
AIUB
SOLUTION

Theory of Computation

Introduction: is a branch of compt science that deals with
→ how efficiently prob can be solved on
a model of computation, using an algorith.

type:
- Automate theory & langu
- Computatity theory
- Complexity theory

① Automate theory & language: deals with the definition & properties of various mathematical model of computation

Models: finite Automate
Content ofree grammers
turing m/c

② Computability theory: It deal's with what can be cannot be computed by the model.

③ Complexity theory: It group the computable problem base on their hardness.

Main purpose of TOC: Develop formula model mathematical model of computation that reflect real world computer.

Basic Defination

① Symbol -

Symbol is a character.

Eg: a, b, c - - - z

0, 1, 2 - - - 9

+, -, %. ..... special characters

② Alphabet:

An alphabet is a finite non empty set of Symbol. It is denoted by Σ

Example:

Eg $\Sigma = \{0, 1\}$ set of binary alphabets

$\Sigma = \{a, b \ldots z\}$ set of all low case letters

$\Sigma = \{+, \&, \%, \ldots\}$ set of all special character

③ String of word:

A String is a finite set sequence of Symbole chosen from some alphabet.

Eg: a) 01110110 is a string from binary alphabate. $\Sigma = \{0, 1\}$

b) aabbaccabb is a string from alphabet

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

Symbol → Alphabet ⇒

3rd:

Empty string: The empty strong is a the strong with zero occurence of symbol (No symbols)

It is denoted by "E" → No symbols

{ }

Length of string:

It is the no: of symboles in the string

It is denoted by |w|

$w = 0|0|1|1|0|0|1|0|1$ from binary alphabate

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

Lenth of string $|w| = 9$

Concatenation of string: Join 2 or more strings

Let, $x = a_1, a_2, a_3 \ldots a_m$

$y = b_1 b_2 b_3 \ldots b_n$

Concatenation of string $xy = a_1 a_2 a_3 \ldots a_n b_1 b_2 b_3$

$A_1 = \{ w : w$ is a binary string containing an odd number of 1s $\}$

$A_2 = \{ w : w$ is binary string containing an even number of 0's $\}$

4th:

Power of an alphabate $\rightarrow \Sigma$

$\Rightarrow$ If Σ is an alphabate. we can express set of all string of certain length from that alphabate by using exponestial notation it is denoted by Σ^k is the set of string of length k.

Eg 1 If $\Sigma = \{0,1\}$ has 2 symboles

(i) $\Sigma^1 = \{0,1\}$ $(\because 2^1 = 2)$ $k = 1$

(ii) $\Sigma^2 = \{00, 01, 10, 11\}$ $(\because 2^2 = 4)$ $k = 2$

(iii) $\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$ $(2^3 = 8)$ $k = 3$

Σ^* is called kleen closer

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

$= \{\varepsilon, 0, 1, 00, 01, 10, 11 - - .\}$

$(\because \Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 - - .)$

with ε symbole.

$\Rightarrow$ The set of strings over an alphabete Σ excluding ε is usally denoted by Σ^+ = kleen plus

$$\therefore (\varepsilon^+ = \varepsilon^* - \{\varepsilon\})$$

$$\varepsilon^+ = \varepsilon^1 \cup \varepsilon^2 \cup \varepsilon^3$$

with out ε symbole

Power of alphabute
- kleen Closure (ε^+)
- kleen plus (ε^+)

Eg:

language (finite set of non empty string)

ε is an alphabet

6th. Finite Automata:

Finite automata is an abstract compaling devie. It is a mathmatical model of a system with discrete inpats, Outputs, States and Set of Transition from state to State that occurs on input symbols from alphabet Σ.

Representation:

1. Graphical | Tran. Diagram — trantable
2. Tabular | tran table
3. Mathematical | Transition or function or Maping func

Formal Defination of Finite Automata:

A finite auto mata is a 5-tuples; they are

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

Q : is a finite set called the states

Σ : is a finite set called the alphabets.

$\delta : Q \times \Sigma \to Q$ is the transition function.

