FLAT :- Formal language & automata theory.

① Introduction to Automata Theory.

② Regular languages & expressions.

③ CFG → context free Grammar & languages.

④ PDA & TM → Turing machine

PDA ↓ push down automata.

⑤ Lex & Yacc

PDA → DPDA Deterministic (one state)

PDA → 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 letters

$\Sigma = \{aa, aaaa, aaaaaa, \ldots\}$

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) $|1011011| = 5$ $|abbabc|$ length $\{\epsilon, 01, 1010, 0011\}$

(1 zero, 1 one) (2 zero, 2 one)

* Concatenation of strings :→

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

$x = a_1 a_2 a_3 \ldots a_i$

$y = b_1 b_2 b_3 \ldots b_j$

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

(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 :→ set of all strings
(L)

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

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.

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. ∈ - NFA

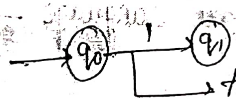
* Representation of finite automata :→

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

TD → Transition diagram

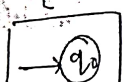
TT → Transition Table.

states



Arc / edges { Transition from q_0 on 1 to q_1

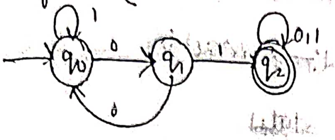
The start state is represented by :



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
states	→ q_0	q_1	q_0
	* q_1	q_1	q_2
	* q_2	q_2	q_2

Input alphabets

Here → indicates start state

* indicates final state

Scanned by CamScanner

DEFINATION :→

DFA :→ ① 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

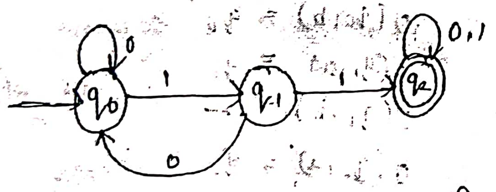
δ :→ Transition function.

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

Transition is mapping from states to input alphabets which yields states

q_0 → start state (Only one start state)

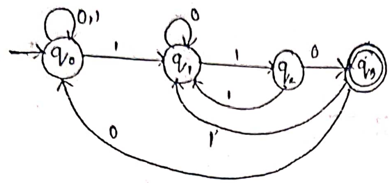
q_f → final state & is subset of Q
(i.e more than one final state)



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

$\Sigma = \{0, 1\}$ q_0 → start state

* Check wether the following transitions constitute DFA or not

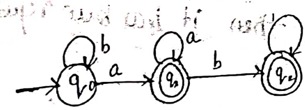
① 

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

Transition table

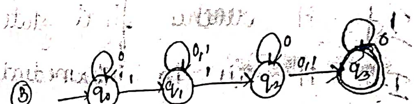
	δ	0	1
Q	$\rightarrow q_0$	q_0	$\{q_0,q_1\}$
	q_1	q_1	q_2
	q_2	q_3	q_1
	$* q_3$	q_0	q_1

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

Ⓐ 

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

Ⓑ 

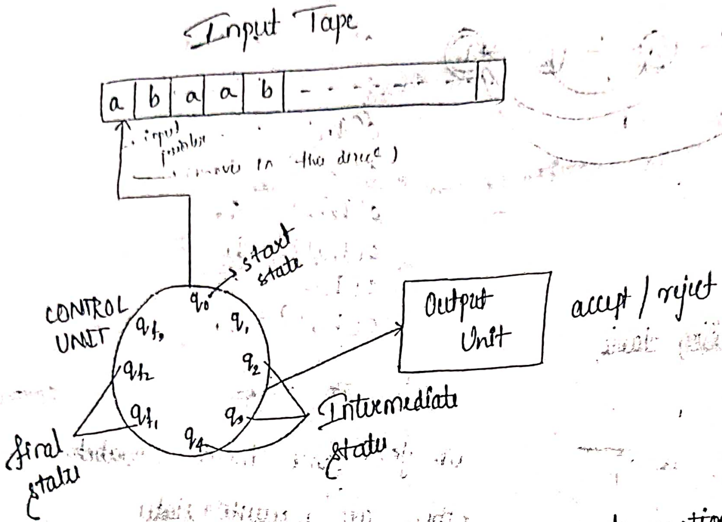
$\Sigma = \{0,1\}$
$Q = \{q_0, q_1, q_2, q_3\}$

Transitions

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

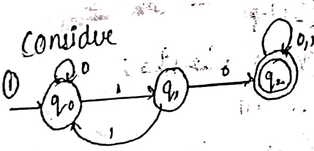
Scanned by CamScanner

* 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



$W_0 = 011010$

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

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

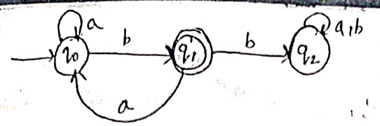
$W_1 = 0110101$

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

$w = abba$

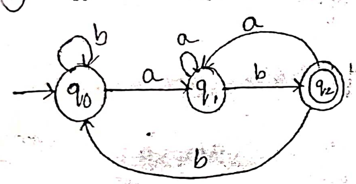
$q_0 \xrightarrow{a} q_0 \xrightarrow{b} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_2$ Rejected

$w = abbabb$

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

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

① abaab ② abb ③ abaa.



Sol: $\Rightarrow$ $Q = \{q_0, q_1, q_2\}$ $\Sigma = \{a, b\}$

$q_0 \rightarrow$ 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^* denotes 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.

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

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

$= \delta(q_0, a)$

$= q_1$

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

$= \delta(q_1, b)$

$= q_2$

For the prefix

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

$= \delta(q_2, a)$

$= q_1$

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

$= \delta(q_1, a)$

$= q_1$

$abaab \to \delta^*(q_0, abaab) = \delta(\delta^*(q_0, abaa), b)$

$= \delta(q_1, b)$

$= q_2$

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

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

$= \delta(q_0, a)$

$= q_1$

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

$= \delta(q_1, b)$

$= q_2$

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

$= \delta(q_2, a)$

$= q_1$

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

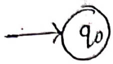
$= \delta(q_1, a)$

$= q_1$

Scanned by CamScanner

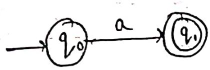
* Transition diagram for the following languages.

① ϕ → Empty language.

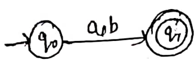


② ϵ → Empty string.

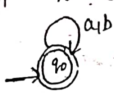
③ DFA to accept only one a



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 recognization 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.

Scanned by CamScanner

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

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

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

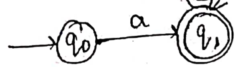
or

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

(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

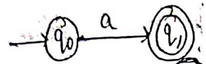
② Construct a DFA to accept strings of a's & b's having exactly one a.

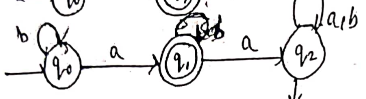
⇒ ~~L = { a, ab, abb, abbb .... }~~

L = { ba, bba, bbba, bab ...... }

① a

② $\Sigma = \{a, b\}$

③ 

④ 

Trap/dead state

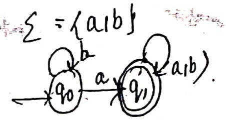
δ	a	b
$\rightarrow q_0$	q_1	q_0
$*q_1$	q_2	q_1
q_2	q_2	q_2

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

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

⇒ ① a

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

$Q = \{q_0, q_1\}$

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

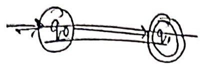
$q_0 \rightarrow$ start state

$q_1 \rightarrow$ final state.

