FLAT :- formal language & automata theory.

① Introduction to Automata Theory.
② Regular language & expression.
③ CFG → context free Grammar & languages.
④ PDA & TM → Turing machine
↓
push down automata.
⑤ Lex & Yacc

PDA ⟨ DPDA Deterministic (one state) / NPDA Non deterministic

* INTRODUCTION TO AUTOMATA THEORY :-

* Basic terminologies.
* NDFA → Non-deterministic finite automata.
* DFA → Deterministic finite automata
* Equivalence of DFA & NDFA
* ϵ-NFA

① ALPHABET :- A finite non empty set of symbols/elements (Σ)

eg $\Sigma = \{0,1\}$, $\Sigma = \{a,b,c,d, ---, z\}$

$\Sigma = \{0,1\}$ → binary alphabet

$\Sigma = \{a,b,c,d ---- z\}$ → set of lower case english letter

$\Sigma = \{$ ~~aa~~ $aa, aaaa, aaaaaa, ...\}$

② STRINGS :- A finite sequence of symbols.
(word)

ϵ, λ → Empty string

e.g ① 01101
② abbabc
③ $\{\epsilon, 011010, 0011\}$

length of string (cardinality / size) :- | |

Scanned by CamScanner

(2) $1011011 = 5$ $|abbabc|$ $\{\epsilon, 01, 1010, 0011\}$ length 0, 2, 4, 4 (1 zero 1 one) (2 zero 2 one)

* Concatenation of strings:→

x is a string having l symbol in it. y is also a string having j number of symbols in it such that

$x = a_1 a_2 a_3 \ldots a_l$

$y = b_1 b_2 b_3 \ldots b_j$

then concatenation of two strings $xy = a_1 a_2 a_3 \ldots a_l b_1 b_2 b_3 \ldots b_j$

(3) power of an alphabet :→

set of all strings of length k is denoted by Σ^k where k is length of the string.

$\Sigma^1 = \{0, 1\}$, $\Sigma^0 = \{\epsilon\}$, $\Sigma^2 = \{00, 01\}$, $\Sigma^3 = \{000, 111\}$

power of an alphabet is also denoted with :→

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots$

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \ldots$

(Non empty finite set of alphabets)

(4) Language (L) :→ set of all strings

$L = \{\epsilon, 0, 1, 00, 11, 1010, 0011, 11001, 11001100 \ldots\}$

Hirarchy or classification of languages

① Type 0 → Recursively enumerable → TM (Turing machine)

② Type 1 → Context sensitive Language → LBA (Linear Bounded automata)

③ Type 2 → Context free Language → PDA (Push down automata)

④ Type 3 → Regular Language → FA (Finite automata)

ϕ → empty Language

Type 0 → phrase structured / Recursively enumerable → TM (Turing machine)

Type 1 → context sensitive Language → LBA → Linear Bounded automata.

Type 2 → Context free language → PDA → push down automata

Type 3 → Regular language → finite automata.

- → DFA
- → NFA / NDFA
- → ϵ-NFA

} abstract machines / computational models

TOC → Theory of computation

abstract → Brief summary.

Scanned by CamScanner

Finite automata is defined as abstract machine, a mathematical computational model which comprises both hardware & software in reality these kind of machine don't have physical existence.

Types :→

① DFA :→ Deterministic final automata

② NDFA :→ Non-Deterministic final automata

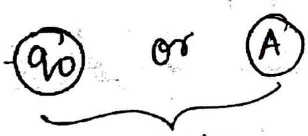
③ ε - NFA

* Representation of finite automata :→

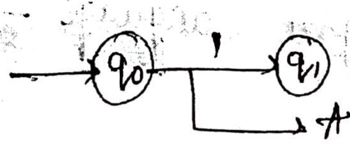
- states
- Input alphabets
- Transition
- start state
- final states

TD → Transition diagram

TT → Transition Table.

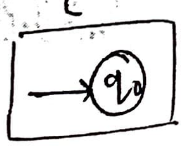


states



Arc / edge { Transition from q_0 on 1 to q_1

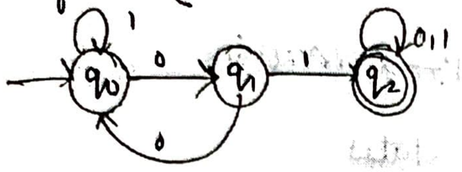
The start state is represented by :-



start state.

Final state is denoted by

* Example for DFA. Identify the start state, final state & transition.



start state → q_0 / q_s
final state → q_2 / q_f
Input alphabets → $\Sigma = \{0, 1\}$.

Transitions :- $\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_0$ $\delta(q_1, 1) = q_2$
$\delta(q_1, 0) = q_0$ $\delta(q_2, 1) = q_2$ $\delta(q_2, 0) = q_2$

* Transition Table :-

q_0 → start state
$q_2, q_1 \subseteq Q$ → final state

Transition Table

$\Sigma = \{a, b\}$
$\delta(q_0, a) = q_1$
$\delta(q_0, b) = q_0$
$\delta(q_1, a) = q_1$
$\delta(q_1, b) = q_2$
$\delta(q_2, a) = q_2$
$\delta(q_2, b) = q_2$

δ	a	b
→ q_0	q_1	q_0
* q_1	q_1	q_2
* q_2	q_2	q_2

states (rows); Input alphabets (columns)

There → indicates start state
* indicates final state

DEFINATION :→

DFA :→ Deterministic finite automata

$M_{DFA} = (Q, \Sigma, \delta, q_0, q_f)$ M :→ Machine

Q :→ finite non empty set of states

Σ :→ finite non empty set of input alphabets

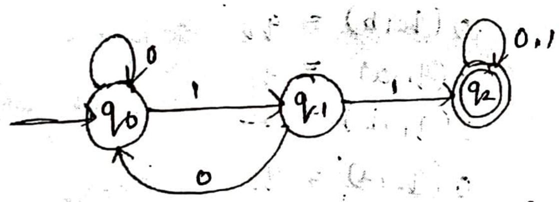
δ :→ Transation function.

