set of all strings ~~of~~ over Σ of length '1' (exactly one)
= {a, b}

$\Sigma^2 = \Sigma \times \Sigma$ (sigma concatinate with sigma)
ie. $\{a, b\} \times \{a, b\} = \{aa, ab, ba, bb\}$

set of all strings of length 2.

Σ^3 = set of all strings of length 3
$= \Sigma^2 \times \Sigma = \Sigma \times \Sigma^2 = \Sigma \times \Sigma \times \Sigma$
$= \{aa, ab, ba, bb\} \{a, b\}$

$= \{aaa, aba, baa, bba, aab, abb, bab, bbb\}$

The coordinality of this string is $|\Sigma^3| = 8$

Σ^n = n length string.

Σ^0 = set of all string of length '0' $\Sigma^0 = \{\epsilon\}$.
It is also called NULL string.

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

Σ^* It is nothing but you in place of star you can substitute any number starting from zero.

so. $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \ldots$

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

$\boxed{\text{infinite}}$

sigma ~~star~~ star is nothing but set of all string possible a, b of all length.

may be called it as mother of languages
— " — universal set.

sigma star contains every thing that include a, b
— " — all possible things possible over a, b

once we define sigma (Σ) and sigma star (Σ^*) next thing is we can define a language.

so we have already seen that Language

L_1
L_2 }
L_3

previously or thing but sets of string.

L_1, L_2 } Finite L_3 - infinite

now we can see this

Σ^* $\boxed{L_1\ L_2\ L_3}$

$L_1 \subseteq \Sigma^*$ L_1 is subset of Σ^*
$L_2 \subseteq \Sigma^*$ L_2 — " —
$L_3 \subseteq \Sigma^*$ L_3 — " —

Language is nothing but any subset it can be finite or infinite which should we define ~~precisely~~ precisely.

How many languages possible over sigma star (Σ^*) infinite.

No. of string possible over sigma infinite
No of lang. ——"—— sigma star infinite.

L is finite.
let us say ex is
$\Sigma = \{a, b\}$
let us say set
$L_1 = \{aa, ab, ba, bb\}$
set of all string length 2.

This is lang & if i give any string

aaa

if i ask question whether this string present in this language or not you could take each string & examine, and you say that it is not there.

Think language is finite we can do it.

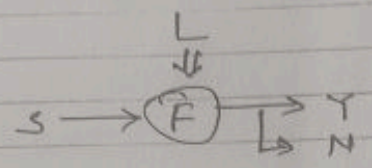
but lang. is infinite.
string start with 'a'
e.g. $L2 = \{a, aa, aaa, ab \ldots\}$

$s = \{baaa\}$

if a i ask you whether this string is present or not (in lang. L2)
Should i
How long i compare this
Dont look at first symbol it is clear this string is not in language L2

But Machine doesn't think this way
it give at method (linear search)

So simple thing is how can you given a language come up with a finite representation some finite representation.

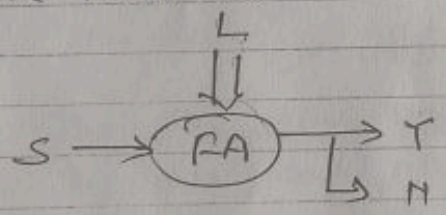


using this if i give a string now you should be able to say the string is in the lang. or string is not in the lang.

That is a simple thing.

Finite Automata

Given a language i will try to construct some finite representation which is called Finite Automata



using this Finite Automata if i give a string this Finite Automata this Finite automata say Yes i accept this string. and No i reject this string.

which means this string is present in this lang. or not.

Before Discussion about what a Finite automata and types of Finite automata i would like to give small example.

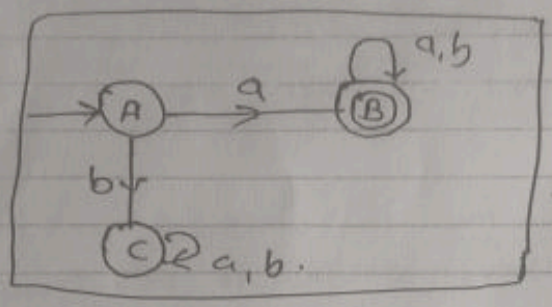
let us say.

↳ a lang which is set of all string which start with a.

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

so i am going to construct a diagram which is call it finite automata.
then we discuss the defination.

FA.



so
↳ even if you right it like this
$\{a, aa, ab, aaa ---\}$ infinite set.

or
use this finite representation both are same

so all this circles are <u>states</u>.

In this case state A, B, & C.
double circle is final state.

Initial state is A
Final state is B

If is take a string
e.g. aab.

Q | check the string present in this lang. or not.

so now i going to do this

initially i starting from A

$A \rightarrow B$ - so a is going to B

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

now upon scanning the entire string started from initial state and we are able to final state.

we reach to the initial state to final state then string is set to be accepted.

eg. now string, bba

is this string D in lang. no. seq string is start with b.

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

but c is not final state

so string is not accepted F.A.

Finite Automata →

Finite automata is a collection of 5-tuple $(Q, \Sigma, \delta, q_0, F)$

Q → is ~~set~~ a finite set of states, which is non empty

Σ → is input alphabet, indicates input set

q_0 → is an initial state and q_0 is in Q.

F → is a set of finial states.

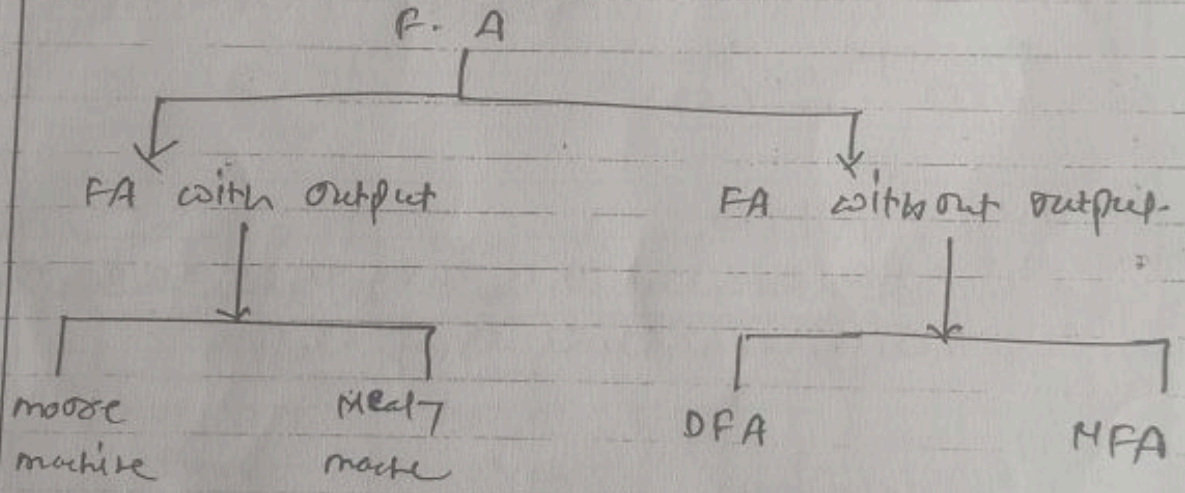
(6) $\delta \; Q \times \Sigma \quad \{A, B, C\} \times \{a, b\} \quad (A,a)\ (A,b)\ (C,a)\ (B,a)\ (B,b)\ (C,b)$

