Mathematical notation & terminology :-

Set - It is a group of objects represented as unit. The object in a set are called element or member of set.

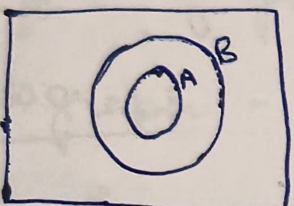
Representation of set : $\{1, \ldots\}$

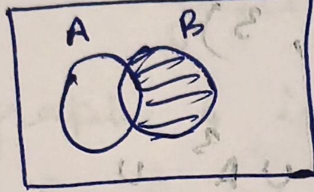
or

$\{x : x \geq 10\}$

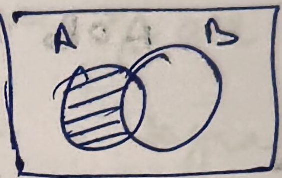
$A \subseteq B \rightarrow$ A is subset of B

$A \subset B \rightarrow$ A is proper subset of B.





B - A



A - B

Sequences & tuples -

A sequence of object is a list of those objects which are in some order. In set order of object does not matter ~~at~~ a lot but it does matter in sequences. Repetition of objects does matter in the case of sequences but it doesn't matter in the case of set. As with the case of set sequences can be finite or infinite, finite sequence is called tuple. A tuple can be called as 2-tuple or 3-tuple or k-tuple depending upon the no. of objects in the sequence.

Power set

Set of all subsets

$A \times B$ -

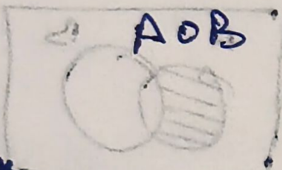
$A = \{1, 2, 3, 4\}$; $B = \{4, 5\}$

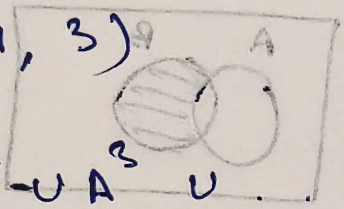
$A \times B = \{(1,4), (1,5), (2,4), (2,5), (3,4), (3,5), (4,4), (4,5)\}$

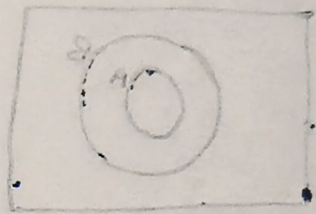
Ring operator

$A \circ B$ $A = \{1, 2\}$ $B = \{2, 3\}$ same

$A \circ B = (1, 3)$







$A^* = A \cup A^2 \cup A^3 \cup \ldots$

$A^+ = A \cup A^2 \cup A^3 \ldots$ - empty

~~Set~~

$A^2 = A \circ A$

Function

A function is an object that set up an input & output relationship. A function takes an input and produces an output $f(a) = b$. Function sometimes also known as mapping. The set of all possible input to a function is called domain of that function. The set of all possible output is called range. $f : D \to R$

Types of function -

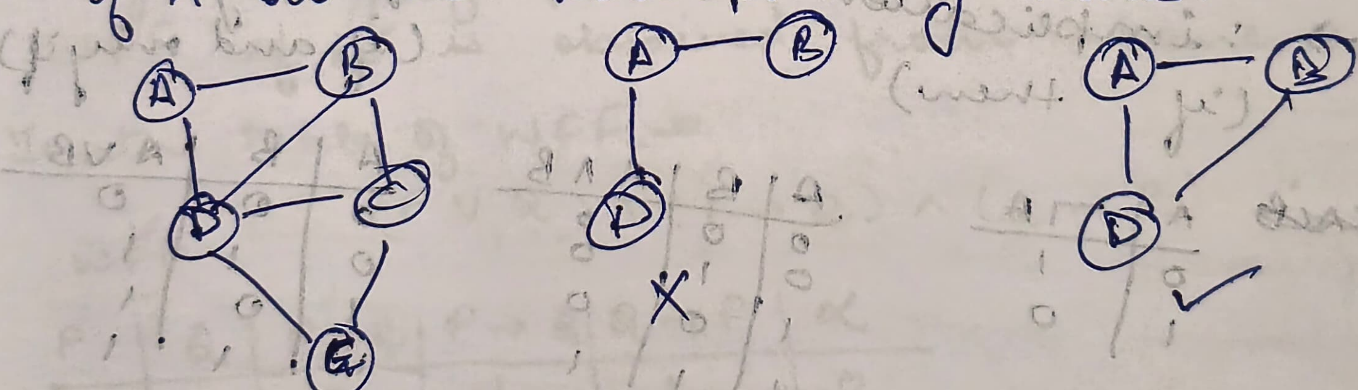
1. One - to - one
2. One - to - many
3. Many - to - one
4. Many - to - many
5. Onto function → Uses all the object of range.