$\delta : Q \times \Sigma = Q$

Transition is mapping from states to input alphabets which yields states

$q_0 \longrightarrow$ start state (Only one start state)

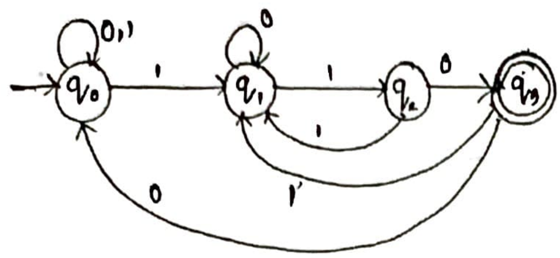
$q_f \longrightarrow$ final state & is subset of Q
(i.e more than one final state)



$Q = \{q_0, q_1, q_2\}$ $q_2 \longrightarrow$ final state

$\Sigma = \{0, 1\}$ $q_0 \longrightarrow$ start state

* Check wether the following transition constitute DFA or not

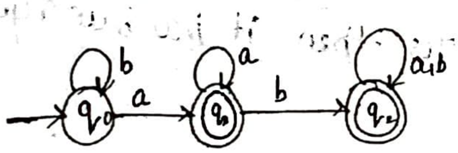
① 

$\delta(q_0, 0) = q_0$
$\delta(q_0, 1) = \{q_0, q_1\}$
$\delta(q_1, 0) = q_1$
$\delta(q_1, 1) = q_2$
$\delta(q_2, 0) = q_3$
$\delta(q_2, 1) = q_1$
$\delta(q_3, 0) = q_0$
$\delta(q_3, 1) = q_1$

Transition table

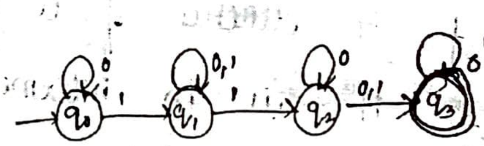
	δ	0	1
Q	$\rightarrow q_0$	q_0	$\{q_0, q_1\}$
	q_1	q_1	q_2
	q_2	q_3	q_1
	$* q_3$	q_0	q_1

As for same input alphabet there are multiple state it is not DFA.

② 

δ	a	b
$\rightarrow q_0$	q_1	q_0
$* q_1$	q_1	q_2
$* q_2$	q_2	q_2

It is DFA

③ 

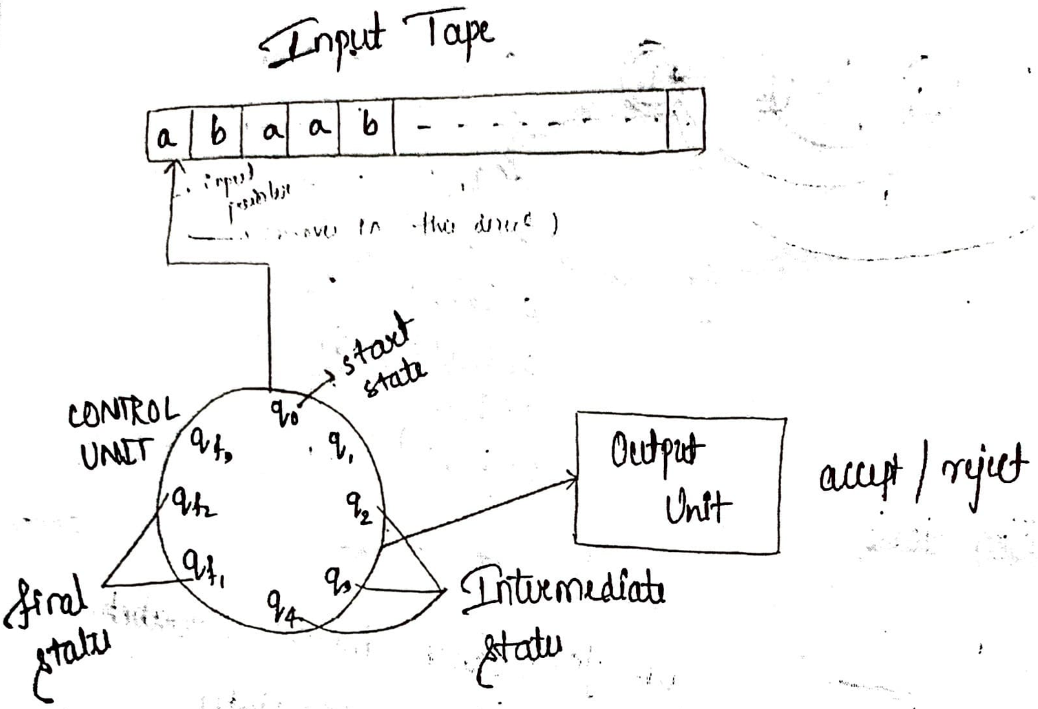
$\Sigma = \{0, 1\}$
$Q = \{q_0, q_1, q_2, q_3\}$

Transitions

$\delta(q_0, 0) = q_0$
$\delta(q_0, 1) = q_1$
$\delta(q_1, 0) = q_1$
$\delta(q_1, 1) = \{q_1, q_2\}$
$\delta(q_2, 0) = \{q_2, q_3\}$
$\delta(q_2, 1) = q_3$
$\delta(q_3, 0) = q_3$
$\delta(q_3, 1) = \emptyset$

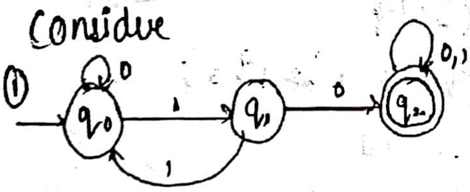
Scanned by CamScanner

* Working procedure of finite automata :-



The machine will be in start state (Assumption).
If it reaches final state then it is been accepted.
If it is in intermediate states then it has been rejected.