$\delta \Rightarrow$ is a transition function or mapping function using this function the next state can be determined.

→ The finite state system represents a mathematical model of a system with certain input.
· The model finally gives certain output.

eg. The very good example of finite state system is a control mechanism of ~~the~~ elevator

This mechanism only remembers the current floor number pressed, it does not remember all the previously pressed numbers.



eg.

2:15 - 3:15

Present no TOC 01/07/2019

5, 7, 8, 11, 13, 16, 17, 18, 19, 20, 23, 24, 25, 26, 36, 44, 54, 65

02/07/2019

7, 8, 9, 10, 11, 13, 14, 16, 18, 20, 21, 22, 23, 24, 25, 26, 31, 37, 38, 39, 42, 45, 46, 53, 54, 55, 56, 65

Punctuation eliminate & display String

45 TOC 04/07/2019

1, 2, 3, 5, 8, 9, 11, 14, 16, 17, 19, 20, 22, 23, 25, 26, 30, 34, 35, 36, 37, 41, 42, 45, 46, 47, 48 53, 55, 59,

2:15-3:15 TOC 08/07/2019

1, 3, 8, 11, 13, 17, 18, 21, 24, 27, 36, 40, 43, 45, 46, 54, 65

Sundarga, Mayurbhanj
(66) (67)

50 6/12
1000 28/12
1000 31/12
250 1 Jan
100 5 Jan
3000 7 Jan
250 11/01

TOC 09/07/2019

1, 2, 3, 5, 8, 9, 10, 11, 12, 13, 14, 18, 19, 21, 22, 24, 25, 26, 30, 34, 35
36, 40, 41, 43, 44, 46, 47, 53, 54, 59, 65, 66, 67

TOC 10/7/2019

1, 3, 5, 8, 9, 10, 11, 12, 13, 14, 15, 18, 21, 22, 24, 25, 30,
36, 46, 54, 66, 67, 68, 69, 70, 71, 72, 75, 76, 77, 78, 79

TOC 11/7/2019

1, 2, 5, 7, 11, 13, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29, 30, 41, 45
46, 54, 59, 65, 66, 67, 68, 71, 72, 73, 74, 75, 77

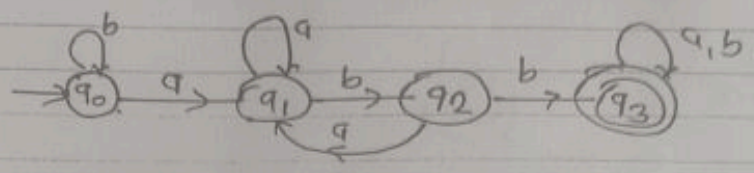
TOC 15/7/2019

1, 3, 4, 9, 13, 14, 17, 22, 23, 24, 25, 30, 35, 36, 37, 39, 40, 41, 43, 45, 49,
54, 55, 56, 67, 68, 71, 72, 75, 76, 77, 78,

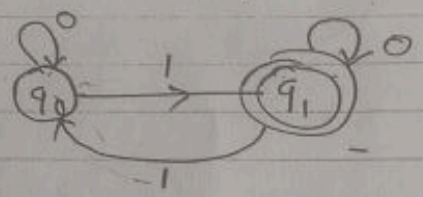
CLASS : Roll No.: Supplement No.: 7

* Construct FA for the strings where abb is a substring

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

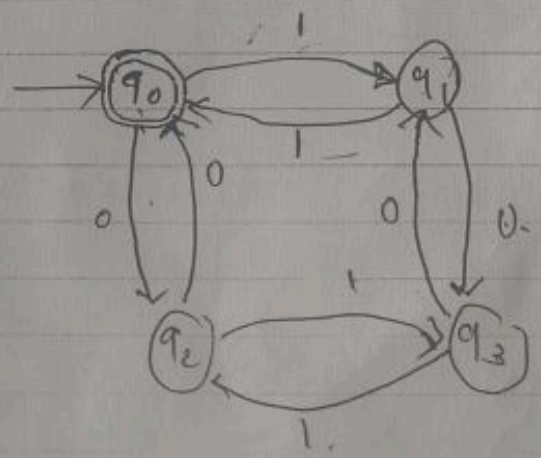


* Design FA for the string odd no. of 1's over the lang. $\Sigma = \{0, 1\}$

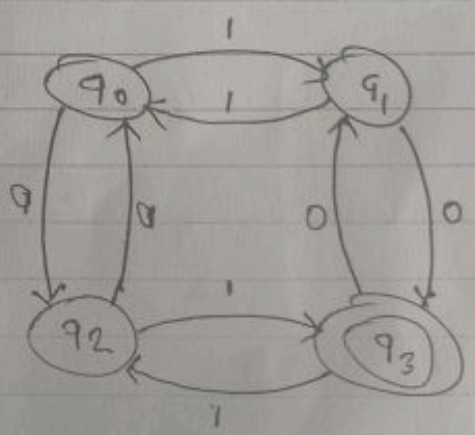


* no. of 1's is even & no. of 0's is even.
no. of 1's is odd & no. of 0's is odd

Situation		State
no. of 0's	no. of 1's	
Even	Even	q_0
Even	odd	q_1
odd	Even	q_2
odd	odd	q_3

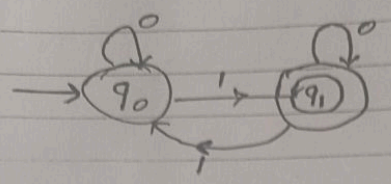


For even no.

