Theory of Computation

Symbol → a, b, c, 0, 1, 2, 3, ...

Alphabet → Collection of symbols denoted by Σ
Eg: Σ = {a, b}
Σ = {0, 1, 2}

String → Sequence of symbols
Eg: a, b, 0, 1, aa, ab, bb, ...

Language → Set of strings
Example: Σ = {0, 1}
L_1 = set of all strings of length 2
= {00, 01, 10, 11} → finite set

L_2 = set of all strings of length 3
= {000, 001, 010, 011, 100, 101, 110, 111} → finite set

L_3 = set of all strings that begin with 0.
= {0, 00, 01, 000, 001, 010, ...} → infinite set

Powers of sigma (Σ)
Σ = {0, 1}
Σ^0 = set of all strings of length 0.
Σ^0 = {ε} ε → epsilon symbol.

Σ^1 = set of all strings of length 1
Σ^1 = {0, 1}

Σ^2 = set of all strings of length 2
$\Sigma^2 = \{00, 01, 10, 11\}$

Σ^3 = set of all strings of length 3
$\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$
⋮

Σ^n = set of all strings of length n

Cardinality → No. of elements in a set
↳ $\Sigma^n = 2^n$ [only if $\Sigma = 0, 1$]

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots \Sigma^n$
$= \{\epsilon\} \cup \{0, 1\} \cup \{00, 01, 10, 11\} \cup \dots$

= set of all possible strings of all length over $\{0, 1\}$
↳ infinite set

C P P L
Date: / / 2022
Page No.

Finite State Machine
OR
Finite Automata

- FA with output
  - Moore Machine
  - Mealy Machine
- FA without output
  - DFA
  - NFA
  - ϵ-NFA

Finite State Machine
→ Simplest model of computation
→ Limited memory

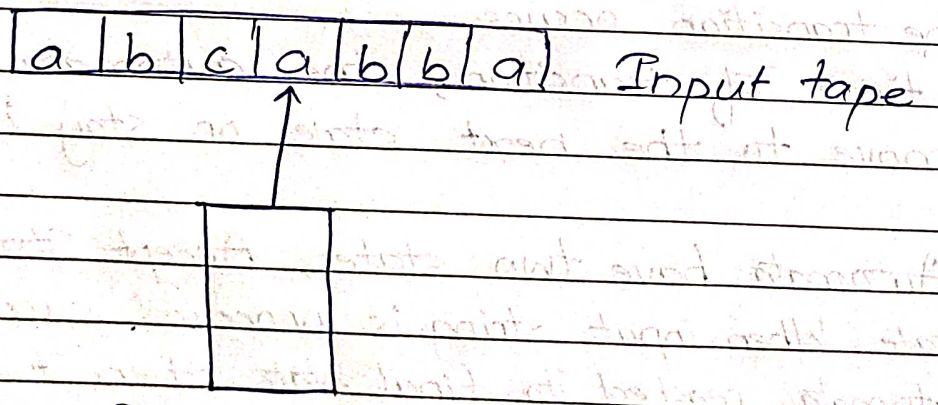
DFA → Deterministic Finite Automata

Finite Automata

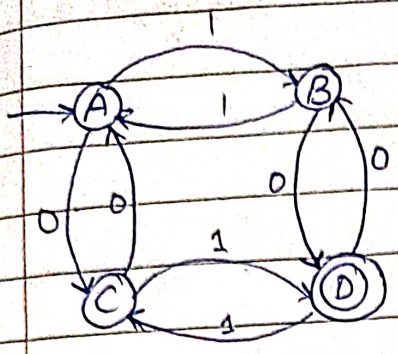
* FA are used to recognize patterns.
* It takes the string of symbol as input and change its states accordingly. When the desired symbol then the transition occurs.
* At the time of transition, the automata can either move to the next state or stay in same state.
* Finite Automata have two states, Accept State or Reject State. When input string is processed successfully and automata reached its final state then it will accept.

Formal definition of FA :-
A finite automata is a collection of 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
$Q \rightarrow$ finite set of states
$\Sigma \rightarrow$ set of input symbols
$q_0 \rightarrow$ initial state
$F \rightarrow$ final state
$\delta \rightarrow$ transition function

Finite Automata Model
FA can be represented by input tape and finite control
Input tape → Linear tape having some number of cells. Each input symbol is placed in a cell.
Finite control → Decides the next state on receiving particular input from input tape.
Reads one input symbol at a time.
Reads from left to right cells one by one.