Consider

① 

$W_0 = 011010$

$\delta(q_0,0) = q_0$

$\delta(q_0,1) = q_1$

$\delta(q_1,1) = q_0$

$\delta(q_0,0) = q_0$

$\delta(q_0,1) = q_1$

$\delta(q_1,0) = q_2$ } → final state

$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2$

OR

$W_1 = 0110101$

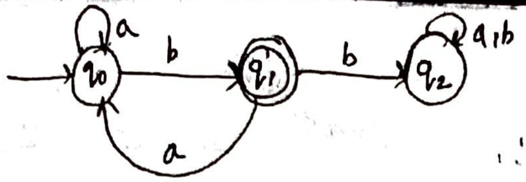
$q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_2$ } final state ∴ accepted

$W_2 = 110011011$

$q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0$ } Rejected

Scanned by CamScanner

(2)



w = abba

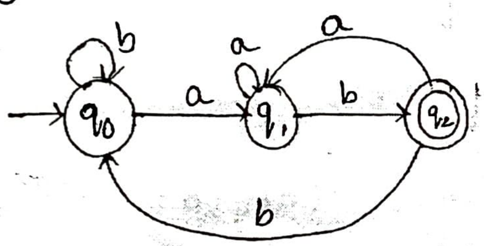
$q_0 \xrightarrow{a} q_0 \xrightarrow{b} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_2$ Rejected

w = abbabb

$q_0 \xrightarrow{a} q_0 \xrightarrow{b} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_2 \xrightarrow{b} q_2 \xrightarrow{b} q_2$.

* What are the moves that are made by the following DFA to accept the following strings.

(1) abaab (2) abb (3) abaa.



Sol^n ⇒ $Q = \{q_0, q_1, q_2\}$ $\Sigma = \{a, b\}$ } V. imp to const

$q_0 \rightarrow$ start state

$q_2 \rightarrow$ final state

$\delta(q_0, a) = q_1$ $\delta(q_1, a) = q_1$ $\delta(q_2, a) = q_1$

$\delta(q_0, b) = q_0$ $\delta(q_1, b) = q_2$ $\delta(q_2, b) = q_0$

(i) ~~ab~~ w : abaab

$\delta(q_0, a) = q_1$ $\delta(q_1, b) = q_2$ $\delta(q_2, a) = q_1$ $\delta(q_1, a) = q_1$ $\delta(q_1, b) = q_2$

Since the machine is in final state the string is accepted by DFA.

(ii) w = abb $\delta(q_0, a) = q_1$ $\delta(q_1, b) = q_2$ $\delta(q_2, b) = q_0$

Since the machine is not in final state the string is rejected or// $q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_0$

Scanned by CamScanner

(ii) abaa

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_1 \xrightarrow{a} q_1$

since the machine is not in final state it is rejected.

* Extended transition function :-

It is denoted by $\boxed{\delta^*(q, w) = p}$

δ^* denote extended transition function.

q is the first parameter which represents current state.

w is a second parameter which represents the input string.

p is a new state obtained after transition.

Let $w = xa$ w is input string.

$\delta(\delta^*(q, x), a)$

$\delta^*(q, \epsilon) = q$

* Using extended transition function write the moves that are made by DFA for the following.

(1) abaab   (2) abb   (3) abaa.

(1) $\delta^*(q_0, \epsilon) = q_0$

$a \rightarrow \delta^*(q_0, a) = \delta(\delta^*(q_0, \epsilon), a)$
$= \delta(q_0, a)$
$= q_1$

$ab \rightarrow \delta^*(q_0, ab) = \delta(\delta^*(q_0, a), b)$
$= \delta(q_1, b)$
$= q_2$

For the prefix

$aba \rightarrow \delta^*(q_0, aba) = \delta(\delta^*(q_0, ab), a)$
$= \delta(q_2, a)$
$= q_1$.

$abaa \rightarrow \delta^*(q_0, abaa) = \delta(\delta^*(q_0, aba), a)$
$= \delta(q_1, a)$
$= q_1$

$abaab \rightarrow \delta^*(q_0, abaab) = \delta(\delta^*(q_0, abaa), b)$
$= \delta(q_1, b)$
$= q_2$

Scanned by CamScanner

② $\delta^*(q_0, \epsilon) = q_0$

for prefix a $\quad \delta^*(q_0, a) = \delta(\delta^*(q_0, \epsilon), a)$

$= \delta(q_0, a)$

$= q_1$

for prefix b $\quad \delta^*(q_0, ab) = \delta(\delta^*(q_0, a), b)$

$= \delta(q_1, b)$

$= q_2$

for prefix b $\quad \delta^*(q_0, abb) = \delta(\delta^*(q_0, ab), b)$

$= \delta(q_2, b)$

$= q_0$

③ $\delta^*(q_0, \epsilon) = q_0$

$\delta^*(q_0, a) = \delta(\delta^*(q_0, \epsilon), a)$

$= \delta(q_0, a)$

$= q_1$

$\delta^*(q_0, ab) = \delta(\delta^*(q_0, a), b)$

$= \delta(q_1, b)$

$= q_2$

$\delta^*(q_0, aba) = \delta(\delta^*(q_0, ab), a)$

$= \delta(q_2, a)$

$= q_1$

$\delta^*(q_0, abaa) = \delta(\delta^*(q_0, aba), a)$

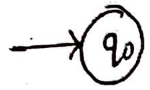
$= \delta(q_1, a)$

$= q_1$

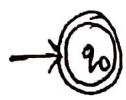
Scanned by CamScanner

* Transition diagram for the following language.

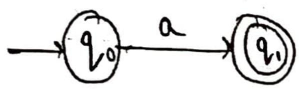
① ϕ → Empty language.



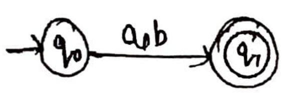
② ϵ → Empty string.



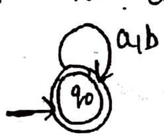
③ DFA to accept only one a



④ DFA to accept one a or one b.



⑤ DFA to accept zero or more number of a's or b's



DFA :→

① pattern recognition problem
② Divisible by k
③ Modulo k counter.

To solve a problem

STEP :→

① Identify minimum string.
② Identify input alphabet
③ Construct a base DFA having start state & final state.
④ Identify the transitions that are not defined in the 3rd step.

