

## C - 2

■ finite state machine:

symbol - a, b, c, 0, 1, 2 ...

Alphabet -  $\Sigma$

collection of symbols :  $\{a, b\}$ ,  $\{0, 1, 2\}$  ...

string - sequence of symbols : a, b, aa, 01, 0...

Language - set of strings :  $\Sigma = \{0, 1\}$

Ex:  $L_1$  = set of all strings  
of length 3

$$= \{000, 001, 010, \dots, 111\}$$

↓  
Finite

visitaturness to laban, that begin with 0

$L_2$  = set of all strings that begin with 0  
 $= \{0, 00, 01, \dots\} \rightarrow$  infinite

powers of  $\Sigma$  :: Suppose,  $\Sigma = \{0, 1\}$

$\Sigma^0$  = set of all strings of length 0 =  $\{\epsilon\}$

$\Sigma^2$  = ... length 2 =  $\{00, 01, 10, 11\}$

$\Sigma^n$  = ... length n

Cardinality: number of elements in a set

$$\Sigma^n = 2^n$$

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \dots$$

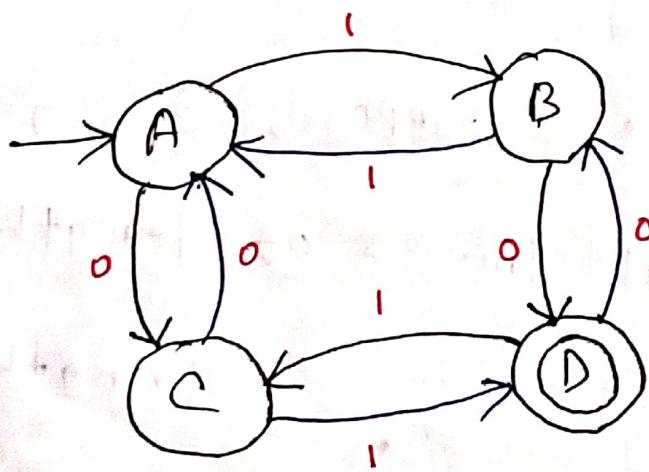
$$= \{\Sigma\} \cup \{0, 1\} \cup \{ \dots \}$$

= set of all possible strings of all lengths  
over  $\{0, 1\}$   $\rightarrow$  infinite

## Deterministic finite Automata (DFA)

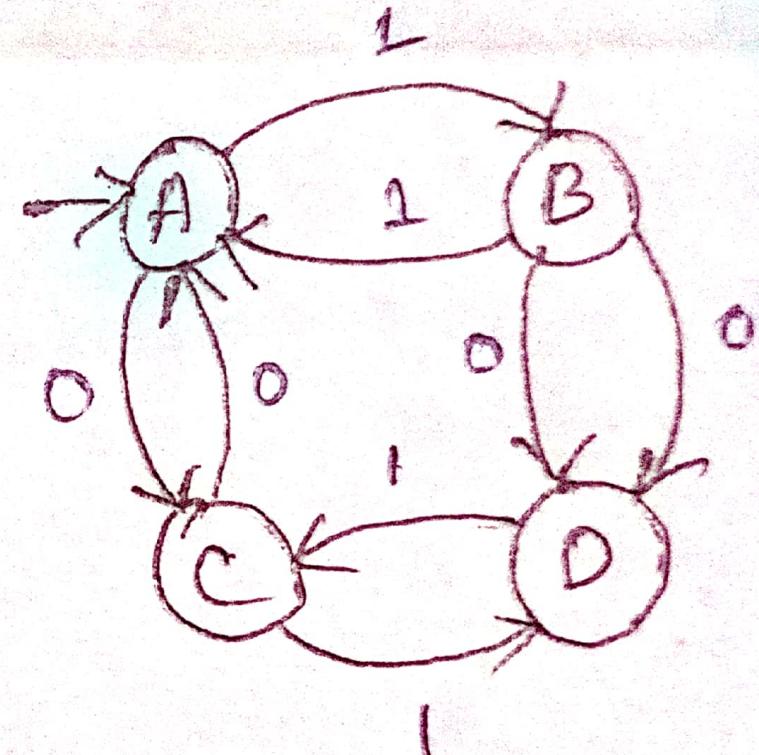
- simplest model of computation
- has very limited memory