Graph - Set of vertices and edges. $G(V, E)$ ↓ undirected

Directed graph : $G(V, \vec{E})$

Degree of graph → highest degree of all the nodes

Subgraph - A graph G is a subgraph of H if the nodes of G is subset of nodes of H and the edges on G are the edges of H on the corresponding nodes.



Path - A path in a graph is a sequence of nodes connected by edges.

Simple path - A path that does not repeat any nodes.

Cycle - Starting and ending node is same in a path.

Simple cycle - It must contain at least 3 nodes and repeats only first & last node.

Alphabets, String & Languages -

Alphabets - Finite set of symbols

Languages →
- Natural language (Eng, Hindi . . .)
- Programming language
- Mathematical language

Boolean logic - It is a mathematical system built on 2 values i.e. true & false (1 & 0).

Boolean operation - ~~The~~

$\neg$ → not / negation

$\wedge$ → and / conjunction

$\vee$ → or / disjunction

$\oplus$ → XOR

$\rightarrow$: implication (if then)

$\leftrightarrow$: equality (if and only if)

A	$\neg$A
0	1
1	0

A	B	A $\wedge$ B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A $\vee$ B
0	0	0
0	1	1
1	0	1
1	1	1

A	B	A $\oplus$ B	A $\leftrightarrow$ B	A $\rightarrow$ B
0	0	0	1	1
0	1	1	0	1
1	0	1	0	0
1	1	0	1	1

Proposition (Statement)

In boolean logic proposition or statement is classified as declarative sentences to which only one value can be assigned (true, false).

Well formed formula (WFF) :-

It is defined as -

- If P is a propositional variable then it is said to be WFF.
  P = Delhi is capital of India
- If α is in WFF, the negation of α is also in WFF.
- If α and β are in WFF then $\alpha \vee \beta$, $\alpha \wedge \beta$, $\alpha \to \beta$, $\alpha \leftrightarrow \beta$ are also to be in WFF.
- Any string of symbol is said to be in WFF if it is obtained from above rules.

Truth table of WFF →

$$\underset{\text{WFF}}{\alpha} = (P \vee Q) \wedge (P \to Q) \wedge (Q \to P)$$

P	Q	$P \vee Q$	$P \to Q$	$Q \to P$	α
0	0	0	1	1	0
0	1	1	1	0	0
1	0	1	0	1	0
1	1	1	1	1	1

Tautology - A WFF is said to be a tautology if last column is true always.

Contradiction - A WFF is said to be a contradiction if last column is false always.

Equivalence of 2 WFF - Two WFF α and β are said to be equivalent if $\alpha \leftrightarrow \beta$ is a tautology. $\alpha \equiv \beta$

$$\alpha = (P \vee Q) \rightarrow ((P \vee R) \rightarrow (R \vee Q))$$

P	Q	R	$P \vee Q$	$P \vee R$	$R \vee Q$	$(P \vee R) \rightarrow R \vee Q)$	α
0	0	0	0	0	0	1	1
0	0	1	0	1	1	1	1
0	1	0	1	0	1	1	1
0	1	1	1	1	1	1	1
1	0	0	1	1	0	0	0
1	0	1	1	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1

α is not a tautology.

Tutorial -1 -

1. If P represents 'This book is good' and Q represents 'The book is cheap', write the following ~~sent~~ sentences in symbolic form:
(a) This book is good & cheap
(b) This book is not good but cheap
(c) This book is costly but ~~cheap~~ good
(d) This book is neither good nor cheap
(e) This book is either good or cheap

2. Construct the truth table for
$$\alpha = (P \vee Q) \rightarrow ((P \vee R) \rightarrow (R \vee Q))$$

3. Show that $(P \wedge Q) \vee (P \wedge \neg Q) \equiv P$.

4. Obtain a disjunctive normal form of
$P \vee (\neg P \to (Q \vee (Q \to \neg R))$

5. For a given formula α, the truth values are given in the table below. Find the principal disjunctive normal form.

P	Q	R	α
T	T	T	T
T	T	F	F
T	F	T	F
T	F	F	T
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

1. (a) $P \wedge Q$ (b) ~~$\neg P \vee Q$~~ $\neg P \wedge Q$ (c) $P \wedge \neg Q$
(d) $\neg(P \vee Q)$ (e) ~~$P \vee Q$~~ ~~$P \oplus Q$ $\neg(P \leftrightarrow Q)$~~ $P \vee Q$

