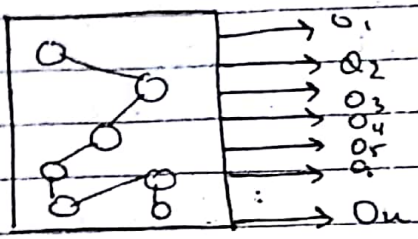
Unit - 1

Finite Automata



Input - An automaton contain finite set of discrete i/p denoted by Σ.

Output - finite automata produces the set of output denoted by Δ.

States - An automaton contain finite set of states denoted by Q.

OUTPUT RELATION - The o/p depends on state & state current input applied on the state, denoted by λ.

STATE TRASITION - Applying the i/p on state that state remain transit to next state denoted by δ.

(DFA) DETERMINISTIC FINITE AUTOMATA.

(Finite State Machine)

Finite automata, or DFA is represented by 5 tuples. $M = (Q, \Sigma, \delta, q_0, F)$

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

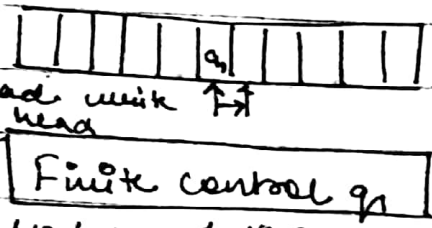
1. Q = Nonempty finite set of states.
2. Σ = Nonempty finite set of i/p symbol
3. δ = is state transition function which maps $Q \times \Sigma \to Q$

i.e $\delta(q, a) = q'$ where $q, q' \in Q$, $a \in \Sigma$

4. q_0 is called the initial state i.e $q_0 \in Q$
5. F → finite set of final states $F \subseteq Q$

The language accepted by finite Automata M is defined as: $L(M) = \{w \mid \delta(q_0, w) = q_f$ where $q_f \in F, w \in \Sigma^{q_f}\}$

Finite Automata can be viewed as a collection of Tape Read / Write head & finite control



Model of finite automata

Input is connected with finite control by read/write head input is (i/p)ed into no. of cell each cell contain one symbol at a time. It is limited in size. Finite control examines the current i/p under the read/write head.

Let a_i, at a present state q_i

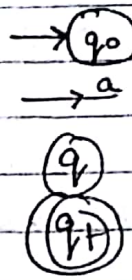
Achintya Pathak

Finite Automata gives 2 output

1. R/W head move one cell right.
2. Finite control change the current state q_i to q_j i.e $\delta(q_i, a_i) = q_j$

Representation of FA

Representation by transition Diagram



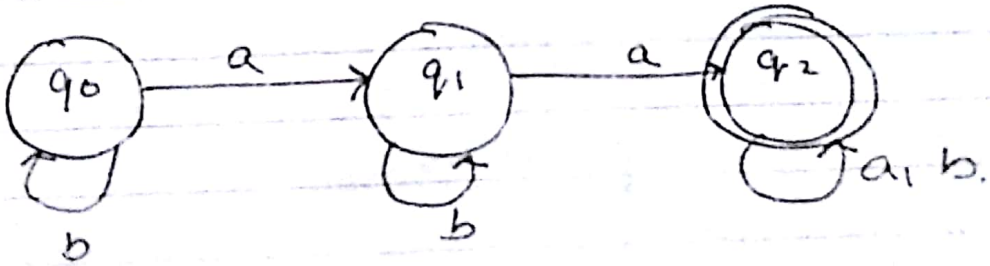
initial state
transition
state
final state

Representation by transition Table

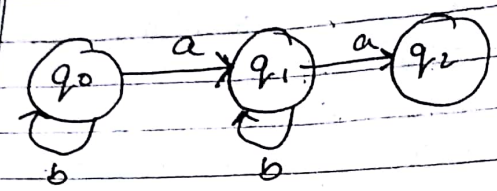
Present state	Next state on input a	Next state on input b
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_0

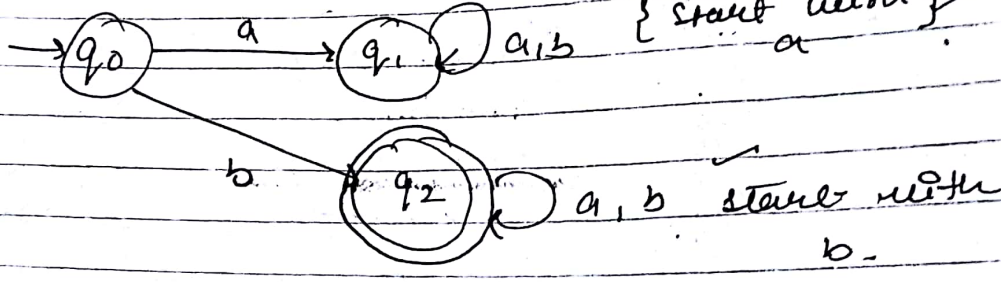
Questions.

es. Create a FA for the set of all strings over $\{a, b\}$ contain at least 2 a's



Ques 2 Design a F.A. for the set of all string over {a, b} starts with b.





Σ	Next state	
	a	b
q_0	q_1	q_2
q_1	q_1	q_1
q_2	q_2	q_2

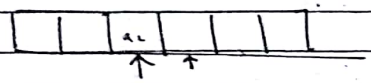
Revision class

pew

Date 29/1

Page

Deterministic Finite Automata



Finite control

$q_1 \rightarrow q_n$

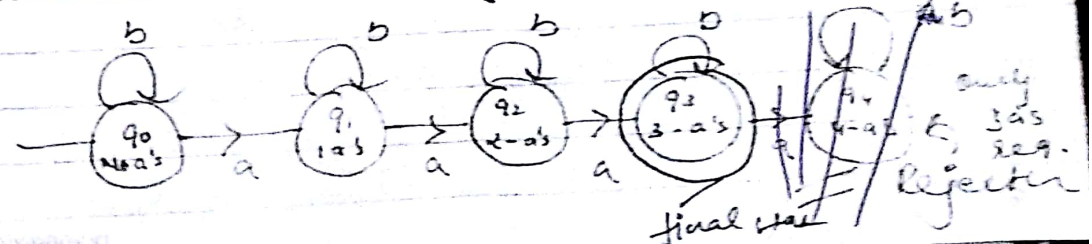
$M = (Q, \Sigma, \delta, q_0, F)$

where

$Q \rightarrow$ Non empty finite set of states

$\Sigma \rightarrow$ Nonempty finite set of i/p symbol

Construct a DFA that accepts all string over {a, b} containing exactly 3a's.



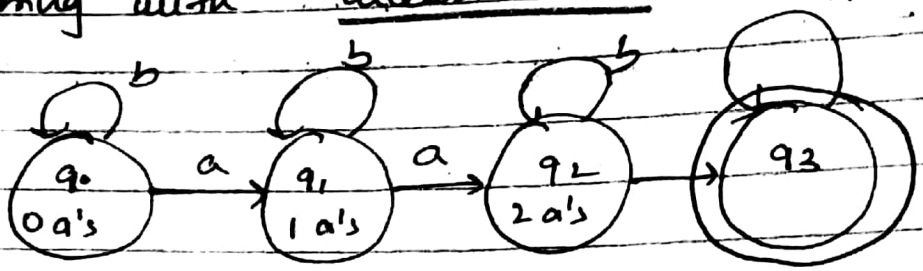
(0) → final state

Check string bbba

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

But q_0 is not final state.

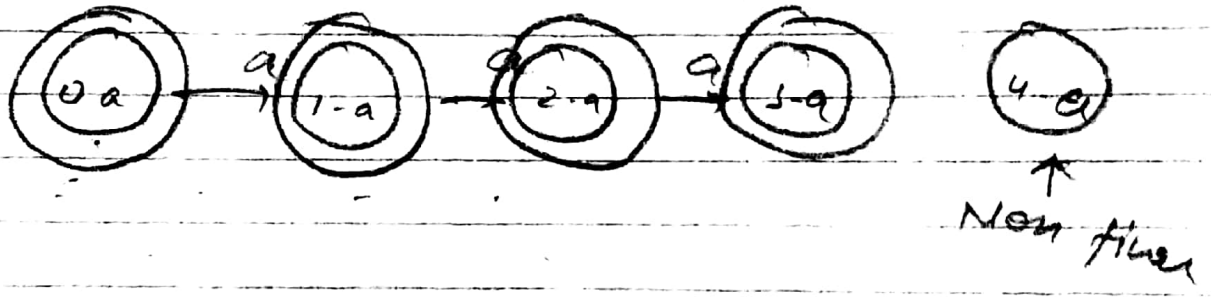
Q Construct a finite automata that accepts all string with atleast 3 a's. a, b



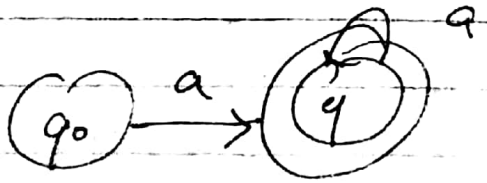
Q " " " at Most 3 a's

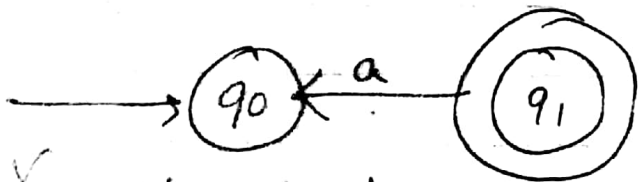
↓

4 final state



Atleast 1 - a





$L(M) = \phi$

No path from initial to final ∴

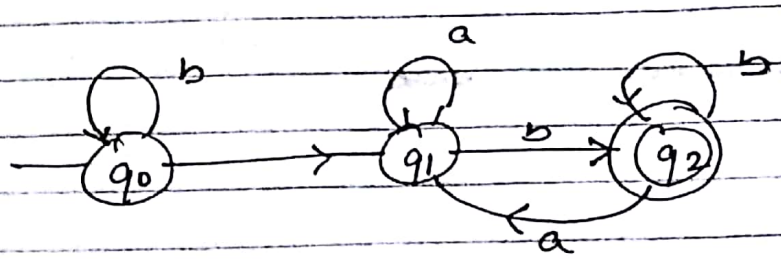
→ (0) → Initial state is also final
$L(M) = \epsilon$

Null string ← ϵ - Epsilon

Properties of δ

(i) $\delta(q, \epsilon) = q$

(ii) $\delta(q, aw) = \delta(\delta(q,a), w)$
$= \delta(\delta(q,w), a)$



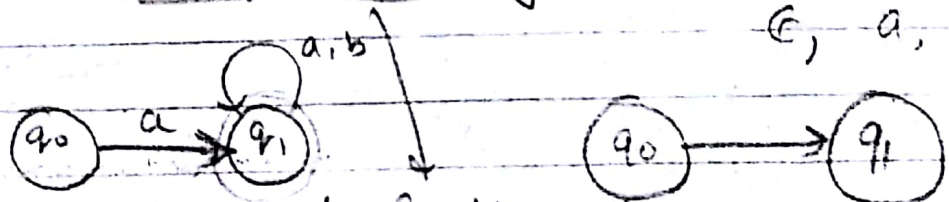
$\delta(q_0, \underbrace{baab}_{w}) = \delta(\delta(q_0, b), aab) =$

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

Construct the DFA

① for the set of all string over $\{a, b\}$

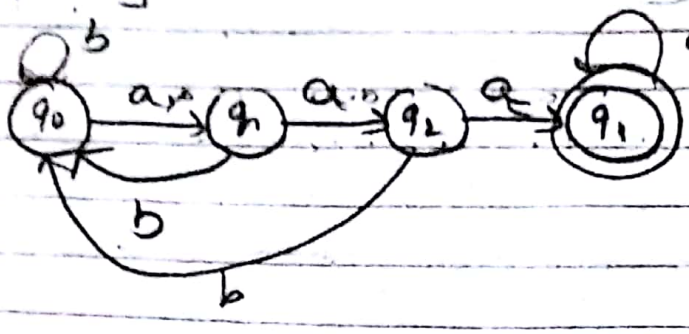
ϵ, a, b, aa, ab, ...