Ex:



## DFA:

- given current state
- we know what the next state is
- no choice/randomness
- simple/easy to design
- only one unique next state



$Q$  - set of all states e.g.:  $\{A, B, C, D\}$

$\Sigma$  - inputs e.g.:  $\{0, 1\}$

$q_0$  - start state e.g.: A

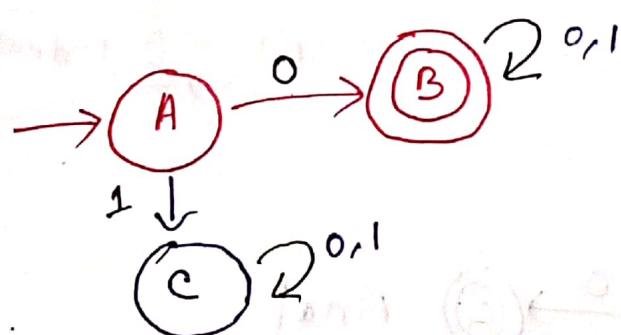
$F$  - set of final state e.g.:  $\{D\}$

$\delta$  - transition function from  $Q \times \Sigma \rightarrow Q$

e.g.:

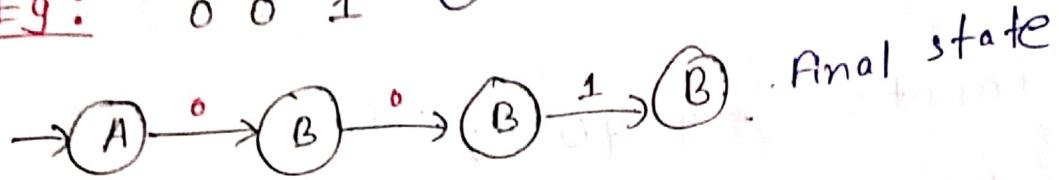
	0	1
A	C	B
B	D	A
C	A	D
D	B	C

Example:  $L_1$  = set of all strings that start with '0'  
with  $\{0, 1\}$

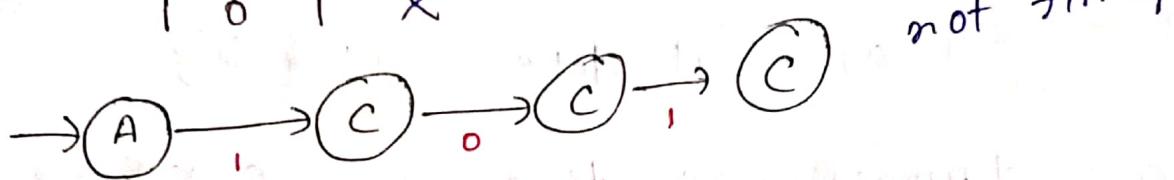


dead state/trap state

Eg: 0 0 1 ✓



1 0 1 X

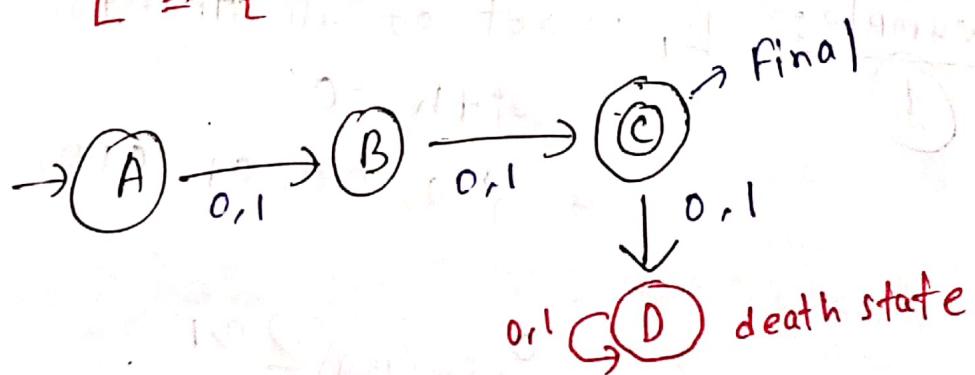


Example: construct a DFA that accepts sets  
② of all strings over  $\{0, 1\}$  of length 2.