$q_0 : \in Q$ is the start state called initial state

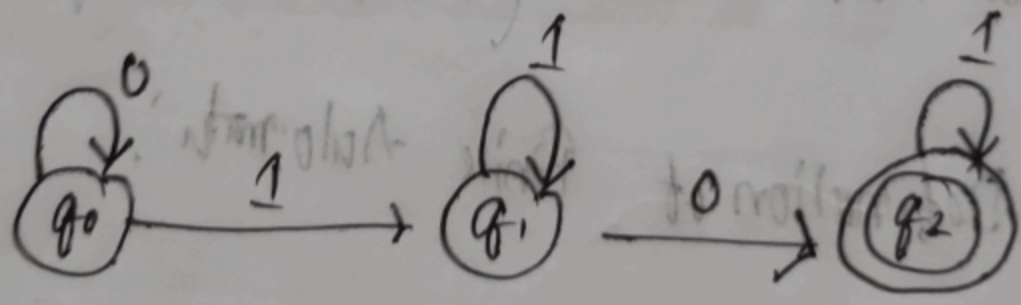
$F \subseteq Q$ is the set of all that states called final state.

7th:

Graphical (transition Diagram)

It is a directed graph associated with vertices of the graph corrosponds of the state of finite automata.

Eg.



$\{0,1\}$ is are input.

Transition table:

It is a basically tabular representation of the transition function that takes two arguments (a state and a symbol) and returns a value (the next state

→ Row corresponds to states

→ Column corresponds to input symboles

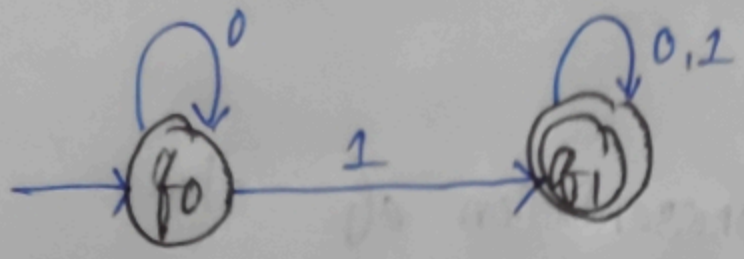
→ Entries corresponds to next states

→ the start state is marked with an arrow

→ the accpt state are maked with star (*)

row represents → State

↓ Colum represents → input.



	0	1
q_0	q_0	q_1
q_1	q_1	q_1

$q_0 \times 0 \rightarrow q_0$

$q_0 \times 1 \rightarrow q_1$

$q_1 \times 0 \rightarrow q_1$

$q_1 \times 1 \rightarrow q_2$

Transition function

- the mapping function of transition function denoted by δ
- Two parameters are passed to this transition function.

transition function returns a state which can be called as next state

δ (Current - state, Current - input state - symbol) = next state

$Q \times \Sigma = Q$

Example $\delta(q_0, a) = q_1$

$\delta(q_0, 1) = q_1$

8th:

→ Compiler desing

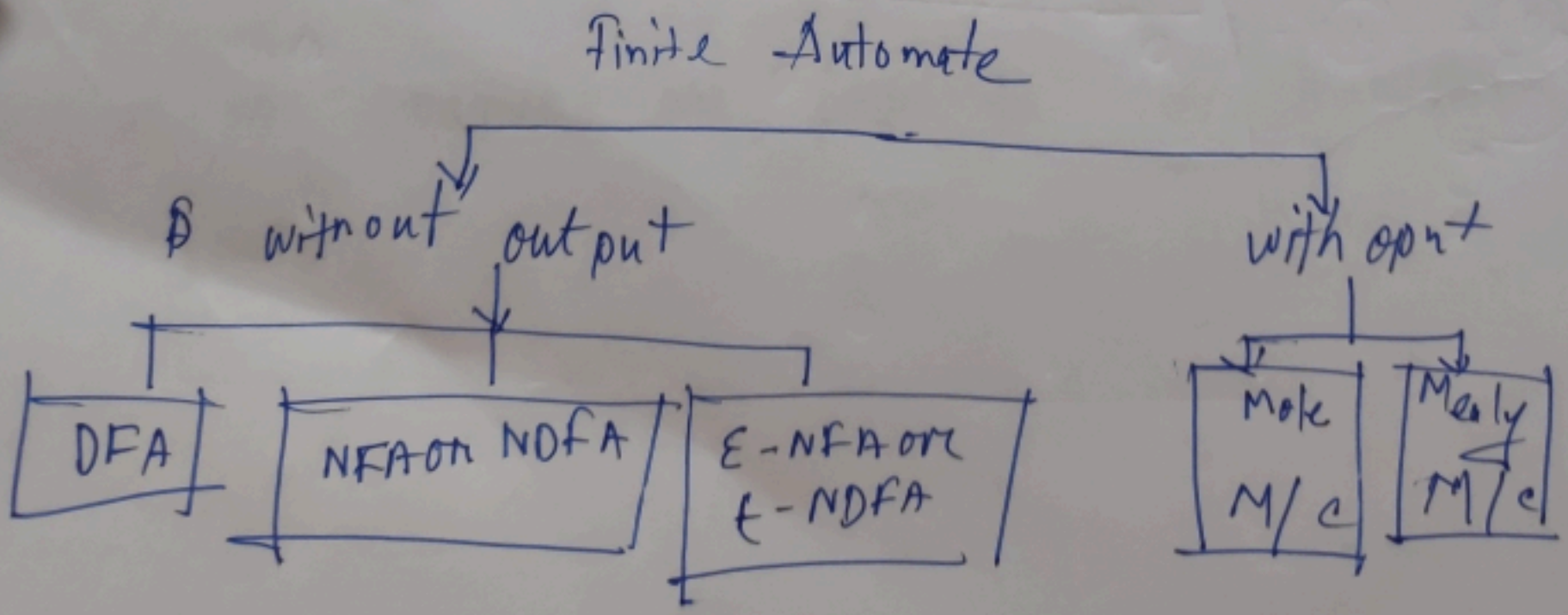
→ In switching theory desing and analysis digital circuits automata theory is applied

→ Desing and analysis of Complex S/w & H/w desing.

→ To prove correctness of the program automata theory is used.

→ To designer finit state machines such as · Mealy & Moore

→ base for formal language that base of programing language.



9th:

DFA

→ read an i/p string one o symbol at a time.

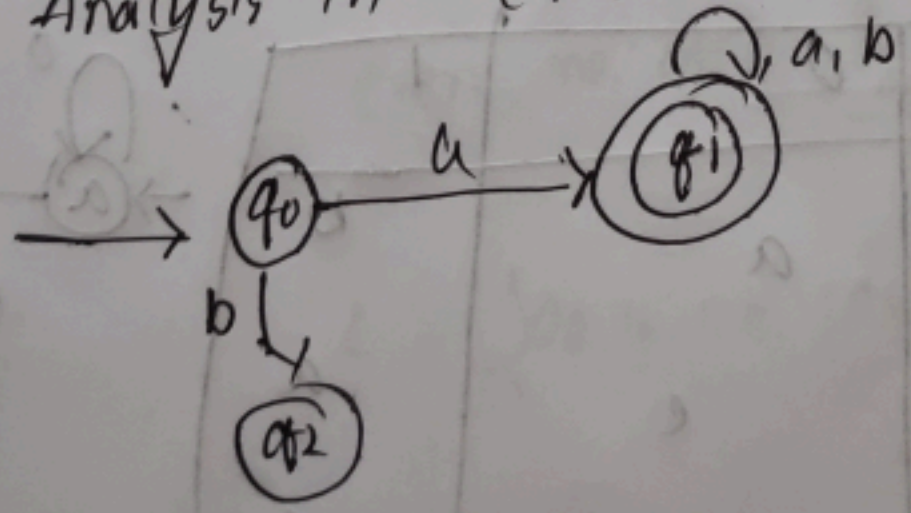
→ relyes to the uniqueness of the computation.

→ olny one path for specific i/p from the current state to the next state.

dose not accept.

→ null more, it canit chang state without any i/p charater

can contain o multiple final states. It is used lexical Analysis in compiler.



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

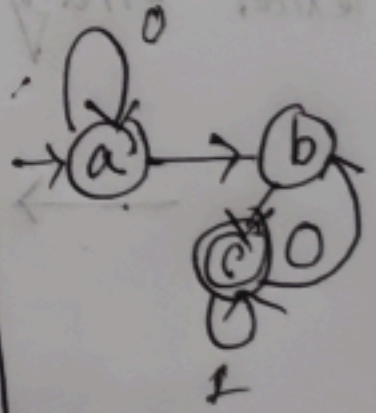
Collection of 5-tuples (Same as FA)

10-th!