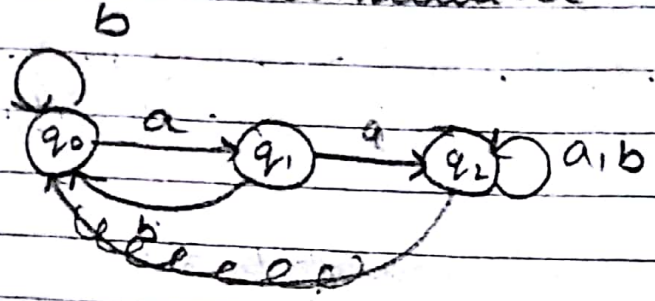
= ϕ, ϵ the data in ...

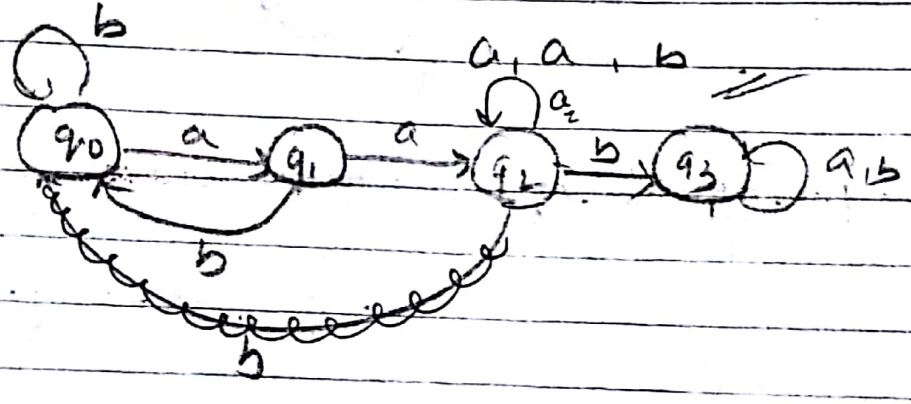
dati

② Containing 3 consecutive a's over {a, b}.

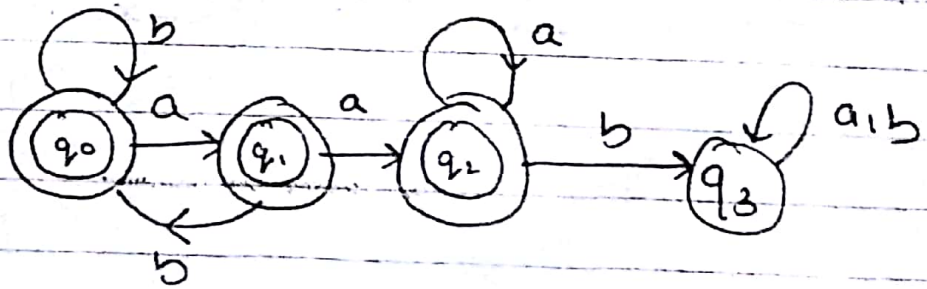


2 consecutive a's.





Q Set of all string over {a, b} not containing aab as substring.

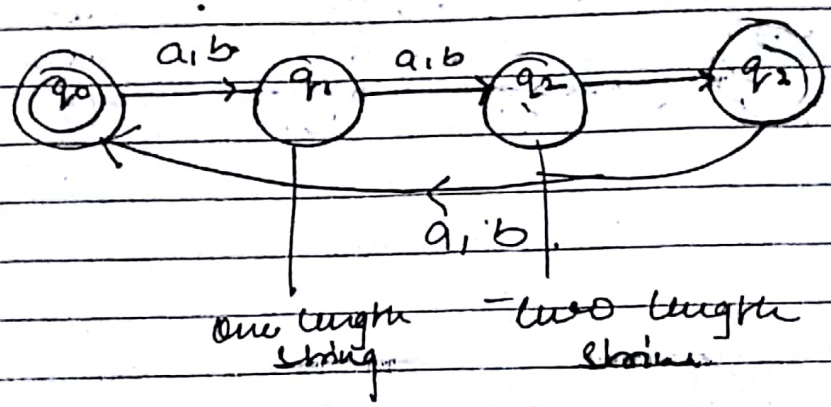


All string over {a, b} whose length is divisible by 3

aaa, bbb, abaaba also ∈ ✓
↑ Null string. 0's divisible by 3.

a + b is one length string.
↑ either

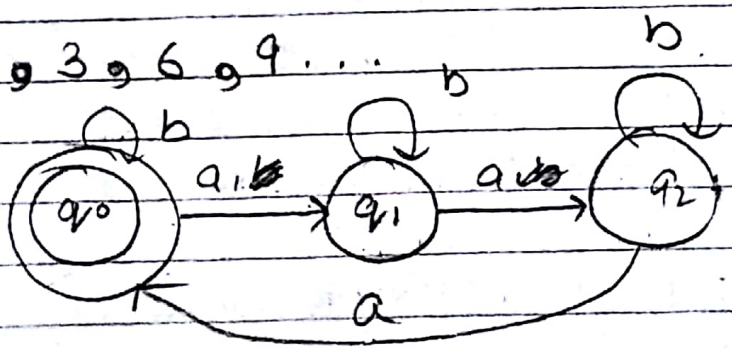
(a+b)(a+b) is two length string.
↳ ⇒ a.a + ab + ba + bb.
2 length



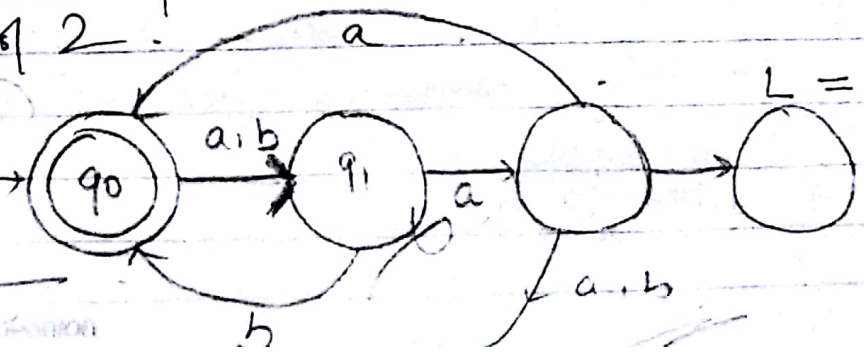
Imp
* No. of a is multiplication of 3.

∈, 3, 6, 9 ...

any no. of b's allow



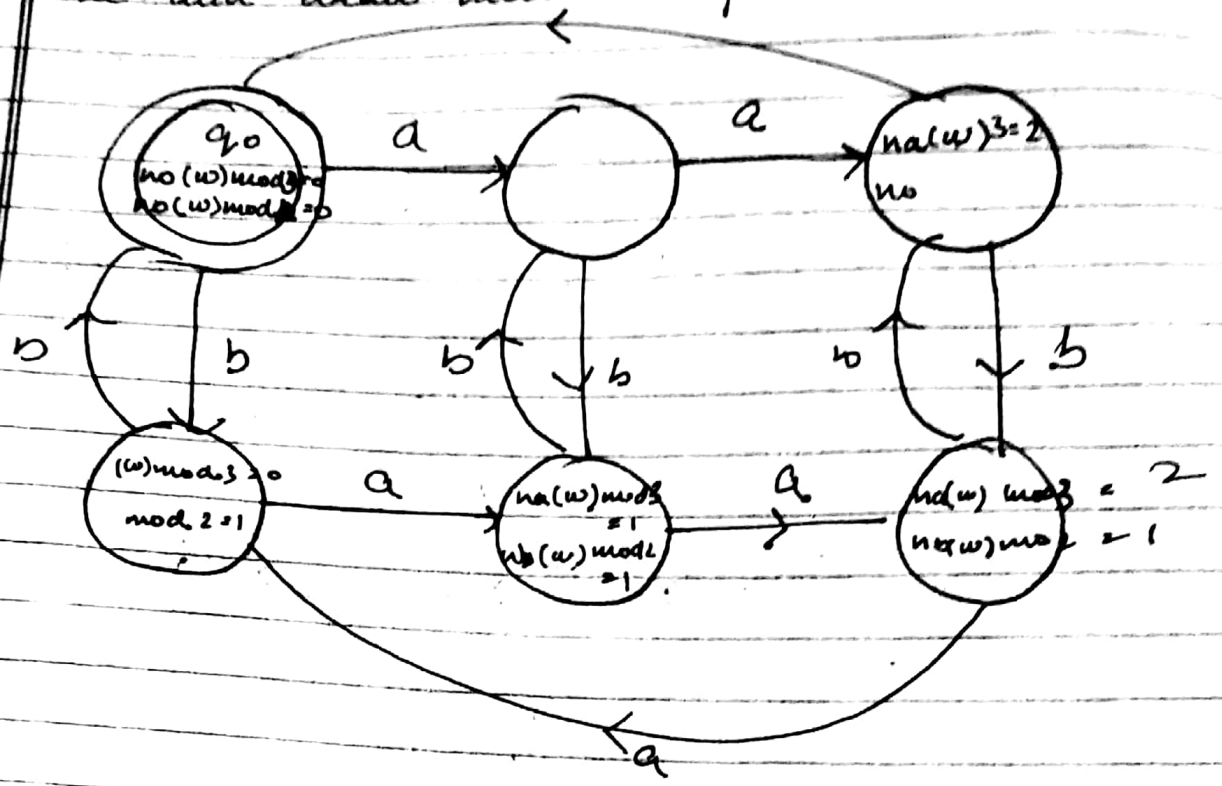
Number of a is (x)tion of 3 & no. of b is (x) of 2.



$L = \{ w \mid n_a(w) \bmod 3 = 0 \text{ and } n_a(w) \bmod 2 = 0 \}$

$w \in \{a, b\}^*$

Here we have to see a & b both. i.e. we will draw individually.



Nondeterministic Finite Automata (NFA)

In NFA applying a single input on a state, there may exist 0, 1 or more than one transition out of the state.

