9/05/22

Automata theory (TOC)

Alphabet :- Alphabet is a set of symbol(s) which is represented by Σ.

String :- String is a sequence of symbols from Σ. It is represented by w.

Language :- Language is a collection of valid strings. It can be two types -

① **finite** :- strings are countable.

eg :- L1 = set of all string of length 2, $\Sigma = \{a, b\}$

$\{aa, ab, bb, ba\}$

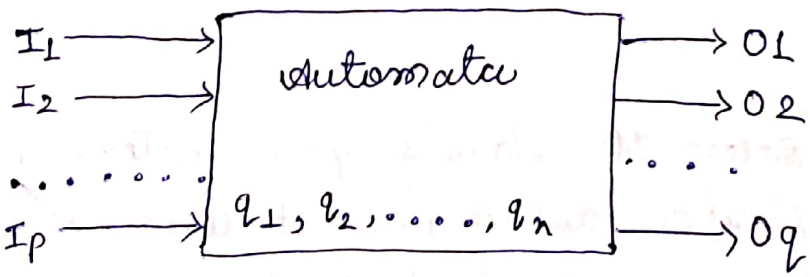
⑪ **Infinite** :- strings are uncountable

eg :- L2 = set of all strings starting with a -

$\{a, ab, aab, aabb, \ldots\ldots\}$

Automata :- An automaton is defined as a system where energy, materials and information are transform, transmitted and used for performing some function without direct participation of human.

eg :- An automatic machine tool, text processors, compilers etc.



[Fig : Model of discrete automata]

Scanned with CamScanner

Inputs: At each of discrete instant of time $T_1, T_2, \ldots, T_p$ input values $I_1, I_2, \ldots, I_p$ each of which one take finite no. of fixed values from the input alphabet Σ, are applied to input side of model.

Output:- $O_1, O_2, \ldots O_q$ are outputs of the model each of which can take finite no. of fixed value from an output O.

states:- At any instant of time automation can be in one of states $q_1, q_2, \ldots q_n$.

state relation:- The next state of an automation at any instant of time is determined by present state and present input.

Output relation:- Output is related to either state only or to both the input and the state.

At any instant of time the automation is in some state. On reading an input symbol the ~~automation~~ automaton moves to next state, which is given by state relation.

11/05/22

Finite Automata :- It is the restricted model of automatic machines.

FA is an abstract model of computer system. It has only finite no. of states hence the finite automata can count only a finite no. of input seneries.

- DFA:- ~~DFA~~ It stands for deterministic finite automata. It can be represented by a quin (5) tuple $(Q, \Sigma, \delta, q_0, F)$

Scanned with CamScanner

where,

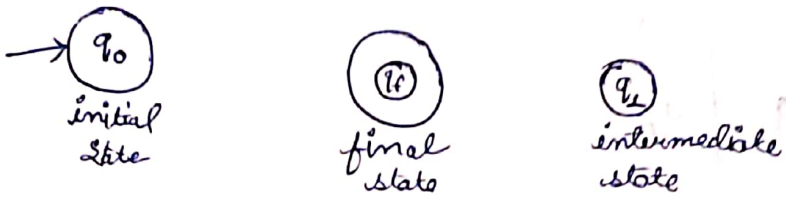
Q is finite non empty set of states

Σ is finite non empty set of input alphabets

δ is function which maps $Q \times \Sigma \to Q$ and is called a transmission transition function.

q_0 ~~below~~ $q_0 \in Q$, q_0 is initial state

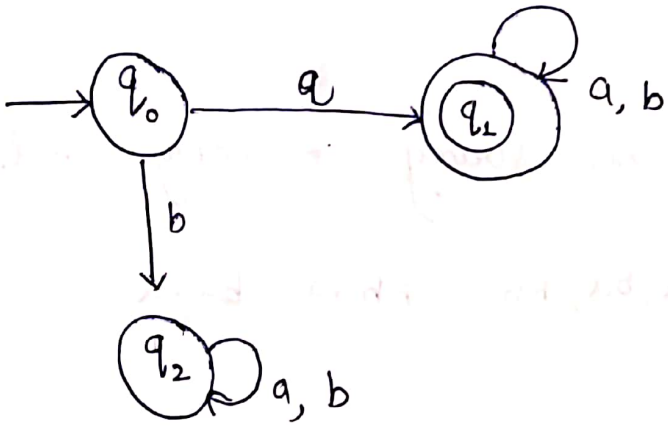
F is subset of Q ($F \subseteq Q$) is the set of final states