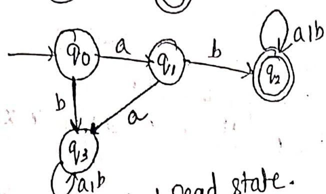
④ Construct a DFA of strings a & b such that sequence should begin with ab.

⇒ ① ab

② $\Sigma = \{a, b\}$ $\quad L = \{ab, ababba, abba, abbba, \ldots\}$



③



Trap state | Dead state.

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

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

$q_0 \to$ start state

$q_2 \to$ final state

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

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

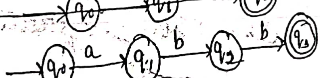
① abb

② $\Sigma = \{a, b\}$

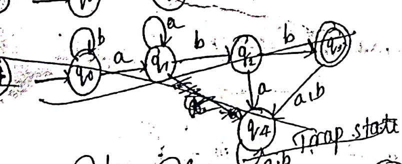
$L = \{abb, ababb, aabb, babb, aababb, bbabb, aaabb, \ldots\}$

③

③



④



Trap state

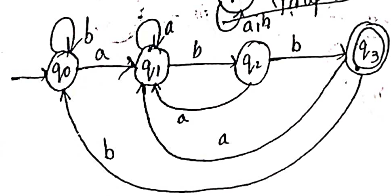
$Q = \{q_0, q_1, q_2, q_3\}$

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

$q_0 \to$ start state

$q_3 \to$ final state

④

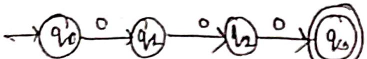


(6) Construct a DFA of strings 0's & 1's. such that it should have three consecutive zeros.

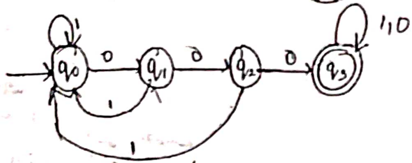
⇒ L = {000, 100011, 10000, 00011 ... }

(2) Σ = {0, 1} (1) 000

(3)



(4)



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

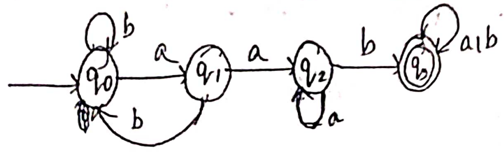
⇒ L { aab, baabaa, aaabb, aaaabb ... }

(1) aab

(2) Σ = {a, b}

(3)

(4)



Scanned by CamScanner

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

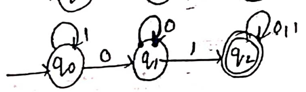
⑧ Construct a DFA to accept strings of 0's & 1's such that somewhere in the sequence 01 should appear in the string.

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

① 01

② $\Sigma = \{0,1\}$

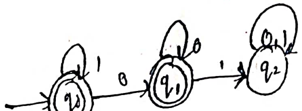
③ 

④ 

δ	0	1
q_0		
q_1		
q_2		

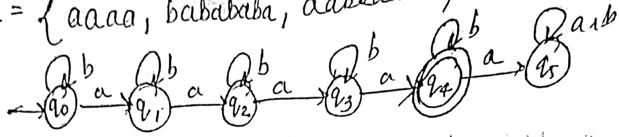
⑨ Construct a DFA to accept strings of 0's & 1's such that strings should not have a substring 01.

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



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

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

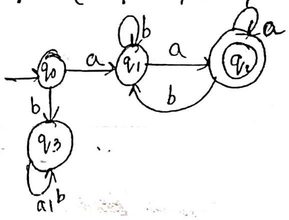


(11) Construct a DFA to accept the following language

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

→ Minimum string :→ aa

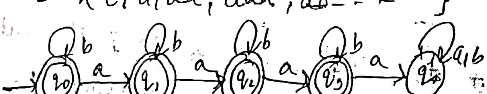
$L = \{aa, aba, aaa, abaa \ldots\}$



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

⇒ ~~L = {aaa, baaa, abaa~~

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

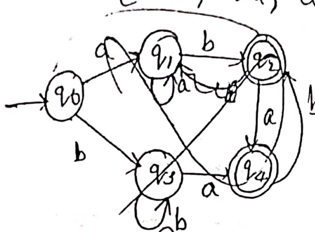


Here they are made final states because to accept zero a's, 1 a, 2a ...

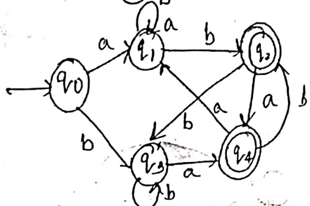
(13) $L = \{ w \in a,b \mid w(ab + ba) \}$ or

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

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

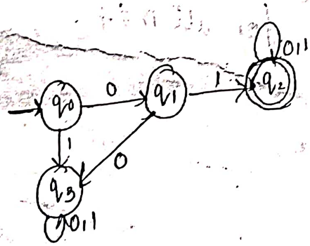


$\delta(q_1, a) =$



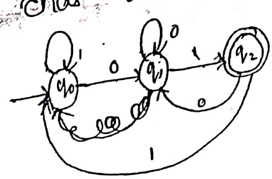
(14) Construct a DFA of strings 0's & 1's which begins with 01.

⇒ $L\{01, 0100, 0110 \ldots\}$



(15) Ends with 01.

⇒

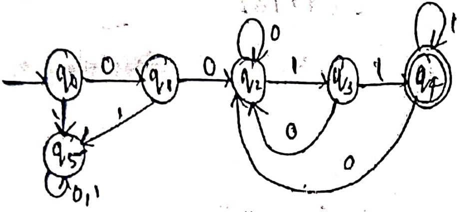


(18) Construct DFA for following languages :-

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

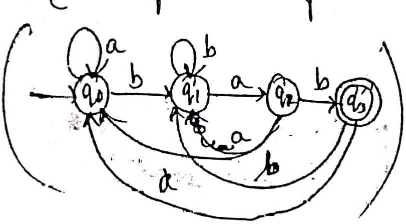
⇒ Begins with atleast two zeroes & ends with atleast two ones.

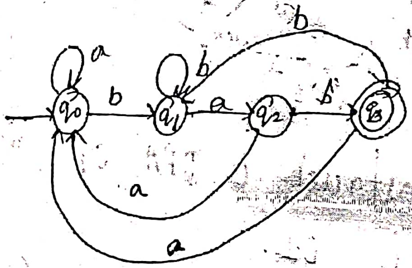
($*$ indicates zero or any times)



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

⇒ strings ending with bab.





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

$\rightarrow$

① DIVISIBLE BY K PROBLEMS :→

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

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

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.

Scanned by CamScanner

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

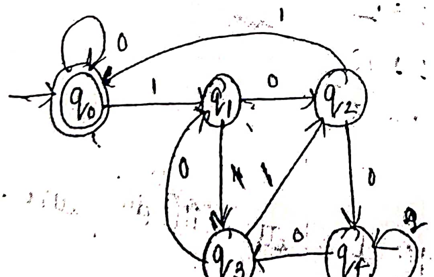
$\Rightarrow$ $r = 2$

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

$d = \{0,1\}$

$k = 5$

i	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.