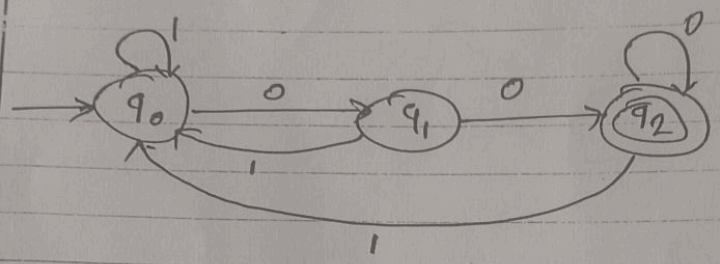


For odd no.

* Odd no. of 1's and any no. of 0's.

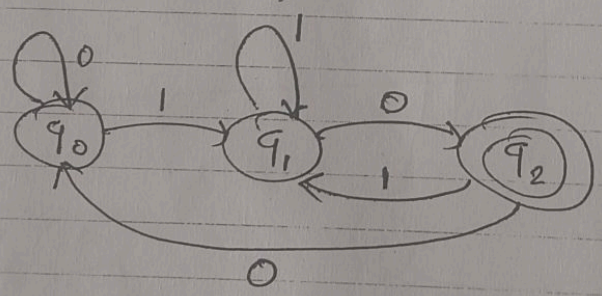


* Design FA to accept the string that always ends with 00.

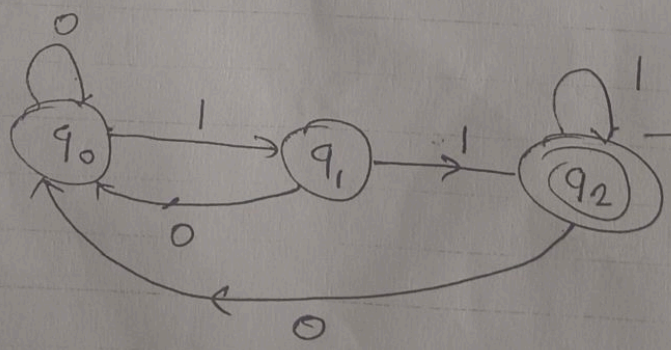


* Design FA for a larg. of string 0 & 1 that
1) Ending with 10
2) Ending with 11
3) Ending with 1

	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_2	q_1
q_2^*	q_0	q_1

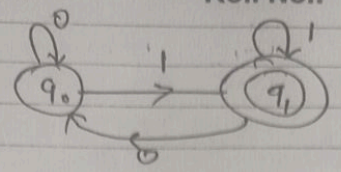


End with 10



	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_2
q_2^*	q_0	q_2

End with 11



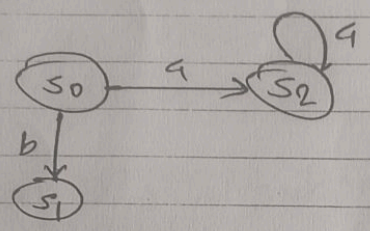
	0	1
$\rightarrow q_0$	q_0	q_1
q_1^*	q_0	q_1

Ends with 1

DFA

The F.A. is called Deterministic FA. if there is only one path for a specific input from current state to next state.

DFA can be shown below



state S_0 for input 'a' there is only one path going to S_2

similarly S_0 there is only one path for input b going to S_1

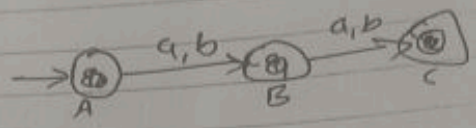
DFA can be represented by the same 5-tuples described in the definition of FSM.
all the above examples are actually the DFA.

Construct DFA that accept set of all strings over $\{a, b\}$ of length 2.

Input alphabet is $\Sigma = \{a, b\}$
identify lang.
$L = \{aa, ab, ba, bb\}$

This is the basic steps.

once you identify this is the lang. take the possible string.
i.e. aa
construct the skeleton



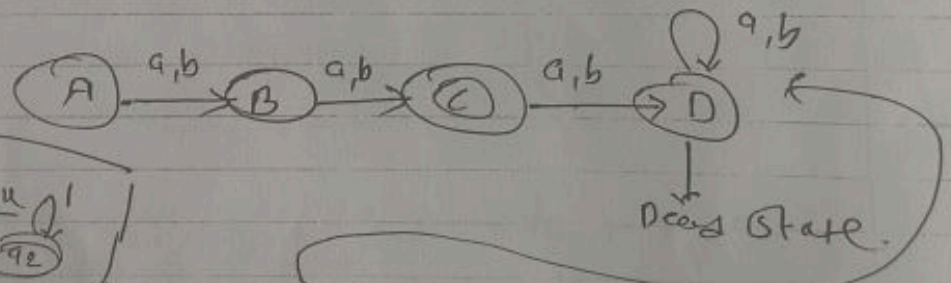
we reach capital C is aa, ab, ba, or bb

lets take any example

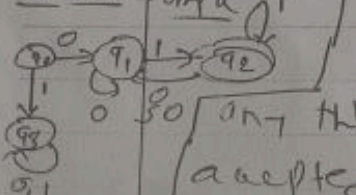
a a
A → B → C

now if you are in C & you get one more a what does it mean. It means that we have seen a string of length 3.

"aaa" is the string in that lang. definitly no. there for if I see either 'a' or 'b' the string length is going to exceed 2 there for I will be going to some other state i.e. D



one more example



start with 0 ends with 1

so any thing after this i should not accepted & going to be in this state

so how is this D acting is once you reached D. it is never going to go back to final state.

so that is way this state is called as Dead State or trap state.

CLASS : Roll No.: Supplement No.: 9

* Design FA to check whether given decimal number is divisible by three.

⇒ Given decimal number is divisible by three we need to take the i/p no. digit by digit.

Also while considering its divisibility by three we have to consider that the possible remainders could be 1, 2, or 0.

The remainder 0 means, it is divisible by 3.