2. $\alpha = (P \vee Q) \to ((P \vee R) \to (R \vee Q))$

P	Q	R	$P \vee Q$	$P \vee R$	$R \vee Q$	$(P \vee R) \to (R \vee Q)$	α
0	0	0	0	0	0	1	1
0	0	1	0	1	1	1	1
0	1	0	1	0	1	1	1
0	1	1	1	1	1	1	1
1	0	0	1	1	0	0	0
1	0	1	1	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1

$A \to B = \sim A \vee B$

3. $(P \wedge Q) \vee (P \wedge \neg Q) \equiv P$

P	Q	$P \wedge Q$	$P \wedge \neg Q$	$(P \wedge Q) \vee (P \wedge \neg Q)$
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	1	1	0	1

$(P \wedge Q) \vee (P \wedge \neg Q) \Rightarrow P \wedge (Q \vee \neg Q)$

$\Rightarrow P \wedge T$

$\Rightarrow P$

$\cancel{(P \wedge Q \vee Q) \wedge (P \wedge Q \vee \neg Q)}$

$\cancel{(P \wedge Q) \wedge (P \wedge T)}$

$\cancel{P \wedge Q \wedge P}$

$\cancel{P \wedge Q}$

$\cancel{Q \wedge P \vee P}$

$\cancel{P}$

4. Sum of product → Disjunctive normal form.

POS → Conjunctive normal form.

$P \vee (\neg P \to (Q \vee (Q \to \neg R)))$

$\Rightarrow P \vee (\neg P \to (Q \vee (\sim Q \vee \neg R)))$

$\Rightarrow P \vee (\neg P \to (\overset{T}{\cancel{[illegible]}}))$

$\Rightarrow P \vee (P \vee \overset{T}{\cancel{\neg R}})$

$\Rightarrow \cancel{R \vee \neg R}\ T$

Equivalence b/w WFF -

If α and β are WFF then if $\alpha \leftrightarrow \beta$ is Tautology then α and β are equivalent.

Q: If P = It is raining, Q = I have the time, R = I will go to a movie. Write sentences using these propositions.

1. If it is not raining and I have time, then I will go to a movie.

   $(\neg P \wedge Q) \rightarrow R$

2. It is raining and I will not go to movie.

   $P \wedge \neg R$

3. I will go to a movie only if it is not raining.

   $\cancel{R \leftrightarrow Q \rightarrow P}$. $R \rightarrow \neg P$

~~Normal forms~~ -

Logical identities -

1. $P \vee P \equiv P$; $P \wedge P \equiv P$
2. $P \vee Q \equiv Q \vee P$; $P \wedge Q \equiv Q \wedge P$
3. $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$; $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$
4. $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$;

   $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
5. $P \vee (P \wedge Q) \equiv P$; $P \wedge (P \vee Q) \equiv P$
6. $\neg (P \vee Q) \equiv \neg P \wedge \neg Q$; $\neg (P \wedge Q) \equiv \neg P \vee \neg Q$
7. $P \equiv \neg(\neg P)$

8. $P \vee \neg P \equiv T$; $P \wedge \neg P \equiv F$

9. $P \vee T \equiv T$; $P \wedge T \equiv P$; $P \vee F \equiv P$; $P \wedge F \equiv F$

10. $(P \to Q) \wedge (P \to \neg Q) \equiv \neg P$

11. $P \to Q \equiv \neg Q \to \neg P$

12. $P \to Q \equiv \neg P \vee Q$

Q: Prove that $(P \wedge Q) \vee (P \wedge \neg Q) \equiv P$

$\Rightarrow P \wedge (Q \vee \neg Q)$ → By using distributive law

$\Rightarrow P \wedge T$ → By using I_8

$\Rightarrow P$ → By using I_9

Normal forms -

① Sum of product (Disjunctive normal form)

② Product of sum (conjunctive normal form)

In normal form there will be only $\wedge, \vee, \neg$.

Disjunctive normal form -

1. Eliminate implication and equality by using logical identities.
2. Eliminate negation before sum or products resulting in the formula which is having negation only before the propositional variable.

Q: $P \vee (\neg P \to (Q \vee (Q \to \neg R)))$

$\Rightarrow P \vee (\neg P \to (Q \vee (\neg Q \vee \neg R)))$ → ②

$\Rightarrow P \vee (\neg P \to T)$

$\Rightarrow P \vee (P \vee T) \Rightarrow T$

$= P \vee Q \vee \neg Q \vee \neg R$ → ans

$\Rightarrow P \vee (P \vee (Q \vee \neg Q \vee \neg R))$ — ③

$\Rightarrow P \vee Q \vee \neg Q \vee \neg R$ — ④ → ans.