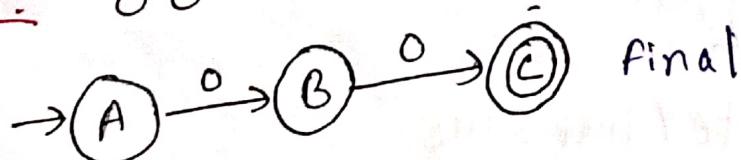
$$\Sigma = \{0, 1\}$$

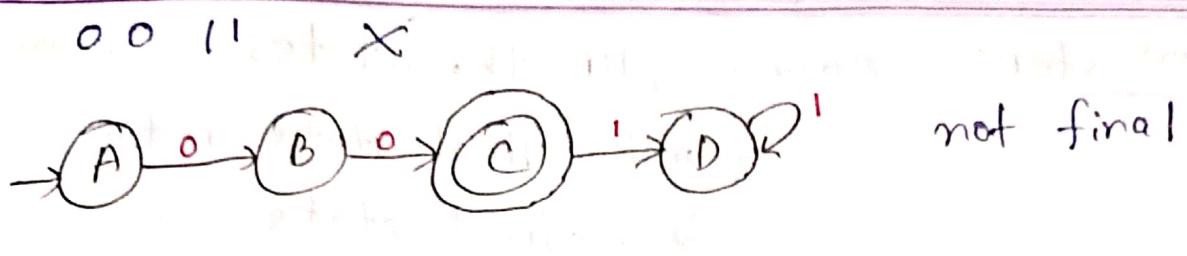
$$L = \{00, 01, 10, 11\}$$

Ans:



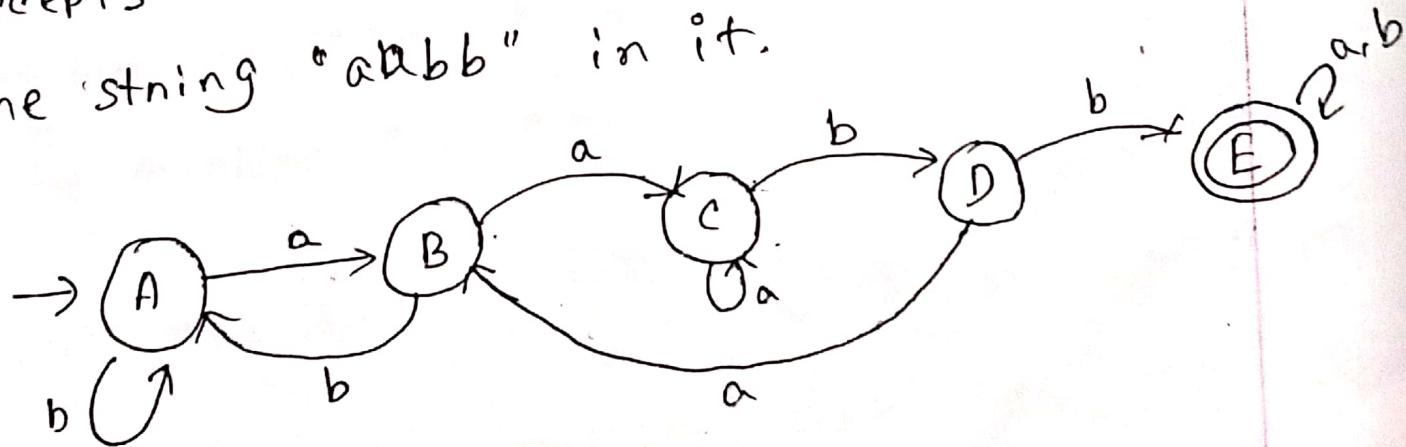
Eg: 0 0 ✓





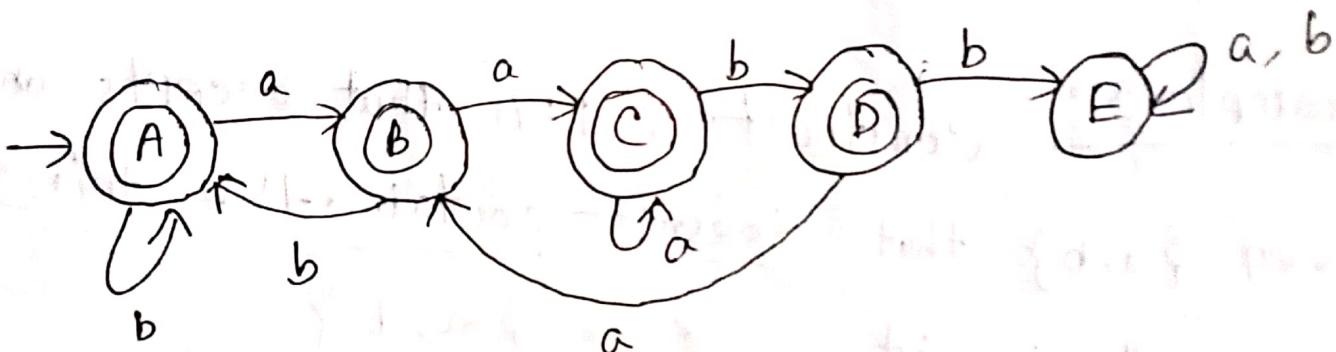
Example 3: Construct a DFA that accepts any strings over  $\{a, b\}$  that doesn't contain the string "aabb" in it.  $\Sigma = \{a, b\}$

Solution: At first, we'll construct a DFA that accepts all strings over  $\{a, b\}$  that contains the string "aabb" in it.

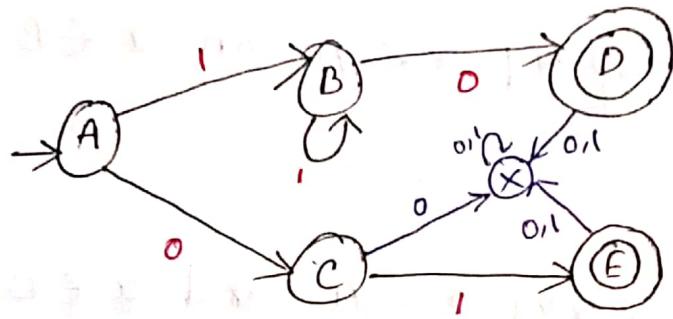


2<sup>nd</sup> step: now - flip the states  
— make final state into non-final state  
— make non-final states into final states

So, the answer is :



By figuring out what a DFA recognizes?



accepts : 1) 0 1

2)  $\frac{111\cdots 1}{A \downarrow B \downarrow D} 0$

$L = \{ \text{accepts the string } 01 \text{ on a string of at least one '1' followed by a '0'} \}$

### \* Regular Language:

A language is said to be a regular language if and only if some finite state machine recognizes it.

not regular :  
- not recognized by any FSM  
- requires memory

\* operations on regular languages:

Union:  $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$

concatenation:

$A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$

Star:  $A^* = \{x_1 x_2 \dots x_k \mid k \geq 0 \text{ and } x_i \in A\}$

Eg:  $A = \{pq, r\}$ ,  $B = \{t, uv\}$

$A \cup B = \{pq, r, t, uv\}$

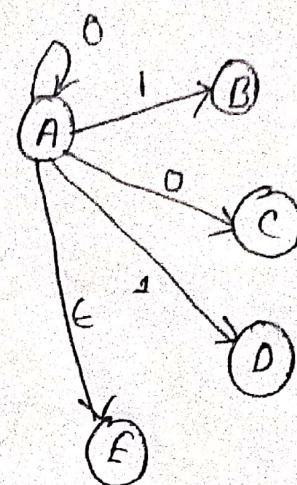
$A \circ B = \{pqt, pquv, rt, ruv\}$

$A^* = \{\epsilon, pqr, r, pqr, ppqr, rr, pqrpq, rrr, \dots\}$

N.B: The class of regular languages is closed under union and concatenation.