17/05/22

Q. Construct a DFA for all strings starting with a over $\Sigma = \{a, b\}$.

Sol



Transition table

state	input	
	a	b
$\to q_0$	q_1	q_2
q_1	q_1	q_1
q_2	q_2	q_2

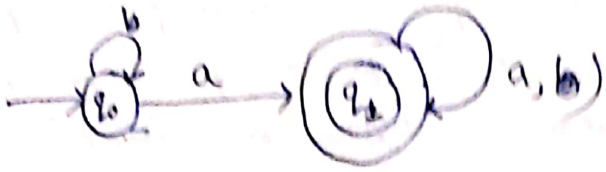
Scanned with CamScanner

Q. Construct a DFA for all string containing

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

a, aa, ab, abb

∴

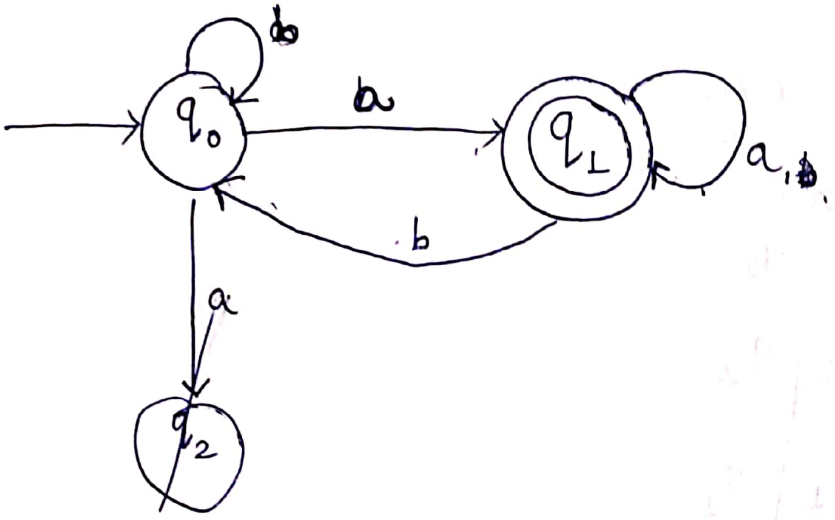


Transition table

state	input a	input b
→ q_0	q_1	q_0
(q_1)	q_1	q_1

Q. Contruct a DFA for all string ending with a.

aba, ba, bba, bbaa, baaa abα



Scanned with CamScanner

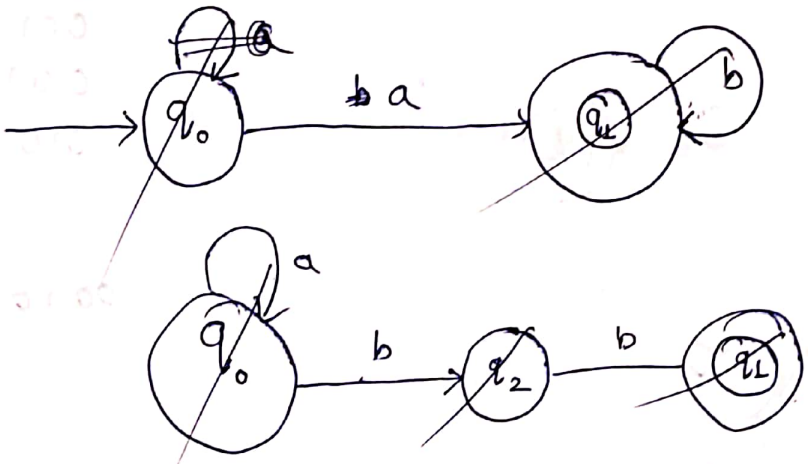
Transition table

state	input a	input b
$\rightarrow q_0$	q_1	q_0
(q_1)	q_1	q_{dead}

18/05/22

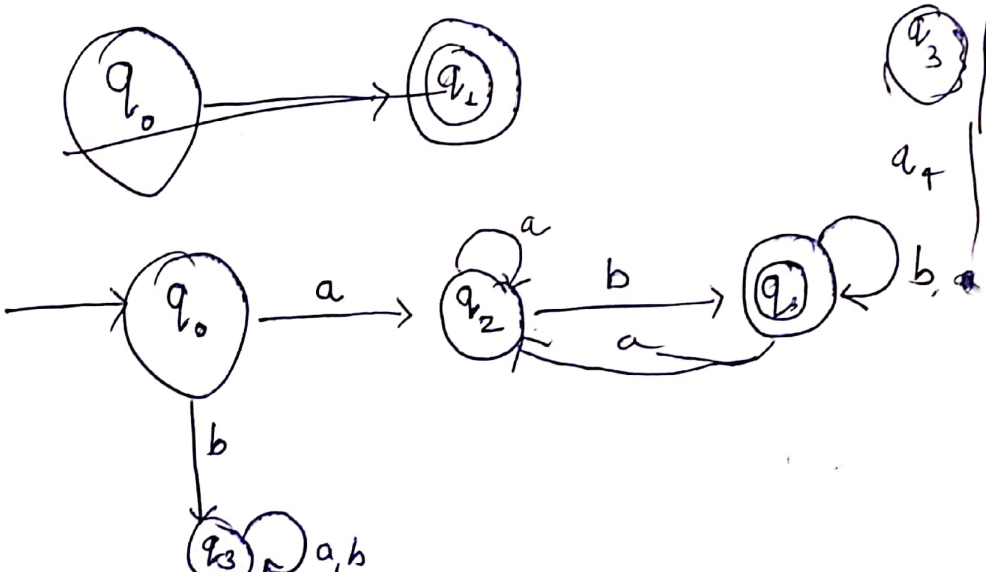
Q. Construct a DFA for all string starting with a and ending with b.

ab, aab, abb



Transition table

state	input a	input b
$\rightarrow q_0$	q_1	q_2
q_2	q_2	q_4
(q_3)	q_3	q_3
q_4	q_4	q_4



Scanned with CamScanner

Q. NFA :- It stands for non-deterministic finite automata. It is quin tuple $(Q, \Sigma, \delta, q_0, F)$

where,

Q is finite non-~~infi~~ empty set of states

Σ finite non-empty set of input alphabet

δ is func transition function $\boxed{Q \times \Sigma \to 2^Q}$

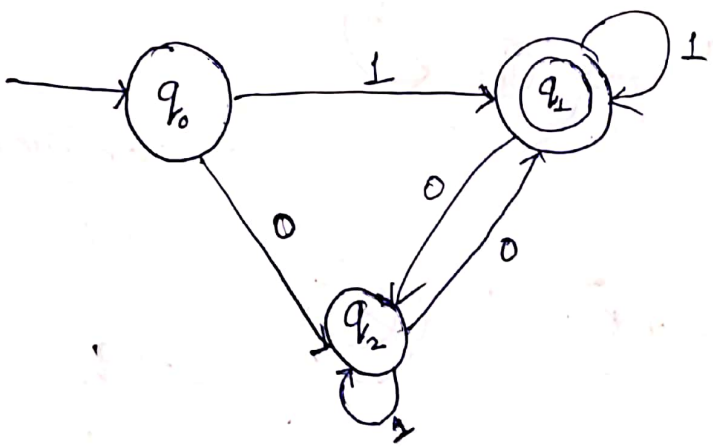
$q_0 = q_0 \in Q$, q_0 is initial state

F is subset of Q ($F \subseteq Q$) is the set of final states

Q. Construct a NFA to accept ~~for all~~ string containing even no. of 0's and any no. 1's $\Sigma = \{0, 1\}$

~~0, 11~~

~~Sol~~



1
001
0011...
01011...
0110
00100

Scanned with CamScanner

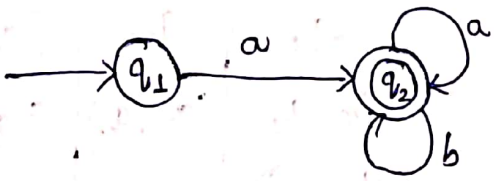
* Equivalance of DFA and NFA :-

For every NFA there exist a DFA which simulates the behaviour of NFA. Alternatively if "L" is in set accepted by NFA, then there exist a DFA which also accepts "L". A DFA can simulate the behaviour of NFA by increasing the number of states i.e. a DFA $(Q, \Sigma, \delta, q_0, F)$ can be viewed as NFA $(Q, \Sigma, \delta, q_0, F)$ by defining.