⑤ Write complete transition diagram with all ~~steps~~ transition & ~~to~~ hence write transition table.

* Construct DFA to accept strings of a having atleast one a.

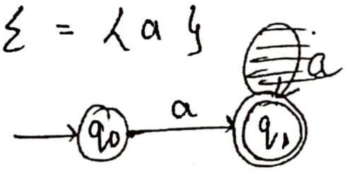
$\Rightarrow$ $L = \{a, aa, aaa, aaaa, aaaaa \ldots\ldots\}$

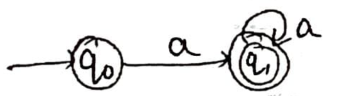
or

$L = \{W : |n_a \geq 1, \; w \in \{a^*\}\}$

① a

② $\Sigma = \{a\}$

③ 

④ 

⑤ $Q = \{q_0, q_1\}$ $\Sigma = \{a\}$ $q_0 \rightarrow$ start state $q_1 \rightarrow$ final state.

δ: $\delta(q_0, a) = q_1$, $\delta(q_1, a) = q_1$

δ	a
$\rightarrow q_0$	q_1
$* q_1$	q_1

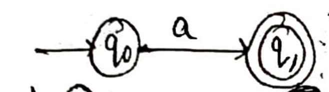
Scanned by CamScanner

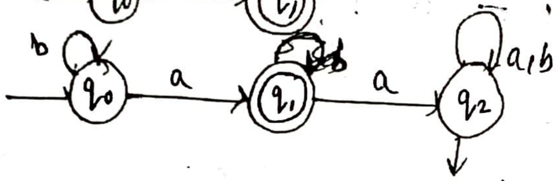
② Construct a DFA to accept strings of a's & b's having exactly one a.

⇒ ~~L = {a, ab, abb, abbb ....}~~
L = {ba, bba, bbba, bab .....}

① a

② $\Sigma = \{a, b\}$

③ 

④ 

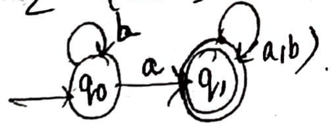
δ	a	b
$\rightarrow q_0$	q_1	q_0
$*q_1$	q_2	q_1
q_2	q_2	q_2

③ Construct DFA to accept strings of a's & b's having atleast one a.

⇒ ① a

L = {a, ab, abb, bab, bba, bbaa, baa, baaa, bbaa, bba ---}

② $\Sigma = \{a, b\}$

③ 

$\delta(q_0, b) = ?$
$\delta(q_1, a) = ?$
$\delta(q_1, b) = ?$

δ	a	b
$\rightarrow q_0$	q_0	q_1
$*q_1$	q_1	q_1

$Q = \{q_0, q_1\}$
$\Sigma = \{a, b\}$
$q_0 \rightarrow$ start state
$q_1 \rightarrow$ final state.

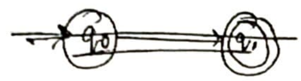
Scanned by CamScanner

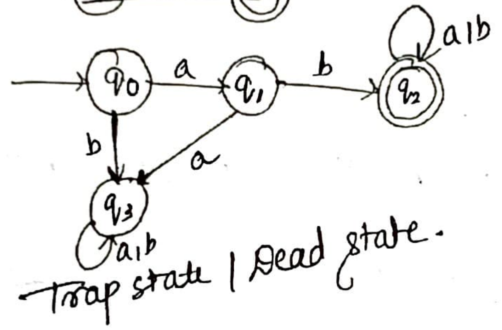
(4) Construct a DFA of strings a & b such that sequence should begin with ab.

⇒ (1) ab

$L = \{ab, ababba, abba, abbba, \dots\}$

(2) $\Sigma = \{a, b\}$

~~(3)~~ 

(3) 

$Q = \{q_0, q_1, q_2, q_3\}$

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

$q_0 \rightarrow$ start state

$q_2 \rightarrow$ final state

δ	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_3	q_2
$* q_2$	q_2	q_2
q_3	q_3	q_3

(5) ~~Construe~~ Construct DFA of strings of a & b such that each & every string should end with abb.

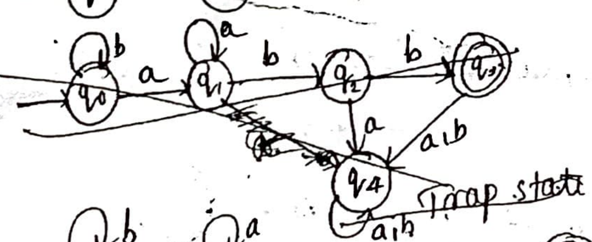
(1) abb

$L = \{abb, ababb, aabb, babb, aababb, bbabb, aaabb, \dots\}$

(2) $\Sigma = \{a, b\}$

~~(3)~~ 

(3) 

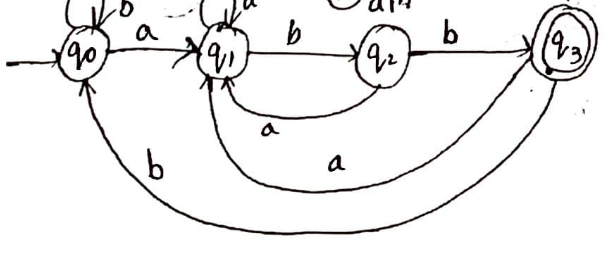
(4) 

$Q = \{q_0, q_1, q_2, q_3\}$

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

$q_0 \rightarrow$ start state.

$q_3 \rightarrow$ final state

(4) 

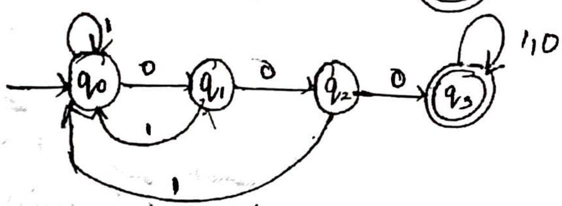
Scanned by CamScanner

(6) Construct a DFA of strings 0's & 1's. such that it should have three consecutive zeros.