⇒ $r = 10$

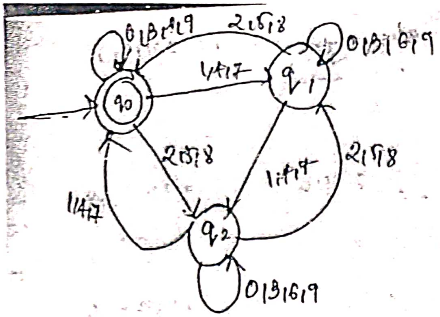
$i = 0, 1, 2$

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

$k = 3$

	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)	$(10 * 2 + 2) \bmod 3 = q_1$



(3) Construct a DFA of strings of 0's & 1's which are divisible by 5 such that DFA should not accept binary number 000

(4) 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

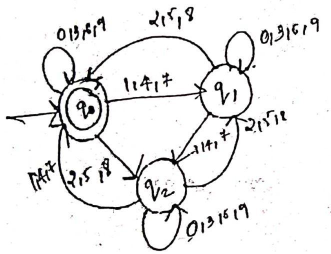
0 → q_0

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.

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

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

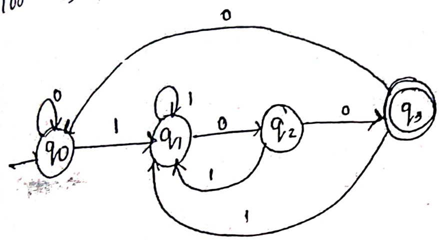
$Q = \{q_0, q_1, q_2, q_3\}$

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

$q_0 \rightarrow SS$ $q_3 \rightarrow fs$

	δ	0	1
ε	→q_0	q_0	q_1
1	q_1	q_2	q_1
10	q_2	q_3	q_1
100	*q_3	q_0	q_1

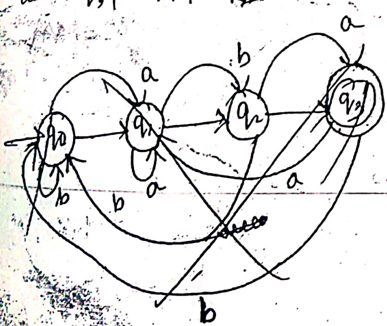
ϕ0, ϕ1
ϕ10, ϕ11
100, 101
1000 1001



ⓔ Ending with aba.

	δ	a	b
ε	→q_0	q_1	q_0
a	q_1	q_1	q_2
ab	q_2	q_3	q_0
aba	*q_3	q_1	q_2

ϕa, ϕb
aa, ab
aba, abb
abaa, abab.



NFA :→ Non deterministic finite automata

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

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

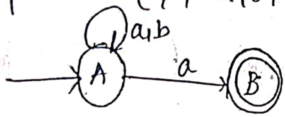
$\delta : Q \times \Sigma = 2^Q$ (power set) ⟶ NFA

Q is set of finite states

power set ⟶ set of all subsets

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

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



$\delta(A, a) = A$

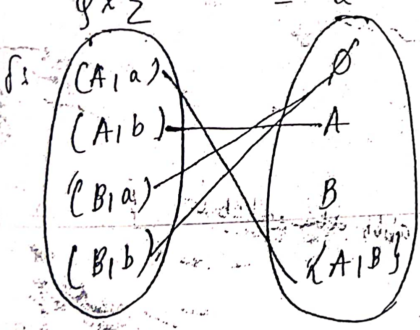
$\delta(A, b) = A$

$\delta(B, a) =$

$\delta(B, b) =$

δ	a	b
→A	{A, B}	A
*B	$\emptyset$	$\emptyset$

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



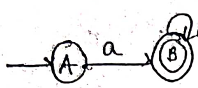
~~Examples of NFA~~ :-

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 .... }

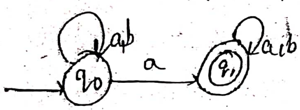


$\delta(A, a) \rightarrow B$

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

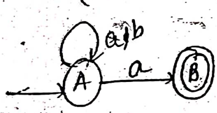
② L = { containing a } over a,b

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



③ L = { ends with a } over a,b

⇒

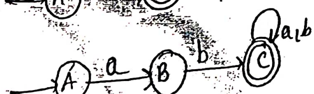


④ L = { starting with ab }

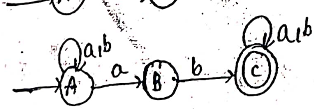
⑤ L = { containing ab }

⑥ L = { ends with ab }

④ L = { abbb, aba, abbb .... }

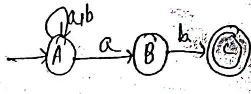


⑤



L = { abab, abbb, bbabb }

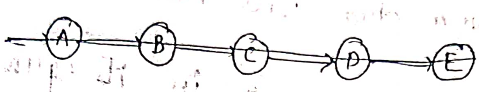
⑥

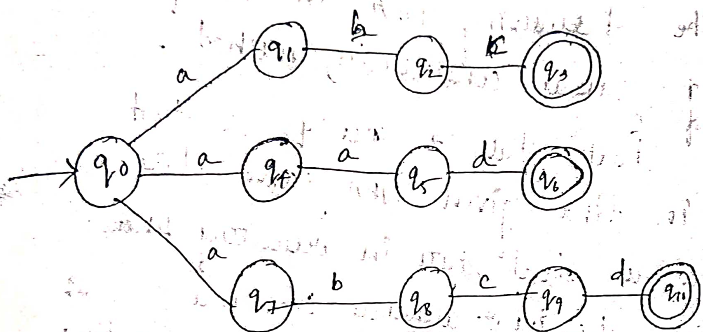


(7) In order to accept the following set of strings

abc
aad
abcd

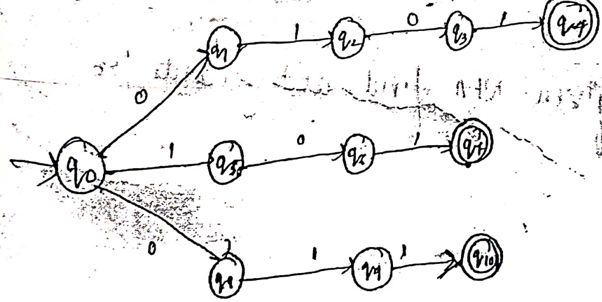
⇒





(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 is accepted by NFA

REGULAR EXPRESSION

ϕ :→ It is a regular expression which denotes Empty language

ϵ :→ It is a regular expression which denotes Empty string.

R is a regular expression corresponding to language L_R.

S is a regular expression then its corresponding language denoted by L_s

& the following cases holds good.

$R+S = L_R \cup L_s$

$R.S = L_R . L_s$

$R^* = (L_R)^*$

$S^* = (L_s)^*$

$*$ closure operator (Highest priority)

$+$ Union operator

$\cdot$ Concatenation operator (Least priority)

* Write regular expressions for the input alphabets ~~a & b~~.

① strings of any number of a's including NULL.

→ $\{\epsilon, a, aa, aaa \dots\}$

a^*

Scanned by CamScanner

② Obtain a regular expression for a string having atleast one a

$\Rightarrow$ a^+ or aa^*

③ strings having one atleast one a or one b

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

④ strings of a's & b's starts with a & ends with abb.

$\Rightarrow$ $a(a+b)^*abb$

⑤ strings of a's & b's having aa as substring.

$\Rightarrow$ $(a+b)^*aa(a+b)^*$

⑥ strings of zeros & ones should end with three consecutive zeros.

$\Rightarrow$ $(0+1)^*000$

⑦ Even number of a's.

$(aa)^*$

⑧ odd number of a's

~~$(aa)^*$~~, $a(aa)^*$

⑨ obtain regular expression of strings of a & b whose length should be equal to 2.

$(a+b)(a+b)$.

⑩ of length ≤ 2.