$$\delta'(q, a) = \{\delta(q, a)\}$$

* Conversion of NFA to DFA

Q construct NFA for all strings starting with 'a' over $\Sigma = \{a, b\}$.

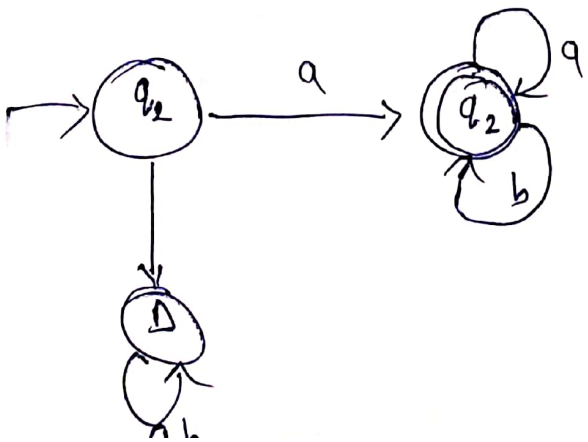


transition table :-

state	a	b
→ q_1	q_2	-/D
((q_2))	q_2	q_2

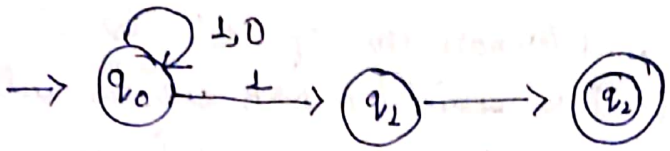
transition table for DFA :

state	a	b
→ q_1	q_2	D
((q_2))	q_2	q_2
D	D	D



Scanned with CamScanner

Q. construct NFA which accepts all the string that end with '10'

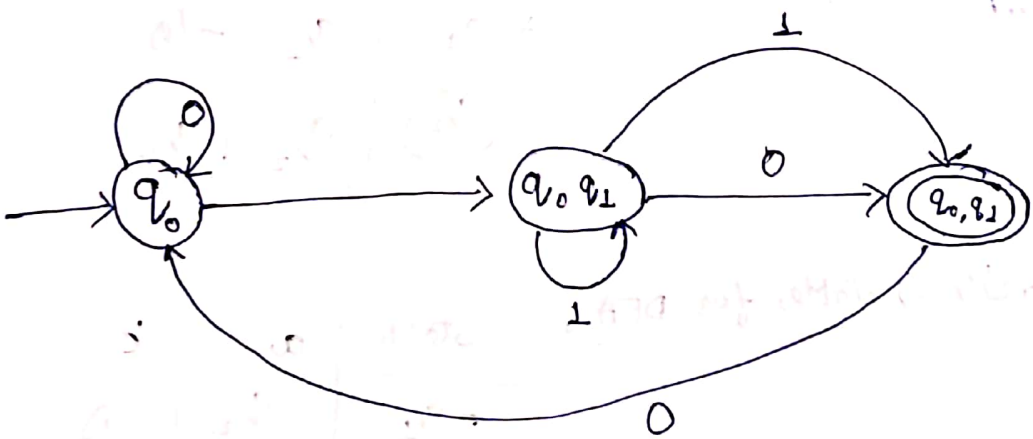


Transition table :

state	0	1
→ q_0	q_0	$\{q_0, q_1\}$
q_1	q_2	ϕ
(q_2)	ϕ	ϕ

Transition table for DFA:

state	0	1
→ q_0	q_0	$\{q_0, q_1\}$
q_0, q_1	(q_0, q_2)	$\{q_0, q_1\}$
(q_0, q_1)	q_0	$\{q_0, q_1\}$



Scanned with CamScanner

* NFA with ε Move / Move

NFA with null transition is denoted by 5 tupples

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

Q = set of states

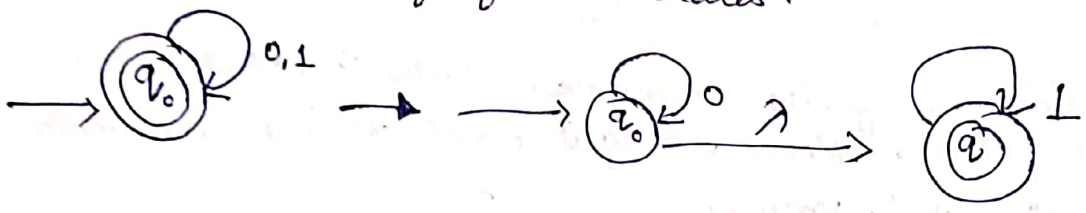
Σ = set of input alphabets

δ = transition function

$$Q \times (\Sigma \cup \{\lambda\}) \longrightarrow 2^Q$$

q_0 = initial state

F = set of final states.



* Transition system containing λ move:

Transition system can be generated by permitting λ transitions or λ moves which are associated with a null symbol λ. These transitions can occurs when no input is applied. But it is possible to con convert transition systems with λ move into can equivalent transition system without λ move.