⇒ L = {000, 100011, 10000, 00011 - - - }

(2) Σ = {0, 1}   (1) 000

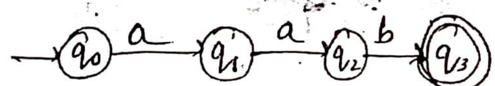
(3) 

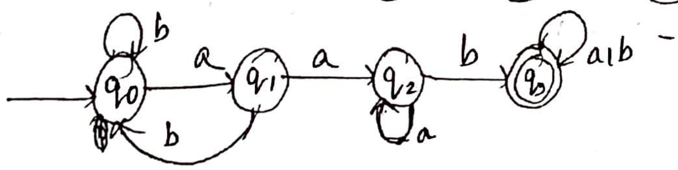
(4) 

(7) Write a DFA of strings a's & b's having a substring aab.

⇒ L { aab, baabaa, aaabb, aaaabb - - - }

(1) aab

(2) Σ = {a, b}   (3) 

(4) 

Scanned by CamScanner

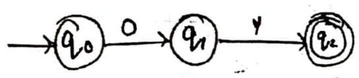
δ	a	b
$\rightarrow q_0$		
q_1		
q_2		
$*q_3$		

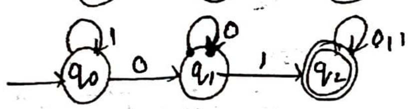
⑧ Construct a DFA to accept strings of 0's & 1's such that somewhere in the sequence 01 should appear in the string.

$\Rightarrow L = \{01, 1011, 00101, 110100, 000101, 0011 \ldots\}$

① 01

② $\Sigma = \{0,1\}$

③ 

④ 

δ	0	1
q_0		
q_1		
q_2		

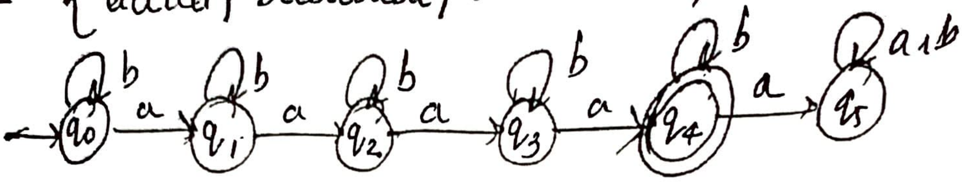
⑨ Construct a DFA to accept strings of 0's & 1's such that strings should not have a substring 01.

* NOTE:- If not keyword appears
change final state → Non final state
Nonfinal state → final state

Scanned by CamScanner

(10) Construct DFA ~~of str~~ having 4 a's for the input alphabet $\Sigma = \{a, b\}$.

⇒ $L = \{aaaa, babababa, aabbaabb, abababa \ldots\}$

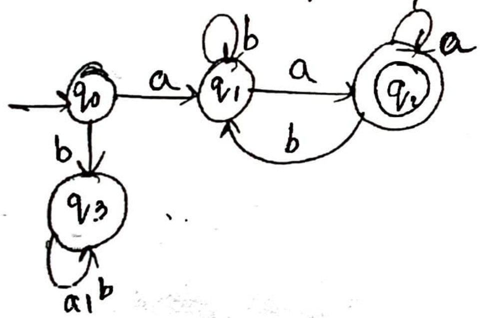


(11) Construct a DFA to accept the following language

$L = \{awa \mid w \in \{a,b\}^*\}$.

→ Minimum string :→ aa

$L = \{aa, aba, aaa, abaa \ldots\}$



(12) a's & b's having atmost 3 a's.

⇒ ~~L = {aaa, baaa, abaa~~

$L = \{\epsilon, a, aa, aaa, ao \ldots\}$

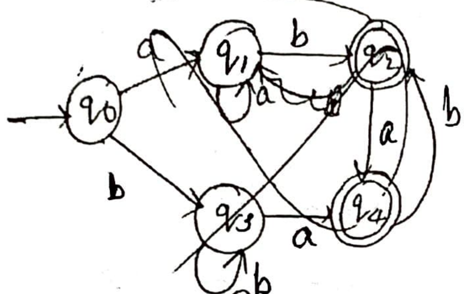
Here they are made final states because to accept zero a's, 1 a, 2a ---

(13) $L = \{ w \in a,b \mid w(ab + ba) \}$

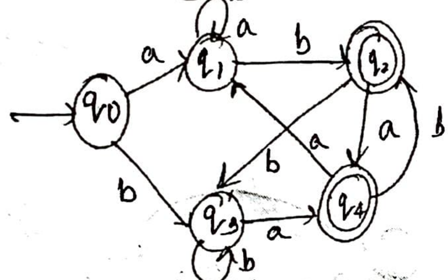
or

⇒ This implies the string should either end with ab or ba

$L = \{ aab, aba, aab, bab, bbab, aaba \ldots \}$

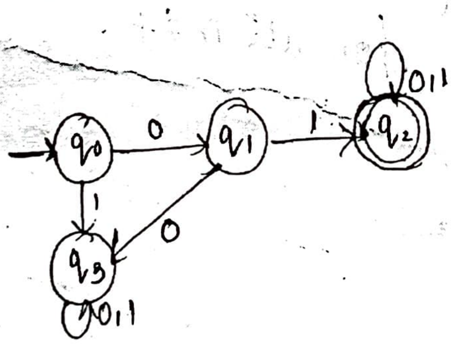


$\delta(q_1, a) =$



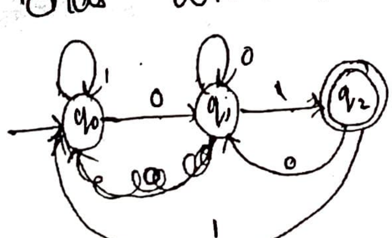
(14) Construct a DFA of strings 0's & 1's which begins with 01.

⇒ $L \{ 01, 0100, 0110 \ldots \}$



(15) Ends with 01.