Mathematically an NFA is represented by 5 tuple

$$M = (Q, \Sigma, \delta, q_0, F)$$ where Q, Σ, q_0, F are similar to DFA & δ is a transition function which maps

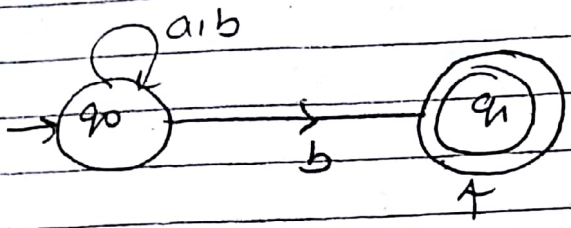
DFA → $Q \times \Sigma \rightarrow 2^Q$

NFA → $Q \times \Sigma \rightarrow Q$

$Q = \{q_0, q_1\}$

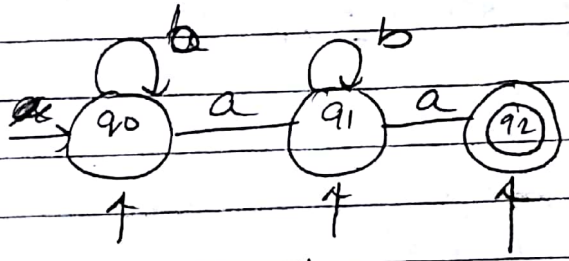
$2^Q = \phi \quad \{q_0\} \quad \{q_1\} \quad \{q_0, q_1\}$

eg 1: Construct a NFA to accept all strings over $\{a,b\}$ Ending with b.



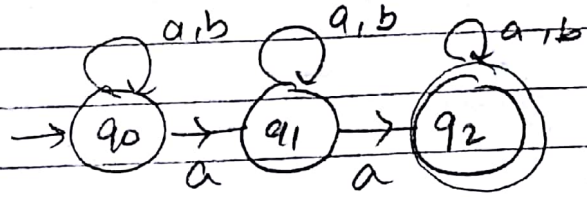
will be ending with b

Const. a NFA at least 2a's over $\{a,b\}$



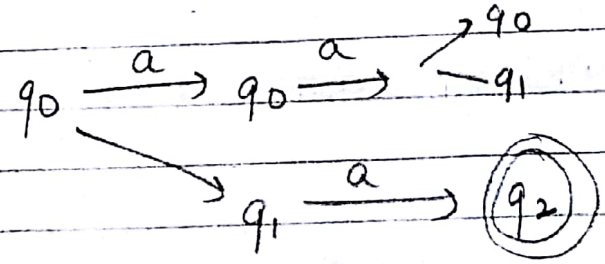
→ DFA

No a. 1a 2 'a

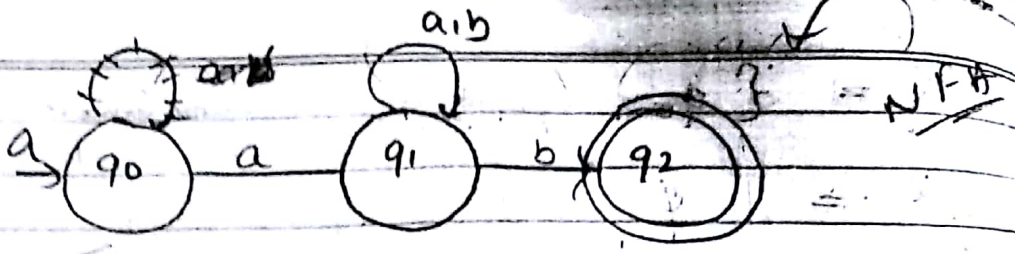


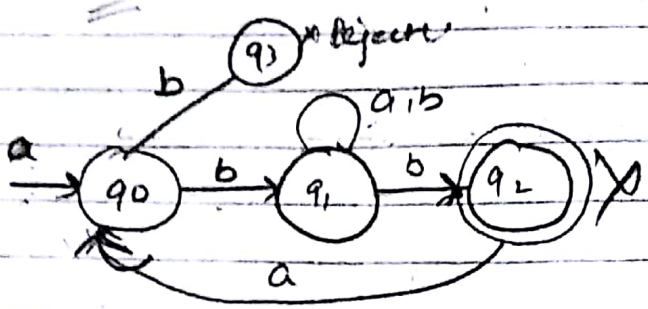
NFA

In NFA, we always have choice to move

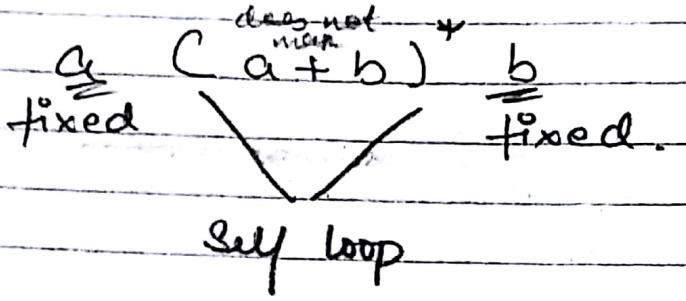


Construct NFA, start with 'a and ends with b.

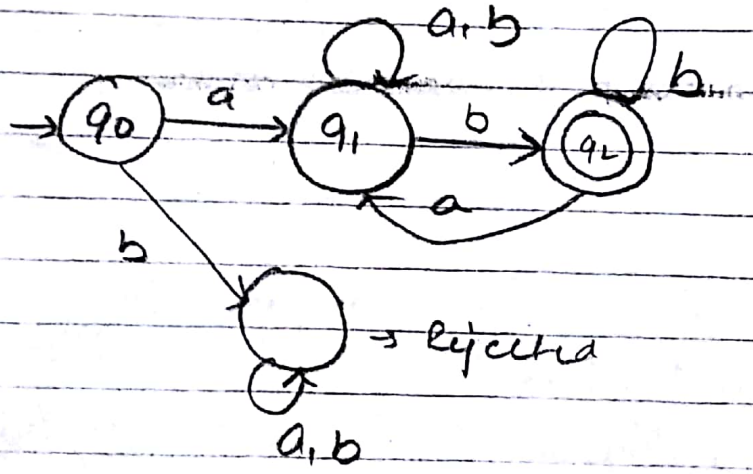




DFA

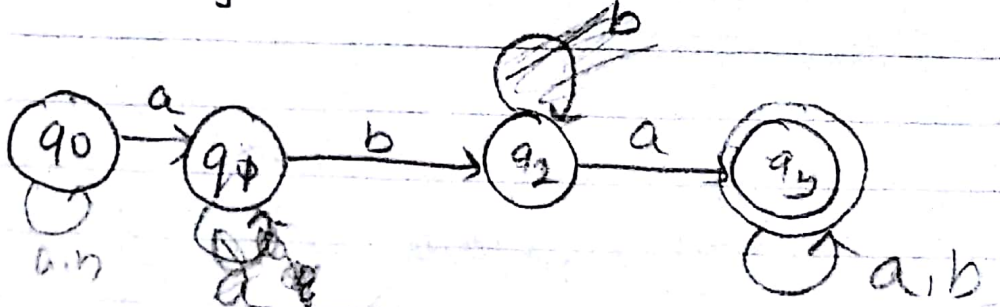


DFA



Q Const NFA containing ~~aba~~ ~~as~~ {a,b,a} as substring.

ac . . aba.



ba ✗

aba ✓

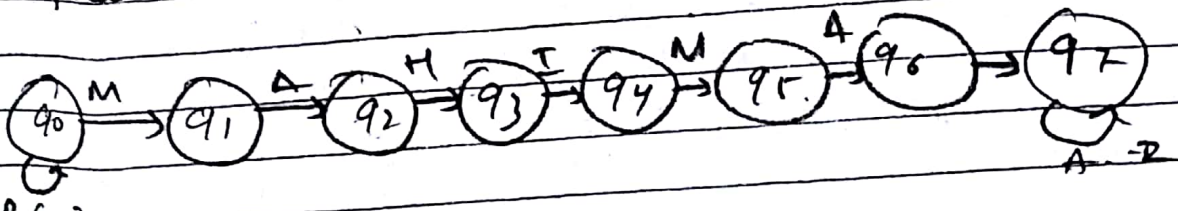
$L^* = \bigcup_{i=0}^{\infty} L^i$

$L^0 = \epsilon$

$L^i = L^{i-1} \cdot L$

$\epsilon, - - -$

Const NFA for the set of all string over alphabet that except your name.



Equivalence of NFA by DFA

Let N be an NFA that excepts the language L then there exists an equivalent DFA M that accept L

i.e $L(M) = L(M')$

$\boxed{M \equiv M'}$

3/2.

Convergence of NFA to DFA.

$M = (Q, \Sigma, \delta, q_0, F)$ be an NFA accepting a language L then there exists an equivalent DFA $M' = (Q', \Sigma, \delta', q_0', F)$ then accept L : Here $M' = (Q', \Sigma, \delta', q_0', F')$ is defined as $Q' \equiv 2^Q$

NFA: $\delta : Q \times \Sigma \to 2^Q$

DFA: $\delta' : Q' \times \Sigma \to Q'$

$2^Q \times \Sigma \to 2^Q$

Let $\{q_1, q_2 \ldots q_n\}$ is Q then Q' will be $\{\epsilon, [q_1], [q_2] \ldots [q_n]\}$

Equivalence of NFA & DFA.

NFA = DFA.

Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA that accepts a language L then there exists an equivalent DFA.

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

$$L(M') = L(M)$$

(1) Every DFA is a subset of NFA

DFA ⊂ NFA

$Q \times \Sigma \to Q$ ⊂ $Q \times \Sigma \to 2^Q$

(2) NFA ⊂ DFA.

$$L(M) = \{w \mid \delta(q_0, w) = q_t \text{ where } q_t \in F\}$$

Proving NFA ≡ DFA.

$$\delta(q, x) = [q_1, q_2 \ldots q_i]$$

iff $\delta'([q], x) = [q_1, q_2 \ldots q_i]$

this can be proved by Induction hypothesis.

Let $w = \epsilon$, $|w| = 0$

$\delta'(q_0', \epsilon) = q_0' = [q_0]$

that means for $w = \epsilon$ string terminated at same state.

Hypothesis → Assume w as n length string & following property is true.

$\delta(q_0, w) = \{q_1, q_2, q_3 \dots q_i\}$.

If $\delta'(q_0', w) = [q_1, q_2, q_3 \dots q_i]$

Inductive step → Assuming string 'wa' $|wa| = n+1$.

$\delta(q_0, wa) = \delta(\delta(q_0, w), a)$

$= \delta(\{q_1, q_2 \dots q_i\}, a) = \{p_1, p_2 \dots\}$

$\delta'(q_0', wa) = \delta'(\delta'(q_0', w), a)$

$= \delta'([q_1 q_2 \dots q_i], a)$

$[p_1 p_2 \dots p_i]$

If w is accepted by NFA.

$\delta(q_0, w) = q_f, q_f \in F$

$\delta'(q_0, wa) = [q_f]$

So Hence Language $L(M') = L(M)$

Hence NFA = DFA.

$\delta'(q_0', \epsilon) = q_0' = [q_0]$

that means for $w = \epsilon$ string terminated at same state.

Hypothesis → Assume w as n length string & following property is true.

$$\delta(q_0, w) = \{q_1, q_2, q_3 \dots q_i\}.$$

iff $\delta'(q_0', w) = [q_1, q_2, q_3 \dots q_i]$

Inductive step → Assuming string 'wa' $|wa| = n+1$.

$$\delta(q_0, wa) = \delta(\delta(q_0, w), a)$$
$$= \delta(\{q_1, q_2 \dots q_i\}, a) = \{p_1, p_2 \dots p_j\}$$

$$\delta'(q_0', wa) = \delta'(\delta'(q_0', w), a)$$
$$= \delta'([q_1, q_2 \dots q_i], a)$$
$$[p_1, p_2 \dots p_i]$$

if w is accepted by NFA

$\delta(q_0, w) = q_f, q_f \in F$

$\delta'(q_0, w) = [q_f]$

So Hence Language $L(M') = L(M)$

Hence NFA = DFA.