Example:-



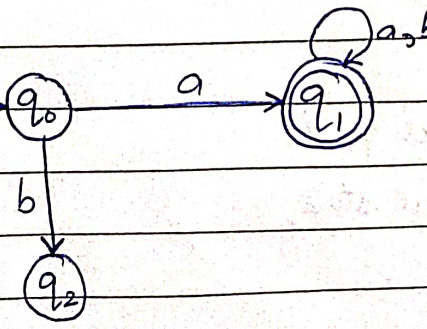
$Q = \{A, B, C, D\}$
$\Sigma = \{0, 1\}$
$q_0 = \{A\}$
$F = \{D\}$
δ :-

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

DFA → Deterministic Finite Automata

* FA is known as DFA if the machine reads an input string one symbol at a time.
* In DFA there is only one path for specific input from current state to next state.
* It doesnot accept null move.
* It can have multiple final state.

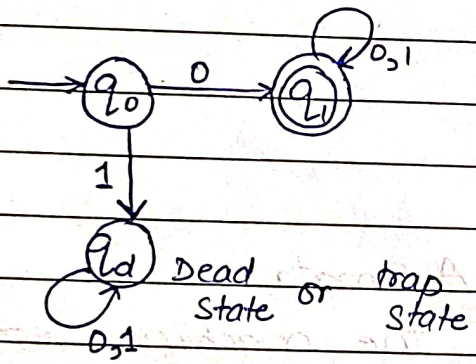
Example



Formal Definition:-
DFA is a collection of 5 tuples $(Q, \Sigma, q_0, F, \delta)$
where:
$Q \rightarrow$ set of states
$\Sigma \rightarrow$ set of inputs
$q_0 \rightarrow$ Initial state
$F \rightarrow$ final state
$\delta \rightarrow$ transition function from $Q \times \Sigma \rightarrow Q$

DFA Example 1

$L1$ = Set of all strings that start with 0
$= \{0, 00, 01, 010, 000, \dots\}$



001

$\delta(q_0, 001)$
↓
$\delta(q_1, 01)$
↓
$\delta(q_1, 1)$
↓
$\delta(q_1, \wedge)$ Accepted as q_1 is final state

classfellow

Date: / / 2022

Page No.

101

$\delta(q_0, 101)$

↓

$\delta(q_2, 01)$

↓

$\delta(q_2, 1)$

↓

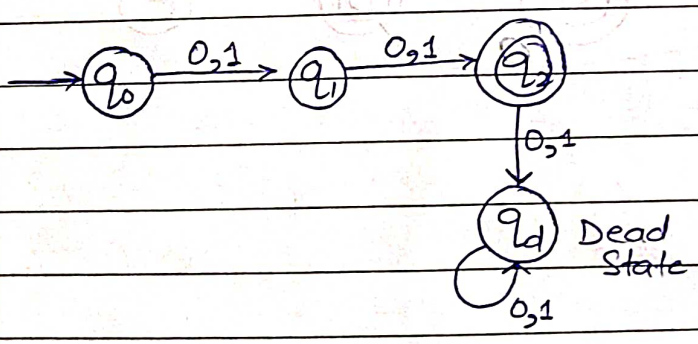
$\delta(q_2, \wedge)$ Rejected as q_2 is not final state

DFA Example 2

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

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

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



00

$\delta(q_0, 00)$

↓

$\delta(q_1, 0)$

↓

$\delta(q_2, \wedge)$ Accepted.

001

$\delta(q_0, 001)$

↓

$\delta(q_1, 01)$

↓

$\delta(q_2, 1)$

↓

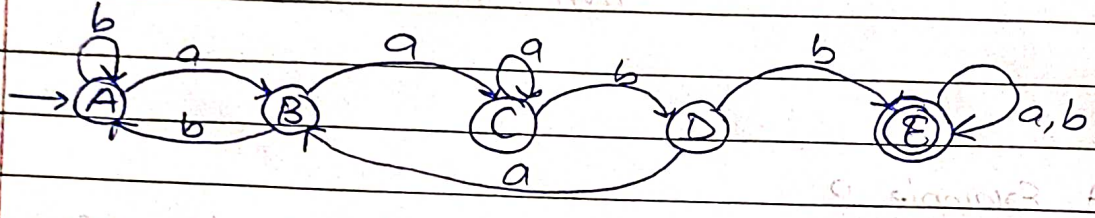
$\delta(q_d, \wedge)$ Rejected

DFA Example 3

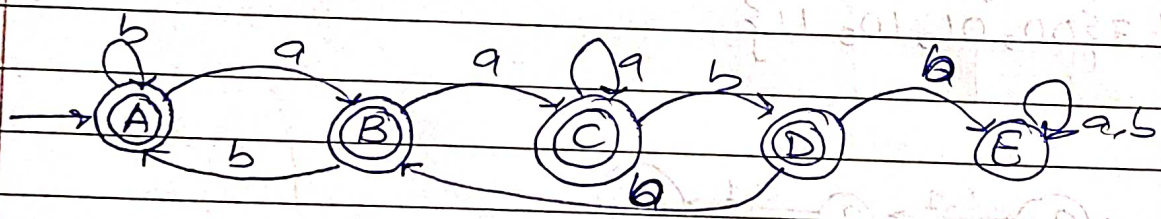
Construct a DFA that accepts any strings over $\{a,b\}$ that does not contain the string aabb in it.

