* Non deterministic Finite Automata (NFA) :-

The FA which has zero (0) (or) one (1) (or) more transitions for any input symbol from any state is called as NFA.

- A NFA can reside in multiple states at the same time.

NFA has 5 tuple in $M = \{Q, \Sigma, \delta, q_0, F\}$

where, Q – Set of all states

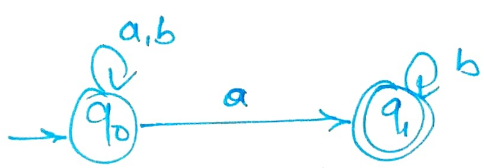
Σ = I/P alphabets

$\delta : Q \times \Sigma \rightarrow 2^Q$ is a transition fun^n

q_0 – initial state

F – Set of all final states

Ex:-



If automata is in q_0 state then for input 'a' the next will be q_0 (or) q_1

δ	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
q_1	ϕ	q_1

$$\delta :- Q \times \Sigma \rightarrow 2^Q = 2^2 = 4$$

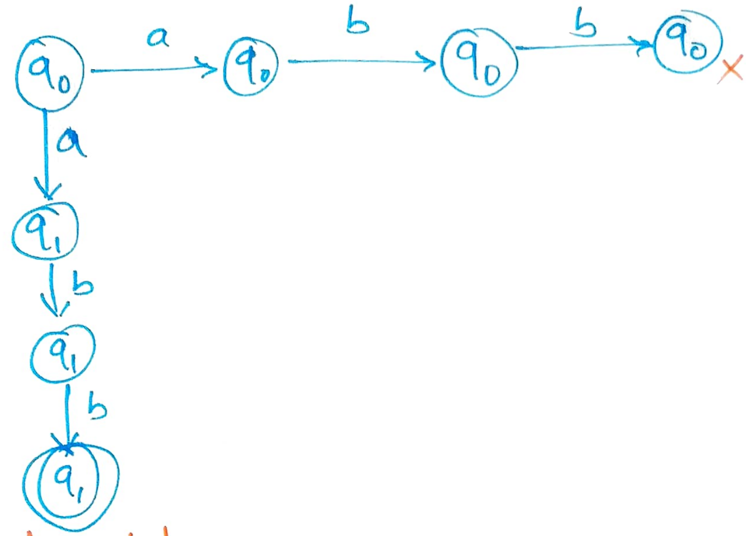
$$2^Q = \{\{q_0, q_1\}, \{q_0\}, \{q_1\}, \phi\}$$

$Q = \{q_0, q_1\}$

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

$F = \{q_1\}$

Ex: $W = abb$



Accepted

Note :-

1) NFA is rough idea to represent regular language

2) Accepting power of NFA = Accepting power of DFA.

3) RL $\Rightarrow$ L(NFA) = L(DFA)

4) Construction of NFA is easier than DFA for any regular language.

5) NFA takes care of only valid inputs & no need to take care of invalid strings.

6) No concept of dead states in NFA

Imp 7) Every DFA is NFA but every NFA can be converted into DFA i.e. there exist equivalence between NFA & DFA.

8) NFA can take move to any no. of states, after taking the input symbols from Σ.

9) In NFA no need to define the transition for each and every input symbol for each and every state.

10) NFA takes more time to recognize a string because of non-determinism.

• Acceptance by NFA :-

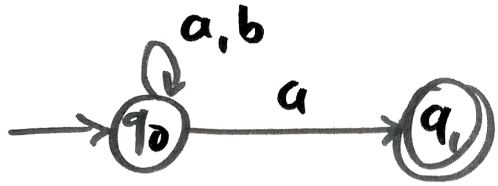
Let x is any string from Σ^* corresponding to x, multiple transition path can exist. If atleast one transition path ends in the final state then the string x is accepted by NFA.

• ⇒ The set of all strings which are accepted by NFA is called as language of NFA.

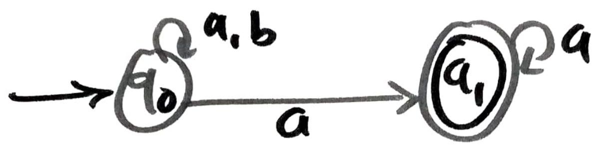
$$\therefore L(NFA) = \{x \in \Sigma^* \mid \delta(q_0, x) = \text{Final state}\} = L(DFA)$$

Note :- NFA is not unique for any regular language.