Scanned by CamScanner

NFA with ϵ transition.

NFA with ϵ transition is extension of NFA by applying the ϵ as an i/p.

ϵ is used to move from one state to another without modifying the string over (sigma) Σ.

Mathematically a NFA with ϵ is represented by 5 tuple.

$$M = (Q, \{\Sigma \cup \epsilon\}, \delta, q_0, F)$$

where Q, Σ, q_0, F are similar to NFA and δ is defined as $Q \times \{\Sigma \cup \epsilon\} \rightarrow 2^Q$

Properties of Transition function δ

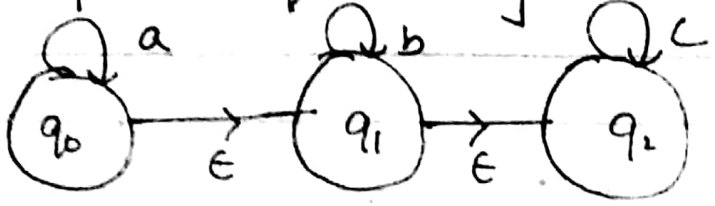
We have to extended δ to $\hat{\delta}$ for NFA with ϵ

1) $\hat{\delta}(q, \epsilon) = \epsilon\text{-CLOSURE}(q)$

$\epsilon\text{-CLOSURE}(q) = \{P\}$.

The set of state P such that we can move from q to P by ϵ.

eg



eg- $\delta(q_0, \epsilon) = \{q_0, q_1, q_2\}$

$\delta(q_2, \epsilon) = \{q_1, q_2\}$

$\delta(q_1, \epsilon) = \{q_1, q_2\}$

$\epsilon . a = a . \epsilon = a$.

② $\hat{\delta}(q, wa) = \epsilon\text{-CLOSURE}(P)$

where $P = \delta(\hat{\delta}(q, w), a)$

eg $\hat{\delta}(q_0, a) = \delta(\hat{\delta}(q_0, \epsilon), a)$

$\epsilon . a$

$\hat{\delta}(q, wa)$

$w = \epsilon$

$= \epsilon\text{-CLOSURE}(\delta(\epsilon\text{-CLOSURE}(q_0), a)$

$= \epsilon\text{-CLOSURE}(\delta(\{q_0, q_1, q_2\}, a)$

$= \epsilon\text{-CLOSURE}(\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a))$

$\epsilon\text{-CLOSURE}(q_0 \cup \phi \cup \phi)$

$\epsilon\text{-CLOSURE}(q_0)$

$\{q_0, q_1, q_2\}$

Summary

Properties

① $\hat{\delta}(q, \epsilon) = \epsilon\text{-CLOSURE}(q)$

$= \{p\}$

② $\hat{\delta}(q, wa) = \epsilon\text{-CLOSURE}(\delta(\hat{\delta}(q_0, w), a))$

Ex 2. $\hat{\delta}(q_0, b)$ with * same diag

$= \epsilon\text{-CLOSURE}(\delta(\hat{\delta}(q_0, \epsilon), b))$

$\epsilon\text{-CLOSURE}(\delta(\epsilon\text{-CLOSURE}(q_0), b)$

$\epsilon\text{-CLOSURE}(\delta(q_0, q_1, q_2\}, b)$

= ε-CLOSURE (δ (q0, b) ∪ δ(q1, b) ∪ δ(q2, b)

ε-CLOSURE (ϕ ∪ q1 ∪ ϕ)

ε-CLOSURE (q1)

Q = Compute δ̂ (q0, ab)

ε-CLOSURE (δ (δ̂ (q0, a), b))

ε-CLOSURE (δ (δ̂ (ε-CLOSURE (q0), a), b)

ε-CLOSURE (δ (ε-CLOSURE (δ (ε-CLOSURE (q0), a)

ε-CLOSURE (δ (ε-CLOSURE (δ ({q0, q1, q2}, a)

ε-CLOSURE (δ (ε-CLOSURE (δ (q0, a) ∪ δ(q1, a) ∪ δ(q2, a)

ε-CLOSURE (δ (ε-CLOSURE (q0), b)

ε-CLOSURE (δ ({q0, q1, q2}, b)

ε-CLOSURE (δ (q0, b) ∪ δ(q1, b) ∪ δ(q2, b))

ε-CLOSURE (ϕ ∪ q1 ∪ ϕ)

ε-CLOSURE (q1)

{q1, q2}

=

CLOSURE - close under.

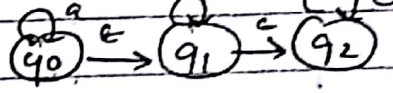
Conversion of NFA with ε to NFA without ε

Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA with ε that accepts a language L then there exists an equivalent NFA $M' = (Q, \Sigma, \delta', q_0, F')$

M' is defined as
Q, Σ, q_0 are similar to NFA with ε
δ' is defined as

$$\delta'(q, a) = \hat{\delta}(q, a) \quad \forall q \in Q, \; a \in \Sigma$$

$F' = F \cup \{q\}$ such that ε-CLOSURE(q) contain final state of M.



NFA

b))

present state	a	b	c	ε
q_0	$\{q_0\}$	–	–	$\{q_1\}$
q_1	ϕ	$\{q_1\}$	–	$\{q_2\}$
q_2	–	–	$\{q_2\}$	–

Eg

Step $\delta(q_0, a)$.

P.S	a	b	c
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	–	$\{q_1, q_2\}$	$\{q_2\}$
q_2	–	–	$\{q_2\}$

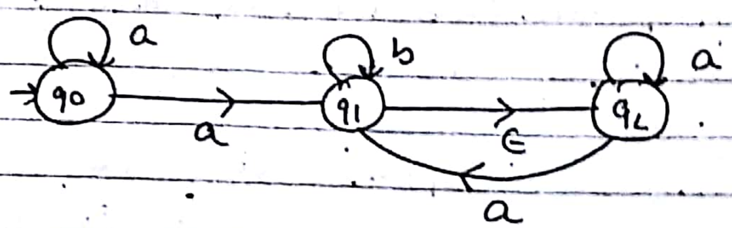
Last state

final state should contain last state.

$F' = F \cup \{q\}$ → such that ε-closure (q) contains final/... state.

q_0 :: ε closure(q_0) = $\{q_0, q_1, q_2\}$

Q Convert this ε NFA to NFA without ε.



make this with diagram.

Present state	Next state a	b	ε
→ q_0	$\{q_0, q_1\}$	–	–
q_1	–	$\{q_1\}$	$\{q_2\}$
q_2	$\{q_2, q_1\}$	–	–

This with formula

P.S.	N.S. a	b
q_0	$\{q_0, q_1, q_2\}$	–
→ q_1	–	$\{q_1, q_2\}$
q_2	$\{q_2, q_1\}$	–

Show this

ε-closure(δ (q_0, a))

8 | Feb.

Unit - 2

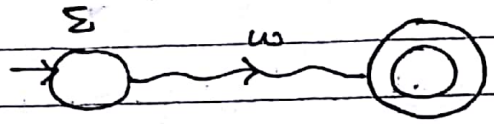
Regular Expression. (RE)

A simple expression which express the Regular language called the regular expression.

↓

A language is Regular language if it is expet accepted by a finite automata.

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a FA. then language accepted by M is defined as $L(M) = \{w \mid$ such that $(q_0, w) = q_f, q_f \in F$ & $w \in \Sigma^*\}$



A regular expression is defined recursively over input alphabets Σ as follows.

(i) ϕ is a regular exp. & define the regular set ϕ.

(ii) ϵ is a r.exp & define the regular set (language) $\{\epsilon\}$

(iii) Let $a \in \Sigma$ then a is a r.e & define the regular set $R = \{a\}$ & & &.

(iv) Let $a, b \in \Sigma$ & a & b are reg. exp that define the regular set R & S respectively then $a+b$, $a \cdot b$, a^* are regular exp. that defines the regular set $R \cup S$, RS, R^* resp.

$L^* = a^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup L^3 \ldots$

ex. $\Sigma = \{a, b\}$ prove $(a^* + ab)(a+b)$ is r.e

Let $a \in \Sigma$ then a^* is a reg. exp

a & b are reg. exp then a.b is also reg exp.

as a^*, ab are reg. exp. then $a^* + ab$ is also r.e

a, b are r.e then $a+b$ is a reg. exp.

$(a^* + ab)$ & $(a+b)$ are reg. exp. then

$(a^* + ab)(a+b)$ is eq. ex.

ex 2. $\Sigma = \{a, b\}$ is $(a^* + ab)(a+b)c$ is regex?

No. Ans c does not belongs to Σ

$a + b \rightarrow$ Either a or b.

$a.b \rightarrow$ a followed by b

$a^* \rightarrow$ Kleen closure of a

Difference

Kleen closure	Positive closure
A Kleen closure of a language L is denoted by $L^* = \bigcup_{i=0}^{\infty} L^i$	A Positive closure of a language L is denoted by $L^+ = \bigcup_{i=1}^{\infty} L^i$

* Imp

$\epsilon \rightarrow$ 0 length string

Consist in * only

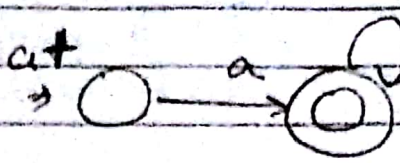
Not in +ve

where $L^0 = \epsilon$

$L^i = L \cdot L^{i-1}$

$L^i = L \cdot L^{i-1}$

a^* 

$a+$ 

Write r.e for following language (Imp

① the set of all string over {a, b}

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

$(a+b)^0, (a+b)^1, (a+b)^2 \ldots$

$\epsilon, a+b, ((a+b)\cdot(a+b)) \ldots$

② The set of all string starts with a over {a, b}

Sol:- $a(a+b)^*$

Contain at least 2 a's over {a, b}

Sol:-

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

Sol

③ Set of all string over {a, b} contains atleast 2 a's :-

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

④ Exactly 2 a's $b^* a b^* a b^*$

Atmost 2a's

no. a's = b^*

1 a's = $b^* a b^*$

2 a's = $b^* a b^* a b$

→ re = $b^* + b^* a b^* + b^* a b^* a b^*$

c. The set of all strings contain even no. of a's

$$b^* + (b^*ab^*ab^*)^*$$

a. All odd no.'s of a's

$$b(b^*ab^*)(b^*ab^*ab^*)^*$$

Q10 All string over $\{a,b\}$ doesn't contain aa as substring.

when no. of a appear = b^*
when a appear in string = ab^*

$\{\epsilon, b, ab(b+ab).(b+ab)$

$(b+ab)^*$ → ending with B.
$(b+ab)^*a$ → ending with a.

sup

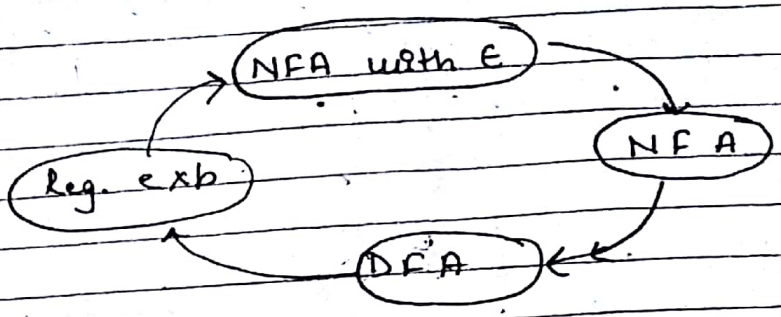
Ques Set of all strings in which string contain atmost one aa.

when no. of a apper = b^*
$|a \rightarrow ab^*$

$$(b+ab)^*(\epsilon+a)+(b+ab)^*aa(b+ba)^*$$

d. Set of all string over $\{a,b\}$, no. of a appear in (x)tion of 3

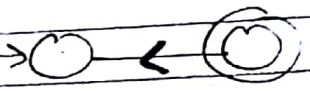
$$(b^*ab^*ab^*ab^*)^* + b^*$$



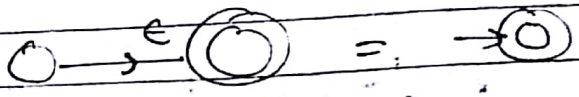
Reg. Exp.