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

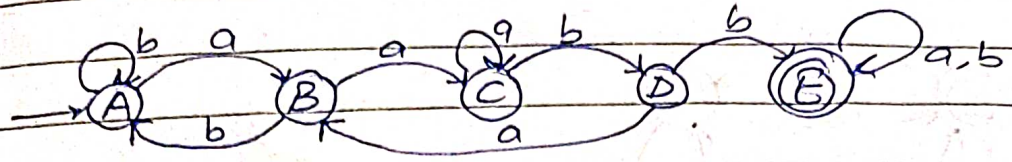
Lets try a simple problem
Construct a DFA that contains aabb



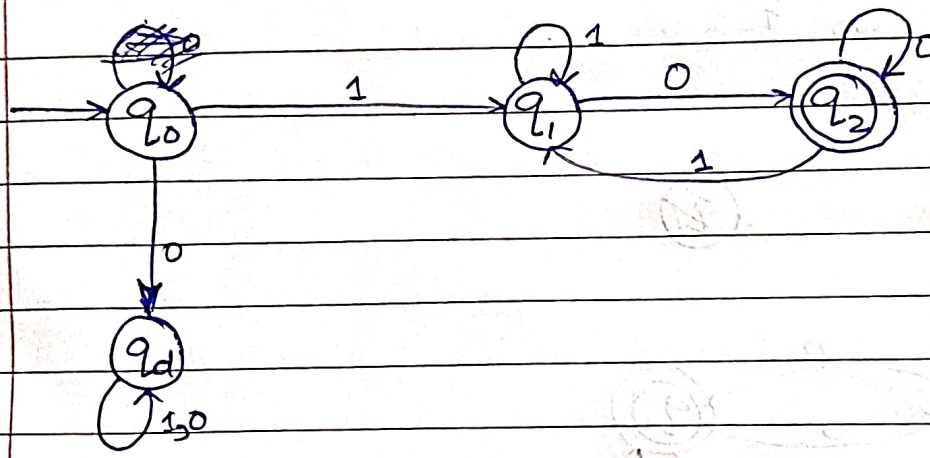
Now flip the states
Make final state to non final states and vice versa



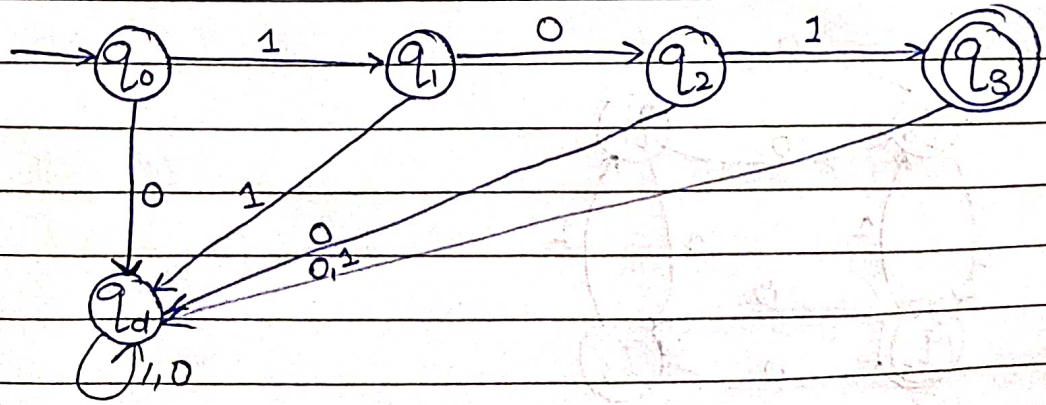
Construct a DFA that accepts all strings over {a,b} that contains string aabb in it.



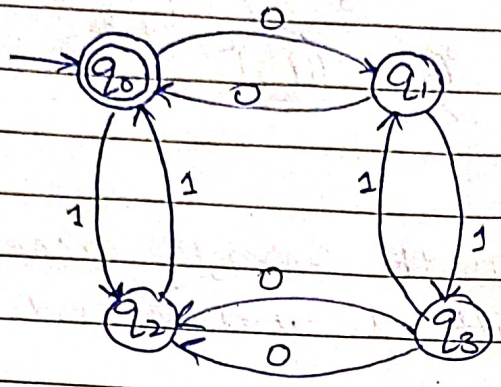
Construct a DFA with Σ={0,1} accepts those strings that start with 1 and ends with 0.
Total states = 3



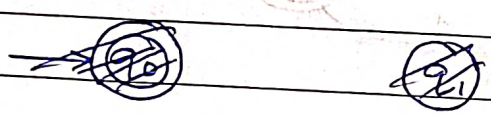
Design a FA with Σ={0,1} accepts the only input 101.
Total States = 4