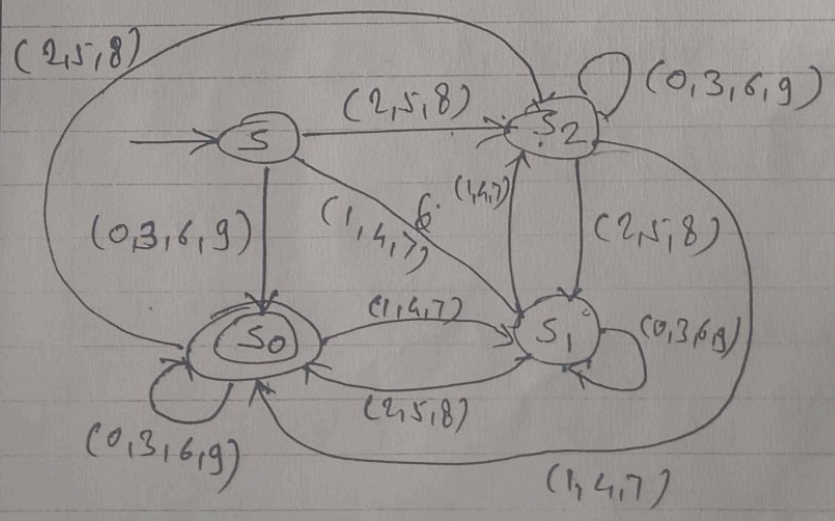
input set is {0, 1, 2, ---9} since decimal no. is a input. ~~we will~~
we will get either remainder 0, 1 or 2 while testing its divisibility by 3.

so we need to group these digits according to their ~~we~~ remainders.

The groups are given below.

remainder 0 group S_0 (0, 3, 6, 9) S_0
— 1 — S_1 (1, 4, 7) S_1
— 2 — S_2 (2, 5, 8) S_2

we have named out these states as S_0, S_1 & S_2
The state S_0 will be the final state as it is remainder 0 state.



lets us test the above FA if the no. is 36

it will process by reading the no. digit by digit.

start $\boxed{3}$ 6 — 252

go to state S_0. — 2

3 $\boxed{6}$ — 5

from S_0 to S_0 — 2

36 — 253

inputs ends & we are in final state S_0.

Eg. 121

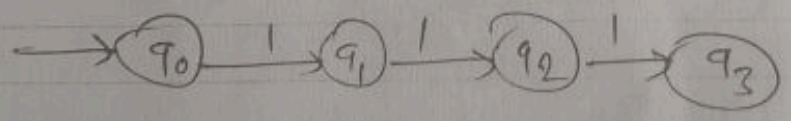
1 S_1 21

12 S_0 1

121 S_1 which is remainder 1 state.

* Design a DFA $L(M) = \{ w \mid w \in \{0,1\}^* \}$ and w is a string that does not contain 3 consequetive 1's.

3 consequetive 1's occur the DFA will be



here two or single consequetive 1 is acceptable.

Non Deterministic Finite Automata (NFA)

→ The concept of NFA is exactly reverse of DFA.

DFA → In DFA there is only one path. for a specific input from current state to next state.

NFA →
when there exists many paths for a specific input from current state to next state.

e.g.

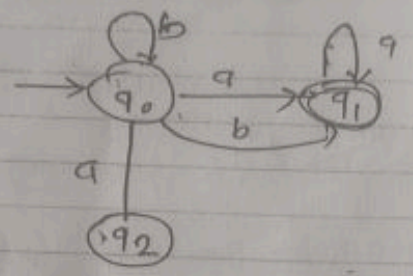


Fig. NFA

Note that the NFA shows from q_0 for input 'a' there are two next states q_1 & q_2.

Similarly from q_0 for i/p b the next states are. q_0 & q_1.

Thus it is not fixed or determined that with a particular input where to go next.
Hence this F.A. is called non deterministic F.A.

Consider the i/p string bba.

input	b	b	a
Path ①	q_0	q_0	q_1
②	q_0	q_0	q_2
③	q_0	q_1	q_1

Thus you cannot take the decision of which path had to be followed for deriving the given string.

NFA → DFA

Def of NFA

It is a collection of 5-tuples.

1) Q – is a finite set of states.
2) Σ – is a finite set of inputs
3) δ – is called next state or transition function.
4) q_0 – is ~~final~~ initial state.
5) F is final state where $F \subseteq Q$.

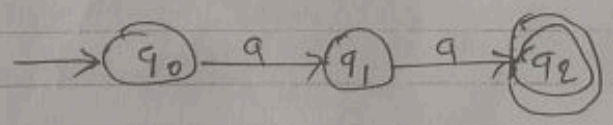
There can be multiple final states.

Question – what is the use of NFA

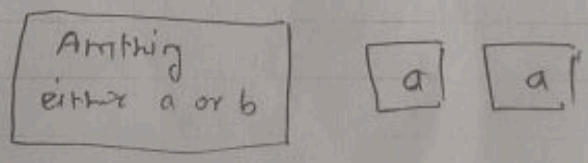
The NFA is basically used in theory of computations because they are more flexible and easier to use than the DFA.

Ex Design NFA to accept strings with a's and b's such that the string ends with 'aa'

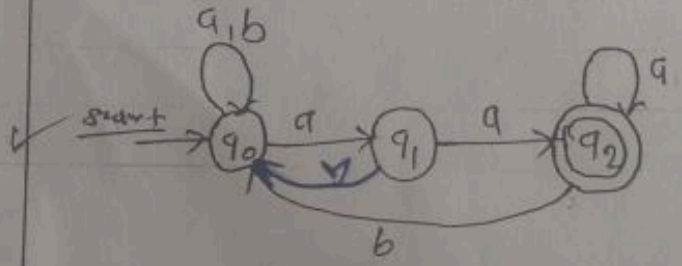
⇒ The simple FA which accepts a string with 'aa' is



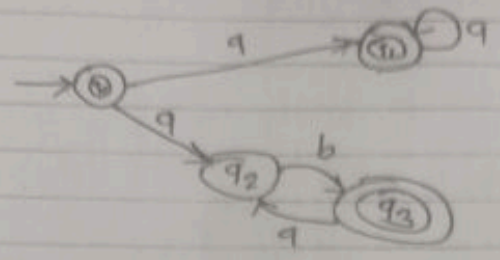
Now there can be a situation where in



Here required NFA is as.



