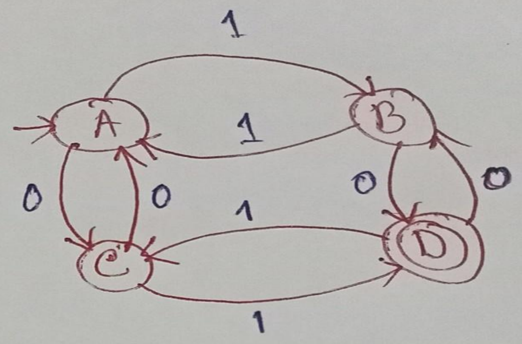
Non-Deterministic FA

Deterministic Finite Automata
↓
DETERMINISM



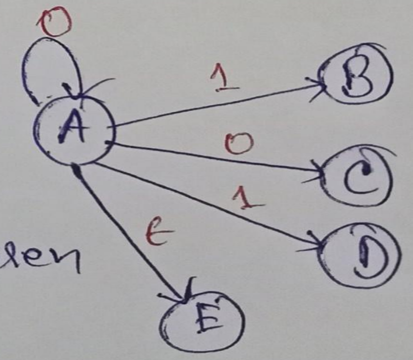
>> In DFA, given the current state we know what the next state will be.

>> It has only one unique next state.

>> It has no choice or randomness.

>> It is simple and easy to design.

Non-deterministic Finite Automata
↓
NON-DETERMINISM

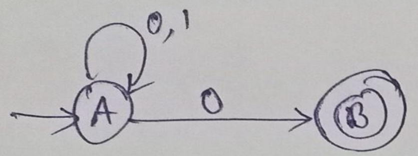


>> In NFA, given the current state there could be multiple next states.

>> The next state may be chosen at random

>> All the next states may be chosen in parallel.

NFA - Formal Definition



L = { Set of all strings that end with 0 }

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

Q = Set of all states — $\{A, B\}$

Σ = inputs — $\{0, 1\}$

q_0 = Start state/initial state — A

F = Set of final states — B

— [illegible]

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

↓

$\left.\begin{array}{l} A \times 0 \rightarrow A \checkmark \\ A \times 0 \rightarrow B \checkmark \end{array}\right]$ · more than one state

$A \times 1 \rightarrow A$

$B \times 0 \rightarrow \emptyset$

$B \times 1 \rightarrow \emptyset$

BS.

$\Rightarrow A \xrightarrow{1} A, B, AB, \emptyset$

$2^2 = 4$

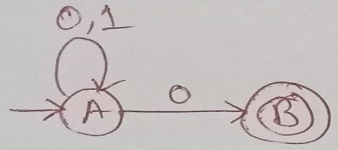
~~3 states~~

$\Rightarrow$ 3 States — A, B, C

$A \xrightarrow{1} A, B, C, AB, AC, BC, ABC, \emptyset$

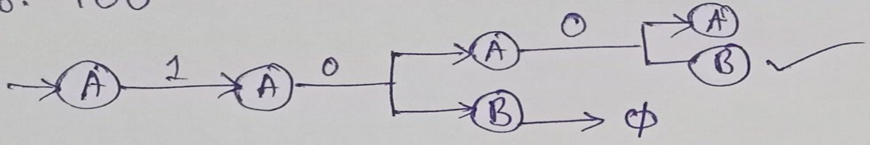
$2^3 = 8$

NFA - Example-1

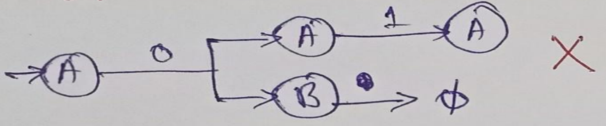


L = { Set of all strings that end with 0 }

Eg. 100



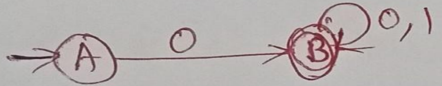
Eg. 01



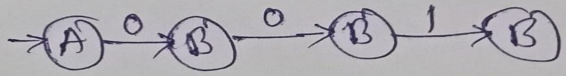
⇒ If there is any way to run the machine that ends in any set of states out of which atleast one state is a final state, then the NFA accepts

Example 2

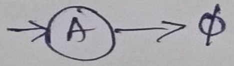
L = { Set of all strings that start with 0 }
= { 0, 00, 01, 000, - - - }