Suppose we want to replace λ move from vertex v_1 to v_2 then we proceed as follows:

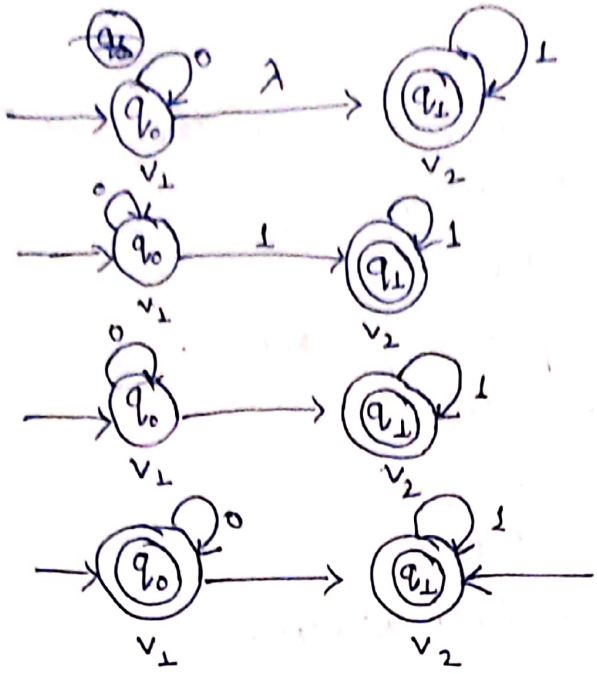
step 1: Find all the edges starting from v_2

step 2: Duplicate all these edges starting from v_1 without changing the edge labels.

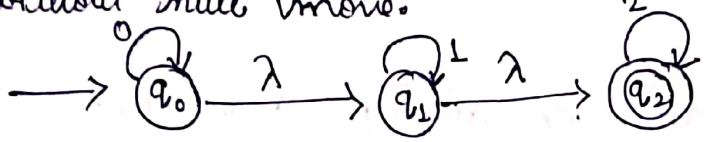
step 3: If v_1 is an initial state make v_2 also as initial state.

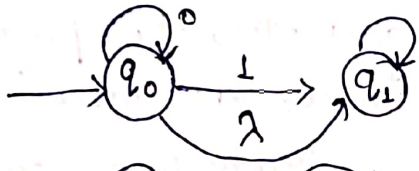
step 4: If v_2 is final state make v_1 final state.

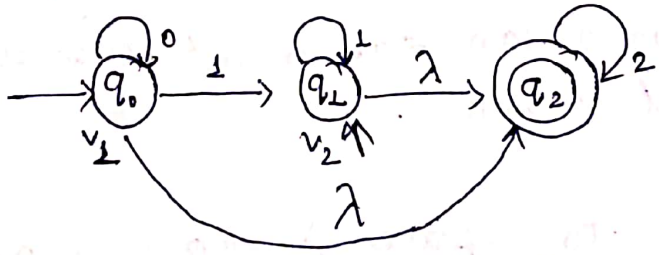
Scanned with CamScanner

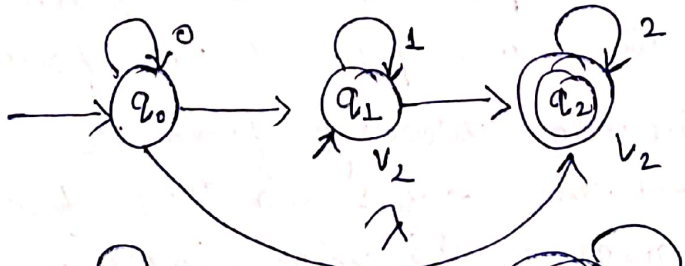


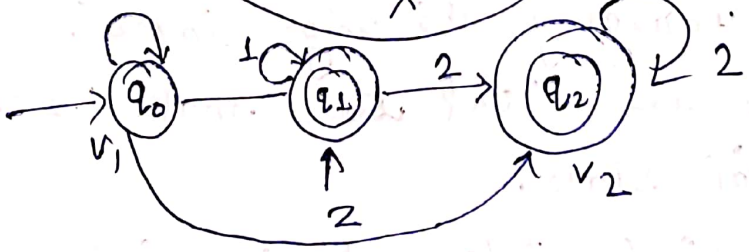
Q. Consider a finite automaton with λ moves given as follows. Obtain an equivalent finite automata without null moves.



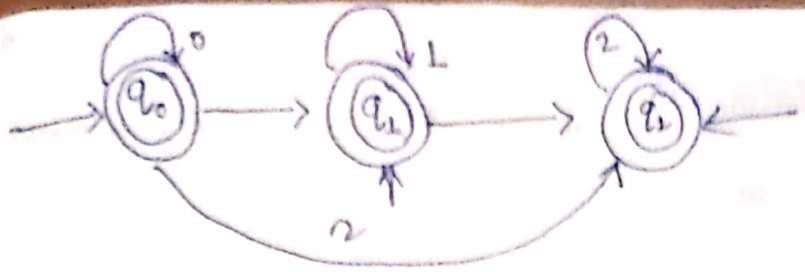








Scanned with CamScanner



* Regular expression:

Formal recursive defination of regular expression over Σ is as follows:-

(i) Any terminal symbol, λ and ϕ are regular expression.

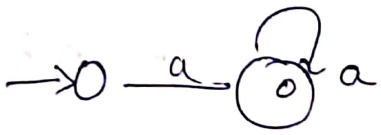
(ii) Union of two regular expression R1 & R2 written as R1+R2 is also a regular exp.

(iii) The concatenation of regular expression R1 & R2 written as $R_1 R_2$ is also a regular expression.

(iv) The iteration of a regular expression written as R^* is also a regular expression.

(v) If R is a regular expression then (R) is also a regular expression.

(vi) A regular expression over Σ are precisly those obtained recursively by application of the rules (i) - (iv) once or several times



Scanned with CamScanner

05/08/2

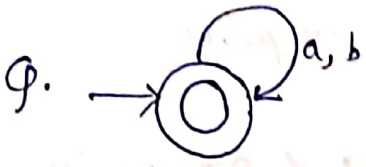
12.