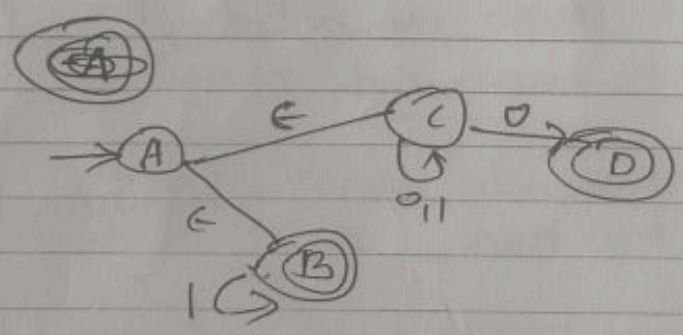
Design NFA for $a^* + (ab)^*$



NFA with ϵ moves.

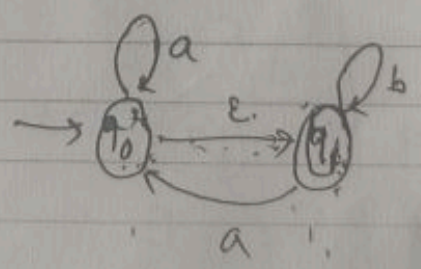
We also allow ϵ transitions ; arrows labeled with the empty string. These allow the NFA to change state without consuming an input symbol.

Ex: Accepts all binary strings where the last symbol is 0 or that contain only 1's.



$a^* b^*$

$L = \{\epsilon, a, b, ab, abab\}$



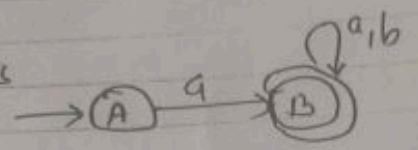
convert NFA to DFA

$L_1 = \{$starts with 'a'$\}$.

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

where L_1 is a language where set of all string start with on 'a' and sigma is $\Sigma = \{a, b\}$

so.
NFA is



Now want to convert DFA.
There are many methods convert NFA to DFA.

but we follow some method call subset construction

Now draw the state transition table for the NFA.

State transition table content rows are State & columns are inputs

	a	b
A	B	ϕ
B	B	B

NFA

Capital A is on small a is going to capital B & capital A is on small b is going to capi ϕ means it is also known as dead configuration.
and every dead configuration in NFA is nothing but dead state in DFA.

i construct the state transition table of DFA

	a	b
A	B	D
B	B	B
D	D	D.

and there is ϕ. ϕ is nothing but dead configuration in NFA. coming to DFA ϕ is translate to DFA Dead state.

ie. D.

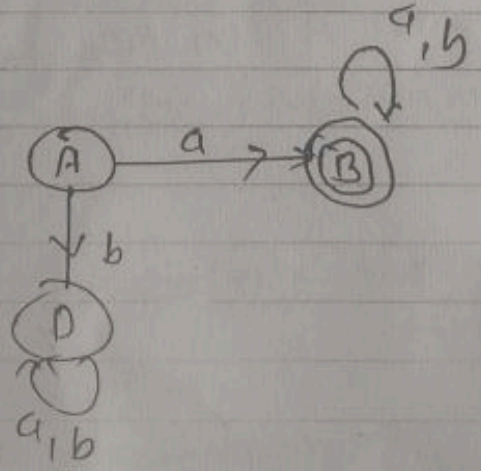
~~and~~ B is now B & B.

and the state table is not complete. without ~~rigting~~ writing state D

if you take any Dude state. Dude state actually has to be a trap state.

once i reach D is should be in the dude state itself.

convert this state transition table, to state transition diagram.

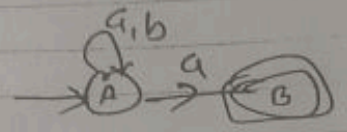


DFA.

In initial state B is a final state
so. here B is a final state.

Ex All string ends with a.

L = { ends with an 'a' }



	a	b
A	{A,B}	{A}
* B	{φ}	{φ}

convert DFA.

	a	b
[A]	[AB]	[A]
[AB]		

A and B both

* In capital A in NFA has two state but in DFA i can't write two state.

every state on every input go to only one state.

∴ can combine this two state write it as [AB].

Note :- [AB] This is a not two state it is a single state.

[AB]

A row & B row

union of A & B for a.

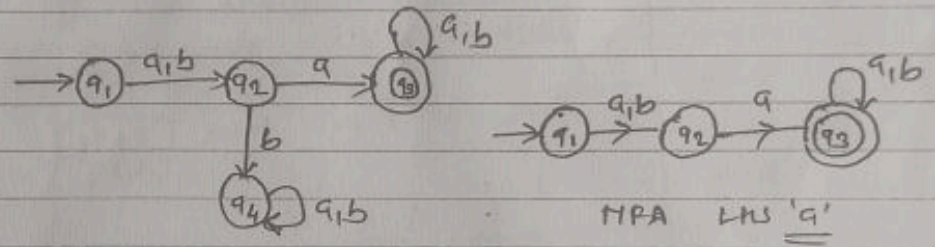
so again it is going to be [AB]

unit of A & B for b

so again it is going to [A]

* Conversion of NFA to DFA for the example " all strings in which second symbol from RHS is 'a'

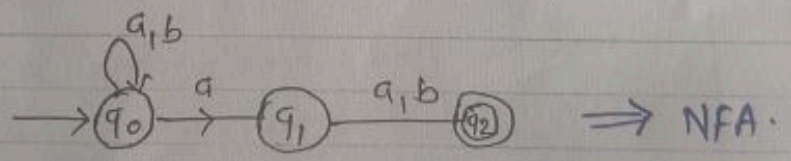
First we see second symbol from LHS 'a' construct DFA.



DFA LHS 'a' NFA LHS 'a'

Now second symbol in RHS 'a' is big complicated construct directly to DFA. so we let we construct first NFA then DFA.

$L = \{ aa, ab, aaa, aab \ldots \}$



$\Rightarrow$ NFA.

NFA

Initially i can see anything.

$\Sigma^* a(a+b)$

$\Sigma^* aa$ $\Sigma^* ab$

first draw state transition table.

	a	b
(A) q_0	$\{q_0, q_1\}$	$\{q_0\}$
(B) q_1	$\{q_2\}$	$\{q_2\}$
(C) $*q_2$	$\{-\}$	$\{-\}$

NFA TST

State transition Table.

	a	b
$[q_0]$ [A]	$[q_0 q_1]$ [AB]	$[q_0]$ [A]

Now ~~capital~~ $[q_0]$ [A] is am already given to row.