R.E	(Set)
ϕ	ϕ
ϵ	$\{\epsilon\}$
a	$\{a\}$
a+b	$\{a,b\}$
a.b	$\{a.b\}$
a^*	$\{\epsilon, a, aa, aaa..\}$

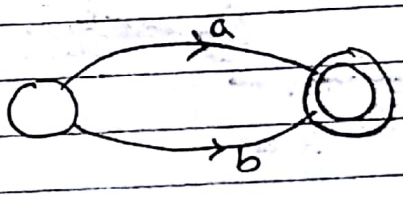
ϕ



ϵ



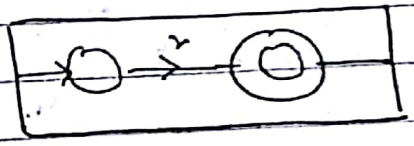
(1) a+b,



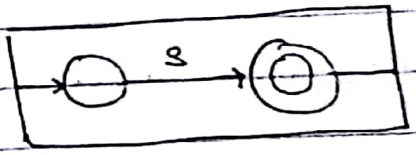
↑ either a ↑ or ↓ b.

a + b

easy (r s)

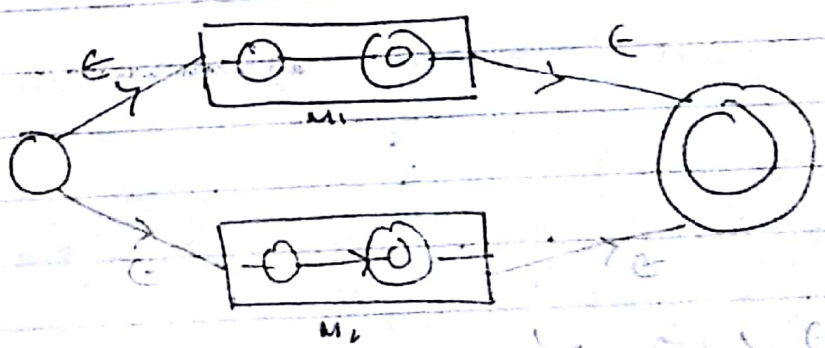


M_1



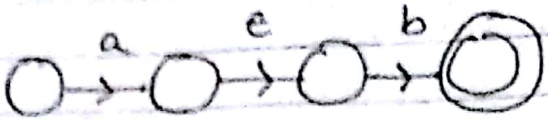
M_2

r + s

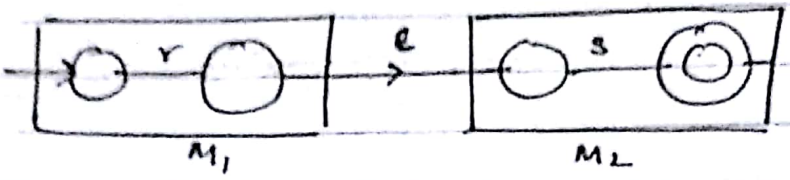


ϵ.r.ϵ→r

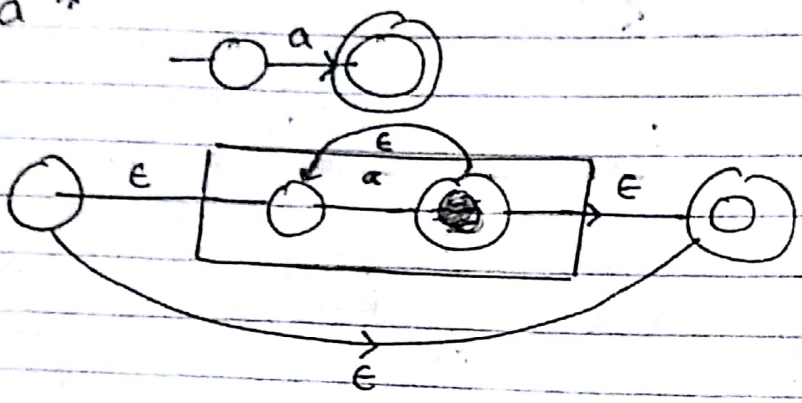
(2) a . b.



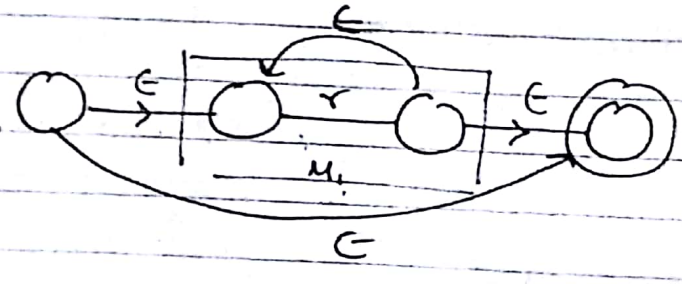
r . s



(3) a *



r *



(4) Construct the NFA with ε for r.e

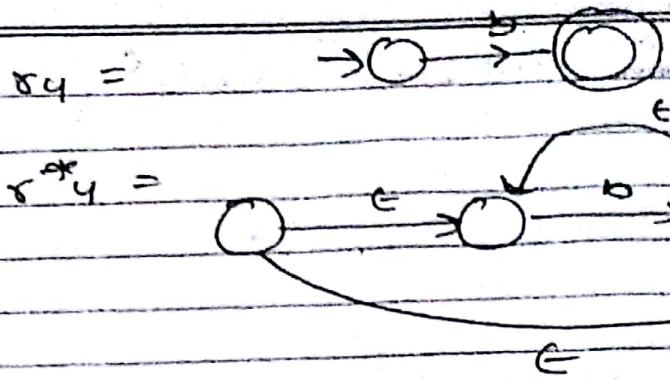
$$\underbrace{b^*}_{r_1}\ \underbrace{abb}_{r_2}\ \underbrace{(a+b)^*}_{r_3}$$

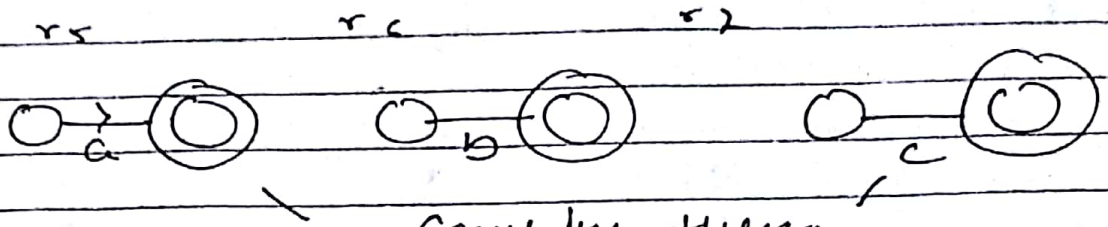
$r_1 = b^* = r_4^*$, $r_4 = b$

$r_2 = abb$, $r_5 = a$, $r_6 = b$, $r_7 = b$

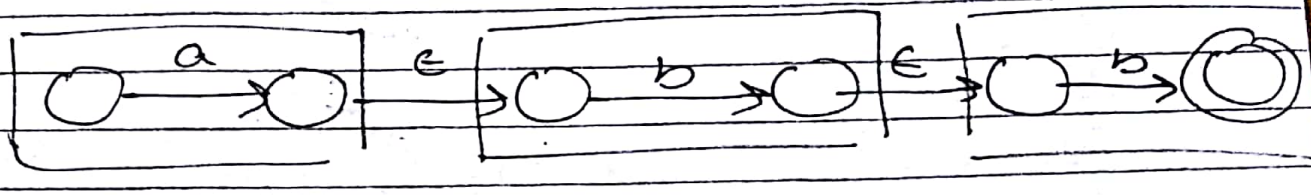
$r_3 = (a+b)^* = r_8^*$

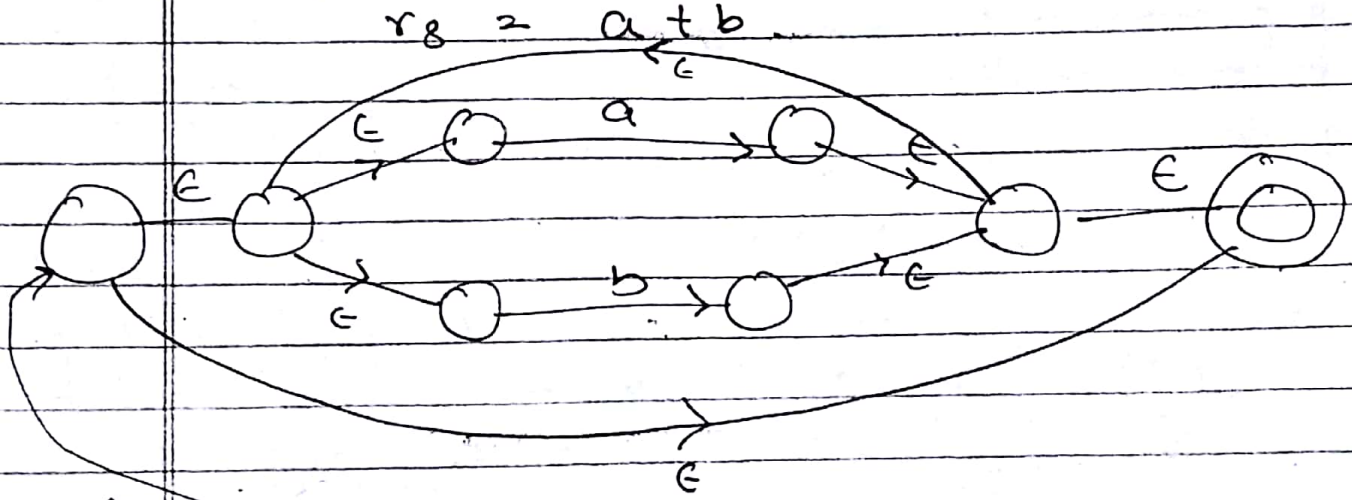
$r_8 = a+b$



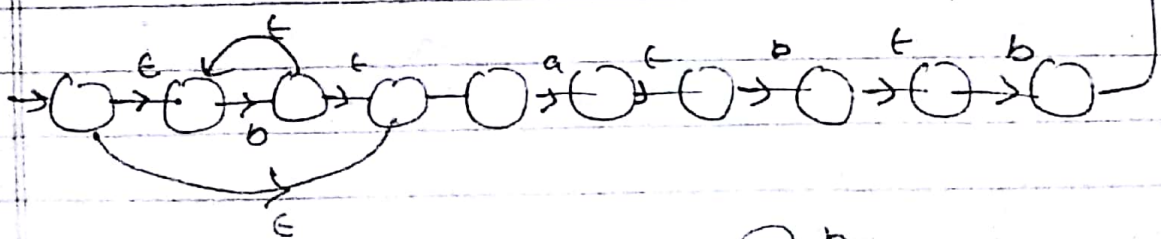


Combine them.