$(\gamma\perp)^*$ Kleene closure (diff. $(\gamma\perp)^*$ include λ i.e zero ∞ repeat

$(\gamma\perp)^+$ positive closure

$(a+b) - 2\,(a,b)$

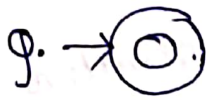
$(a.b) - 1\,(ab)$



$\Rightarrow (a+b)^*$

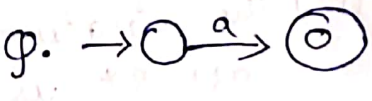
$(ab)^*$

~~$(a)^*(b)^*$~~

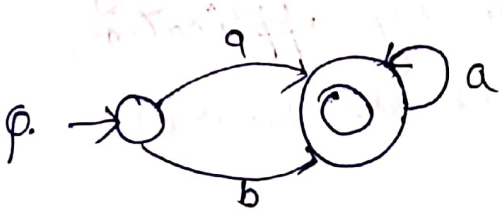


$= \lambda$

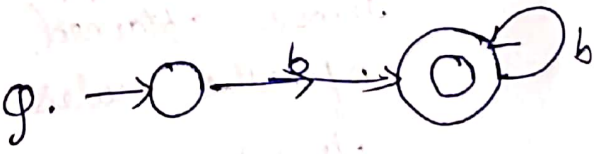
$= \phi$



$= a$



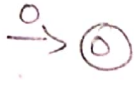
$= (a+b)(a)^*$



$= b\,b^*$

01	00
001	100
011	1100
	0100

$0(0+1)^*1$



1^+0^* $(0+1)^*0$

7

Scanned with CamScanner

(i) set of all of strings 0's and 1's ending with 00.

$\Rightarrow (0+1)^*00$

(ii) set of all string 0's and 1's begining with zero and ending with 1.

$\Rightarrow 0(0+1)^*1$

(iii) set of all string of 0's and 1's that end with 3 consicutive 1's.

$\Rightarrow (0+1)^*111$

(iv) set of all string of 0's and 1's containing atleast one 0.

$\Rightarrow (1+0)^*0(1+0)^*$ $\qquad (1)^*(0)^{0+}$

$(1)^*(0)^+$

$(1+0)^*0^+$

0
01
10
010
101

06/08/22

Arden's Theorem :- Let P and Q be two regular expression over Σ. if P does not contain λ, then the following equation in R, $R = Q + RP$ as a unique solution given by

$R = QP^*$

$R = Q + RP$

$R = Q + (Q+RP)P$

$R = Q + QP + RP^2$

$R = Q + QP + (Q+RP)P^2$

$R = Q + QP + QP^2 + RP^3$

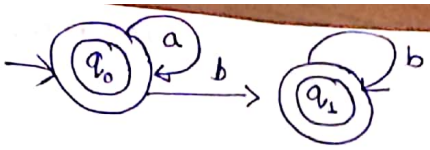
$= Q + QP + QP^2 + (Q+RP)P^3$

$R = Q + QP + QP^2 + QP^3 + RP^4$

$R = Q + P^3$

Scanned with CamScanner

Q.



$q_0 = \lambda + q_0 a$

$q_0 = \lambda a^*$ $\quad [\because \lambda R = R]$

$q_0 = a^*$

$q_1 = q_0 b + q_1 b$

$q_1 = a^* b + q_1 b$

$q_1 = a^* b b^*$

now,

$q_0 + q_1$

$= a^* + a^* b b^*$

$= a^* (\lambda + b b^*)$ $\quad [\because \lambda + b b^* = b^*]$

$= a^* b^*$

Q.

Scanned with CamScanner

Identities of regular expression

(i) $\phi + R = R$

(ii) $\phi R = R\phi = \phi$

(iii) $\lambda R = R\lambda = R$

(iv) $\lambda^* = \lambda$ and $\phi^* = \lambda$

(v) $R + R = R$

(vi) $R^* R^* = R^*$

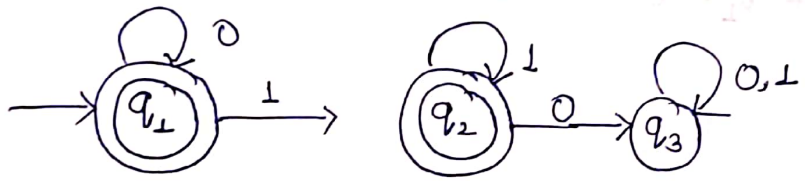
(vii) $RR^* = R^* R$

(viii) $\lambda + RR^* = R^*$

(ix) $(PQ)^* P = P(QP)^*$

(x) $(P+Q)^* = (P^* Q^*)^*$

Q.



$q_1 = \lambda + q_1 0$

$= \lambda 0^*$

$= 0^*$

$q_2 = q_1 1 + q_2 1$

$= 0^* 1 + q_2 1$

$= 0^* 1 1^*$

$q_3 = q_2 0 + q_3 0 + q_3 1$

$= 0^* 1 1^* + q_3 0 + q_3 1$

$= 0^* 1 1^* 0^*$

~~$q_3 = \phi$~~

now, $q_1 + q_2 + q_3$

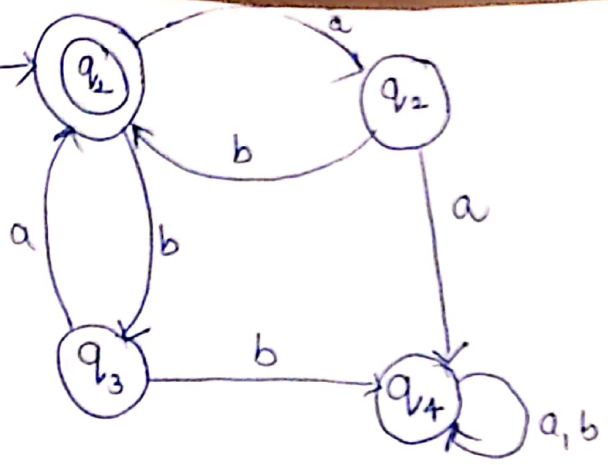
$= 0^* + 0^* 1 1^* + \phi$

$= 0^* (\lambda + 1 1^*) + \phi$

$= 0^* 1^* + \phi$

Scanned with CamScanner

Q.