Principle disjunctive normal form

Finite State Automata (FSA) -

It is 5-tuple quantity of Q = finite set of state, Σ = finite set of alphabets, δ = transition funcⁿ $= \Sigma \times Q = Q$

q_0 = initial state, F = final state $F \subseteq Q$

FSA can be designed with the help of -

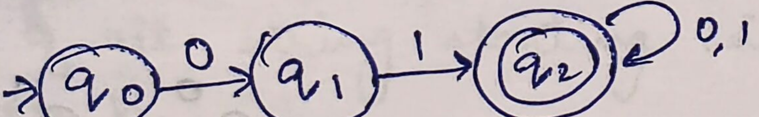
① Transition graph

② Transition table

Transition graph -

○ → state ◎ → final state

→○ → initial state ⟶ → Transition from one state to another

eg: 

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

$\delta = Q \times \Sigma = Q$ → (in form of table)

$F = q_2$ (Final state can be more than 1)

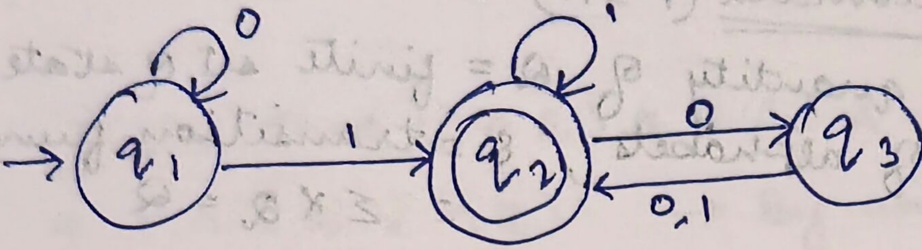
→ only one initial state

→ Transition table

state ↓ / I/P →	0	1
→ q_0	q_1	–
q_1	–	q_2
q_2	q_2	q_2

Q:

	0	1
$\rightarrow q_1$	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2



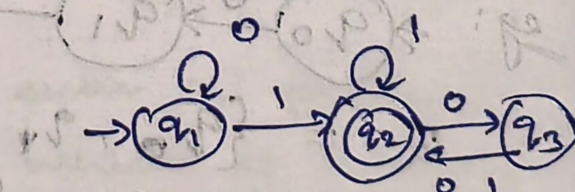
Language of a FA -

String is accepted if it ends in the final state.

word/string is valid if by processing the last symbol of that string the automata must be in final state.

If A is the set of all string that machine M accepts, we say that A is the language of that machine and it is written as $L(M) = A$.

Acceptibility of a string -



$x = 10011 \rightarrow$ accepted

$\delta(q_1, 10011) = (q_2, 0011)$

$= (q_3, 011)$

$= (q_2, 11)$

$= (q_2, 1)$

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

↳ final state.

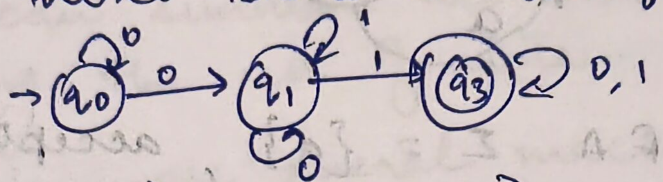
FA → DFA (Deterministic finite state automata)

FA → NDFA (Non-deterministic FSA)

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

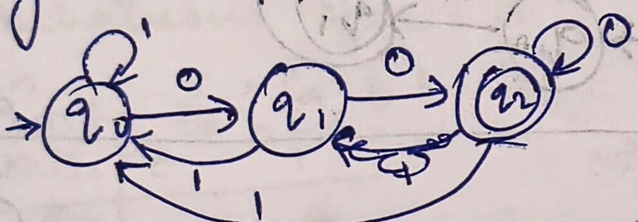
In NDFA, for current input and current state, the next state is not unique.
In DFA next state is unique.

NDFA →

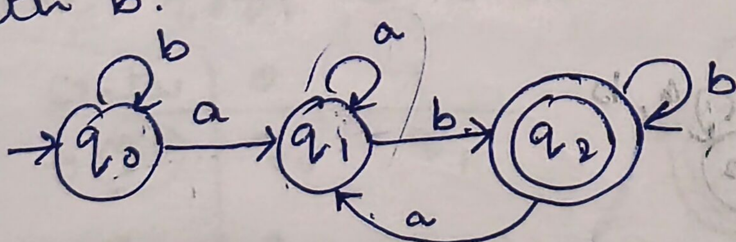


$\delta(q_0, 0) = (q_0\ q_1)$

Q: Design a FA over $\Sigma = \{0,1\}$, the set of all string that ends in 00.