⇒

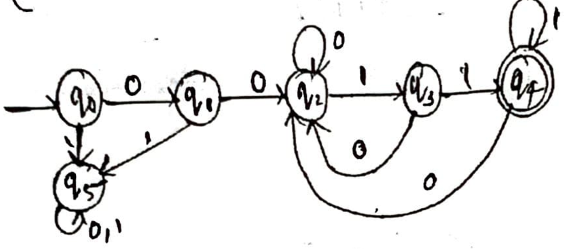


(16) Construct DFA for following languages :-

(1) $L = \{w \in (0+1)^* \mid 00(0+1)^*11\}$

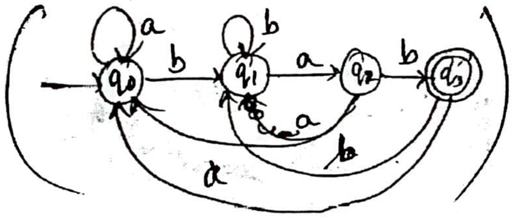
⇒ Begins with atleast two zeros & ends with atleast two ones.

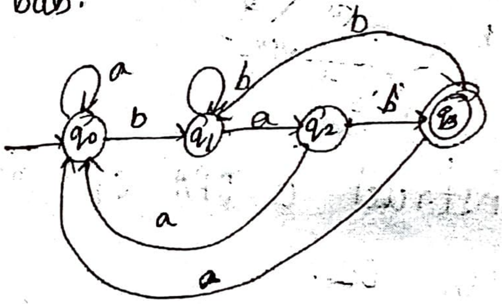
(* indicates zero or any times)



(2) $L = \{w \in \{a,b\} \mid w bab\}$

⇒ strings ending with bab.





(17) DFA's of a's & b's which accepts all the strings of length a^n for all $n \neq 4$.

⇒

(18) Multiples of two of a's $\Sigma = \{a, b\}$

→

① DIVISIBLE BY K PROBLEMS :→

$\delta(q_i, a) = q_j$ $\qquad Q \times \Sigma = Q$

$j = (r * i + d) \bmod k$

r :→ radix input

i :→ remainder obtained after dividing by k.

d :→ digits

k :→ divisor

for binary radix input :→ 2

decimal radix input :→ 10

" :→ 3, {0, 1, 2}

NOTE :→ start state & final state will be same.

① Construct a DFA of strings 0's & 1's where each strings are represented as binary numbers which are divisible by 5.

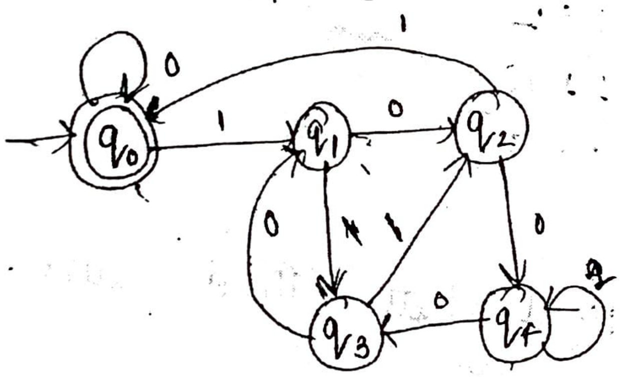
⇒ $r = 2$

$i = 0, 1, 2, 3, 4$

$d = \{0, 1\}$

$k = 5$

	d	$(r * i + d) \bmod k = j$	$\delta(q_i, a) = q_j$
$i = 0$	0	$q_0 = (2*0+0) \bmod 5 = 0$	$\delta(q_0, 0) = q_0$
	1	$q_0 = (2*0+1) \bmod 5 = 1$	$\delta(q_0, 1) = q_1$
$i = 1$	0	$q_1 = (2*1+0) \bmod 5 = 2$	$\delta(q_1, 0) = q_2$
	1	$q_1 = (2*1+1) \bmod 5 = 3$	$\delta(q_1, 1) = q_3$
$i = 2$	0	$q_2 = (2*2+0) \bmod 5 = 4$	$\delta(q_2, 0) = q_4$
	1	$q_2 = (2*2+1) \bmod 5 = 0$	$\delta(q_2, 1) = q_0$
$i = 3$	0	$q_3 = (2*3+0) \bmod 5 = 1$	$\delta(q_3, 0) = q_1$
	1	$q_3 = (2*3+1) \bmod 5 = 2$	$\delta(q_3, 1) = q_2$
$i = 4$	0	$q_4 = (2 \times 4+0) \bmod 5 = 3$	$\delta(q_4, 0) = q_3$
	1	$q_4 = (2 \times 4+1) \bmod 5 = 4$	$\delta(q_4, 1) = q_4$



② Construct a DFA of strings of decimal numbers that are divisible by 3.

$\Rightarrow$ $r = 10$

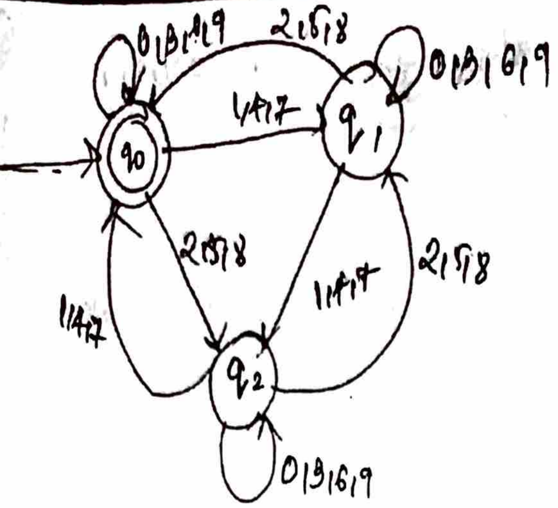
$i = 0, 1, 2$

$d = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$k = 3$

	d	$j = (r * i + d) \bmod k$	$\delta(q_i, a) = q_j$
$i = 0$	0	$q_0 = (10 * 0 + 0) \bmod 3 = 0$	$\delta(q_0, 0) = q_0$
	1	$q_0 = (10 * 0 + 1) \bmod 3 = 1$	$\delta(q_0, 1) = q_1$
	2	$q_0 = (10 * 0 + 2) \bmod 3 = 2$	$\delta(q_0, 2) = q_2$
	3	q_0	
	4	q_0	
	5	q_0	
	6	q_0	
	7	q_0	
	8	q_0	
	9	q_0	