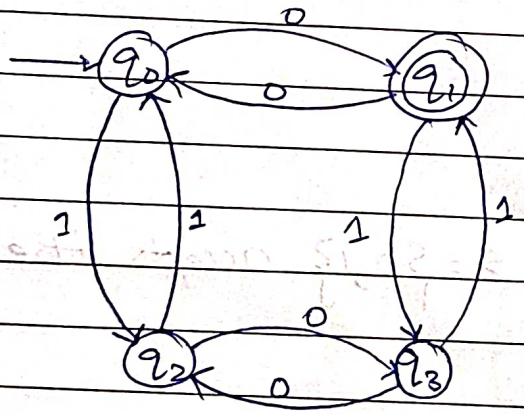


Design FA with $\Sigma = \{0, 1\}$ accepts even no. of 0's and even no. of 1's.

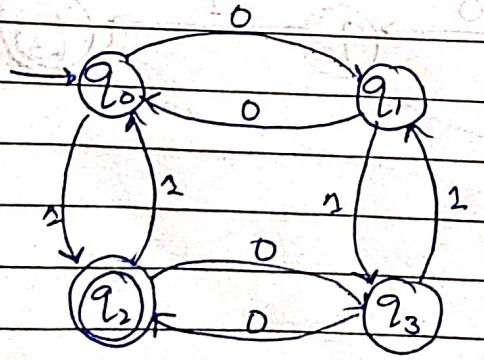


Odd 0 & Even 1

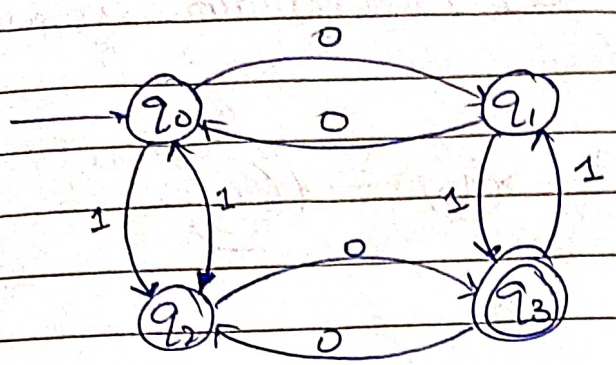




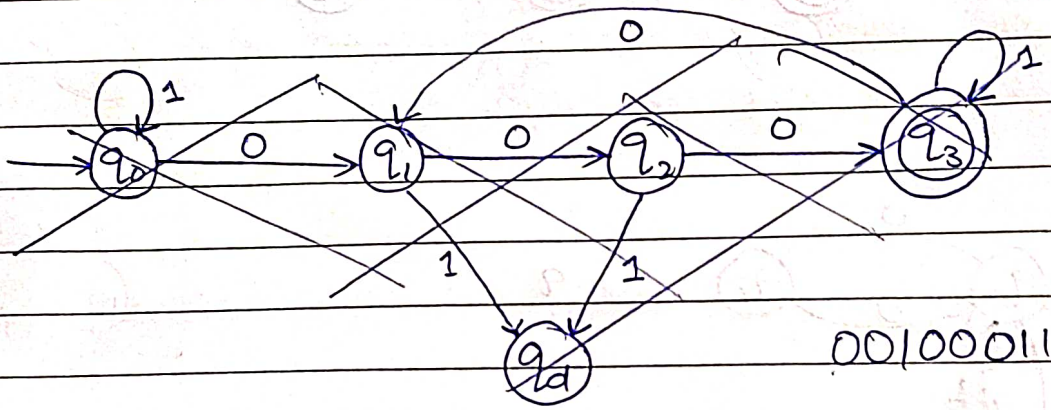
Even 0 and Odd 1



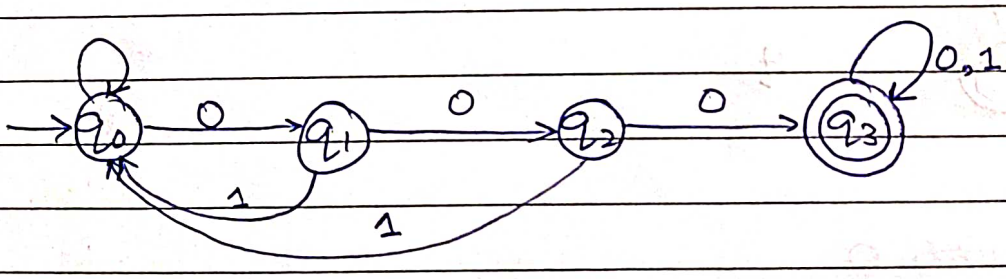
Odd 0 and Odd 1



Design FA with $\Sigma = \{0, 1\}$ accepts the set of all strings with three consecutive 0's.