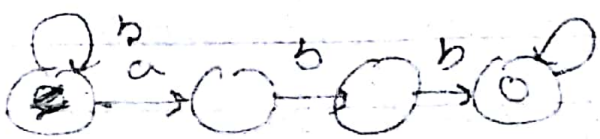




Join them



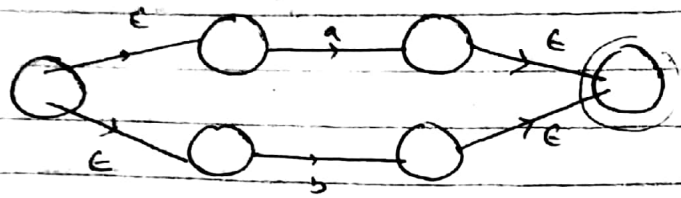
final



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

$r_1 \quad r_2 \quad r_3$

$r_1 = (a+b)^* = r_4^* = (a+b)$

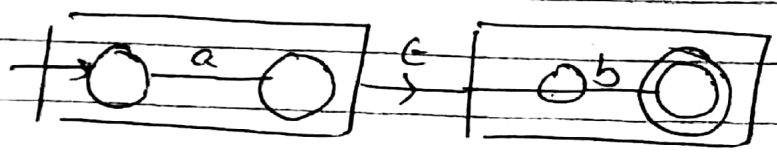


$r_2 = ab + a$

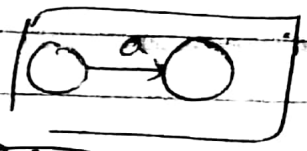
$r_5 = ab$

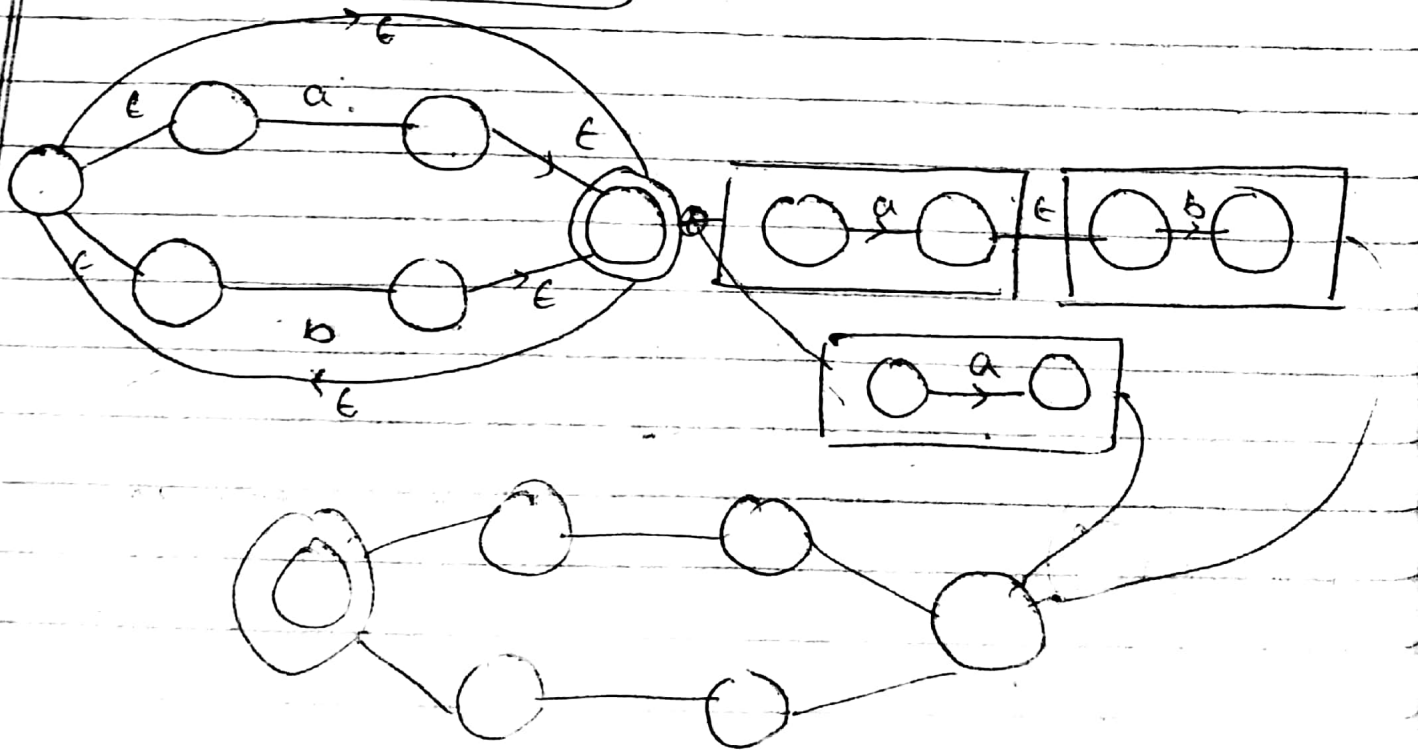
$r_6 = a$

r_5

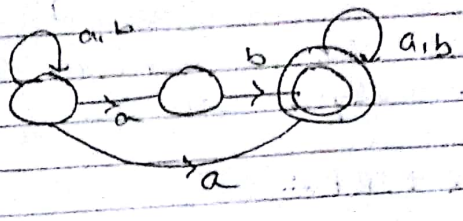


r_6





NFA without ϵ



Identities for R.ϵ

1) $\phi + R = R + \phi = R$
2) $\phi . R = R . \phi = \phi$
3) $\epsilon . R = R . \epsilon = R$
4) $\epsilon^* = \epsilon$, $\phi^* = \epsilon$
5) $R + R = R$
6) $R^* R^* = R^*$
7) $R . R^* = R^* . R$
8) $(R^*)^* = R^*$
9) $\epsilon + R . R^* = R^*$
10) $(PQ)^* P = P(QP)^*$
11) $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
12) $(P+Q)R = PR + QR$
13) $R(P+Q) = RP + RQ$

2 Mark *** Lab

Arden's theorem :-

Let $R = Q + RP$ be a regular expression then it has an unique solution $R = QP^*$

Proof - $R = Q + RP$

Again Put R in RHS

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

$R = Q + QP + RP^2$

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

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

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

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

and so on we put $R = Q + RP$ in eq. we get

$R = Q + QP + QP^2 + QP^3 + QP^4 + \ldots$

$R = Q(\epsilon + P + P^2 + P^3 + P^4 + \ldots)$

$R = QP^*$

s. Using identity, solve

$R = \epsilon + 1^*(0|1)^*(1^*(011)^*)^*$

$\epsilon + RQ^* = R^*$

$R = 1^*(011)^*$

$\epsilon + R$

$\epsilon + 1^*(011)^*(1^*(011)^*)^* = (1^*(011)^*)^*$

$\epsilon + R R^* = R^*$

$= (1^*(011)^*)^*$

Identity 11.

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

$P = 1$

$Q = (011)^*$

$= (1 + 011)^*$

1 is common. take 1 common. but right side is

$= ((\epsilon + 01)1)^*$ ✓

↓

String with no two zeros consecutively.

↓↓ Construction of RE for FA's ↓↓

Let $M = (\{q_0, q_1, q_2 \cdots q_n\}, \Sigma, \delta, q_0, F)$ be a FA then there exists a regular lang L accepted by FA defined as $L(M)$ is exp. by a r.e.

Regular exp from FA is constructed by Algebric Method.

Algebric method following equations are defined a_{ij} denotes the transition from state q_i to q_j

$q_0 = q_0 a_{00} + q_1 a_{10} + q_2 a_{20} + \cdots q_n a_{n0} + \epsilon$ — (1)

$q_1 = q_0 a_{01} + q_1 a_{11} + q_2 a_{21} + \cdots q_n a_{n1}$ — (2)

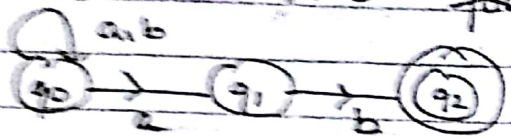
$q_n = q_0 a_{0n} + q_1 a_{1n} + q_2 a_{2n} + \cdots q_n a_{nn}$ —

a language accepted by FA is defined as

$$L(M) = \{w \mid \delta(q_0, w) = q_f,\ q_f \in F\}$$

So we have to solve the above eq for final state i.e. q_f

Prob Construct the RE for following FA.



Solution

$$q_0 = q_0(a+b) + q_1 \cdot \phi + q_2 \phi + \epsilon$$
$$q_0 = q_0(a+b) + \epsilon \quad \text{— (1)}$$

$$q_1 = q_0 \cdot a + q_1 \cdot \phi + q_2 \cdot \phi$$
$$q_1 = q_0 \cdot a \quad \text{— (2)}$$

$$q_2 = q_0 \cdot \phi + q_1 \cdot b + q_2 \cdot \phi$$
$$q_2 = q_1 b \quad \text{— (3)}$$

Solving eq 1.

$$q_0 = \epsilon + q_0(a+b)$$
$$\downarrow$$
$$R = Q + RP$$

eq is in the form of $R = Q + RP$ which has unique sol. i.e $R = QP^*$

$q_0 = \epsilon.(a+b)^* = (a+b)^*$

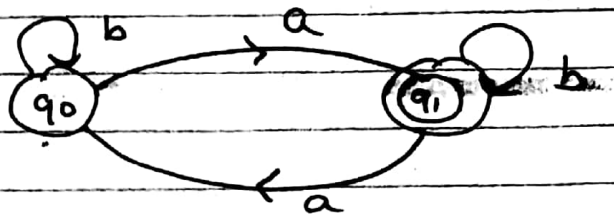
Put the q_0 in eq (2) we get

$q_1 = q_0 a = (a+b)^* a$

Now put q_1 in eq (3)

$q_2 = (a+b)^* ab.$

Q Construct R.E for finite auto.