(1) let DFA be $Q = \{a, b, c\}$ $q_0 = \{a\}$

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

Present State	0	1
→ a	a	b
b	c	a
* c	b	c

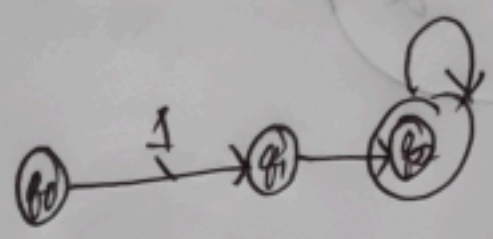


Example 2: Dosing DFA with $\Sigma = \{0,1\}$ accepts those strings which start with 1 & ends with 0.

$L = \{10, 1000, 100, 1010, \ldots\}$

min length = 2 number of states = min length + 1

= 2 + 1 = 3



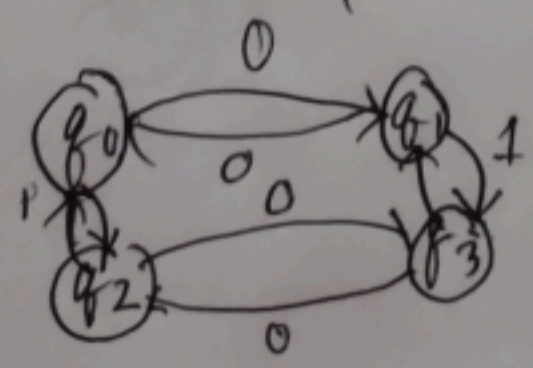
10 — Acc

100

Exam – 3: DFA $\Sigma = \{0,1\}$ accepts

Even no. of 0 & Even no. of 1's

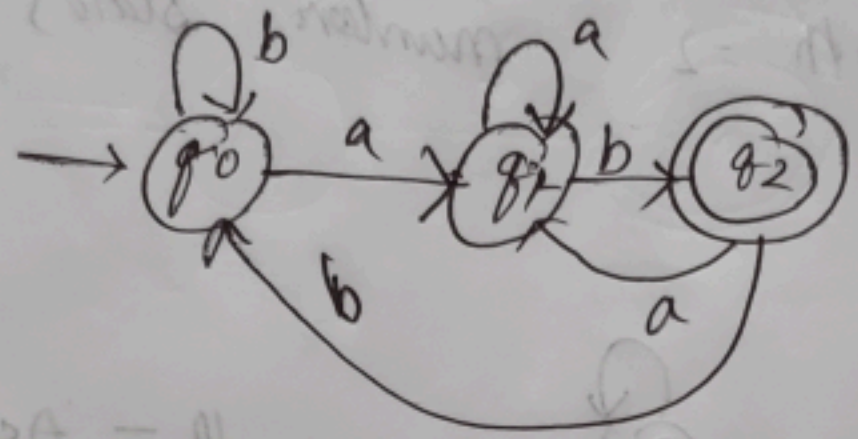
Sol: $L = \{00, 11, 110, 100, 110 \ldots\}$



Eg. 4 DFA accepting all string over {a, b} Ending with "ab" ?

$L = \{ab, aab, bab.$

min length = 2 no of state = 2+1
= 3



ab

~~11th~~:

~~DFA~~:

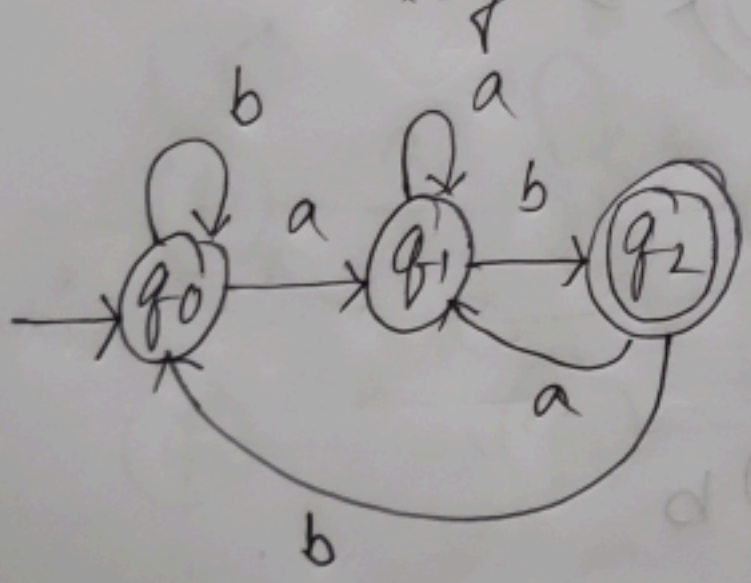
10

12: Construct DFA accepting all strings over {a, b} Ending with "ab"?

Solution: $L = \{ab, aab, bab, \text{-----}?\}$

min length = 2 no state = 2+1

= 3



13th:

13-th:

NFA (NOn Deterministic Finite Automata) NDFA

→ where there exits many path for specific i/p from the current state to the next state.

→ Easy to construct NFA from DFA for a given regular language.

→ Every NFA is not DFA, but Each NFA can be translated into DFA.

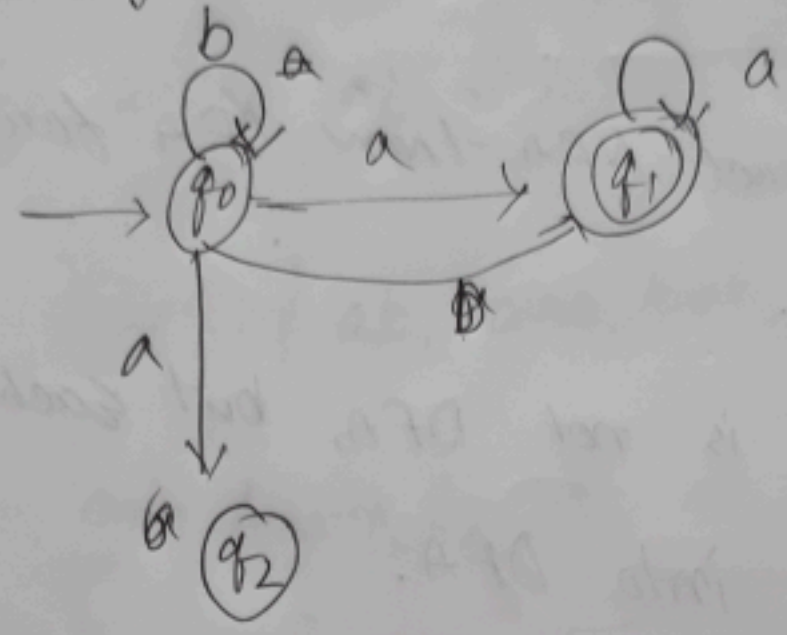
DFA – only on path

NFA – Many path.

NFA is defined in the same way as DFA but two exceptions.

- It contains multiple next state &
  → It contains ε transitions

Eg-1



Formal Defination of NFA:

Same as DFA
but different transtion function

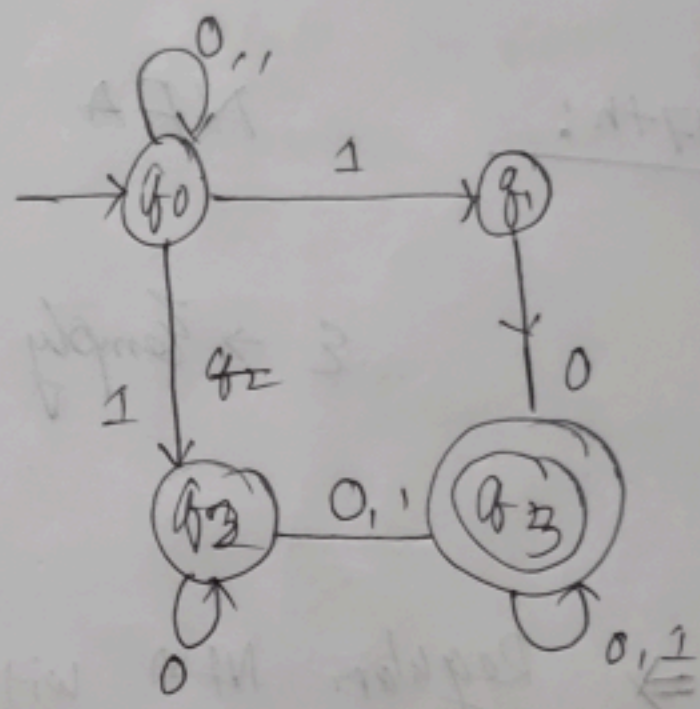
$$\delta : Q \times \Sigma \rightarrow 2^{Q}$$