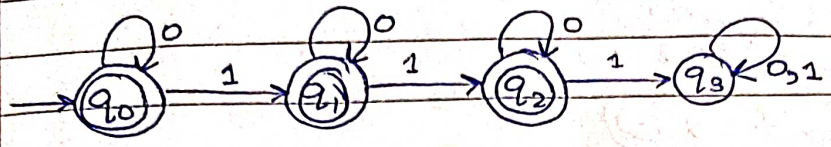


00100011

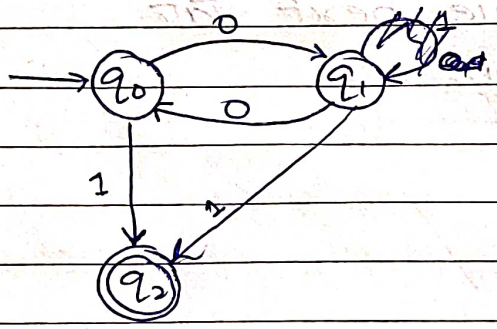


Design a DFA $L(M) = \{w | w \in \{0,1\}^*\}$ and w is a string that does not contain consecutive 1's

$L = \{\varepsilon, 01, 1, 010, 0010, 1010 \cdots\}$

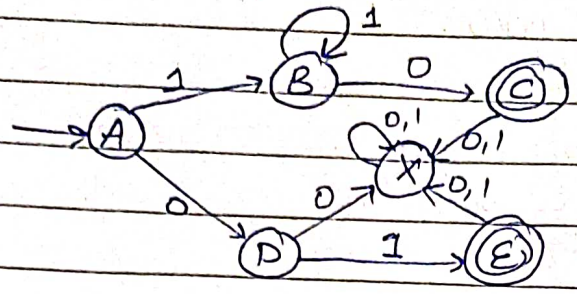


Design a PA with $\Sigma = \{0,1\}$ accepts the strings with an even number of 0's followed by single 1.



00100...

L = {Accepts the string 01 or a string of atleast one '1' followed by a '0'}



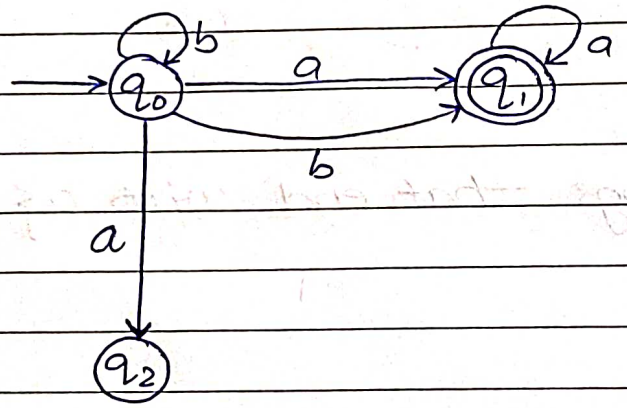
Properties of DPA

* In DPA, given the current state we know what next state it will be
* It has no choices or randomness.
* It has only one unique next state
* Simple and easy to design.

NFA → Non-deterministic finite Automata

* It is easy to construct an NFA than DFA for a given regular language.
* FA is called NFA when there exist many paths for specific input from current state to the next state.
* Every NFA is not a DFA but each NFA can be translated into DFA.
* NFA is defined in the same way as DFA but with the following two exceptions it contains multiple next states and it contains ε transitions.

Example:-



Formal Definition of NFA:-

NFA also has five states same as DFA but different transition functions

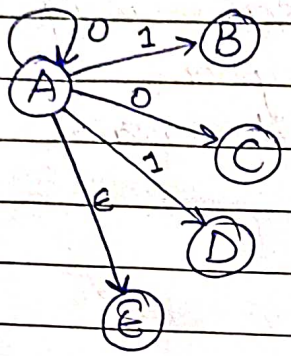
$\delta : Q \times \Sigma \rightarrow 2^Q$

Q: set of states
Σ: set of input symbols
q_0: Initial state
F: final state
δ: transition function.

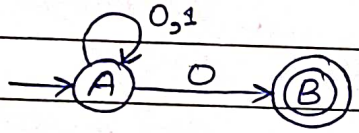
Properties of NFA:-

* In NFA, given current state there could be multiple next states.
* The next state may be chosen at random.
* All next states may be chosen in parallel.

Example:-



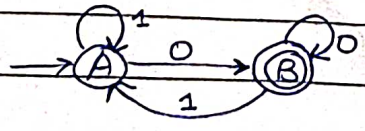
L={set of all strings that end with 0}



Difference from DFA

- from A we have transition for 0 more than 1 time
- no transitions are there on B but still it is complete.