	d	$(r * i + d) \bmod k = j$
$i = 0$	(0, 3, 6, 9)	$(10 * 0 + 0) \bmod 3 = q_0$
	(1, 4, 7)	$(10 * 0 + 1) \bmod 3 = q_1$
	(2, 5, 8)	$(10 * 0 + 2) \bmod 3 = q_2$
$i = 1$	(0, 3, 6, 9)	$(10 * 1 + 0) \bmod 3 = q_1$
	(1, 4, 7)	$(10 * 1 + 1) \bmod 3 = q_2$
	(2, 5, 8)	$(10 * 1 + 2) \bmod 3 = q_0$
$i = 2$	(0, 3, 6, 9)	$(10 * 2 + 0) \bmod 3 = q_2$
	(1, 4, 7)	$(10 * 2 + 1) \bmod 3 = q_0$
	(2, 5, 8)	$(10 * 2 + 2) \bmod 3 = q_1$



) Construct a DFA of strings of 0's & 1's which are divisible by 5 such that DFA should not accept binary number 0000.

(7) Construct a DFA of string of 0's, 1's & 2's which are represented as combination of 0's, 1's & 2's divisible by 4.

Divisibility problem :-

① divisible by 3 / decimal number.

⇒ Remainders are 0, 1, 2

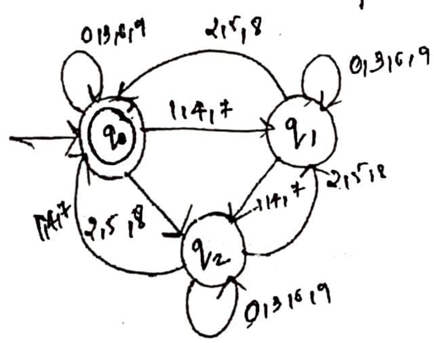
0 → q_0

1 → q_1

2 → q_2

$d = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.$

δ	(0, 3, 6, 9)	(1, 4, 7)	(2, 5, 8)
q_0	q_0	q_1	q_2
q_1	q_1	q_2	q_0
q_2	q_2	q_0	q_1



② Construct a DFA all strings over $\Sigma = \{0, 1\}$ ends with 100.

⇒ q_0 is start state or initial state. q_1 is a state ending with one q_2 is a state ending with 10, q_3 100.

$q_1 \rightarrow 1$ $q_2 \rightarrow 10$, $q_3 \rightarrow 100$

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

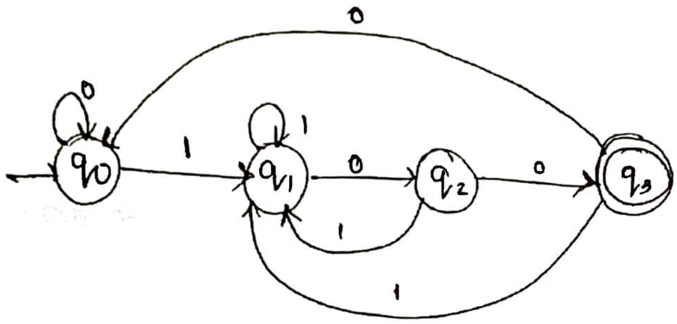
$Q = \{q_0, q_1, q_2, q_3\}$

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

$q_0 \rightarrow SS$ $q_3 \rightarrow FS$

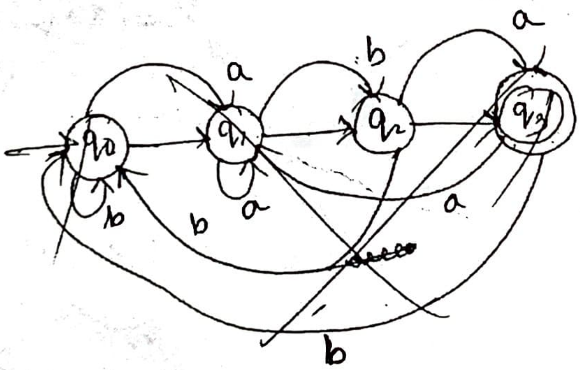
Scanned by CamScanner

	δ	0	1	
ε	$\rightarrow q_0$	q_0	q_1	~~0~~0, ~~0~~1
1	q_1	q_2	q_1	~~0~~10, ~~0~~11
10	q_2	q_3	q_1	~~1~~100, ~~0~~101
100	$* q_3$	q_0	q_1	~~1~~000, 1~~0~~01



② Ending with aba.

	δ	a	b	
ε	$\rightarrow q_0$	q_1	q_0	~~ε~~a, ~~ε~~b
a	q_1	q_1	q_2	~~a~~a, ab
ab	q_2	q_3	q_0	aba, ~~ab~~b
aba	$* q_3$	q_1	q_2	ab~~a~~a, ~~ab~~ab



Scanned by CamScanner

(3) Construct a DFA of strings of 'a's & 'b's starting with aba.

Scanned by CamScanner

NFA :→ Non deterministic finite automata

$M_{NFA} = \{Q, \Sigma, \delta, q_0, F\}$

$\delta : \Sigma \times Q = Q$ (DFA)

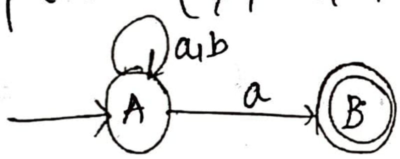
$\delta : Q \times \Sigma = 2^Q$ (power set) ⟶ NFA.

Q is set of finite states.

power set ⟶ set of all subsets.