$$q_1 = \lambda + q_2 b + q_3 a +$$

$$q_2 = q_1 a$$

$$q_3 = q_1 b$$

$$q_4 = q_2 a + q_3 b + q_4 a + q_4 b$$

$$q_1 = \lambda + q_1 ab + q_1 ba$$

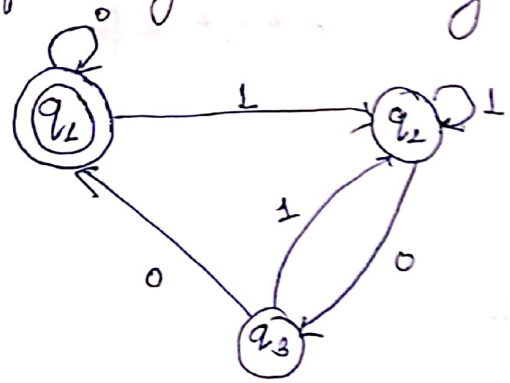
$$q_1 = \lambda + q_1 ab$$

$$R = Q + RP$$

$$q_1 = \lambda ab$$

$$\boxed{q_1 = ab}$$

Q. Construct a regular expression corresponding to the following state diagram.

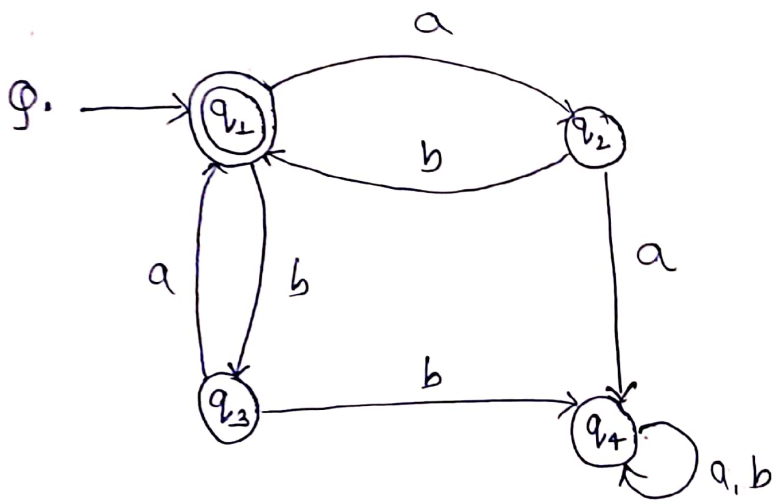


$$q_1 = \lambda + q_1 0 + q_3 0$$

$$q_2 = q_1 1 + q_2 1 + q_3 1$$

$$q_3 = q_2 0$$

Scanned with CamScanner



$q_1 = \lambda + q_2 b + q_3 a$ ——— (i)

$q_2 = q_1 a$

$q_3 = q_1 b$

$q_4 =$

Scanned with CamScanner

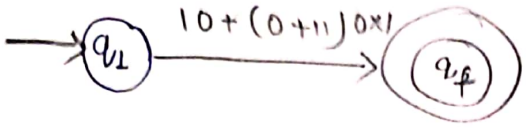
22/08/22

RE to FA

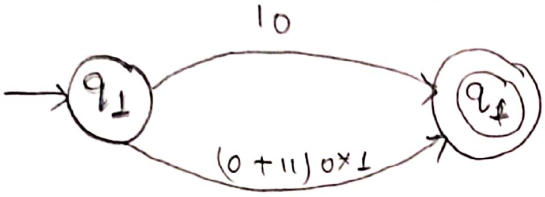
Q construct a DFA reduced state equivalent to regular expression.

$10+(0+11)0^*1$

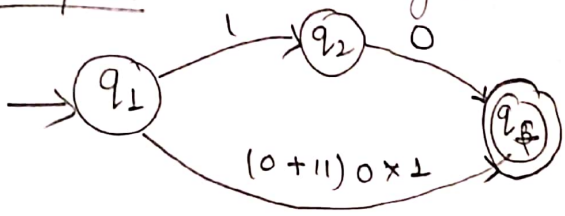
step 1:



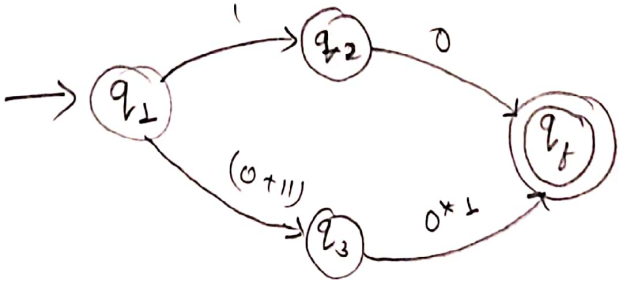
step 2:- eliminating + symbol from $10+(0+11)0^*1$



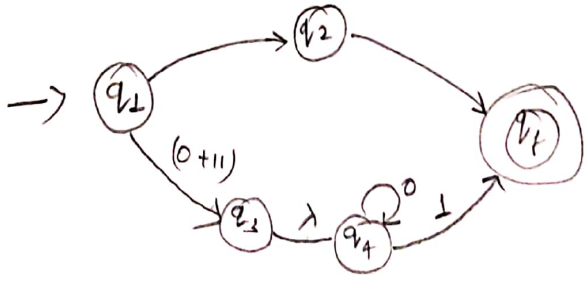
step 3:- eliminating concatination from 10