$\Rightarrow$ $L = \{\epsilon, a, b, ab, ba, aa, bb\}$ $\quad (\epsilon+a+b)(\epsilon+a+b)$

or $(\epsilon+a+b)^2$

Operations on Regular Expressions :-

① $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$

② $A \cdot B = \{x \mid x \in A \text{ \& } x \in B\}$

③ $A^* = \{x \mid x_1, x_2, x_3 \ldots\ldots x_k \mid k \geq 0\}$
$x_k \in A$

$A = \{pq, r\}$ $B = \{t, uv\}$

$A \cup B = \{pq, r, t, uv\}$

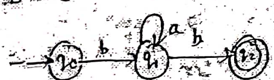
$A \cdot B = \{pqt, pquv, rt, ruv\}$

$A^* = \{pq, r, pqr, rpq, pqpq, pqpqpq, rrr \ldots\}$

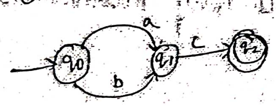
$B^* = \{$

Write DFA for the following regular expressions

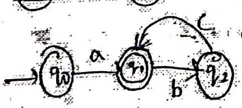
① ba^*b $L = \{bb, bab, baab \ldots\}$ (if * comes self loop)



② $(a+b)c$ $L = \{ac, bc\}$



③ 
$a(bc)^*$ $L = \{a, abc, abcbc$



Obtain regular expression for strings of a & b for the block of 4 consecutive symbol atleast 2 a's should be there

$a(a+b)(a+b)a$

$+$

$(a+b)\,\underline{a}\;\;\underline{a}\,(a+b)$

$+$

$\underline{a}\,(a+b)\,\underline{a}\,(a+b)$

$+$

$\underline{a}\,\underline{a}\,(a+b)\,(a+b)$

$+$

$(a+b)\,(a+b)\,\underline{a}\;\underline{a}$

$+$

$(a+b)\,\underline{a}\,(a+b)\,\underline{a}$

2] string of a's & b's having string of length atleast 2

$\Rightarrow$ $(a+b)(a+b)$ or $(a+b)^2$ $\rightarrow$ exact 2

$(a+b)(a+b)(a+b)^*$ $\rightarrow$ atleast 2.

3] write regular expression

$L = \{ a^{2n} b^{2m} \mid n, m \geq 0 \}$

$\Rightarrow (aa)^*(bb)^*$

4] odd length & multiples of 3, mod 3.

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

7] $|W| \geq 3$ atleast

$(a+b)^3 \, (a+b)^*$

8] Obtain a regular expression of strings of a's & b's such that 3 symbol from left hand should be b.

$\Rightarrow R = (a+b).(a+b).b\,(a+b)^*$

$R = (a+b)^2 . b . (a+b)^*$

9] From the right hand side 28th symbol should be a.

$\Rightarrow (a+b)^* . a . (a+b)^{27}$

10]

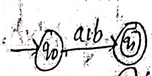
10] Write finite automata for the following regular expression.

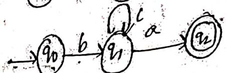
ϵ → Empty string 

ϕ → Empty Language 

a^* :- 

$(a+b)^*$ 

$(a+b)$ 

$b(c)^*a$ 

Minimize the DFA :-

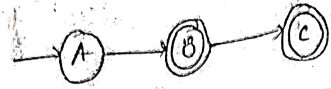
Table filling algorithm :-

* Indistinguishable / Equivalent states } Equivalence of states in FSM | FA
* Distinguishable states

Indistinguishable states :- If they are two states p & q on reading some input if the yield a final state then the pair (p,q) is called indistinguishable pair.

(1) $\left.\begin{array}{l}\delta(p,a) \in F \\ \delta(q,a) \in F\end{array}\right\} \Rightarrow (p,q) \rightarrow$ Indistinguishable pair.

(2) $\delta(p,a) \rightarrow \notin F \Rightarrow (p,q)$
$\delta(q,a) \rightarrow \notin F$

p,q → states   a → input   F → final state e.g. 

$(B,C) \in F$ ∴ IDP

~~(A,B)∉F~~

* Distinguishable states :→ (DP)

$\left.\begin{array}{l}\delta(P,a) \notin F \\ \delta(Q,a) \in F\end{array}\right\}$ DP $\left.\begin{array}{l}\delta(P,a) \in F \\ \delta(q,a) \notin F\end{array}\right\}$ DP

* Minimize the following ~~DFA~~ DFA ~~by~~ by applying table filling algorithm.

①

δ	a	b
→A	B	F
B	G	C
*C	A	C
D	C	G
E	H	F
F	C	G
G	G	E
H	G	C

start from 2nd state..

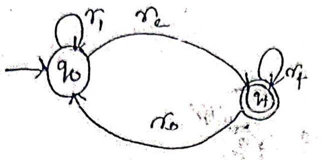
B	X						
C	X	X					
D		X	X				
E		X	X				
F	X	X	X				
G		X	X			X	
H	X		X	X	X		X
	A	B	C	D	E	F	G

start from 1st state to last ~~but~~ one.

δ	a	b
(A,B)	(B,G)	(F,C)
(A,D)	(B,C)	(F,G)
(A,E)	(B,H)	(F,F)
(A,F)	(B,C)	(F,G)
(A,G)	(B,G)	(F,E)
(A,H)	(B,G)	(F,C)

δ	a	b
(B,D)		
(B,E)		
(B,F)		
(B,G)		
(B,H)		

STATE ELIMINATION METHOD

① 

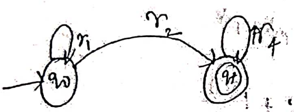
$r_1\ r_2\ r_3\ r_4$ } Regular Expressions

(RE) Regular Expression $= r_1^* r_2 (r_4 + r_3 r_1^* r_2)^*$

$q_0 \rightarrow$ start state $\quad q_f \rightarrow$ final state

② 

$RE = r^*$

③ 

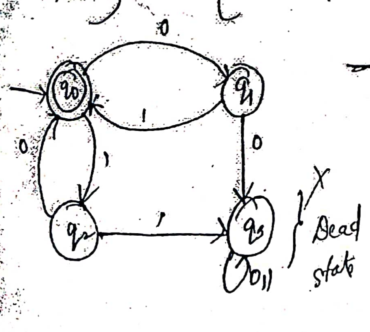
$RE = r_1^* r_2 r_4^*$

No connectivity between final & start start.

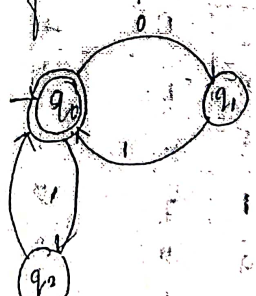
④ When start state becomes final state Regular expression can be written as.

$RE = r_1^* + r_1^* r_2 r_4^*$

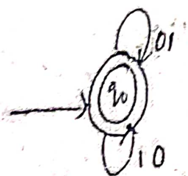
* Obtain the regular expression using state elimination method for the given finite automata.