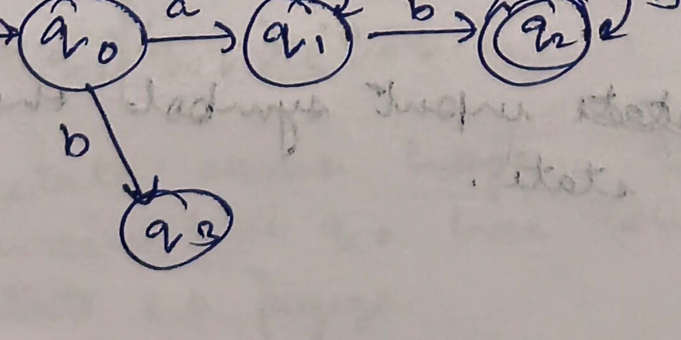


Q: Design FA over $\Sigma = \{a, b\}$, that accept set of all string starting with a & ending with b.

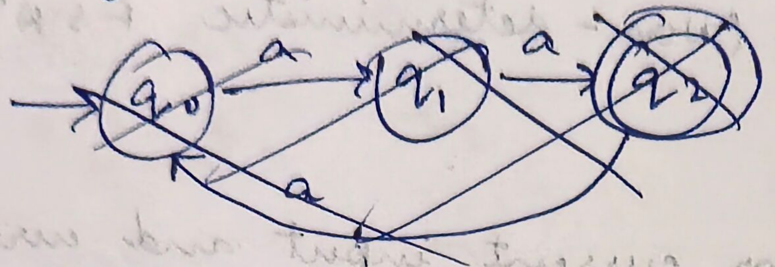


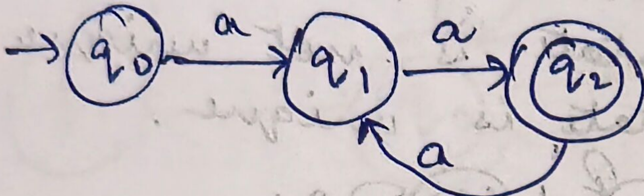
→ bab accept ho raha hai

ans :

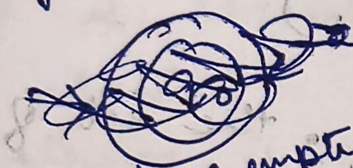


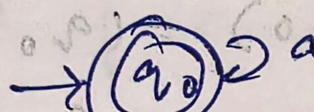
Q: Design FA $\Sigma = \{a\}$, that accept even no. of a.





Q: Design FA $\Sigma = \{a\}$, accepts $a^n : n \geq 0$.

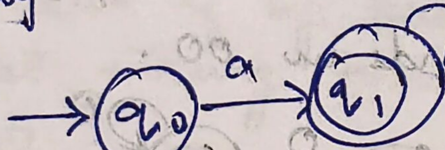




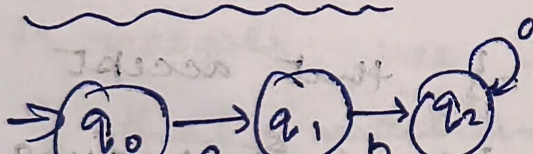
including empty string

Excluding empty string

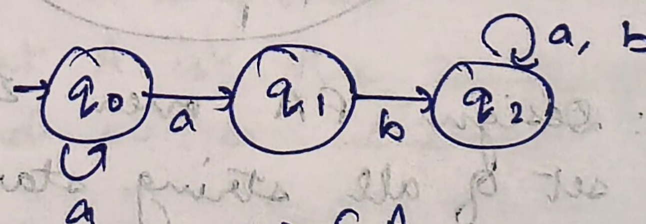
$n > 0$



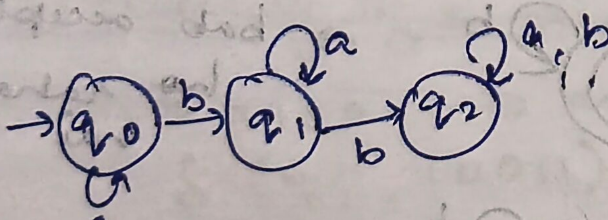
NDFA vs DFA.