## Non-deterministic finite Automata (NFA)

- given the current state
- multiple next states
- next state may be random
- all next states may be chosen in parallel



\* Here,  
 $\epsilon$  - empty string  
 (not found in DFA)

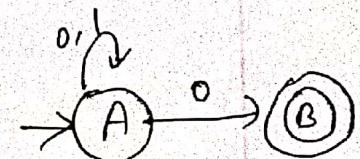
$Q$  - set of all states -  $\{A, B\}$

$\Sigma$  - inputs -  $\{0, 1\}$

$q_0$  - initial state - A

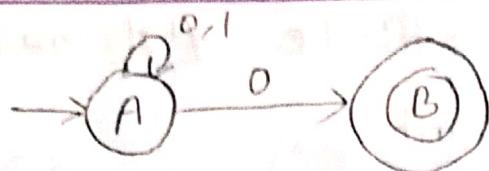
$f$  - set of final states -  $\{B\}$

$\delta$  -  $Q \times \Sigma \rightarrow 2^A$



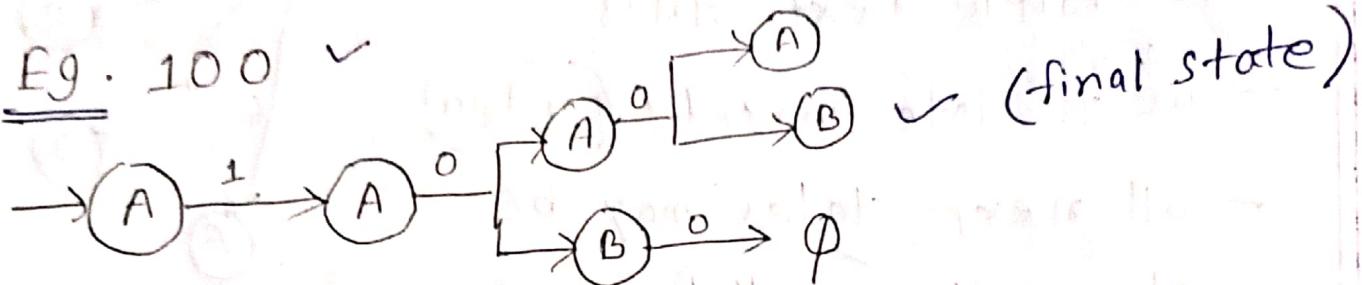
Explanation:  $A \times 0 \rightarrow A$ ,  $A \times 1 \rightarrow B$   
 $B \times 0, 1 \rightarrow \emptyset$

Ex 1:

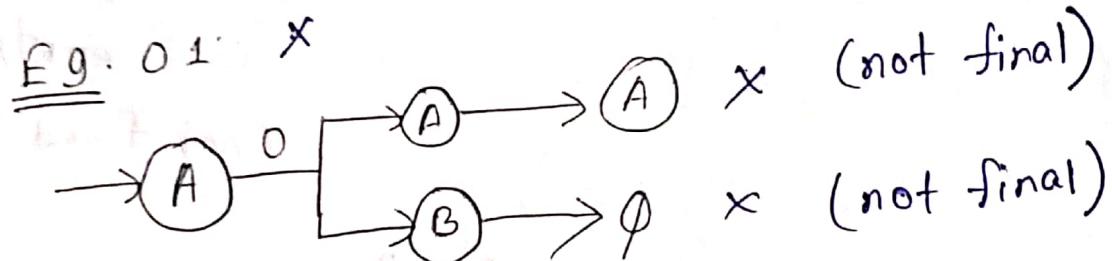


$L = \{ \text{set of all strings that end with } 0 \}$

Eg. 100 ✓



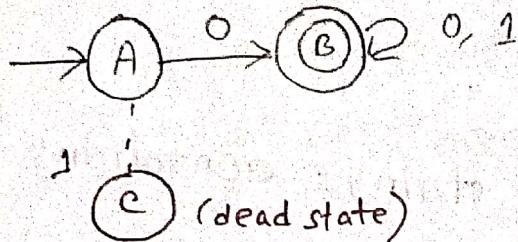
Eg. 01 ✗



N.B: If there is any way to run the machine that ends in any set of states

out of which at least one state is a final state, then the NFA accepts the string.