→

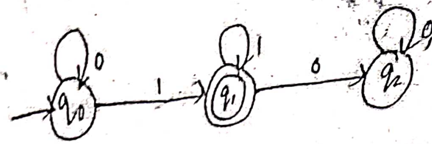


Scanned by CamScanner

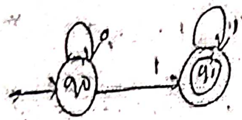


RE = $(01+10)^*$

(2)

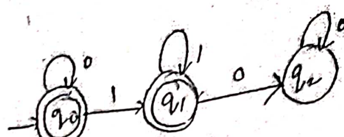


$\Rightarrow$

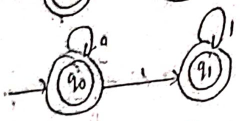


$\boxed{RE = 0^* 1 1^*}$

(3)



$\Rightarrow$



$0^* + 0^* 1 1^*$

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

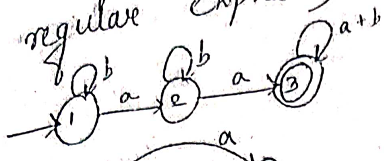
$0^* 1^*$

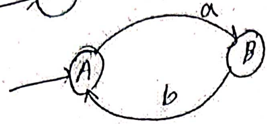
Using table filling algorithm minimize the given DFA

δ	a	b
$\rightarrow$ A	B	E
B	C	F
* C	D	H
D	E	H
E	F	I
* F	G	B
G	H	B
H	I	C
* I	A	E

* Using state elimination method obtain the regular expression :-

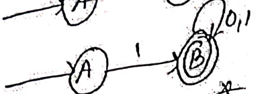
(1)





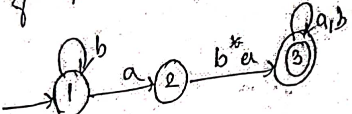
RE = a+b.

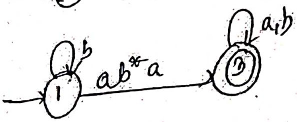
a^*b.



$1.(0+1)^*$

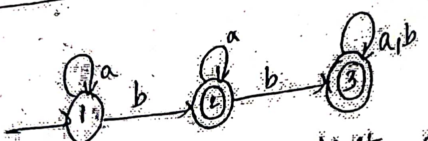
In state elimination method except start & final state the intermediate states should be eliminated by preserving the inputs.





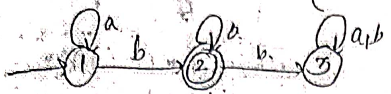
$\boxed{RE = b^* ab^* a\ (a+b)^*}$

(2)



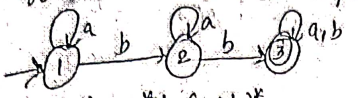
⟹ split to two finite automata with different final states.

2 as an final state



a^*ba^* (∵ 3 is trap state)

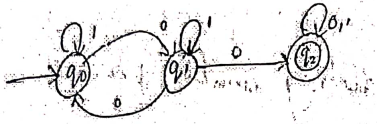
with 3 as an final state

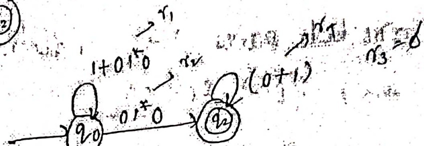


$a^*ba^*b(a+b)^*$

$$\boxed{\therefore\ RE = a^*ba^* + a^*ba^*b(a+b)^*}$$

(3)





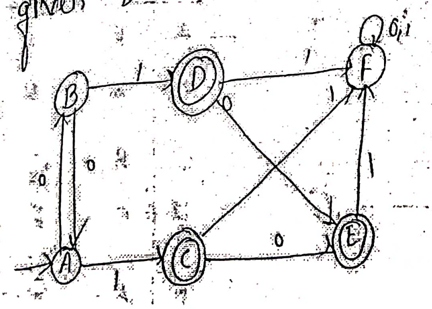
⇒ $\cancel{1^*(0+0)1^*0(0+1)^*}$

$\cancel{1^*01^*0(0+1)^*}$

$(1+01^*0)^*01^*0(0+1)^*$

* Using table filling algorithm minimize the given DFA

⇒



δ	0	1
→A	B	C
B	A	D
*C	E	F
*D	E	F
*E	E	F
F	F	F

B					
C	X	X			
D	X	X			
E	X	X			
F	X	X	X	X	X
	A	B	C	D	E

δ	0	1
(A,B)	(B,A)	(C,D)
(A,F)	(B,F)	(C,F)
(B,F)	(A,F)	(D,F)
(C,D)	(E,E)	(F,F)
(C,E)	(E,E)	(F,F)
(D,E)	(E,E)	(F,F)

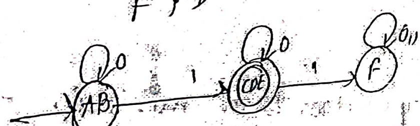
Unmarked pairs :→ Indistinguishable pairs

(AB) (DC)(EC)(ED)

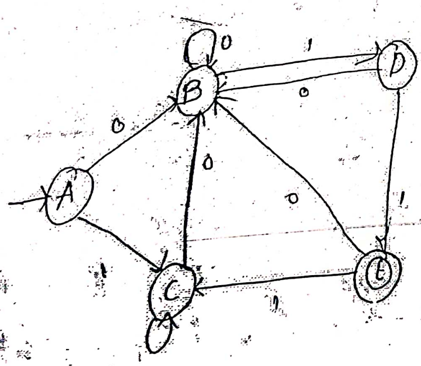
→ (AB) (CDE) } IDP

F } D

δ	0	1
→AB	AB	CDE
*CDE	CDE	F
F	F	F



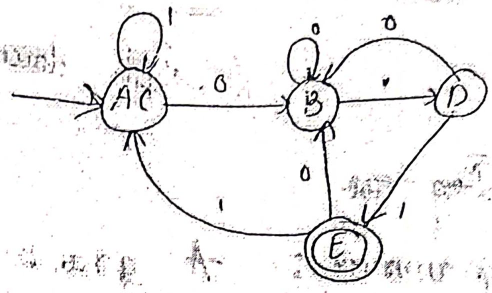
②



δ	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
*E	B	C

Scanned by CamScanner

B	X			
C		X		
D	X	X	X	
E	X	X	X	X
	A	B	C	D



CONVERSION FROM NFA to DFA

1. subset construction method.
2. Lazy evaluation method

Every DFA is NFA but vice versa is not true.

* convert the following NFA in its equivalent DFA using subset construction method.

⟹ 1) STEP 1] start state :→ since q_0 is start state in the given NFA same state continues as start state in resulting DFA.

2] Identify the input alphabets :→ since $\Sigma = \{a, b\}$ for the given problem the same thing constitutes as input alphabets for the resulting DFA.

3] Determine final state :→ since q_2 is final state in given NFA, if q_2 presents in any of the subset the entire set is considered as final state.

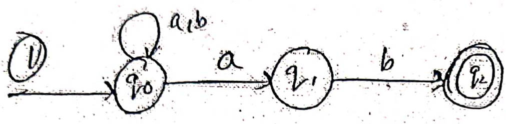
4) For the given NFA find out subsets for the states.

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

$M_D = \{\phi, q_0, q_1, q_2, \{q_0, q_1\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}$

⑤ Identify the transition for all the subsets in the resulting DFA.