Ex:-



$\Rightarrow L_1 = Xa$



$\Rightarrow L_2 = Xa$

$L_1 = L_2$

x

① Construct the NFA that accept all of the strings 0's & 1's where every string -

ⓐ Starts with 0

ⓑ Starts with 0

ⓒ ends with 10

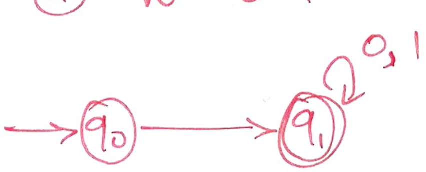
ⓓ ends with 0

ⓔ ends with 01

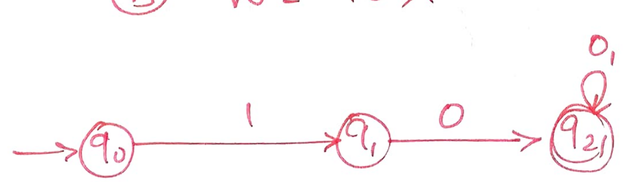
ⓕ contain the substring 101

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

ⓐ $W = 0X$



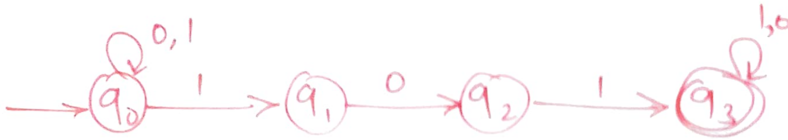
ⓑ $W = 10X$



ⓒ $W = X0$

ⓓ $W = \cancel{10}X01$

(d) $W = x\,101\,x$



Assignment Questions:

① Construct the NFA that accept all the strings of 0's & 1's where every string –

(a) Starts & ends with '0'.
(b) Starts & ends with same symbol.
(c) Starts & ends with different symbol.

② Construct NFA that accept all the strings of 0's & 1's where
(a) 4th symbol from left end is zero.
(b) 5th symbol from right end is one.

③ Construct the NFA that accept all the strings of 0's & 1's where every string
(a) ends with 10 (or) 01
(b) Starts with 10 (or) 01
(c) Contain the substring 10 (or) 01

④ Construct NFA $\Sigma = \{0, 1\}$ where ~~every~~ the length of string is -

ⓐ Exactly 2 ⓑ atmost 2 ⓒ atleast 2

ⓓ Divisible by 3 [0 (mod 3)]

Note :- $\Sigma = \{0, 1\}$

No.	Language	DFA	NFA
1.	$W = Sx$, $\|S\| = n$	$n+2$	$n+1$
	$\|w\| = xS$, $\|S) = n$	$n+1$	$n+1$
	$W = xSx$, $\|S\| = n$	$n+1$	$n+1$
2.	Starts & ends with same symbol.	5	4
	Starts & ends with diff^n symbol	5	4
3.	n^{th} symbol from left	$n+2$	$n+1$
	n^{th} symbol from right	2^n	$n+1$
4.	$\|w\| = n$	$n+2$	$n+1$
	$\|w\| \leq n$	$n+2$	$n+1$
	$\|w\| \geq n$	$n+1$	$n+1$
	$\|w\| = r \pmod{n}$	n	n

* Conversion of NFA to DFA :-

The process of NFA to DFA Conversion is called as subset Conversion.

Algorithm : Let $M = (Q, \Sigma, \delta, q_0, F)$ - NFA

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

① Initial State :

$$q_0' = q_0$$

No Change in initial State

② Construction of δ' :

$$\delta'(q_0, x) = \delta(q_i, x)$$

$$\delta'(q_0, q_1, \cdots q_n, x) = \bigcup_{i=0}^{n} \delta(q_i, x)$$

Start the construction of δ' with the initial state and continue for every new state & stop the construction whenever no new state occurs.

③ Final State :-

Every subset which Contains the final state of NFA is a final State in DFA.

Note: The DFA which is obtained from NFA may or may not be minimal.

Ex:-

① NFA :

δ	a	b
$\rightarrow q_0$	$\{q_0, q_2\}$	$\{q_1\}$
q_1	$\{q_2\}$	$\{q_0, q_1\}$
q_2	$\{q_0\}$	ϕ

$\rightarrow$ $\delta(q_0, a) = \{q_0, q_2\}$ // New state

$$\delta'\{(q_0, q_2), a\} = \delta'(q_0, a) \cup \delta'(q_2, a)$$
$$= \{q_0, q_2\} \cup \{q_0\}$$
$$= \{q_0, q_2\}$$

$$\delta'\{(q_0, q_2), b\} = \delta'(q_0, b) \cup \delta(q_2, b)$$
$$= \{q_1\} \cup \phi$$
$$= \{q_1\}$$ // New state