Ex 2:  $L = \{ \text{set of all strings that start with } 0 \}$   
 $= \{ 0, 00, 01, 010, 0000, \dots \}$

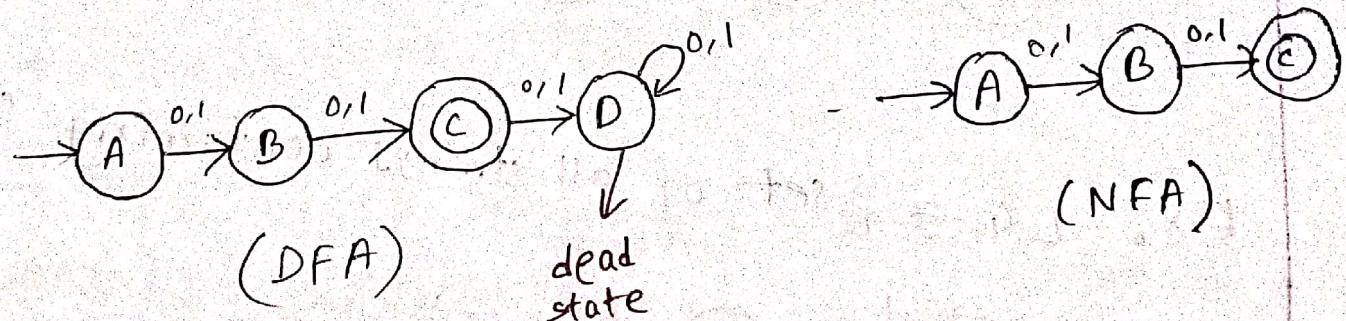


\* In case of NFA,  
dead state doesn't  
need to be mentioned.

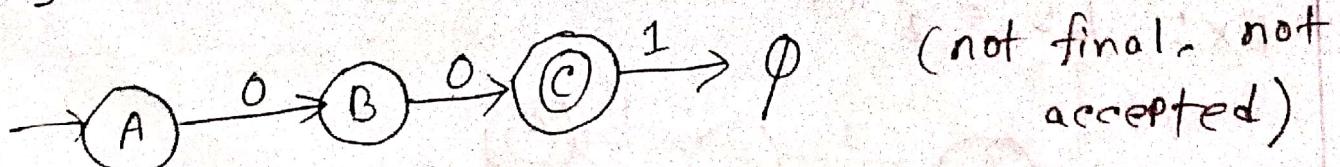
Ex 3: Construct a NFA that accepts sets of all strings over  $\{0, 1\}$  of length 2.

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

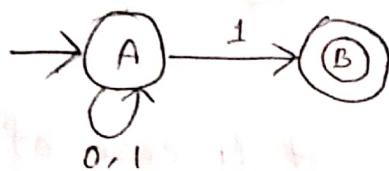
$$L = \{00, 01, 10, 11\}$$



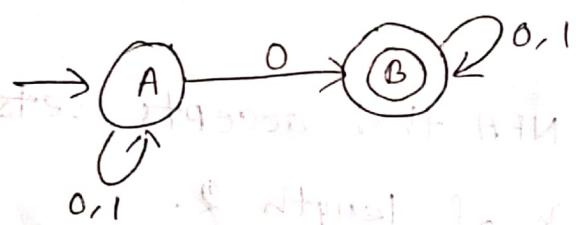
Eg. 001



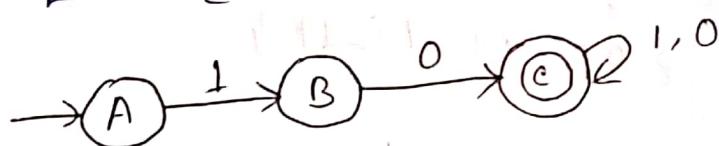
Ex:  $L_1 = \{ \text{set of all strings ending with '1'} \}$