But i didn't given the row for $[q_0 q_1]$ [AB]

Those for i will take $[q_0 q_1]$ [AB]

	a	b
$[q_0]$	$[q_0 q_1]$	$[q_0]$
$[q_0 q_1]$		

Now after taking $[q_0 q_1]$ carike the transition on here

Now $[q_0 q_1]$ is two state. so. we are seen the NFA table. rows A & B take the union of this two state,

$A = \{A, B\}$ & $B = \{C\}$ for symbol 'a'

Now $A \cup B = \{A, B, C\}$

	a	b
$[q_0]$	$[q_0 q_1]$	$[q_1]$
$[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_0 q_2]$

what are the state given to reach.

$[q_0q_1]$, $[q_0]$, $[q_0q_1]$ have given the row

$[q_1]$, now $[q_1]$ have given the row

But didn't given the row $[q_0q_1,q_2]$ & $[q_0q_2]$

Let's first $[q_0q_2]$

	a	b
$[q_0]$	$[q_0q_1]$	$[q_0]$
$[q_0q_1]$	$[q_0q_1,q_2]$	$[q_0q_2]$
$[q_0q_2]$	$[q_0q_1]$	$[q_0]$

now union q_0 & q_2

	a	b
$[q_0]$	$[q_0q_1]$	$[q_0]$
$[q_0q_1]$	$[q_0q_1,q_2]$	$[q_0q_2]$
* $[q_0q_2]$	$[q_0q_1]$	$[q_0]$
* $[q_0q_1,q_2]$	$[q_0q_1,q_2]$	$[q_0q_2]$

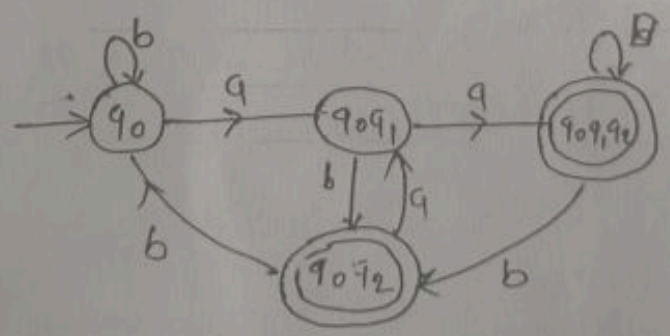
DFA

Now take the union of q_0 & q_1 & q_2

Now draw the state transition diagram for DFA.

How many state are there. (4 states)

$[q_0]$, $[q_0q_1]$, $[q_0q_2]$, $[q_0q_1,q_2]$

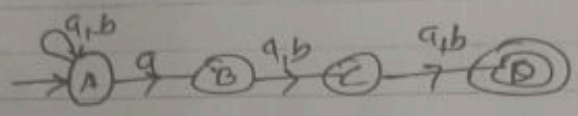


Final state is NFA is q_2 so. Final state in DFA is any state which is containing q_2 is final state.

* convert NFA to DFA to accept ~~the~~ all the string in which third symbol from RHS is 'a'

Σ^* a aa
ab
ba
bb

First construct NFA.



	a	b
A	{A,B}	{A}
B	{C}	{C}
C	{D}	{D}
*D	{ }	{ }

NFA

		a	b
*	→ [A]	[AB]	[A]
(AUB)	[AB]	[ABC]	[AC]
(AUC)	[AC]	[ABD]	[AD]
*	[AD]	[AB]	[A]
	[ABC]	[ABCD]	[ACD]
*	[ABD]	[ABC]	[AC]
*	[ACD]	[ABD]	[AD]
*	[ABCD]	[ABD]	[ACD]

"AD"

DFA

Q1 What is the final state in DFA
Ans the common D is the final state.

~~In NFA "n" state are required then we should ...~~

If an minimal NFA contain 'n' state. minimal DFA for same language contain 2^n state.

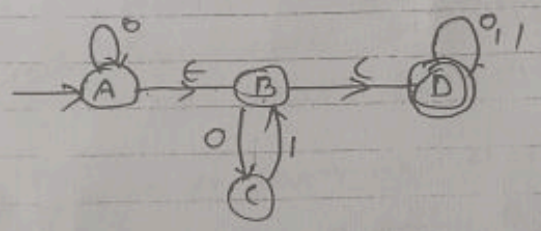
$$\boxed{'n' \longrightarrow '2^n'}$$

Question :- What is the maximum no. of state possible for DFA to solve the problem if NFA is able to solve the probe with 'n' state.

Ans:- $n \rightarrow 2^n$ (2 power n)

* Conversion from NFA with ϵ to NFA without ϵ

ϵ-NFA - special kind of NFA.
ϵ - empty string.



Define $\epsilon NFA = (Q, \Sigma, \delta, q_0, F)$

Here δ is different.

δ in NFA you remember that

$$\delta : Q \times \Sigma \rightarrow 2^Q.$$

If i some state (Q) & if is some input Σ i was going to 2^Q. or i will going to a subset of all state

~~ϵNFA~~ only diff betn ϵNFA & NFA is here i can have ϵ

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

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

which means From particular state (Q) on seeing some symbol (Σ) i can go to some state.

on particular state (Q) without seeing anything a transition is possible.

Mean we can move one state to another state without any symbol.

$\boxed{\text{Property of } \epsilon \text{ closure}}$

let us see.

ϵ - closure (A) =

ϵ closure of any state is nothing but what are all the state which ~~we~~ you can reach only on see ϵ from A.

ϵ - closure (A) is nothing but what are all the state which you can reach from capital A. only on see ϵ.

Always is can reach A on seeing ϵ

i.e. $A \xrightarrow{\epsilon} A$

~~MOO~~ $\cancel{A \rightarrow B}$

MOO A can reach B on seeing ϵ

$A \xrightarrow{\epsilon} B$

as well as B can reach D on seeing ϵ

$B \xrightarrow{\epsilon} D$

$\therefore$ ϵ closure of A is = {A, B, D}

ϵ closure (A) = {A, B, D}

$A \xrightarrow{\epsilon} A$

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

This is ϵ closure.

Now convert ε-NFA to NFA

First so i will go with state transion table

	0	1
A	—	

so i am not having ε so i am writing about 0 & 1

so that we seen particular entry. it is nothing but what does A capital (A) on zero (0) where it does go.

ie. $\delta(A, 0)$ in the NFA

No conversion.