NDFA



NDFA



DFA

In DFA for every ~~state~~ input symbol there must be a transition state.

state	a	b
$\rightarrow q_0$	$(q_0 q_1)$	q_2
q_1	q_0	q_1
q_2	–	$(q_0 q_1)$

Create new state and put a self loop on it

$\rightarrow$

State	a	b
$\rightarrow q_0$	$(q_0 q_1)$	q_2
~~q_1~~	~~q_0~~	~~q_1~~
~~q_2~~		~~$q_0 q_1$~~
$(q_0 q_1)$	$(q_0 q_1)$	$(q_1 q_2)$
$(q_1 q_2)$	q_0	$(q_0 q_1)$
q_1	q_0	q_1
q_2	q_3	$q_0 q_1$
q_3	q_3	q_3

Tutorial-2 -

1. Let $R = \{(1, 2), (2, 3), (2, 4)\}$ be a relation in $\{1, 2, 3, 4\}$. Find R^+.

2. If $A = \{a, b\}$ and $b = \{b, c\}$. Find -
   - (a) $(A \cup B)^*$
   - (b) $(A \cap B)^*$
   - (c) $A^* \cup B^*$
   - (d) $A^* \cap B^*$
   - (e) $(A - B)^*$
   - (f) $(B - A)^*$

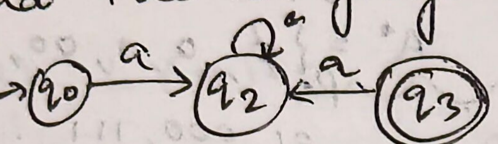
3. A finite automaton M has state set $Q = \{q0; q1; q2\}$. Its input alphabet is $\Sigma = \{0; 1\}$, with $q0$ being the initial state and $F = \{q_0\}$. The transition function δ is given by the following transition table.

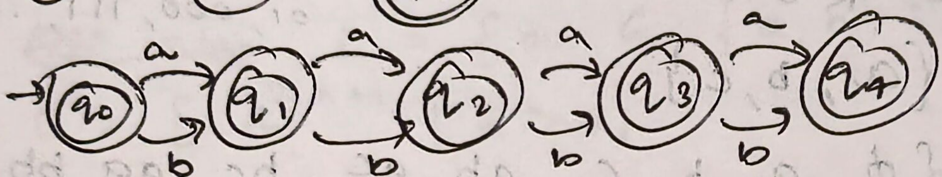
δ	0	1
q_0	q_2	q_1
q_1	q_1	q_0
q_2	q_2	q_2

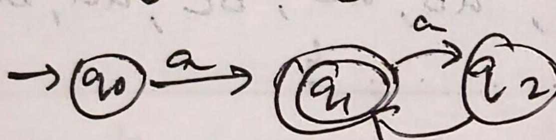
Give the state diagram for M and describe the language L (M).

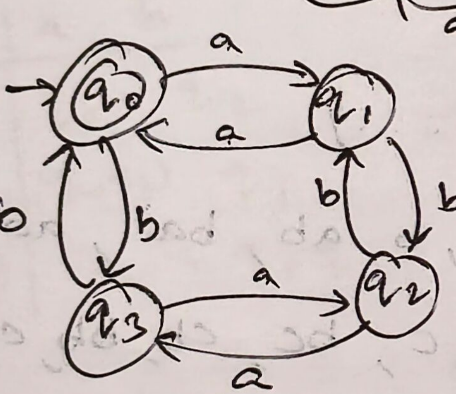
4. Sheep go "baa!" or "baaa!" or "baaaa!" and so on. They do not go "baaa" or "ba!" or "!" or "". Construct a finite automaton (with input alphabet $\Sigma = \{a; b; !;\}$ that recognizes "sheep-talk".

5. Find the language of following FA :

(a) 

(b) 

(c) 

(d) 

6. Find the FA for the following language :-

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

(c) $L = \{a, b, aa, bb, aaa, bbb, \ldots\}$

(d) $L = \{$ set of all strings over $\{0, 1\}$ starting with 0 ending with 1 $\}$.

1. $R^+ = R \cup R^2 \cup R^3 \ldots$

$R = \{(1,2), (2,3), (2,4)\}$

$A.B = \{\langle a, b\rangle \mid \langle a, c\rangle \in A$ and $\langle c, b\rangle \in B$ and $\langle a, b\rangle \notin A, B\}$.

$R^2 = R.R = \{(1,3), (1,4)\}$

$R^3 = R^2 . R = \phi \qquad R^4 = R^3 . R = \phi$

$R^+ = R \cup R^2 = \{(1,2), (2,3), (2,4), (1,3), (1,4)\}$

2. $A = \{a, b\} \qquad B = \{b, c\}$

$A = \{0, 1\}$

$A^* = \{\phi, 0, 1, 00, 11, 10, 01, 000, 111, \ldots\}$

(a) $(A \cup B)^*$

~~$R^* = R \cup R^2 \cup R^3 \ldots R^n$~~

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

$(A \cup B)^* = \{\phi, a, b, c, ab, ac, bc, aa, bb, cc, aab, abc \ldots\}$

(b) $(A \cap B) = \{b\}$

$(A \cap B)^* = \{\phi, b, bb, bbb \ldots\}$

(c) $A^* \cup B^*$

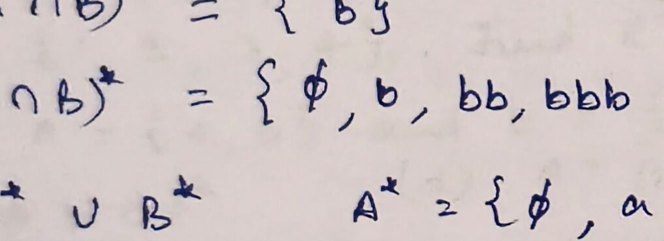
$A^* = \{\phi, a, b, ab, ba, aa, bb, \ldots\}$