Eg, 001 ✓



Eg, 101 ✗



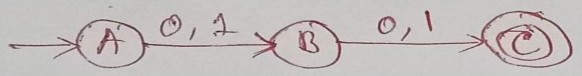
Dead Configuration

NFA – Example – 3

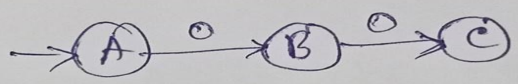
» Construct a NFA that accepts sets of all strings over $\{0,1\}$ of length 2

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

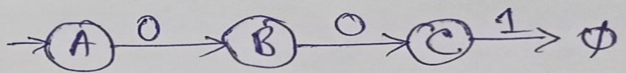
$L = \{00, 01, 10, 11\}$



Eg, 00 ✓

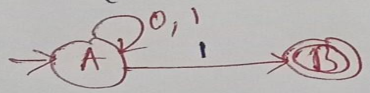


Eg, 001 ✗



Example 4

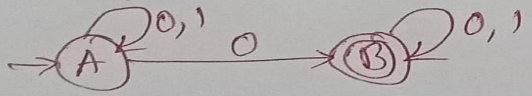
Ex a) $L1 = \{$ Set of all strings that ends with '1' $\}$



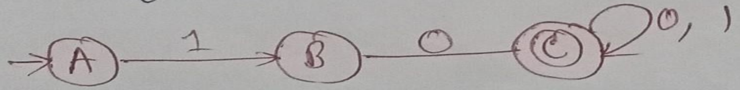
→ 01, 001, 0001 → 0^*1 / 1

→ 101, 1101, ~

Ex b) $L2 = \{$ Set of all strings that contain '0' $\}$

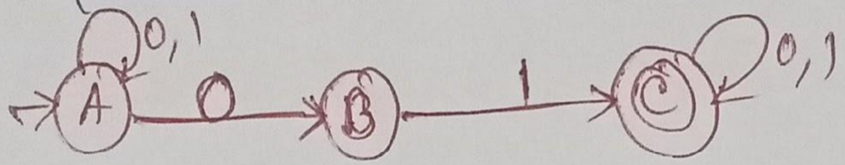


Ex c) $L3 = \{$ Set of all strings that starts wit '10' $\}$

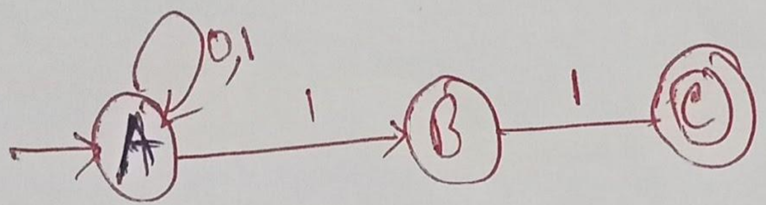


NFA – Example 4

Ex 4) L4 = { Set of all strings that contain '01' }



Ex 5) L5 = { Set of all strings that ends with '11' }



Assignment: If you were to construct the equivalent DFAs for the above NFAs, then tell that how many minimum number of states would you use for the construction of each of the DFA's

Conversion of NFA to DFA

Every DFA is an NFA, but not vice versa

But there is an equivalent DFA for every NFA

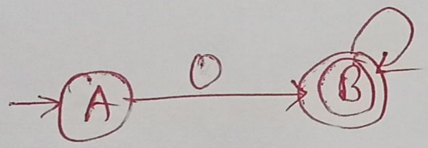
DFA $\delta = Q \times \Sigma \to Q$

NFA $\delta = Q \times \Sigma \to 2^{Q}$

NFA $\cong$ DFA

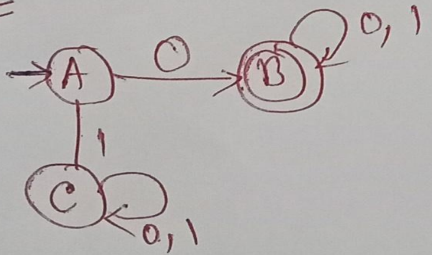
L = { Set of all strings over (0, 1) that starts with '0' }