From state A what happens on zero.

so in terms of what happens on zero directly i see what is ε closure of A. which means what are all the state that are reachable from A only on seeing ε..

Then 0 & then ε.

ie. $\underset{A}{\epsilon^*} \quad 0 \quad \epsilon^*$

what is exactly happen is.

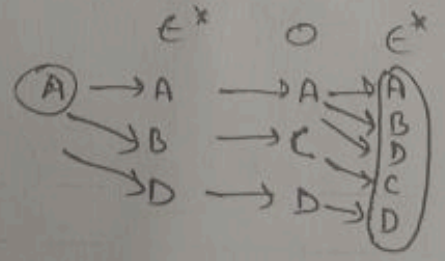
$\boxed{\epsilon\text{-closure}(\delta(\epsilon\text{ closure}(A), 0))}$ ← This is the procedure

it is nothing but i can also find out this entry capital A on zero. in NFA.

(A) (0)

I have to find out

capital A on ϵ^* again apply zero and again apply to ϵ^* (ε closure)



(1) capital A ε closure. one is A then B and D.

(2) capital A on zero. B on zero D on zero

again find out ϵ^*

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

ϵ^* 1 ϵ^*

$A \to A \to \phi$
$\to B \to \phi$
$\to D \to D \to D$

B (0,1)

ϵ^* 0 ϵ^*

$B \to B \to C \to C$
$\to D \to D \to D$

ϵ^* 1 ϵ^*

$B \to B \to \phi$
$\to D \to D \to D$

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

C (0,1)

ϵ^* 0 ϵ^*

$C \to C \to \phi$

ϵ^* 1 ϵ^*

$C \to C \to B \to B$
$\to D$

	0	1
C	{Φ}	{B,D}

D (0,1)

ϵ^* 0 ϵ^*

$D \to D \to D \to D$

ϵ^* 1 ϵ^*

$D \to D \to D \to D$

	0	1
D	{D}	{D}

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

NFA without ϵ

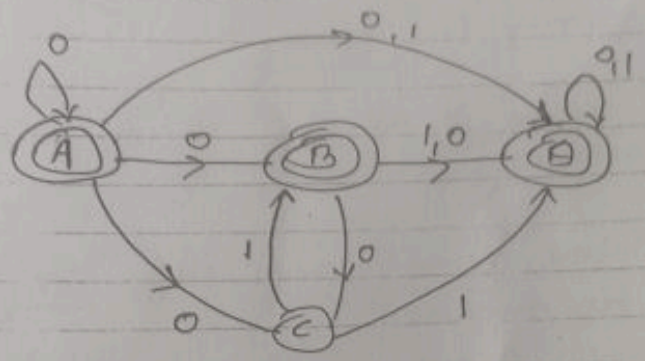
conversion of NFA to DFA no. of states are increased.

But conversion of NFA ∈NFA to NFA no. of state are not increased.

Now what B the final state.

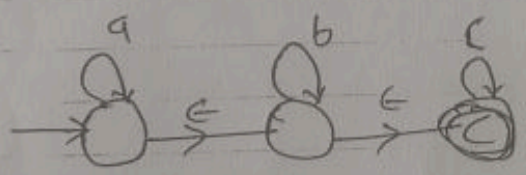
So when we convert ∈NFA to NFA. the final states are going to increased.

So whatever state that can reached to final state only on epsilon (∈) is going to be final state.



NFA without ∈

* Example 2 ∈NFA to NFA.



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

ϵ^* a ϵ^*

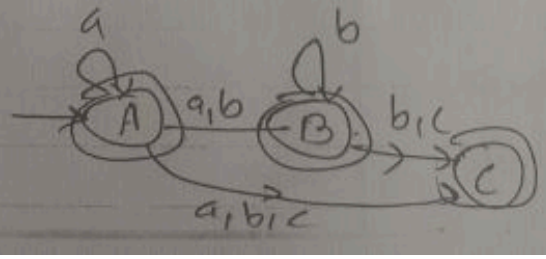
A → A → A → A
→ B → ϕ → B
→ C → ϕ → C

ϵ^* b ϵ^*

A → A → ϕ
→ B → B → B
→ C → ϕ → C

ϵ^* c ϵ^*

A



* FA with output

- The FA is a collection of $(Q, \Sigma, \delta, q_0, F)$ where Q is a set of states including q_0 as a start state.

- In FA after reading the input string if we get final state then the string is said to be "acceptable"

- If we do not get final state then it is said that string is "rejected".

- That means there is no need of output for the finite Automata.

- The "accept". or "reject" acts like "Yes" or "no" output for the machine.

- But if there is a need for specifying the output other than yes or no then in such a case we require finite automata along with output

- There are two types of FA with output
  1) Moore Machine
  2) Mealy machine.

FA with O/P (Deterministic)

- Moore M/C
- Mealy M/C

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

Q - Finite set of states

Σ - i/p alphabet

$\delta \rightarrow$ Transition Function.

$Q \times \Sigma \rightarrow Q$.

$q_0 \rightarrow$ initial state

Δ - O/P alphabet (capital delta)

λ - O/P function (lambda)

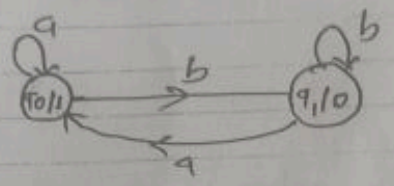
Moore M/C :-

- Moore machine is a finite state m/c in which next state is decided by current state and current i/p symbol.

- The o/p symbol at a given time depends only on the present state of the machine.

- In moore m/c a λ is a function from $Q \rightarrow \Delta$ (Q to capital delta). which means for every state output is associated.

e.g.



$\lambda : Q \rightarrow \Delta$

so λ say that every state output is associated.

For this diagram

q_0 output associated is 1

$(q_0 \rightarrow 1)$

$q_1 \rightarrow 0$ (q_1 output associated is 0).

Mealy Machine :-

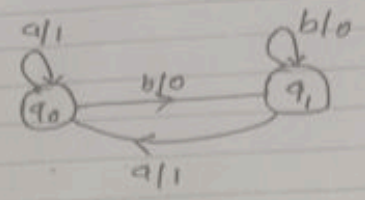
- Mealy M/C is a machine in which output symbol depends upon the present input symbol and and present state of the machine.