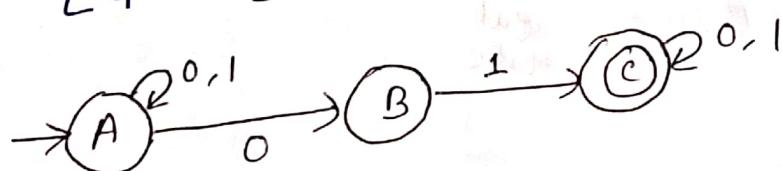
Ex:  $L_2 = \{ \text{set of all strings containing '0'} \}$



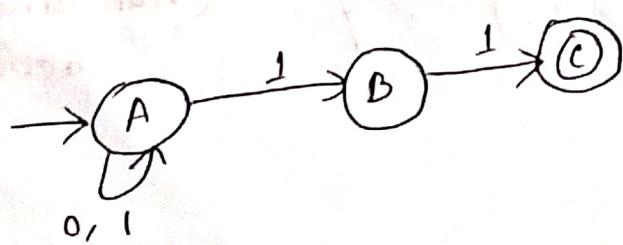
Ex:  $L_3 = \{ \text{set of all strings starting with '10'} \}$



Ex:  $L_4 = \{ \text{set of all strings containing '01'} \}$



Ex:  $L_5 = \{ \text{set of all strings ending with '11'} \}$



## Conversion of NFA to DFA

N.B.: Every DFA is an NFA, but not every NFA is a DFA. But there is an equivalent DFA for every NFA.

Explanation: DFA :  $\delta = Q \times \Sigma \rightarrow Q$

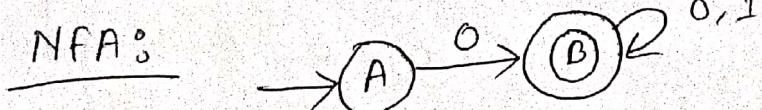
NFA :  $\delta = Q \times \Sigma \rightarrow 2^Q$



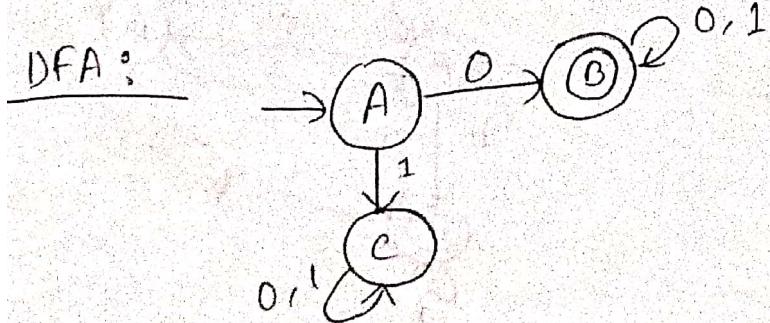
$Q$  also included  
in  $2^Q$

So, DFA is included in NFA

\*\*  $L = \{ \text{set of strings over } \{0,1\} \text{ starting with } 0 \}$   
 $\Sigma = \{0, 1\}$



	0	1
B	B	$\emptyset$
B	B	B



	0	1
B	B	C
B	B	B
C	C	C

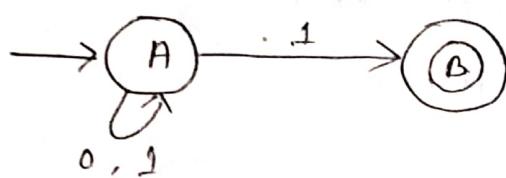
↓ dead state

(subset construction method)

Ex : 1

$L = \{ \text{set of strings ending with 1} \}$

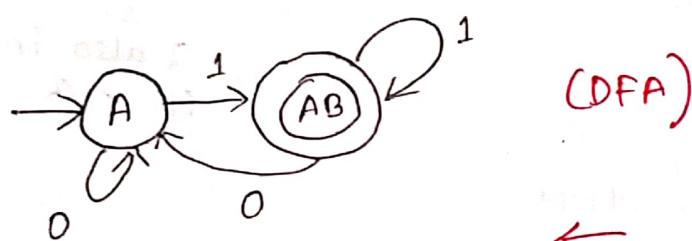
$$\Sigma = \{0, 1\}$$



(NFA)



	0	1
A	{A}	{A, B}
B	$\emptyset$	$\emptyset$



(DFA)