step 4: eliminating concatinati from $(0+11)0^*1$

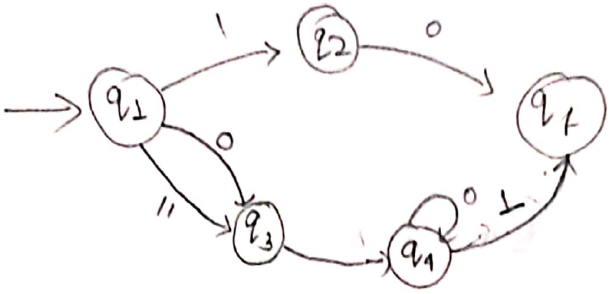


step 5: eliminating * from 0^*1

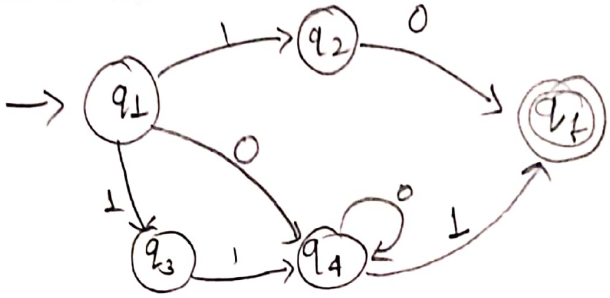


Scanned with CamScanner

step 6:- eliminating + from (0+11)



step 7:-



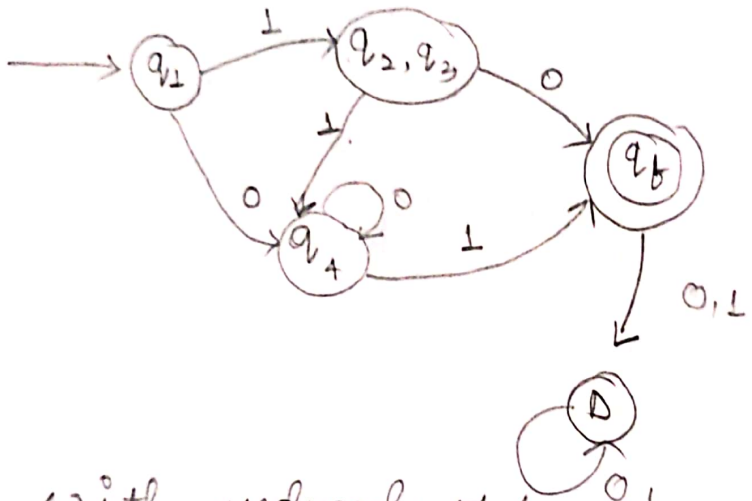
Transition table of NFA

states	0	1
→ q_1	q_4	q_2, q_3
q_2	q_f	ϕ
q_3	ϕ	q_4
q_4	q_4	q_f
(q_f)	ϕ	ϕ

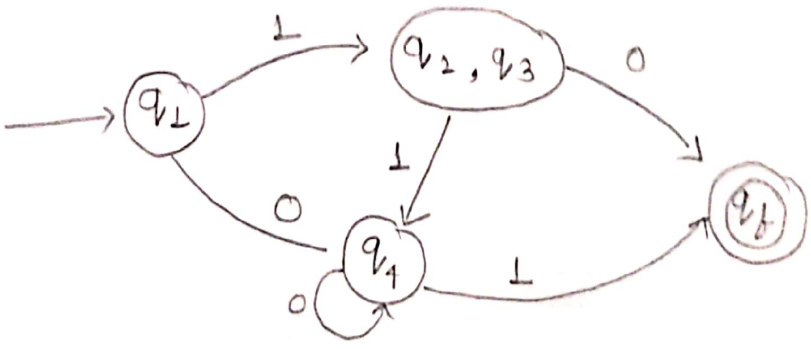
Scanned with CamScanner

Transition table of DFA

states	0	1
→ q_1	q_4	q_2, q_3
q_4	q_4	q_f
(q_2, q_3)	q_f	q_4
q_f	D	D
D	D	D



* With reduced state

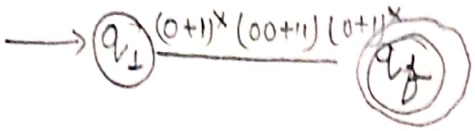


Scanned with CamScanner

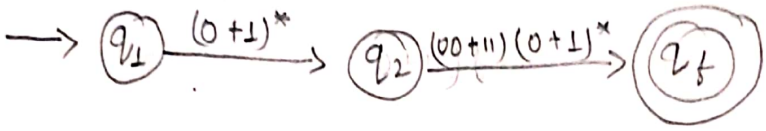
Q. construct the finite automata equivalent to regular expression