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

$P(A) = \{\phi, a, b, c, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$



$\delta(A, a) = A$

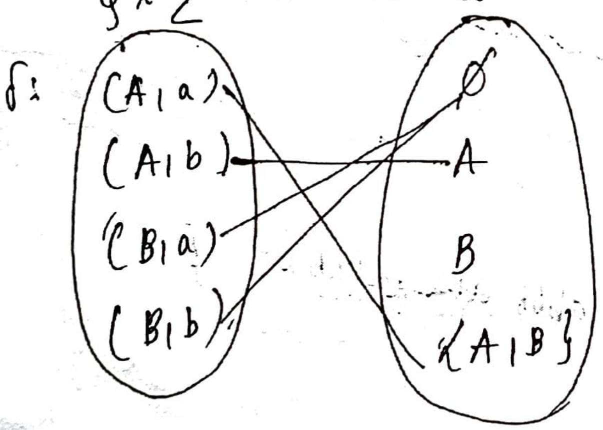
$\delta(A, b) = A$

$\delta(B, a) =$

$\delta(B, b) =$

δ	a	b
→A	$\{A, B\}$	A
*B	ϕ	ϕ

$Q \times \Sigma \quad = \quad 2^Q$



~~Examples of NFA~~

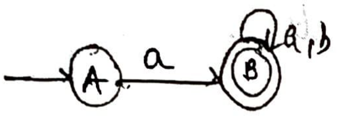
Scanned by CamScanner

EXAMPLES :->

(1) Obtain an NFA over strings a,b where all the strings starts with a.

→ (1) L = { starts with a }

L = { a, ab, abb, aabb . . . }

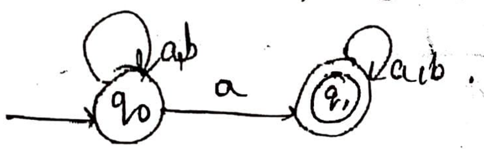


$\delta(A, a) \rightarrow B$

$\delta(B, b) \rightarrow B$

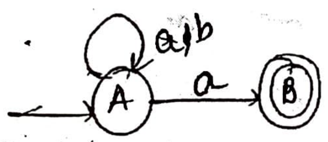
(2) L = { containing a } over a,b.

⇒ ~~L = { aba, a~~ L = { aba, bab, bba, ab, ba . . . }



(3) L = { ends with a } over a,b

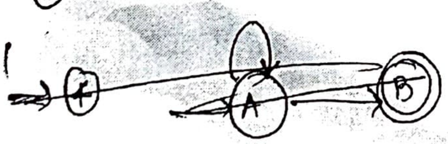
⇒



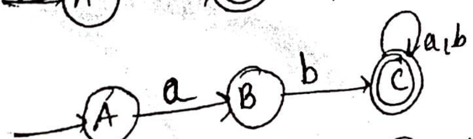
(4) L = { starting with ab }

(5) L = { containing ab }

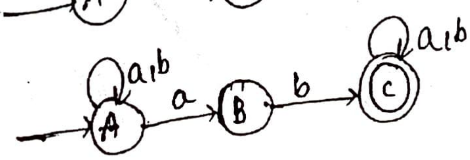
(6) L = { ends with ab }



(4) L = { abbb, aba, abbb . . . }

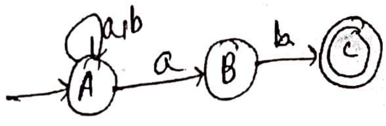


(5)



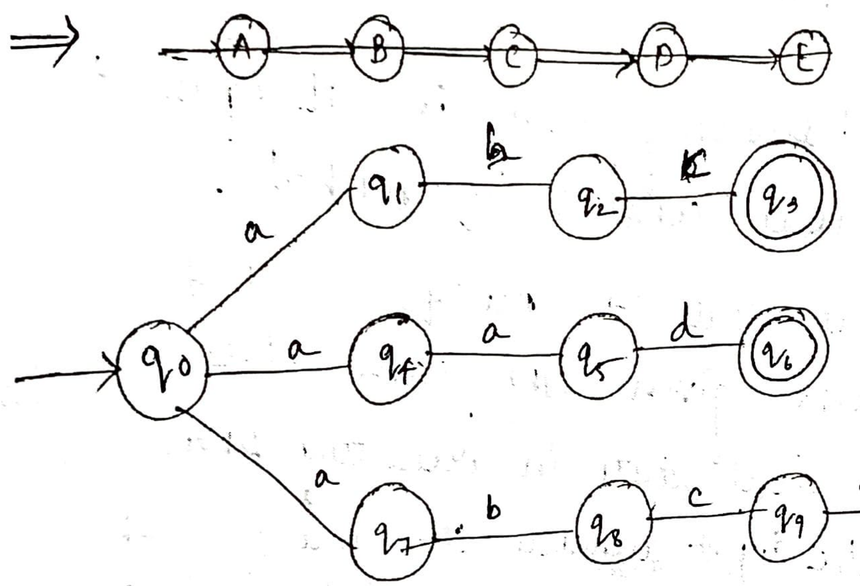
L = { abab, abbb, bbabb }

(6)



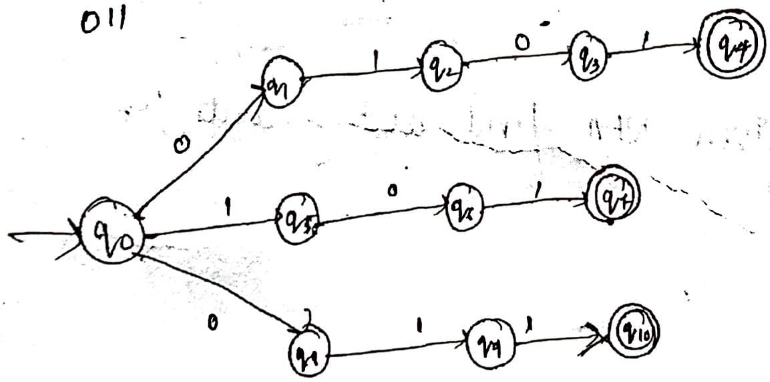
(7) In order to accept the following set of strings

abc
aad
abcd



(8) Construct NFA for given set of strings over 0,1 in order to accept

0101
101
011



NOTE :→ While processing the string or any sequence in a result or in a set of states atleast one state should be accepting state or final state then the string or sequence its accepted by NFA

Scanned by CamScanner