δ	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	q_0
q_1	ϕ	q_2
$* q_2$	ϕ	ϕ

(1) 

① $\delta(\phi, a) = \phi$ $\quad$ $\delta(\phi, b) = \phi$

② $\delta(q_0, a) = \{q_0, q_1\}$ $\quad$ $\delta(q_0, b) = q_0$

③ $\delta(q_1, a) = \phi$ $\quad$ $\delta(q_1, b) = q_2$

④ $\delta(q_2, a) = \phi$ $\quad$ $\delta(q_2, b) = \phi$

⑤ $\delta(\{q_0, q_1\}, a)$
$= \delta(q_0, a) \cup \delta(q_1, a)$
$= \{q_0, q_1\} \cup \phi$
$= \{q_0, q_1\}$

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

⑥ $\delta(\{q_0, q_2\}, a)$
$= \delta(q_0, a) \cup \delta(q_2, a)$
$= \{q_0, q_1\} \cup \phi$
$= \{q_0, q_1\}$

$\delta(\{q_0, q_2\}, b)$
$= \delta(q_0, b) \cup \delta(q_2, b)$
$= q_1 \cup \phi$
$= q_1$

⑦ $\delta(\{q_1, q_2\}, a)$
$= \delta(q_1, a) \cup \delta(q_2, a)$
$= \phi \cup \phi$
$= \phi$

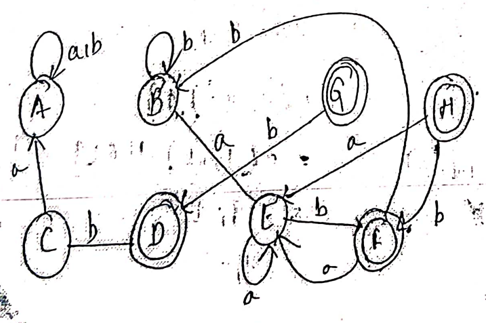
$\delta(\{q_1, q_2\}, b)$
$= \delta(q_1, b) \cup \delta(q_2, b)$
$= q_2 \cup \phi$
$= q_2$

(8) $\delta(\{q_1, q_2, q_3\}, a)$ $\quad\quad$ $\delta(\{q_1, q_2, q_3\}, b)$

$= \delta(q_1, a) \cup \delta(q_2, a) \cup \delta(q_3, a)$ $\quad = \{q_0, q_2\}$

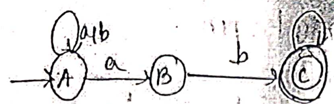
$= \{q_0, q_1\}$

	δ	a	b
A	ϕ	ϕ	ϕ
B	$\rightarrow q_0$	q_0q_1	q_0
C	q_1	ϕ	q_2
D	$* q_2$	ϕ	ϕ
	~~q_3~~	~~ϕ~~	
E	$\{q_0, q_1\}$	~~ϕ~~ q_0q_1	q_0q_2
F	$* \{q_0, q_2\}$	q_0q_1	q_0
G	$* \{q_1, q_2\}$	ϕ	q_2
H	$* \{q_0, q_1, q_2\}$	q_0q_1	q_0q_2



A is null

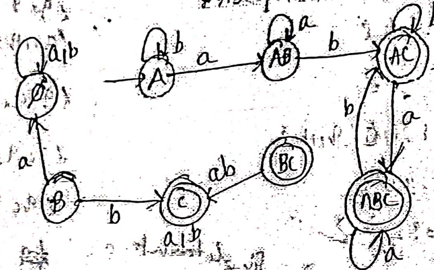
(2) Convert this NFA into its equivalent DFA.



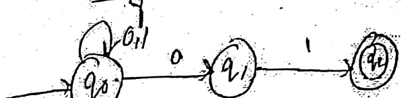
$\Rightarrow$

δ	a	b
$\rightarrow A$	{A, B}	A
B	$\emptyset$	C
*C	C	C

δ	a	b
$\rightarrow \emptyset$	$\emptyset$	$\emptyset$
A	{A, B}	A
B	$\emptyset$	C
*C	C	C
AB	{AB}	AC
*AC	{ABC}	AC
*BC	C	C
*ABC	ABC	AC



* Lazy evaluation method :-



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

$\delta(q_0, 1) = \{q_0\}$

$\delta(\{q_0 q_1\}, 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$

$= \{q_0 q_1\} \cup \emptyset = \{q_0 q_1\}$

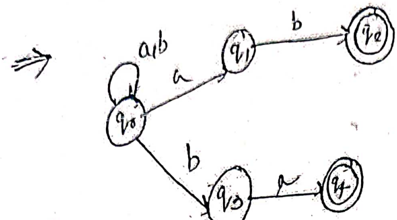
$\delta(\{q_0 q_1\}, 1) = \delta(q_0, 1) \cup \delta(q_1, 1)$

$= q_0 \cup q_2 = \{q_0 q_2\}$

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

Scanned by CamScanner

① Construct an NFA that ends with ab & ba. Convert it into its equivalent DFA.



δ	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0 q_3\}$
q_1	ϕ	q_2
$* q_2$	ϕ	ϕ
q_3	q_4	ϕ
$* q_4$	ϕ	ϕ

① for input a

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

for input b

$\delta(q_0, b) = \{q_0 q_3\}$

② $\delta(\{q_0, q_1\}, a) = \delta(q_0, a) \cup \delta(q_1, a)$

$= \{q_0 q_1\} \cup \phi$

$= \{q_0 q_1\}$

$\delta(\{q_0 q_1\}, b) = \delta(q_0, b) \cup \delta(q_1, b)$

$= \{q_0 q_3\} \cup q_2$

$= \{q_0 q_3 q_2\}$

Singletaint

$\{q_0\}$
$\{q_0, q_1\}$
$\{q_0 q_3 q_2\}$
$\{q_0 q_3\}$
$\{q_0 q_1 q_4\}$

③ $\delta(\{q_0, q_3\}, a) = \delta(q_0, a) \cup \delta(q_3, a)$

$= \{q_0 q_1\} \cup q_4$

$= \{q_0 q_1 q_4\}$

$\delta(\{q_0 q_3\}, b) = \delta(q_0, b) \cup \delta(q_3, b)$

$= \{q_0 q_3\} \cup \phi$

$= \{q_0 q_3\}$

④ $\delta(\{q_0 q_3 q_2\}, a) = \delta(q_0, a) \cup \delta(q_3, a) \cup \delta(q_2, a)$

$= \{q_0 q_1\} \cup q_4 \cup \phi$

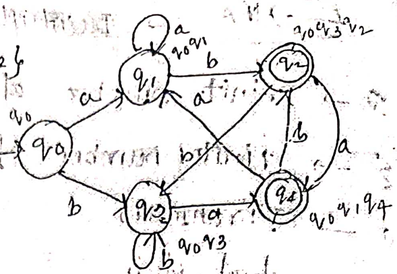
$= \{q_0 q_1 q_4\}$

$\delta(\{q_0 q_3 q_2\}, b) = \{q_0 q_3\}$

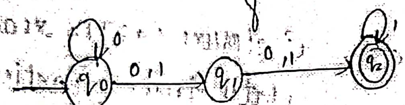
Scanned by CamScanner

(5) $\delta(\{q_0 q_1 q_4\}, a) = \{q_0 q_1\}$

$\Rightarrow \delta(\{q_0 q_1 q_4\}, b) = \{q_0 q_3 q_2\}$



(2) Convert the given NFA into its equivalent DFA



δ	0	1
$\rightarrow q_0$	$\{q_0 q_1\}$	q_1
q_1	q_2	q_2
$* q_2$	$\emptyset$	q_2

① $\delta(q_0, 0) = \{q_0 q_1\}$

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

② $\delta(\{q_0 q_1\}, 0) = \{q_0 q_1 q_2\}*$

$\delta(\{q_0 q_1\}, 1) = \{q_1 q_2\} * \rightarrow B$

③ $\delta(q_1, 0) = q_2 * \rightarrow q_2$

$\delta(q_1, 1) = q_2$

④ $\delta(q_2, 0) = \emptyset$

$\delta(q_2, 1) = q_2$

⑤ $\delta(\{q_0 q_1 q_2\}, 0) = \{q_0 q_1 q_2\}$

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

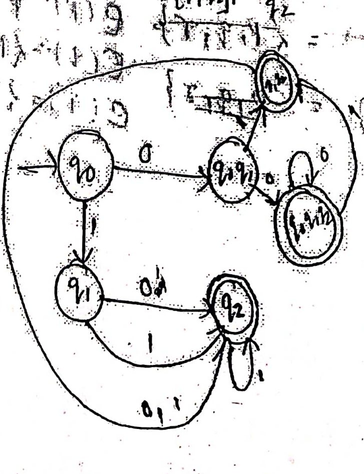
⑥ $\delta(\{q_1 q_2\}, 0) = q_2$

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

DFA

δ	0	1
$\rightarrow q_0$	$[q_0 q_1]$	q_1
q_1	q_2	q_2
$[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_1 q_2]$
$* q_2$	$\emptyset$	q_2
$* [q_0 q_1 q_2]$	$[q_0 q_1 q_2]$	$[q_1 q_2]$
$* [q_1 q_2]$	q_2	q_2