$\delta'(q_1, a) = \{q_2\}$ //

$\delta'(q_1, b) = \{q_0, q_1\}$ // New

$$\delta'\{(q_0, q_1), a\} = \delta'(q_0, a) \cup \delta'(q_1, a)$$
$$= \{q_0, q_2\} \cup \{q_2\}$$
$$= \{q_0, q_2\}$$

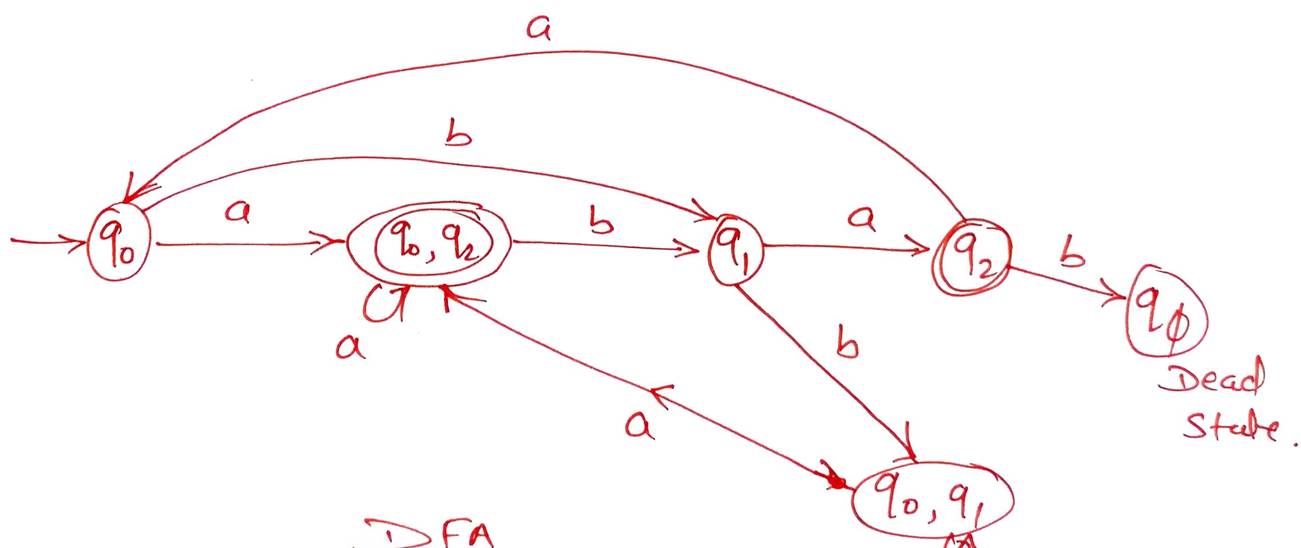
$$\delta'\{(q_0, q_1), b\} = \delta'(q_0, b) \cup \delta'(q_1, b)$$
$$= q_1 \cup \{q_0, q_1\}$$
$$= \{q_0, q_1\}$$

$\delta'(q_2, a) = q_0$

$\delta'(q_2, b) = \phi$ (DS)

DFA

δ'	a	b
$\rightarrow q_0$	$\{q_0, q_2\}$	q_1
$(q_0, q_2)^*$	$\{q_0, q_2\}$	q_1
q_1	q_2	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\{q_0, q_1\}$
q_2^*	$\{q_0\}$	ϕ (DS)
ϕ	ϕ	ϕ



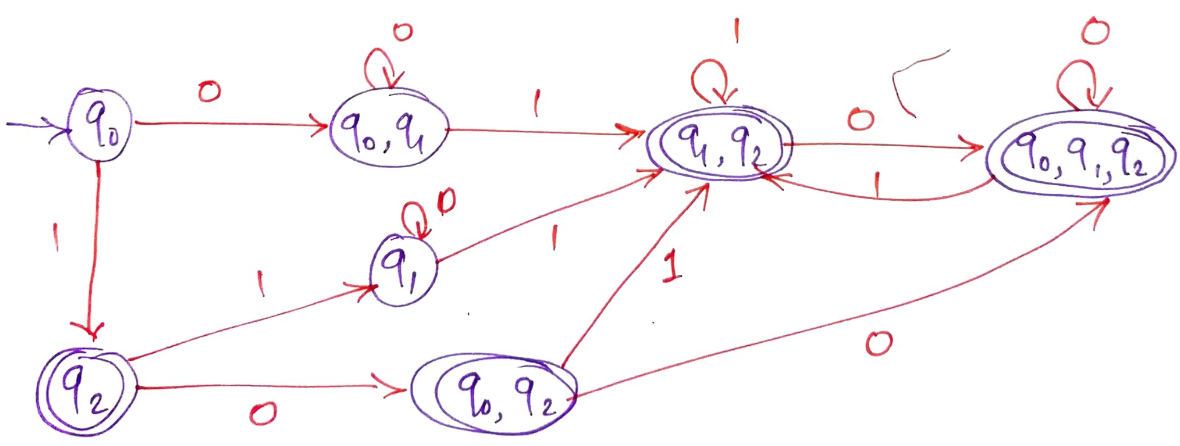
DFA

② NFA :