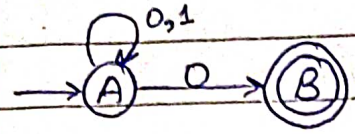
For the same if we construct DFA, it will be like this:



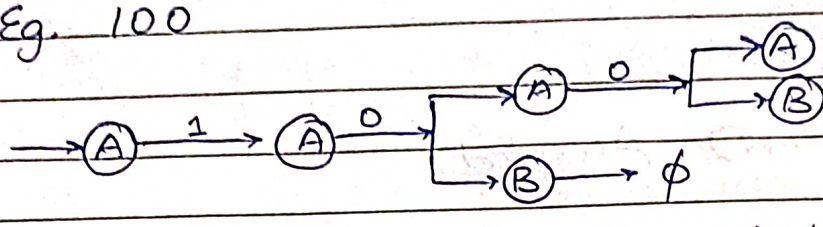
In this we have to show all the leftover inputs but this is not the case with NFA

classfellow NFA Examples Solved

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



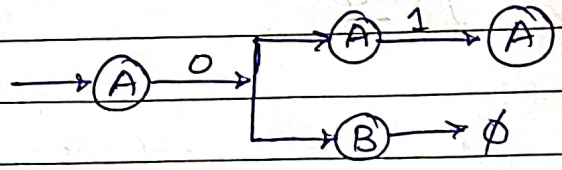
Eg. 100



as one of the states is an accepting states

=> It is accepted

Eg 01

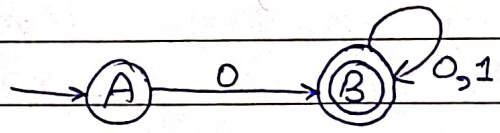


as it is not an accepting state

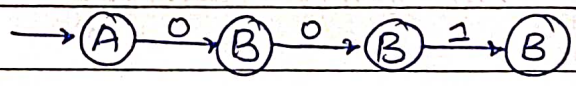
=> It is rejected

Set of all strings that starts with 0

L = {0, 00, 01, 000, ...}

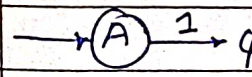


Eg 001



Accepted as B is final state

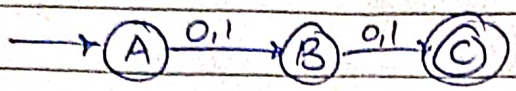
Eg 101



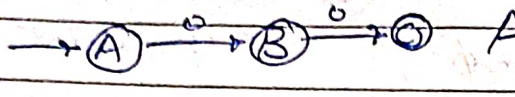
Rejected

Dead configuration

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

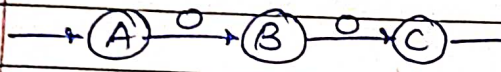


Eg 00



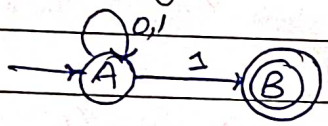
Accepted

Eg 001

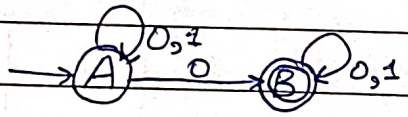


Rejected

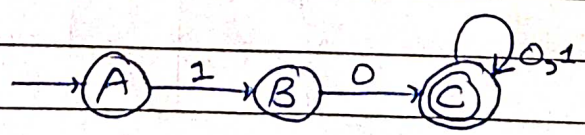
L={set of all strings that ends with 1}



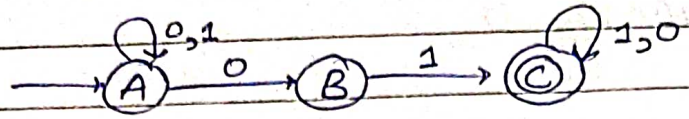
L={set of all strings that contains 0}



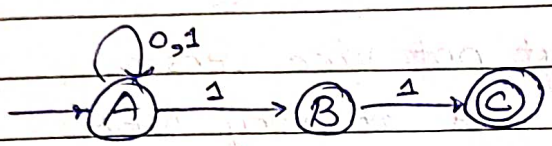
L={set of all ~~str~~ strings that ~~start~~ with 10}



L={set of all strings that contains 01}

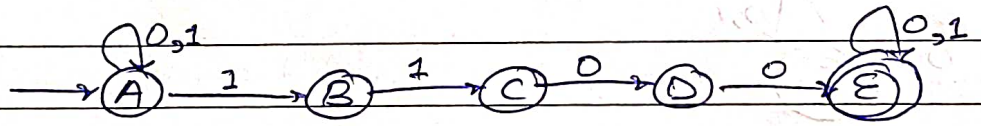


L={set of all strings that ends with 11}

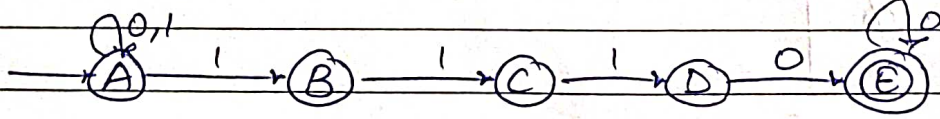


Conversion of

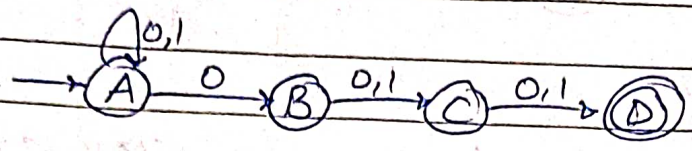
Design an NFA with Σ={0,1} in which double 1 is followed by double 0



Design an NFA in which all string contains a substring 1110



Design an NFA with $\Sigma = \{0, 1\}$ accepts all strings in which the third symbol from the right end is always 0.