$B^* = \{\phi, b, c, bc, cb, bb, cc, \ldots\}$

$A^* \cup B^* = \{\phi, a, b, c, ab, ba, aa, bb, bc, cb, cc, \ldots\}$

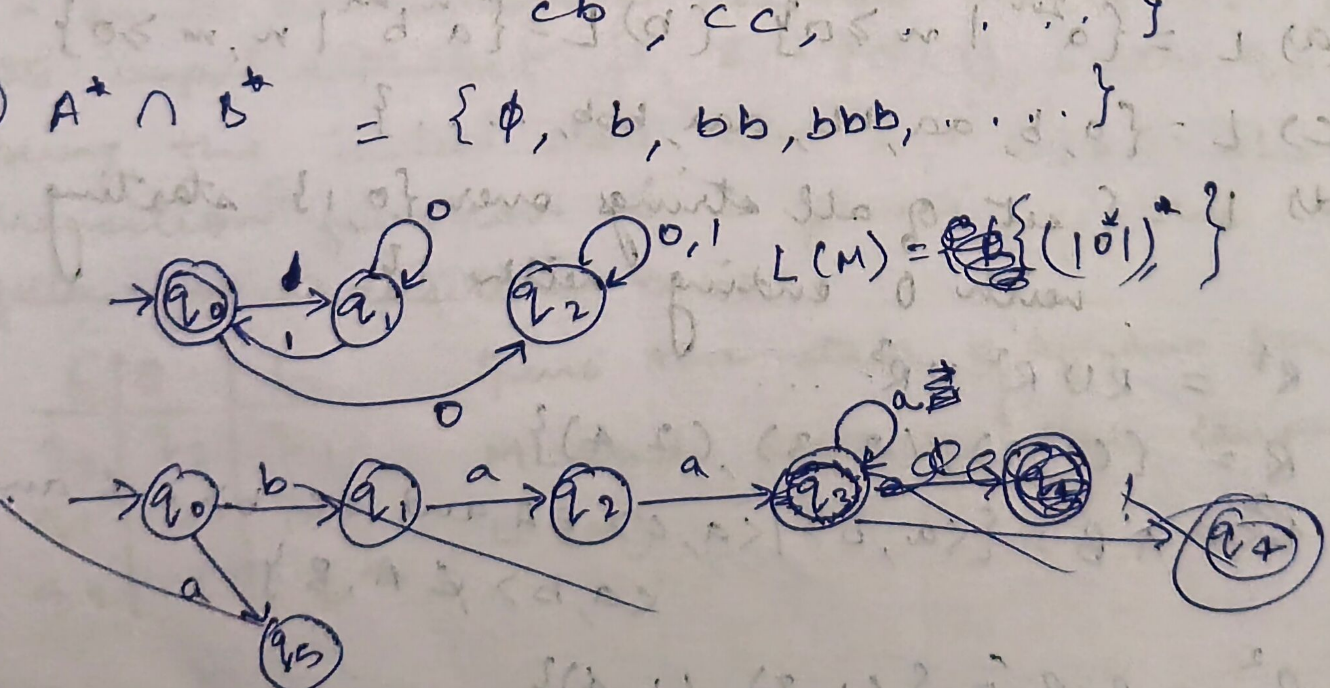
(d) $A^* \cap B^* = \{\phi, b, bb, bbb, \ldots\}$

3.



$L(M) = \{(10^*1)^*\}$

4.



Q:

	a	b
$\rightarrow q_0$	q_0	q_2
q_1	$(q_0 q_2)$	q_1
q_2	q_1	$q_0 q_1$

$\rightarrow$

	a	b
$\rightarrow q_0$	q_0	q_2
q_2	q_1	$(q_0 q_1)$
$(q_0 q_1)$	$(q_0 q_2)$	$(q_1 q_2)$
$(q_0 q_2)$	$(q_0 q_1)$	$(q_0 q_1 q_2)$
$(q_1 q_2)$	$(q_0 q_1 q_2)$	$(q_0 q_1)$
$(q_0 q_1 q_2)$	$(q_0 q_1 q_2)$	$(q_0 q_1 q_2)$
q_1	$(q_0 q_2)$	q_1

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

__Regular language__ - Set of strings that are accepted by the FA.

* $\Sigma = \{0, 1\}$ $0^n 1^n \rightarrow$ ~~not~~ FA can't be made.