NFA



	0	1
A	B	ϕ
B	B	B

DFA



	0	1
A	B	C
B	B	B
C	C	C

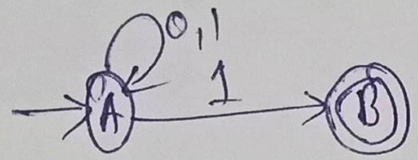
C - Dead state / Trap state

Conversion of NFA to DFA - Examples

L = {Set of all strings over (0, 1) that ends with '1'}

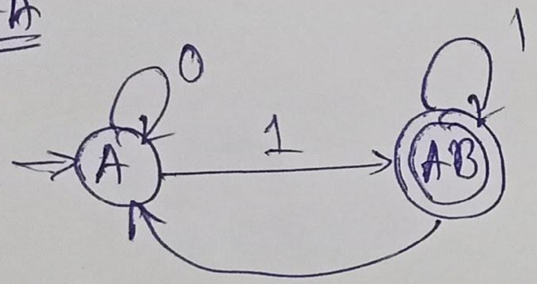
$\Sigma = \{0, 1\}$

NFA



	0	1
A	{A}	{A, B}
B	ϕ	ϕ

DFA



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

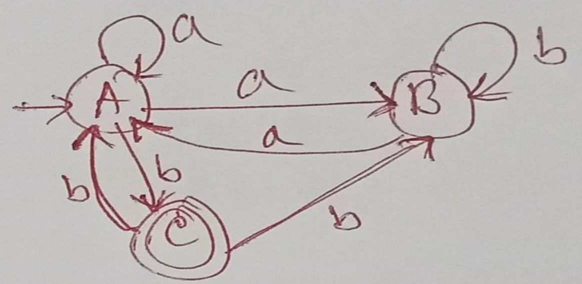
AB - Single State

Subset Construction method

Conversion of NFA to DFA – Examples

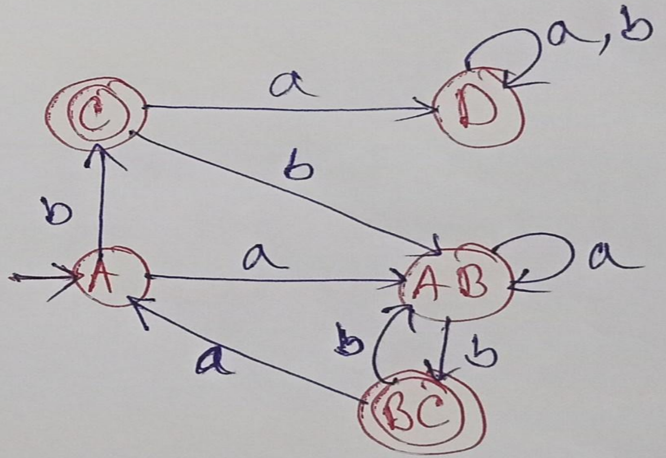
⇒ Find the equivalent DFA for the NFA given by
$M = [\{A, B, C\}, (a, b), \delta, A, \{C\}]$ where δ is given by:

	a	b
→A	A, B	C
B	A	B
C	–	A, B



DFA

	a	b
→A	AB	C
AB	AB	BC
BC	A	AB
C	D	AB
D	D	D

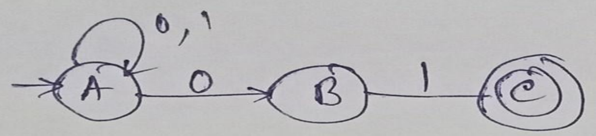


Assignment: Try to find out what does this NFA and its DFA accept.

Conversion of NFA to DFA - Examples

⇒ Given below is the NFA for a language
L = {Set of all strings over (0, 1) that ends with '01'}
Constructs its equivalent DFA

NFA



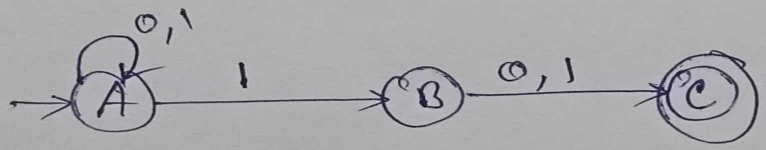
	0	1
→A	A, B	A
B	Φ	C
(C)	Φ	Φ

DFA ?

⇒ Design an NFA for a language that accepts all strings over {0, 1} in which the second last symbol is always '1'. Then converts it to its equivalent DFA

Eg. 0010
0110
110011
000111

NFA



	0	1
→A		
B		
(C)		

DFA ?