square represents one state i.e. single only for DFA



ϵ NFA :→ quintuple having $Q, \Sigma, \delta, q_0, F$

Q → finite number of states

Σ → finite number of input alphabets

δ → Transitions

q_0 → start state

F → final state

$\delta : Q \times \{\Sigma \cup \epsilon\} = 2^Q$

ϵ closure :→ The reachable state from a particular state on ϵ transition

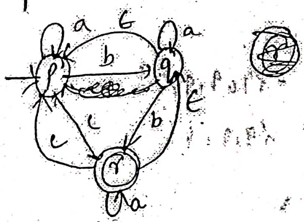
$\epsilon(q_0) = \{q_0\ q_1\ q_2\}$

$\epsilon(q_1) = \{q_1, q_0\}$

$\epsilon(q_2) = \{q_2\}$

Write ϵ closure for given transition table :→

δ	ϵ	a	b	c
→ p	ϕ	p	q	r
q	p	q	r	ϕ
* r	q	r	ϕ	p



~~$\epsilon(p) = \{p, q, r\}$~~

~~$\epsilon(q) = \{p, q, r\}$~~

$\epsilon(p) = p$

$\epsilon(q) = \{p, q\}$

$\epsilon(r) = \{p\ q\ r\}$

Scanned by CamScanner

* Convert the given ∈ NFA into its equivalent DFA :-

→ 

$E(q_0) = \{q_0 q_1 q_2\}$ $E(q_1) = \{q_1 q_2\}$, $E(q_2) = q_2$

$\delta(\{q_0 q_1 q_2\}, a) = q_0 = E(q_0) = \{q_0 q_1 q_2\} = A$

$\delta(\{q_0 q_1 q_2\}, b) = q_1 = E(q_1) = \{q_1 q_2\} = B$

$\delta(\{q_0 q_1 q_2\}, c) = q_2 = E(q_2) = q_2 = C$

$\delta(\{q_1 q_2\}, a) = \emptyset$ ~~$= E(q_0) = \{q_0 q_1 q_2\}$~~

$\delta(\{q_1 q_2\}, b) = q_1 = E(q_1) = q_1 q_2 = B$

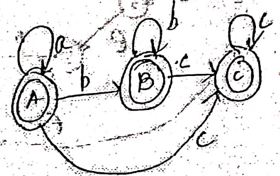
$\delta(\{q_1 q_2\}, c) = q_2$ $E(q_2) = q_2 = B$

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

$\delta(q_2, b) = \emptyset$

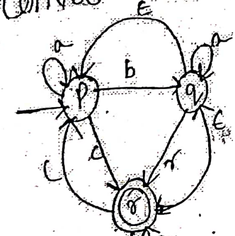
$\delta(q_2, c) = q_2 = E(q_2) = q_2 = C$

δ	a	b	c
→*A	A	B	C
*B	∅	B	C
*C	∅	∅	C



As q_2 is present in all state all states are final state.

* Convert into its equivalent DFA :-



δ	∈	a	b	c
→p	∅	p	q	r
q	p	q	r	∅
*r	q	r	∅	p

$E(P) = P$

$\delta(P, a) = P = E(P) = P$

$\delta(P, b) = q = E(q) = \{Pq\}$

$\delta(P, c) = r = E(r) = \{Pqr\}$

$\delta(\{Pq\}, a) = \{Pq\}$

$\delta(\{Pq\}, b) = \delta(P, b) \cup \delta(q, b)$

$= q \cup r$

$= \{Pq\} \cup \{Pqr\}$

$= Pqr$

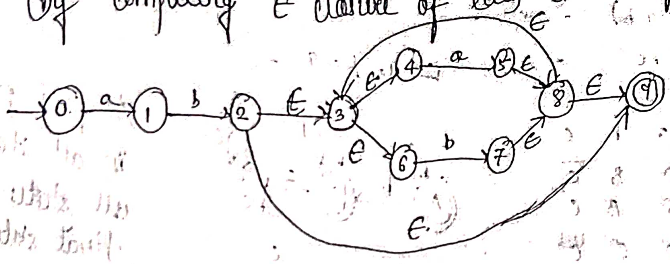
$\delta(\{Pq\}, c) = \{Pqr\}$

$= \{Pqr\}$

$\delta(\{pqr\}, a) = pqr$
$\delta(\{pqr\}, b) = pqr$
$\delta(\{pqr\}, c) = pqr$

δ	a	b	c
$\to p$	p	pq	pqr
pq	pq	pqr	pqr
*pqr	pqr	pqr	pqr

* Convert the given ϵ-NFA into its equivalent DFA by computing ϵ closure of each & every state.



$\Rightarrow$ $E(0) = 0$
$E(1) = 1$
$E(2) = \{2, 3, 4, 6, 9\}$
$E(3) = \{3, 4, 6\}$
$E(4) = \{5, 8, 9, 3, 4, 6\} = \{3, 4, 5, 6, 8, 9\}$
$E(6) = 6$
$E(7) = \{3, 4, 6, 7, 8, 9\}$
$E(8) = \{3, 4, 6, 8, 9\}$
$E(9) = 9$

$\delta(0,a) = 1 = E(1)$

$\delta(0,b) = \phi$

$\delta(1,a) = \phi$

$\delta(1,b) = 2 = E(2) = \{2,3,4,6,9\}$

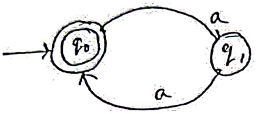
$\delta(\{2,3,4,6,9\}, a) = \{$

$\delta(\{2,3,4,6,9$

COUNTER PROBLEMS

① Construct a DFA where $\Sigma = \{a\}$ accepting only even number of a's.

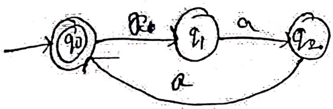
$\Rightarrow L = \{\epsilon, aa, aaaa \ldots\}$ $|W| \bmod 2 = 0$



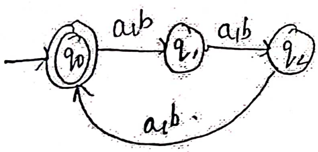
q0 is final state to accept 0 number of a's.

② $|W| \bmod 3 = 0$

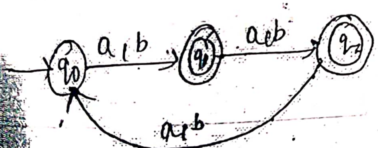
$\Sigma = \{a\}$



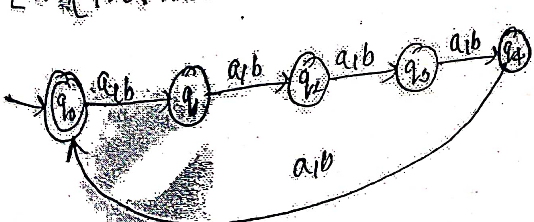
③ $|W| \bmod 3 = 0$



④ $L = \{|W| \bmod 3 \neq 0\}$ $\Sigma = \{a,b\}$



⑤ $L = \{|W| \bmod 5 = 0\}$ $\Sigma = \{a,b\}$

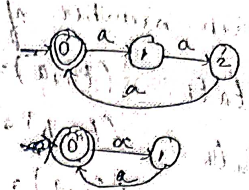


Scanned by CamScanner

(6) $L = \{ |w| \bmod 3 \geq |w| \bmod 2 \}$. $\Sigma = \{a\}$

$\Rightarrow$ $L = \{ |w| \bmod 3 \geq |w| \bmod 2 \}$

$\downarrow$ M_1 $(0,1,2)$ $\downarrow$ M_2 $(0,1)$.