	0	1
A	{A}	{AB}
AB	{A}	{AB}

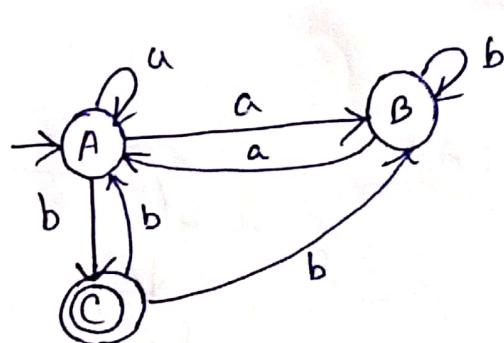
AB - single state

Ex 2: Find the equivalent DFA for the NFA

given by  $M = [\{A, B, C\}, \{a, b\}, \delta, A, \{C\}]$

where  $\delta$  is given by :

	a	b
A	A, B	C
B	A	B
C	$\emptyset$	A, B



DFA

	a	b
$\rightarrow A$	AB	C
AB	AB	BC
$\text{BC}$	A	AB
$\text{C}$	D	AB
D	D	D

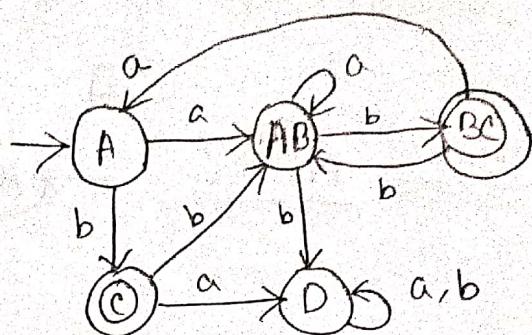
Final

\* NFA  $\leftrightarrow$  C final state

From: DFA  $\leftrightarrow$  C final state

Because C state, dfa has

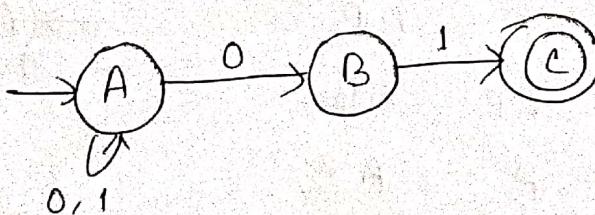
final state  $\leftrightarrow$  C.



Ex 3: Given below is the NFA for a language.

$L = \{\text{set of all strings over } \{0,1\} \text{ that ends with } '01\}$  Construct its eq. DFA.

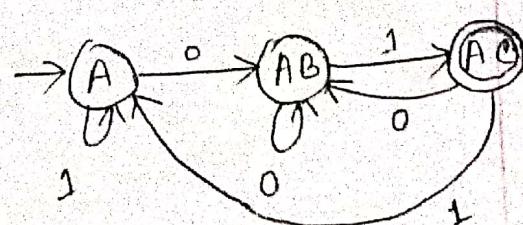
NFA:



	0	1
$\rightarrow A$	A, B	A
B	$\emptyset$	C
C	$\emptyset$	$\emptyset$

DFA:

	0	1
$\rightarrow A$	AB	A
AB	AB	AC
AC	AB	A

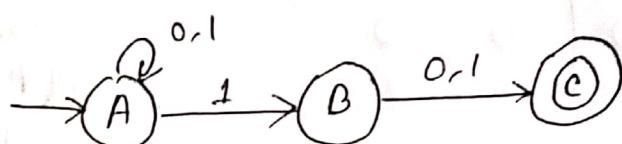


- \* In NFA  $\rightarrow$  2 states are possible
- \* In DFA  $\rightarrow$  (More than 1 state) 2 states are not possible

Ex 4: Design an NFA for a language that accepts all strings over  $\{0,1\}$  in which the second last symbol is always '1'. Then convert it to its eq. DFA.

Ans:

NFA:



(state diagram)

	0	1
A	A	A, B
B	C	C
C	$\emptyset$	$\emptyset$

(transition table)

DFA:

	0	1
A	A	AB
AB	AC	ABC
AC	A	AB
ABC	AC	ABC