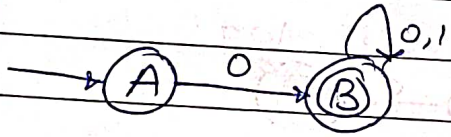
Conversion of NFA to DFA

Every DFA is an NFA but not vice versa.
But there is equivalent DFA for every NFA.

$$NFA \cong DFA$$

$L = \{$set of all strings over $\{0, 1\}$ that starts with 0$\}$
$\Sigma = \{0, 1\}$

NFA



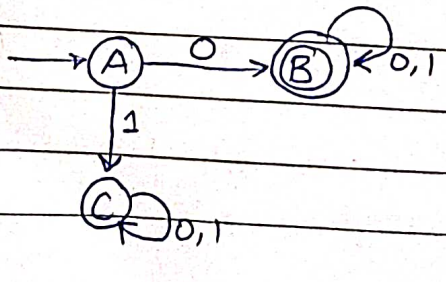
Transition Table:-

	0	1
A	B	ϕ
B	B	B

DFA

	0	1
A	B	C
B	B	B
C	C	C

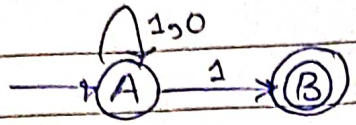
C → dead state



CPPL
Date: / / 2022
Page No.

classfellow Conversion Examples Solved

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



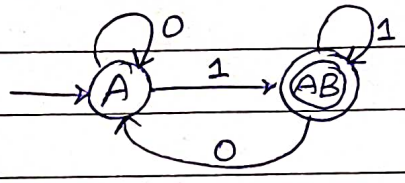
Transition table for NFA

	0	1
A	{A}	{A, B}
B	ϕ	ϕ

Transition table for DFA

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

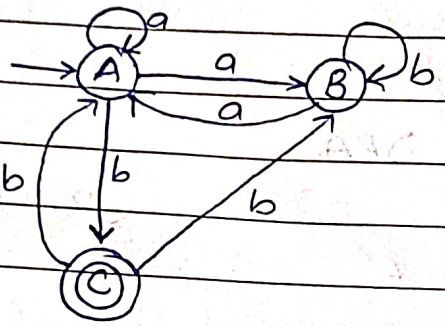
To write for AB → A∪B
↳ union



Conversion Method used here is Subset Construction Method.

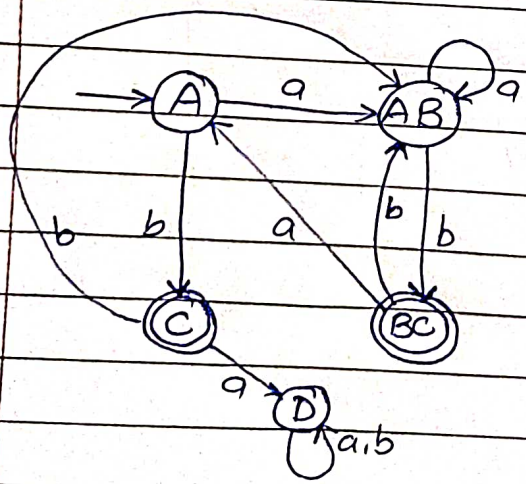
Find the equivalent DFA for the given NFA by $M = [\{A, B, C\}, (a, b), \delta, A, \{C\}]$ where δ is given by:-