$M_1 \times M_2 = ((0,0), (0,1), (1,0), (1,1), (2,0), (2,1))$

$\delta(\overset{q_0}{(0,0)}, a) = \delta\overset{M_1}{(0,a)}\ \overset{M_2}{(0,a)}$

$= \underset{q_1}{\delta(1,1)}$.

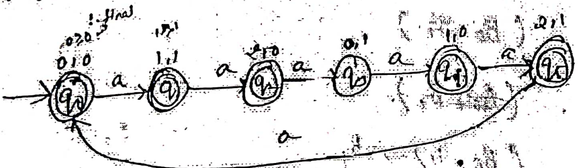
$\delta((1,1),a) = (2,0) \rightarrow q_2$

$\delta((2,0),a) = (0,1) \rightarrow q_3$

$\delta((0,1),a) = (1,0) \rightarrow q_4$

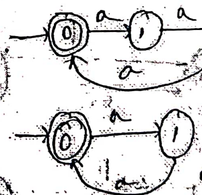
$\delta((1,0),a) = (2,1) \rightarrow q_5$

$\delta((2,1),a) = (0,0)$

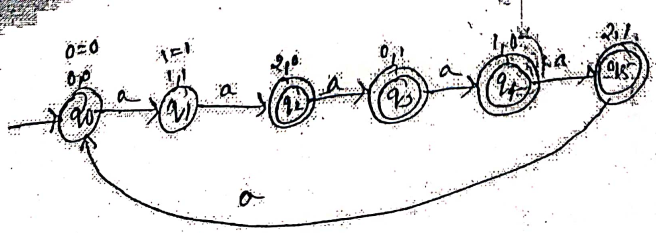


(7) $L = \{ |w| \bmod 3 \neq |w| \bmod 2 \}$ $M_1 \rightarrow (0,1,2)$

$\Rightarrow$ $L = \{ |w| \bmod 3 \neq |w| \bmod 2 \}$



$M_1 \times M_2 = ((0,0), (0,1), (1,0), (1,1),$
$(2,0), (2,1))$



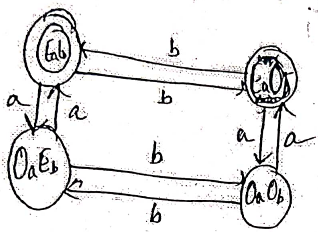
⑧ Construct a DFA to have even number of a's & even number of b's over a,b.

⇒ $E_a E_b$ $(E_a, a) = O_a$

$E_a O_b$ $(E_b, b) = O_b$

$O_a E_b$ $(O_a, a) = E_a$

$O_a O_b$ $(O_b, b) = E_b$.



ⓐ $N_a |W| \bmod 3 \geq N_b |W| \bmod 2$

$\downarrow$ $\downarrow$

0, 1, 2 0, 1

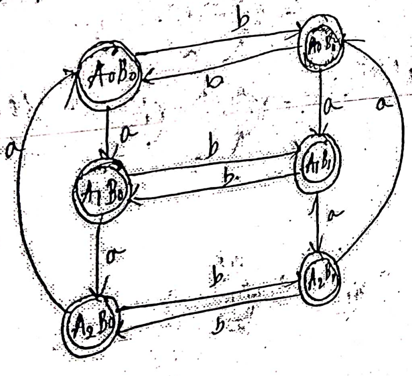
$(A_0\ A_1\ A_2)$ $(B_0\ B_1)$

~~(B_0, B_1)~~.

$(A_0, a) = A_1$ $(B_0, b) \rightarrow B_1$

$(A_1, a) = A_2$ $(B_1, b) \rightarrow B_0$

$(A_2, a) = A_0$



Scanned by CamScanner

(10) $\{ w \mid N_a(w) \bmod 5 = 0$ and $N_b(w) \bmod 3 = 0 \}$

$(A_0\, A_1\, A_2\, A_3\, A_4)$ $(B_0\, B_1\, B_2)$

$\Rightarrow \{ (A_0 B_0)\ (A_0 B_1)\ (A_0 B_2)\ (A_1 B_0)\ (A_1 B_1)\ (A_1 B_2)\ (A_2 B_0)\ (A_2 B_1)$
$(A_2 B_2)\ (A_3 B_0)\ (A_3 B_1)\ (A_3 B_2)\ (A_4 B_0)\ (A_4 B_1)\ (A_4 B_2) \}$

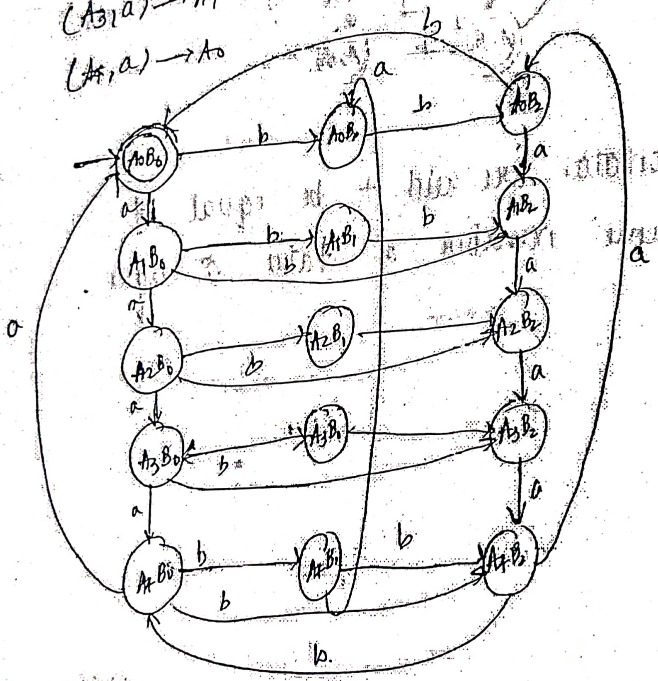
$(A_0, a) \rightarrow A_1$ $(B_0, b) \rightarrow B_1$

$(A_1, a) \rightarrow A_2$ $(B_1, b) \rightarrow B_2$

$(A_2, a) \rightarrow A_3$ $(B_2, b) \rightarrow B_0$

$(A_3, a) \rightarrow A_4$

$(A_4, a) \rightarrow A_0$



(11) $L = \{ w \in (a+b)^* \mid n_a(w) \bmod 3 \geq n_b(w) \bmod 2 \}$

$\Rightarrow$

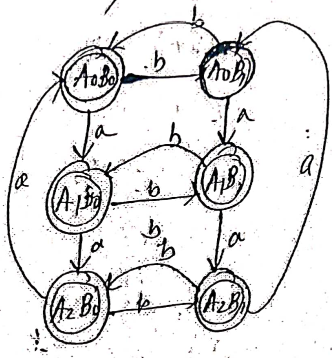
0, 1, 2   0, 1

$A_0 A_1 A_2 \times B_0 B_1$

$(A_0B_0)\ (A_0B_1)$

$(A_1B_0)\ (A_1B_1)$

$(A_2B_0)\ (A_2B_1)$



NOTE :→

Two finite automatas are said to be equal if they have same number of states & edges.