$q_0 = q_0.b + q_1 a + \epsilon$ — (1)

$q_1 = q_0 a + q_1 b.$ — (2)

We have to solve for q_1, but it contains q_0, ∴ first solve for q_0.

eq ko bus Arden's theorem ke form mai lana h. i.e. $R + Q + RP$

$q_0 = \underbrace{q_0.b}_{RP} + \underbrace{q_1 a}_{Q} + \epsilon.$

$\downarrow$

$R = a + RP$

$q_0 = (\epsilon + q_1 a) b^*$

Put q_0 in eq (2)

$q_1 = (\epsilon + q_1 a) b^* a + q_1 b$

$q_1 = b^*a + q_1ab^*a + q_1b$

$q_1 = \underset{\downarrow}{b^*a} + \underset{\downarrow}{q_1}\underset{\downarrow}{(ab^*a + b)}$

$R = Q + RP$

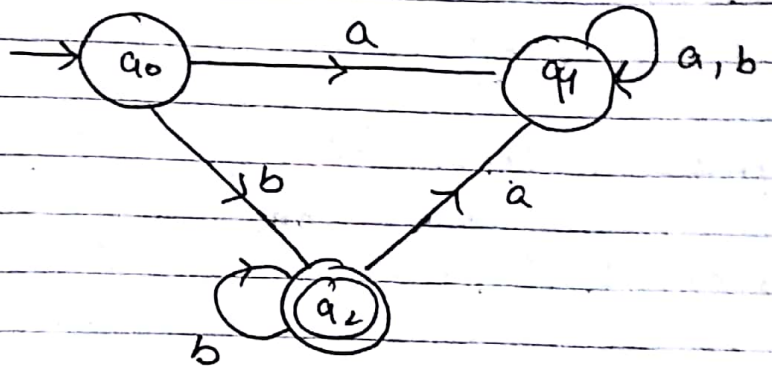
$q_1 = b^*a(ab^*a + b)^*$

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

1 + even

∴ 2 odd as.

Ques



$q_0 = q_0(\phi) + q_1(\phi) + \epsilon$ — (1)

$q_1 = q_0(a) + q_1(a+b) + q_2(a)$

$q_2 = q_0(b) + q_2(b)$

We have to solve for q_2 but it contains q_0.

$q_0 = q_0(\phi)$

$q_2 = \epsilon \cdot b + q_2 b$

$b + q_2 b$

$R = QP^*$ $\quad Q = q_0(b)$

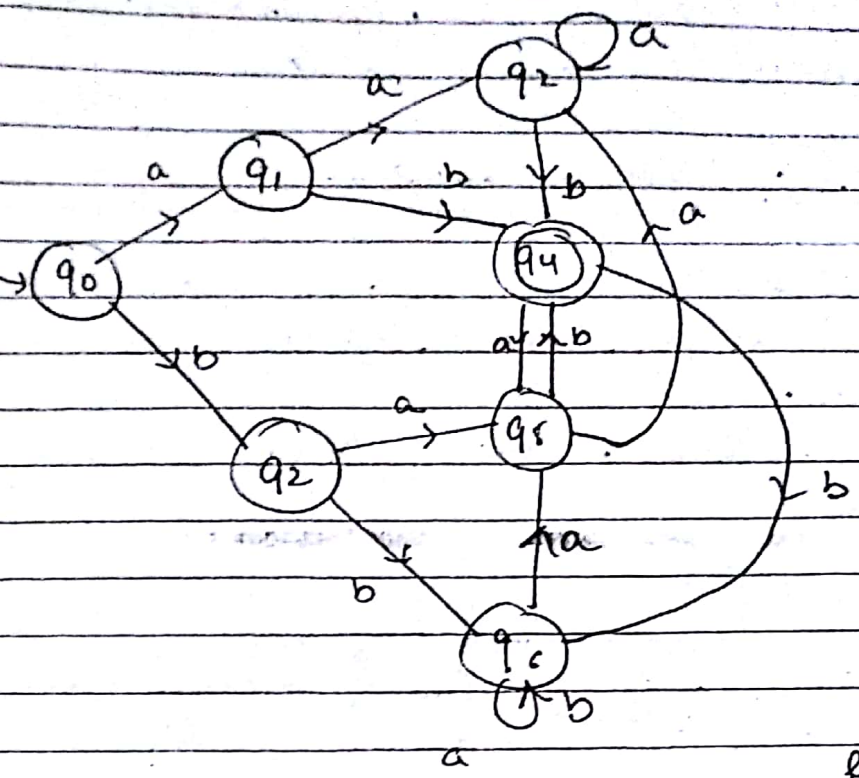
$q_2 = q_0(b)b^*$

⇒ All strings $q b, \ldots b b^*$.

Date :- 15/Feb.

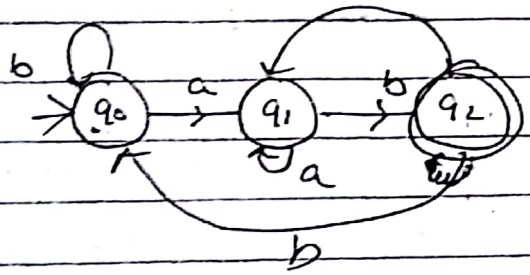
eliminating unused states & reducing no. of state.

Minimization of F.A.



→ See last strings.

leading with as.



Equivalent state

Two states let q_1 & q_2 are equivalent to each other

if $\delta(q_1, a)$ & $\delta(q_2, a)$ for each i/p a belong to ϵ, i.e $a \in \Sigma$, belong to set of final state or set of nonfinal state.

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

$$q_1 \equiv q_2$$

Partition Set of State:

π_K → denotes the partition set of K-equivalent classes of states.

* **K – equivalent states**

Two states let q_1 & q_2 are K equivalent if $\delta(q_1, x)$ & $\delta(q_2, x)$ for $x \in \Sigma^*$ and $|x| = k$ belong to same set of final state or non final state

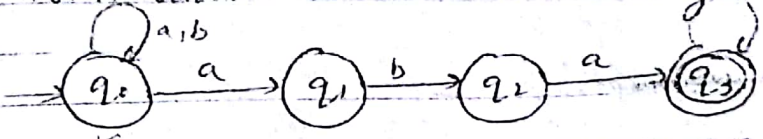
$\delta(q_0, a) = q_1$
$\delta(q_1, a) = q_2$
} Both are non final state
∴ equivalent.

$\delta(q_0, \epsilon) = q_0$
$\delta(q_1, \epsilon) = q_1$
} — They are zero equivalent.

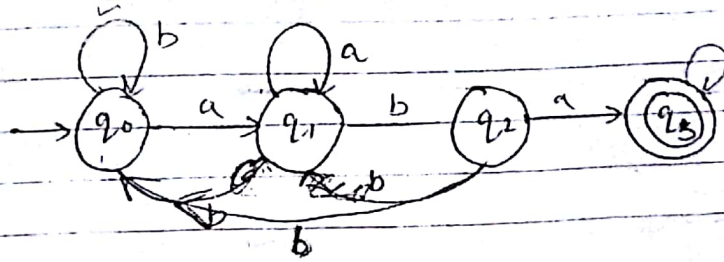
Date

Construct NFA contain aba as sub string

NFA

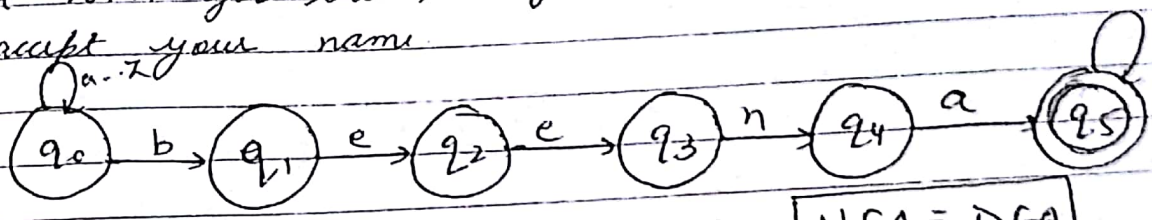


(aba)



DFA

Construct NFA for the set of all string over alphabet that accept your name



DFA ⊆ NFA ; NFA ⊆ DFA ; **NFA ≡ DFA**

Equivalence of NFA & DFA

Let M be an NFA that accepts the language L then there exists an equivalent DFA M' that accepts the language L.

i.e, $L(M) = L(M')$

$\boxed{M \equiv M'}$

Conversion of NFA to DFA

3/2/18 $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA accepting a language L then there exists an equivalent DFA $M' = (Q', \Sigma, \delta', q_0', F')$ that accept L. Here $M' = (Q', \Sigma, \delta', q_0', f')$ is defined as

$Q' = 2^Q$ ∴ NFA : $\delta : Q \times \Sigma \to 2^Q$

Let $\{q_1, q_2 \cdots q_i\}$ is in Q then Q' will be $[q_1, q_2 \cdots q_i]$ DFA : $\delta' : Q' \times \Sigma \to Q'$

$2^Q \times \Sigma \to 2^Q$

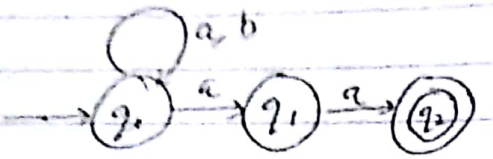
$q_0' = [q_0]$ corresponding to $\{q_0\}$

$F' = F \cup [q]$ if $[q]$ contain final state of M

$\delta'([q_1, q_2 \ldots q_i], a) = \{P_1, P_2 \ldots P_i\}$

if and only if $\delta(\{q_1, q_2, q_3 \ldots q_i\}, a) = \cup \delta(q_i, a)$

$= \{P_1, P_2 \ldots P_i\}$

Q.) 

Construct following NFA

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

$M' = Q', \Sigma, \delta', q_0', F'$

$Q' = \{\phi, [q_0], [q_1], [q_2], [q_0q_1], [q_0q_2], [q_1q_2], [q_0q_1q_2]\}$

$q_0' = [q_0]$

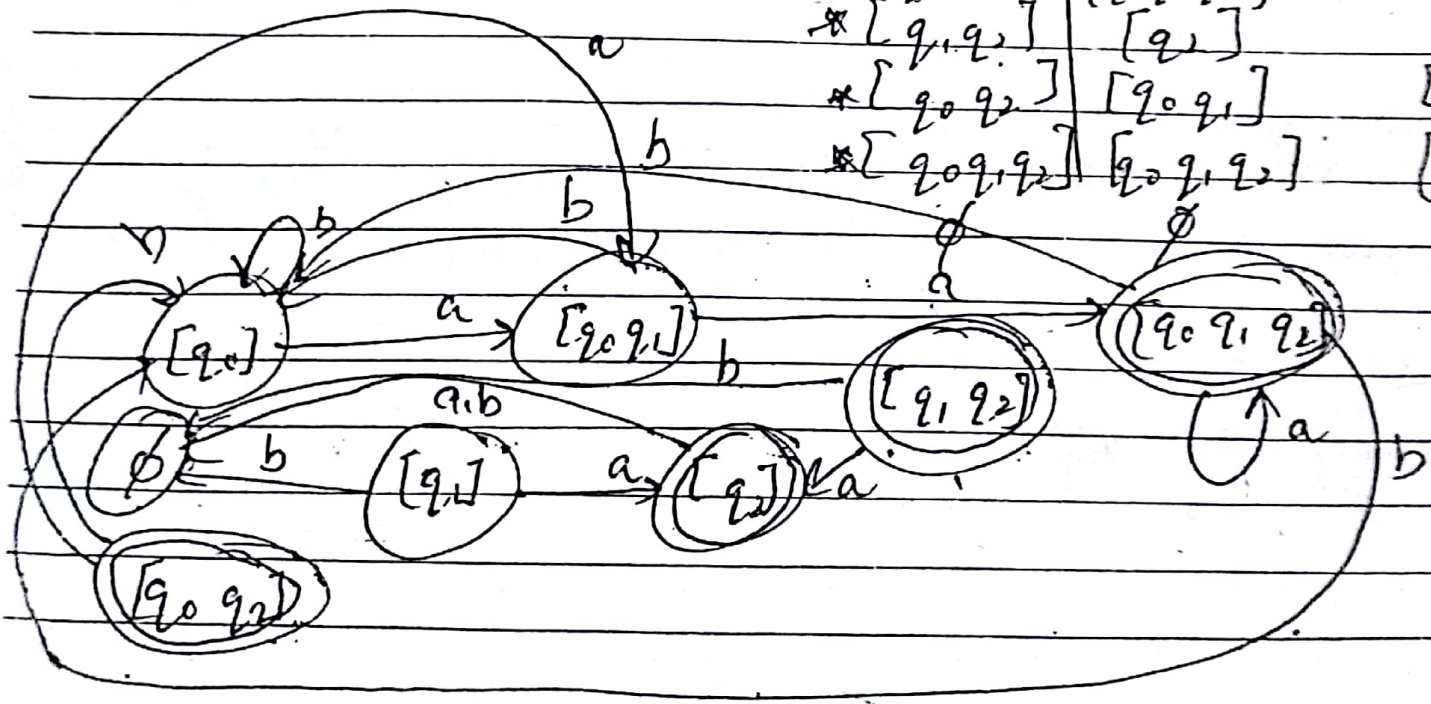
$F' = \{[q_2], [q_0q_2], [q_1q_2], [q_0q_1q_2]\}$