$(0+1)^*\ (00+11)\ (0+1)^*$

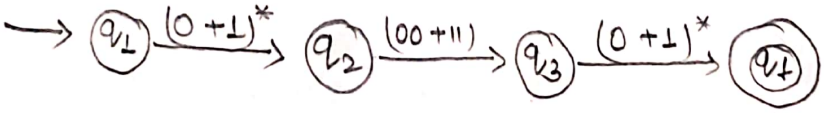
⇒ step 1 :-



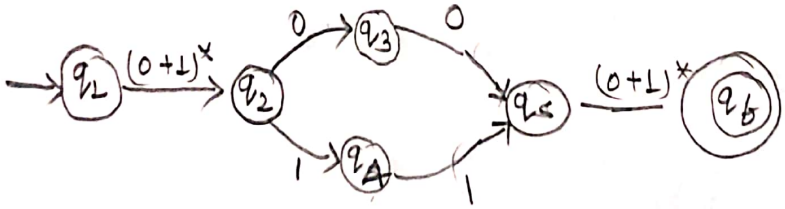
step 2 :-



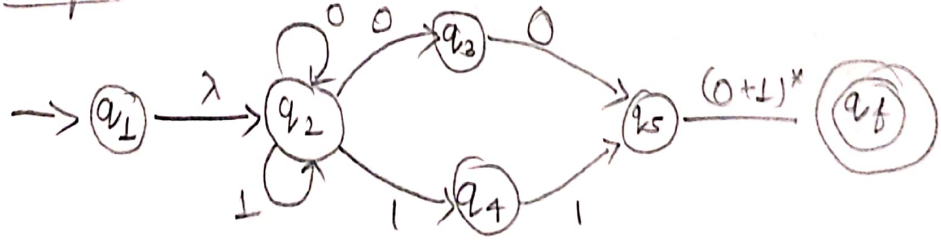
step 3 :-



step 4 :-

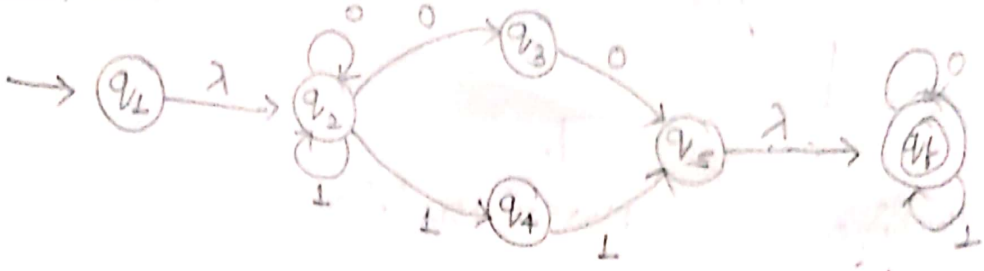


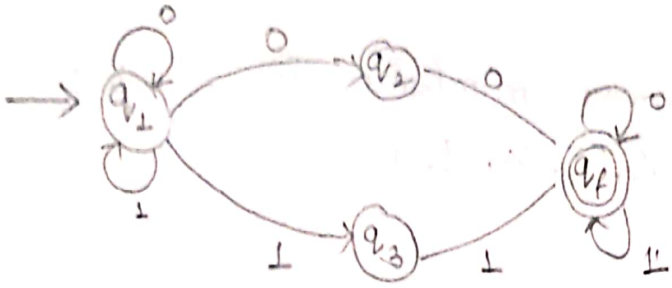
step 5 :-



Scanned with CamScanner

Step 6 :-



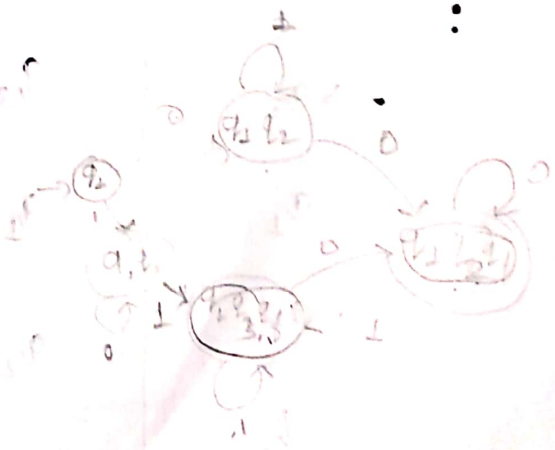


Transition table with NFA

states	0	1
→ q_1	q_1, q_2	q_1, q_3
q_2	q_f	—
q_3	—	q_f
q_f	q_f	q_f

Transition table with DFA

states	0	1
→ q_1	q_1, q_2	q_1, q_3
$q_1 q_2$	$q_1 q_2 q_f$	q_1, q_3
$q_1 q_3$	$q_1 q_2$	$q_1 q_3 q_f$
$q_1 q_2 q_f$	$q_1 q_2 q_f$	$q_1 q_3 q_f$
$q_1 q_3 q_f$	$q_1 q_2 q_f$	$q_1 q_3 q_f$



Scanned with CamScanner

23/08/22

FA with output

- Mealy Machine
- Moore Machine

Moore Machine :- The moore machine is 6 tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$

where,

Q = is finite set of states

Σ = input alphabet

Δ = output "

δ = transition function $\Sigma \times Q$ into Q

λ = output function mapping Q into Δ

q_0 = initial state

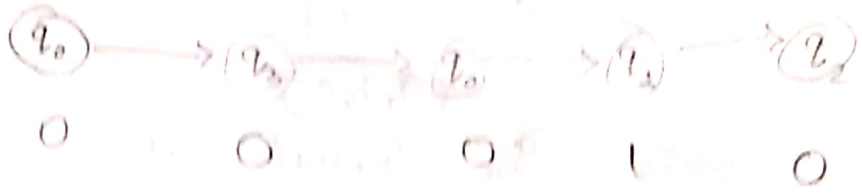
example :- The initial state q_0 is marked with an arrow, the following tables defines δ, λ.

Present state	Next state		output
	a = 0	a = 1	λ
$\rightarrow q_0$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

Scanned with CamScanner

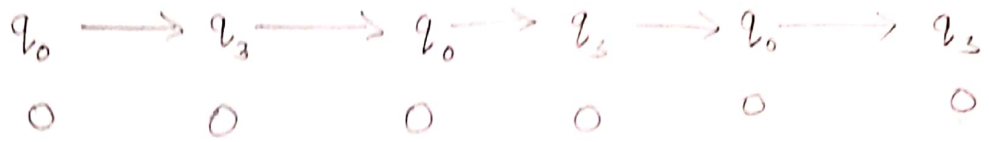
eg:- 0111 → 00010

Transition state



= (0111 → 00010) Ans.

Q. 01010



= (01010 → 000000) Ans.

Mealy Machine :- It is 6 tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$ where,

Q = finite set of states

Σ =

Δ =

δ =

λ = output function mapping $\Sigma \times Q$ into Δ

q_0 = initial state

Scanned with CamScanner

example:- Mealy Machine $Q = \{q_0, q_1\}$

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

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

q_0 = Initial state

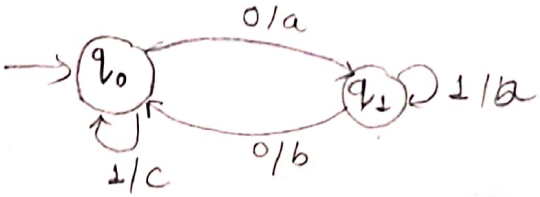
~~$S = S($~~

$\delta(q_0, 0) = q_1$, $\delta(q_0, 1) = q_0$

$\delta(q_1, 0) = q_0$, $\delta(q_1, 1) = q_1$

$\lambda(q_0, 0) = a$, $\lambda(q_0, 1) = c$

$\lambda(q_1, 0) = b$, $\lambda(q_1, 1) = a$



= 0011

$\rightarrow q_0 \longrightarrow q_1 \longrightarrow q_0 \longrightarrow q_0$

a b c c

Scanned with CamScanner