	a	b
→ A	A, B	C
B	A	B
(C)	–	A, B



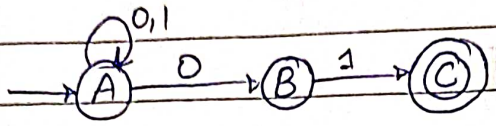
Transition table for DFA

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



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

NFA



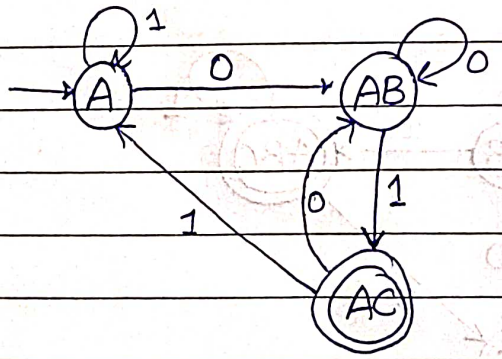
Transition table for NFA

	0	1
→A	A,B	A
B	∅	C
(C)	∅	∅

Transition table for DFA

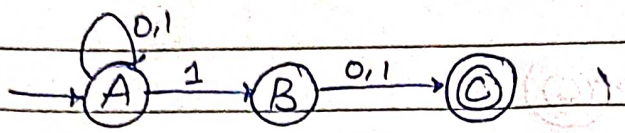
	0	1
→A	AB	A
AB	AB	AC
(AC)	AB	A

DFA



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 equivalent DFA

NFA



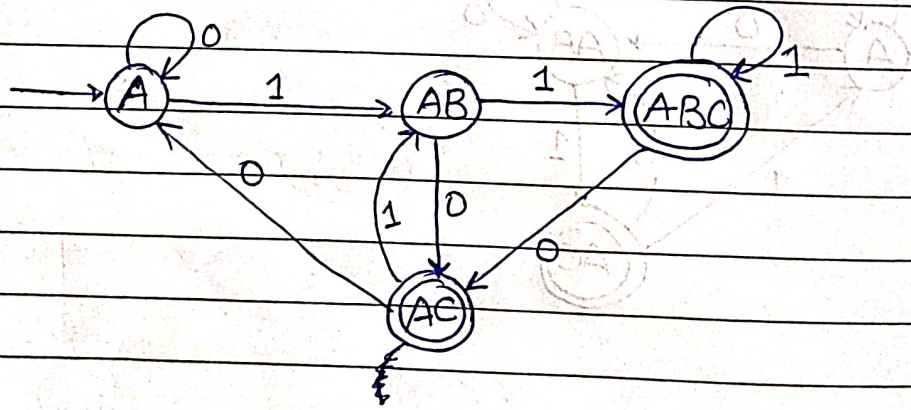
Transition table for NFA

	0	1
→ A	A	A, B
B	C	C
(C)	-	-

Transition table for DFA

	0	1
→ A	A	AB
AB	AC	ABC
(AC)	A	AB
(ABC)	AC	ABC

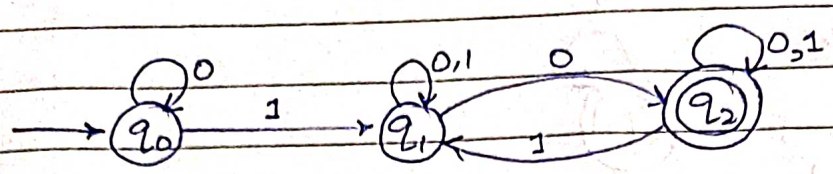
DFA



classfellow

Convert the given NFA to DFA

NFA



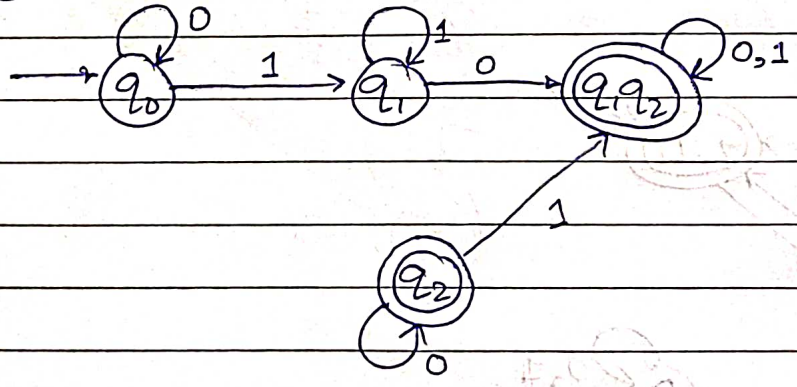
Transition table for NFA

	0	1
→ q_0	q_0	q_1
q_1	$q_1 \to q_2$	q_1
(*) q_2	q_2	$q_1 \to q_2$

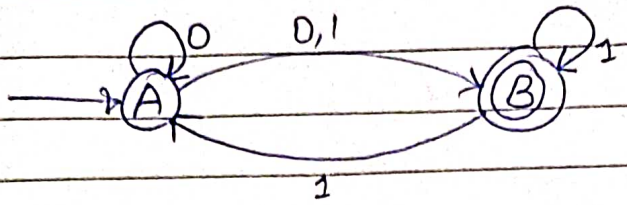
Transition table for DPA

	0	1
→ q_0	q_0	q_1
q_1	$q_1 q_2$	q_1
(*) $q_1 q_2$	$q_1 q_2$	$q_1 q_2$
(*) q_2	q_2	$q_1 q_2$

DPA



Construct the given NFA to DFA



Transition table for NFA

	0	1
→ A	A, B	B
(B)	–	A, B

Transition table for DFA

	0	1
→ A	AB	B
(AB)	AB	AB
(B)	D	AB
D	D	D