δ	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	q_2
q_1	q_1	$\{q_1, q_2\}$
q_2	$\{q_0, q_2\}$	q_1

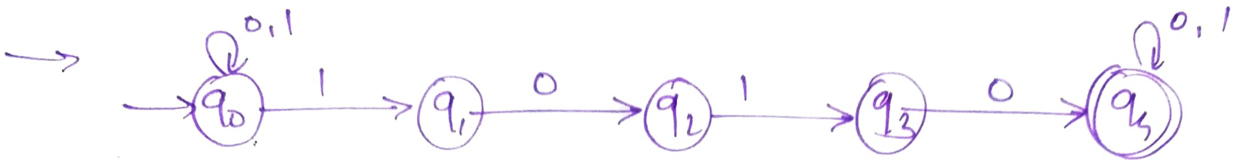
→ DFA

δ'	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	q_2
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_1, q_2\}$
$\{q_1, q_2\}^*$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$
$\{q_0, q_1, q_2\}^*$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$
q_2^*	$\{q_0, q_2\}$	q_1
$\{q_0, q_2\}^*$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$
q_1	$\{q_1\}$	$\{q_1, q_2\}$



DFA.

③ NFA $\Sigma = \{0, 1\}$ Every string contain substring 1010.



δ	0	1
$\rightarrow q_0$	q_0	$\{q_0, q_1\}$
q_1	q_2	ϕ
q_2	ϕ	q_3
q_3	q_4	ϕ
q_4^*	q_4	q_4

DFA

δ'	0	1
$\rightarrow q_0$	q_0	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\{q_0, q_1\}$
$\{q_0, q_2\}$	q_0	$\{q_0, q_1, q_3\}$
$\{q_0, q_1, q_3\}$	$\{q_0, q_2, q_4\}$	$\{q_0, q_1\}$
$\{q_0, q_2, q_4\}^*$	$\{q_0, q_4\}$	$\{q_0, q_1, q_3, q_4\}$
$\{q_0, q_4\}^*$	$\{q_0, q_4\}$	$\{q_0, q_1, q_4\}$
$\{q_0, q_1, q_3, q_4\}^*$	$\{q_0, q_2, q_4\}$	$\{q_0, q_1, q_4\}$
$\{q_0, q_1, q_4\}^*$	$\{q_0, q_2, q_4\}$	$\{q_0, q_1, q_4\}$

4) NFA :

δ	0	1
→ P	P, Q	R
Q	R	R
R	S	Q
S*	S	S

Find DFA

* ε-NFA (or) NFA with ε-Moves:-

The NFA which has a transition even for empty string 'ε' is called as a ε-NFA (or) NFA with ε-Moves.

- Machine can make transition without any input.

ε-NFA is a 5 tuple Machine

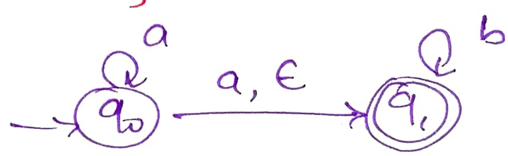
$$M = \{Q, \Sigma, \delta, q_0, F\}$$

$\delta :- Q \times \Sigma\{\epsilon\} \Rightarrow 2^Q$ is a transition fun^n.

* when we find a string through a path, ε is there then ε symbols are discarded.