Q : finite set of states
Σ : finite set of the i/p symbols.
q_0 : initial state
F : final state

δ = Transition function.

Example: 3 Desing NFA to transition table:

State	0	1
q_0	q_0, q_1	q_0, q_2
q_1	q_3	ϵ
q_2	q_1, q_3	q_3
$*q_3$	q_3	q_3



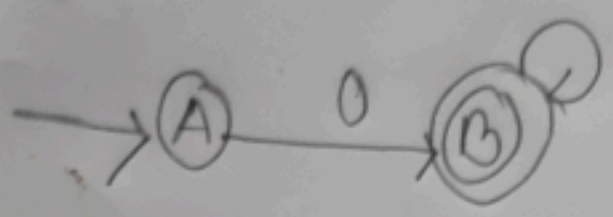
Example: NFA

$L = \{$ Set of All strings that start with 0 $\}$

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

Sol $L = \{0, 00, 01, 001, 010, 000 \ldots$

min length = 1



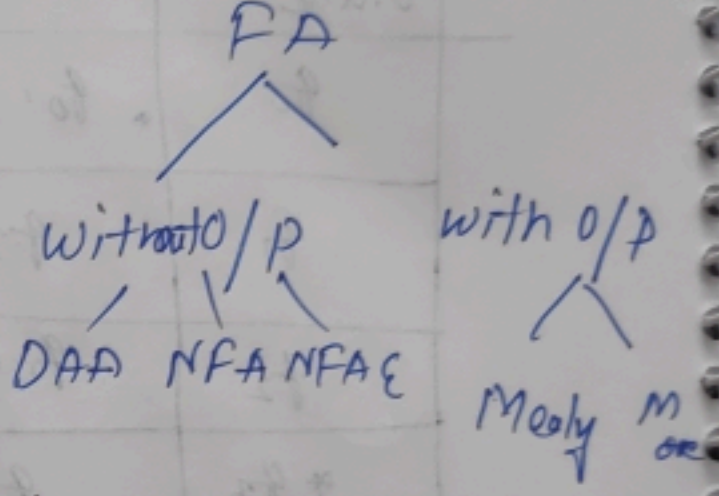
Start with 0.

0 0 1

A —1→ ϕ dead configuration.

17th: NFA with ε

ε → Empty Symbol.



⇒ Regular NFA with 5 tuples..

$\{Q, \Sigma, \delta_0, F, \delta\}$

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

→ ε NFA with 5 tuples

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

Note: Every state on ε ∈ goes to itself.

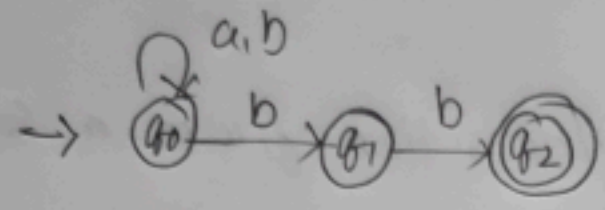
18th:

DFA vs NFA

DFA	NFA
1. Transition from a state is a single particular state	1. Transition from a state to multiple next state in each i/p symbols. Hence it.
2. Empty string transition are not seen in DFA	2. It permit empty string.
3. DFA is sub set of NFA	3. it Need to convert
4. $\delta : Q \times \Sigma = Q$ Eg: $\delta(q_0, a) = \{q_1\}$	4. $\delta \; Q \times \Sigma = 2^Q$ $\delta(q_0, a) = \{q_1, q_2\}$
5. Requires more space	5. Less space
6. Strong acceptance	6. A string is accepted by NDFA. it at least one all possible transition ends in final.