- In mealy machine output is associated like this

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

For a state for a given input there will be some o/p.

e.g.



For
state q_0 given input 'a' then o/p will be '1'

$(q_0, a) \rightarrow 1$
$(q_0, b) \rightarrow 0$
$(q_1, a) \rightarrow 1$
$(q_1, b) \rightarrow 0$

Remember is

* Moore machine is going to associate a symbol or output with the state itself.

* Mealy machine is going to associate output as the combination of current state as well as the input.

e.g.
Moore m/c

In moore m/c i/p is ab then what is happen

a b

initially i will be in state q_0. whenever i am in state q_0 o/p associated is '1' ~~is~~ there for 1 will be printed.

a b

$q_0 \rightarrow q_0 \rightarrow q_1$
$\downarrow \quad \downarrow \quad \downarrow$
1 1 0

So on seeing input is $\boxed{ab}$ moore m/c printing the o/p is $\boxed{110}$

string of length n is input then the the o/p produce string of length is $n+1$.

$$n \longrightarrow (n+1)$$

Even without seeing anything q. going to produce something. that is a region region

* eg. (Moore M/C)

input is a b.

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

output is associated in input.

$$\boxed{\text{'n'} \longrightarrow \text{'n'}}$$

given n bit input the output will be n bit

* Moore Machine example

counting the occurrence of substring ab.

construct a moore m/c that takes set of all strings over $\{a, b\}$ as i/p and prints '1' as o/p for every occurance of 'ab' as a substring.

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

ie. whenever is see ab i print a '1' as output

so output is $\Delta = \{0, 1\}$

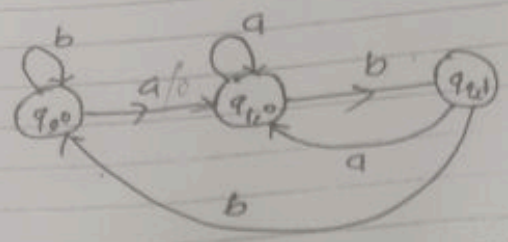
ab print 1

ab ab
1 1

ab bb ab
1 1

in this example we construct DFA is ending with ab.

ie.



according to our problem specification if C D made to output 1 that can solve our problem.
and remaining state you can simply given '0' as the input.

See the example

input is a b

a b

A → B → C

0 0 1

ip se ab ab

a b a b

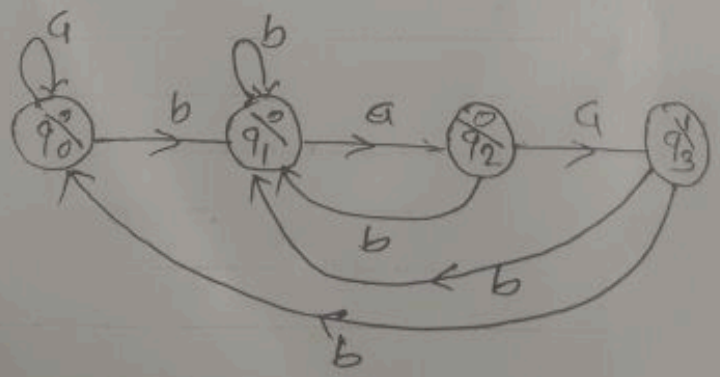
A $\xrightarrow{a}$ B $\xrightarrow{b}$ C $\xrightarrow{a}$ B $\xrightarrow{b}$ C

0 0 1 0 1

(2) ~~baa~~ occurences of substring 'baa'

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

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



b

let us say b a a

$q_0 \xrightarrow[0]{b} q_1 \xrightarrow[0]{a} q_2 \xrightarrow[1]{a} q_3$ (1)

Let us baabaa

$$q_0 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$$

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

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

Mealy Machine

Ex Construct a mealy machine that takes binary numbers as i/p and produces 2's complement of that no. as o/p.

- Assume the string is read LSB to MSB and end carry is discarded.

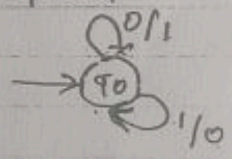
i/p $\Sigma = \{0, 1\}$ Now produce 2'complement of binary
o/p $\Delta = \{0, 1\}$ no. there o/p is equal to binary no. we should get 2's complement of no.

Suppose

1011

First calculate 1's complement

i.e. 0100



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

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

Let us say i have a no.

1100

Find 2's complement

↳ First to find 1's complement

i.e. 0011 – 1's com.

Then add 1

0100 – 2's com

1011

0100 – 1's

1

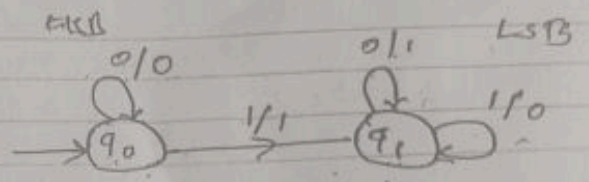
0101 – 2's complement

11101100

00010011 → 1's

1

00010100



initialy state q_0 in this state i will be seen 0 and i am leavin as 0. whenever is see a 1 i remember that i seen 1 & print 1 and go to next state.
and this state (q_1) will actually complement every thing.

eg. 1 1 0 0 (M, L)

0 0 1 1 — 1's compl

1

0 1 0 0 — 2's complement

Read from LSB

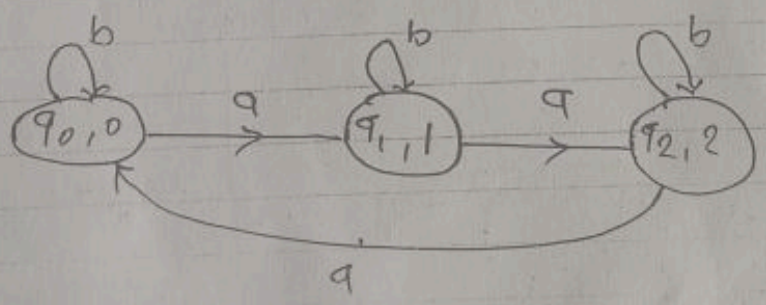
0 0 1 1

0 0 1 1

$q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1$

0 0 1 0

* ~~count no. of a's % 3~~



In moore m/c o/p is associated with state.

so move to the o/p on to the transisions conversion is in two ways

1) Transition diagram
2) Transion Table.