ex:- ① $L = \{a^m b^n \mid m, n \geq 0\}$

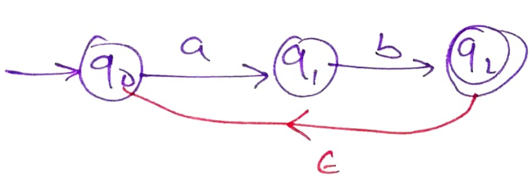
→ $a^m \epsilon b^n$



② $L = \{(ab)^n \mid n \geq 1\}$

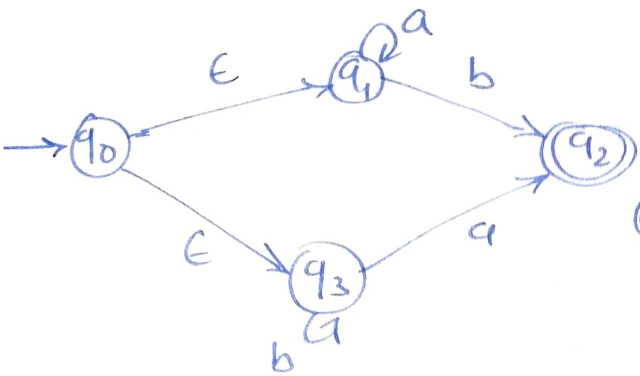
$L = \{ab, abab, \cdots\}$

abεab

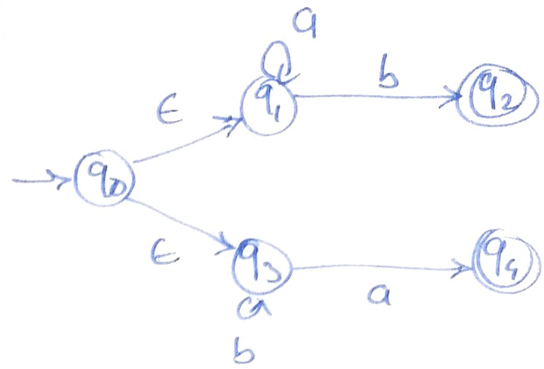


③ $L = \{a^n b \cup b^m a \mid m, n \geq 0\}$

→



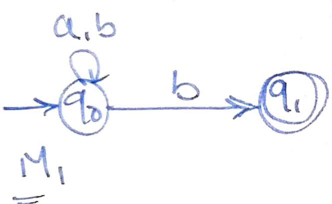
((OR))

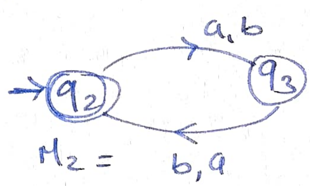


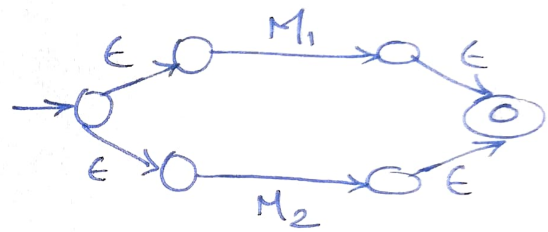
④ L = Set of all strings of a's & b's where every string ending b (or) the length is even.

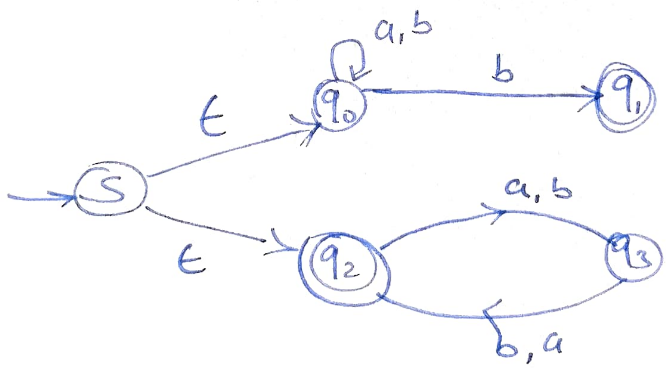
→ $\Sigma = \{a, b\}$

X b $|w| = 0 \pmod 2$



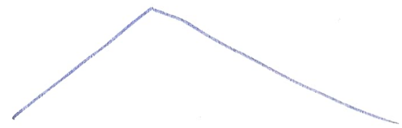




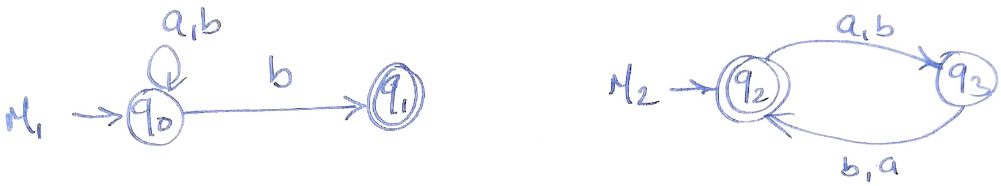


⑤ L = set of all strings of a's & b's where the length of string is even and ends with b

→ $\Sigma = \{a, b\}$



Xb $\qquad$ $|w| = 0 \pmod 2$



Note:

① $E(\epsilon\text{-NFA}) = E(NFA) = E(DFA)$

② Representing a RL by ε-NFA is easier than NFA & DFA

③ Inclusion & exclusion of ε-transitions will not affect the language of NFA.