δ' = NFA

Present state	Next state a	Next state b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	ϕ
q_2	ϕ	ϕ

DFA

Present state	a	b
$[q_0]$	$[q_0q_1]$	$[q_0]$
$[q_1]$	$[q_2]$	ϕ
$*[q_2]$	ϕ	ϕ
$[q_0q_1]$	$[q_0q_1q_2]$	$[q_0]$
$*[q_1q_2]$	$[q_2]$	ϕ
$*[q_0q_2]$	$[q_0q_1]$	$[q_0]$
$*[q_0q_1q_2]$	$[q_0q_1q_2]$	$[q_0]$



How many address are necessary to address 2 MB of memory and 2 KB of memory?

→ 2^{21}

→ 2^{11}

Specify the no of registers & memory cells in 128 × 4 memory

No. of registers = 128 → 4 bit storage ; memory cell = 512

Addr. lines = 7

The memory map of a 4 KB memory chip begins at the location 2000H. Specify the addr. of the last location on the chip and the number of pages in the chip.

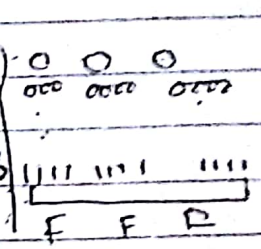
(registers)

4 KB → 12 addr. lines

2000 + 0000H = 2000

2000 + FFFH = 2FFFH

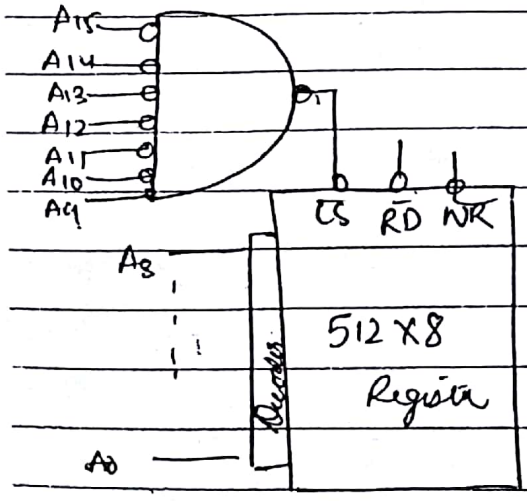
2FFF

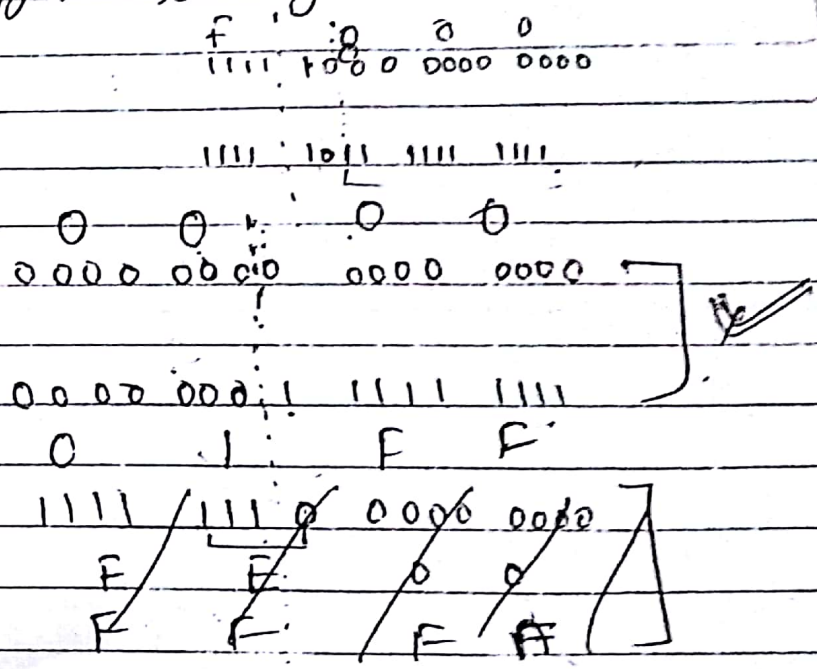


The memory address of the last location of 1 KB memory chip is given as $(FBFF)_{16}$. Specify the starting address.

1 KB → 2^{10}

10 addr. → F800



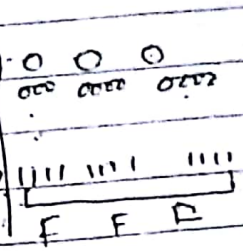


How many address are necessary to address 2MB of memory and 2KB of memory?

1.) Specify the no of registers & memory cells in 128 X 4 memory
No. of registers = 128 → 4 bit storage ; Memory cell = 512
Addl. lines = 7

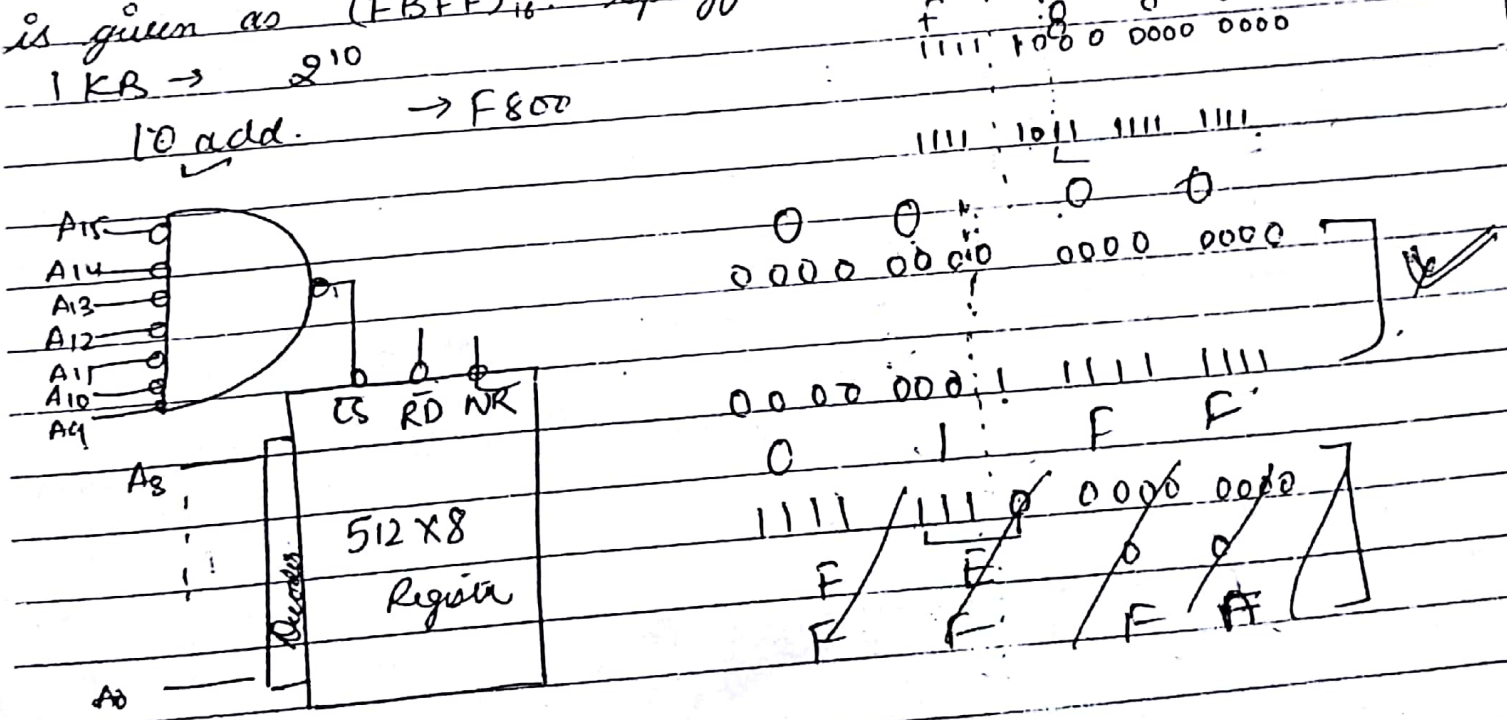
2) The memory map of a 4KB memory chip begins at the location 2000H. Specify the addl. of the last location on the chip and the number of pages in the chip.
(registers)

4 KB → 12 addl.
2FFF
2000 + 0000H = 2000
2000 + FFFH = 2FFFH



3) The memory address of the last location of 1 KB memory chip is given as $(FBFF)_{16}$. Specify the starting address.

1 KB → 2^{10}
10 addl. → F800



Date: 3/2/12

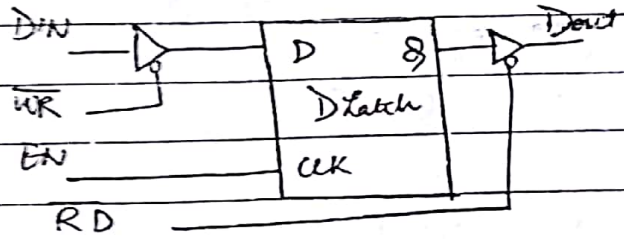
Memory:

Memory is an essential component of a micro computer system. It stores binary instruction and data for the micro processor. The memory is made up of registers and each register has a group of flip flops that store bits of information. These flip-flops are called memory cell. In a memory chip all registers are arranged in a sequence & identify by binary no called memory address.

To communicate with memory microprocessor unit should be - able to - select the chip or device - Identify the register - Read from or write in register.

NOTE: The MPU uses its address lines to send the address of a memory register and uses the data bus and control lines to read from or write into that register.

Flip-Flop or Latch as a storage element:



This latch which can store one binary bit is called as memory cell.

Qn) How many address locations can be address by a micro proc with 14 address lines.

0000 0000 0000 0000 → 0000

0011 1111 1111 1111

3 FFF 12