10th:

Conversion of NFA to DFA



=> NFA

1st Step: Construct NFA transition table

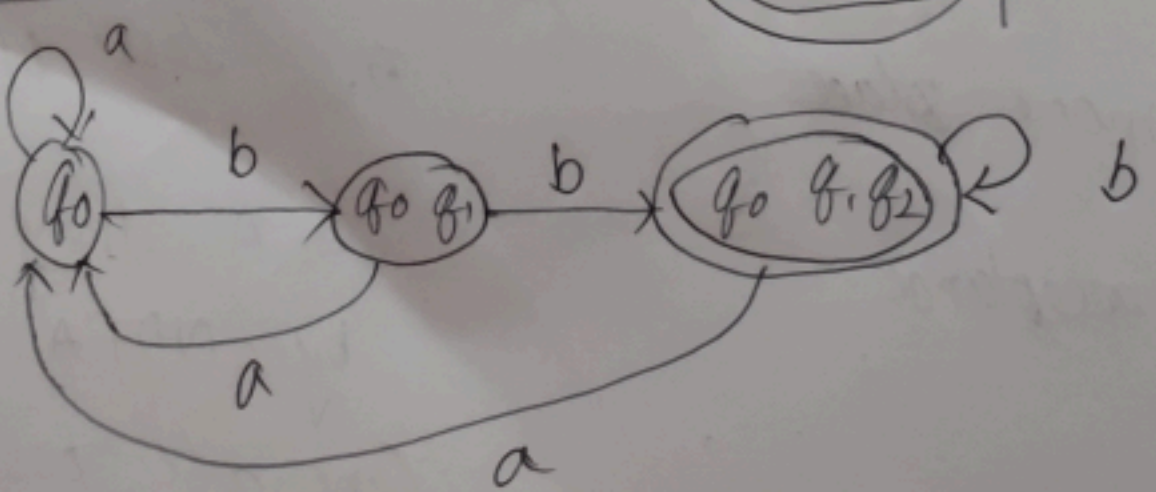
	a	b
→ q_0	q_0	$\{q_0, q_1\}$
q_1	–	q_2
(q_2)	–	–

NFA:

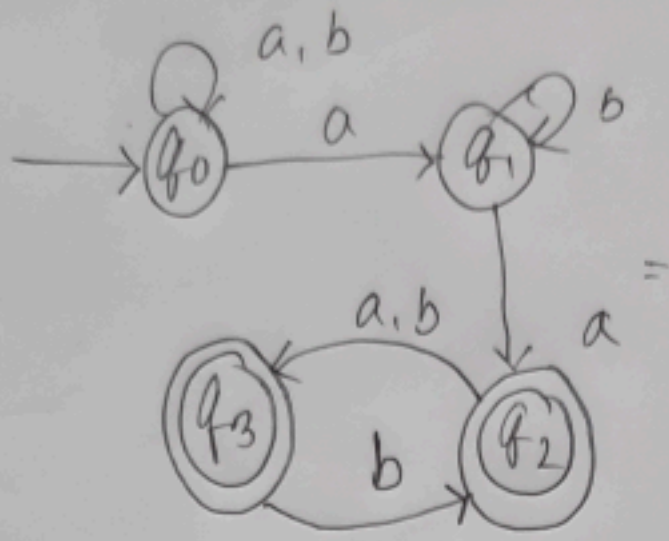
DFA

δ	a	b	S
→ q_0	q_0	$[q_0, q_1]$	
$[q_0, q_1]$	$[q_0]$	$[q_0 q_1 q_2]$	
($[q_0 q_1 q_2]$)	$[q_0]$	$[q_0 q_1 q_2]$	

DFA Diagram



20th :



=

	a	b
q_0	$q_1 q_0$	q_0
q_1	q_2	q_1
q_2	q_3	q_3
q_3		q_2

DFA :

γ	a	b
q_0	$[q_0 q_1]$	q_0
$[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_0 q_1]$
$[q_0 q_1 q_2]$	$[q_0 q_1 q_2 q_3]$	$[q_0 q_1 q_3]$
$[q_0 q_1 q_2 q_3]$	$[q_0 q_1 q_2 q_3]$	$[q_0 q_1 q_3 q_2]$
$[q_0 q_1 q_3]$	$[q_0 q_1 q_2]$	$[q_0 q_1 q_2]$