__Melay machine__ $\rightarrow$ The FA have ~~boundary~~ binary output i.e. either they accept the string or they don't accept the string. This acceptibility of strings was decided on basis of reaching final state from initial state. If we remove this restriction and construct a model where output can be chosen from other alphabet. This output function $z(p)$ is dependent on present state $q(t)$ and present input symbol $x(t)$. So we can write -

$$z(t) = \lambda(q(t) . x(t))$$ where, λ is called output function and this machine is called Melay machine. ~~If z(t) = λ(q~~

- If $z(t) = \lambda(q(t))$, i.e. output is dependent on present state only, then that machine is called Moore machine.

Moore machine - It is a 6 tuple quantity of $Q, \Sigma, \Delta, \delta, \lambda, q_0$. Machine is used to produce output on basis of input.

Q = finite set of states ; Σ = finite set of input ~~alphabet~~

Δ = finite set of output alphabet δ = transition function

λ = output function mapping

$\delta = Q \times \Sigma = Q$ $\qquad \lambda = Q \to \Delta$ $\qquad \lambda(\wedge) = 0$ ↳ empty

eg:

Present state	Next state $a=0$	Next state $a=1$	O/P
$\to q_0$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

If input is of n length then output will be of n+1.

Input string = 0111 O/P = 00010

$q_0 \xrightarrow{0/0} q_3 \xrightarrow{1/0} q_0 \xrightarrow{1/0} q_1 \xrightarrow{1/1} q_2 \xrightarrow{\wedge/0} q_2$

Input = 110110 $q_0 \xrightarrow{1/0} q_1 \xrightarrow{1/1} q_2 \xrightarrow{0/0} q_2 \xrightarrow{1/0} q_3 \xrightarrow{1/0} q_0 \xrightarrow{0/0} q_3 \xrightarrow{\wedge/0} q_3$

→ O/P = 0100000

Melay machine –

Present	Next			
	a=0	λ	a=1	λ
→ q_1	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

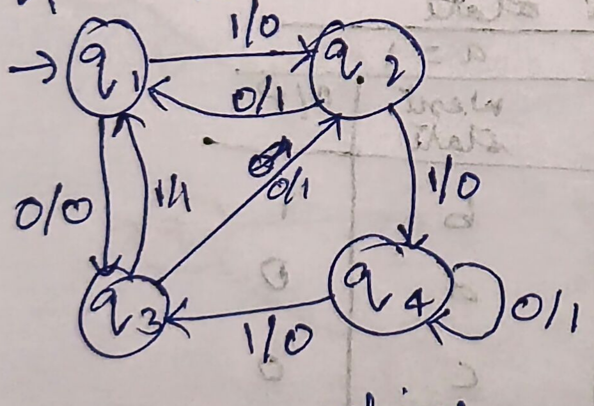
If we process empty string, we get empty.
If n is length of I/P then O/P will be of n.

Input = 0011 O/P = 0100^ $\lambda(\wedge) = \wedge$

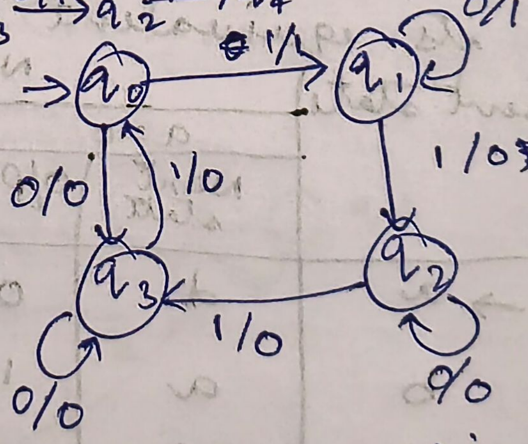
$q_1 \xrightarrow{0/0} q_3 \xrightarrow{0/1} q_2 \xrightarrow{1/0} q_4 \xrightarrow{1/0} q_3 \xrightarrow{\wedge/\wedge} q_3$

Input = 110101 O/P = 001010^

$q_1 \xrightarrow{1/0} q_2 \xrightarrow{1/0} q_4 \xrightarrow{0/1} q_4 \xrightarrow{1/0} q_3 \xrightarrow{0/1} q_2 \xrightarrow{1/0} q_4 \xrightarrow{\wedge/\wedge} q_4$



Melay machine graph

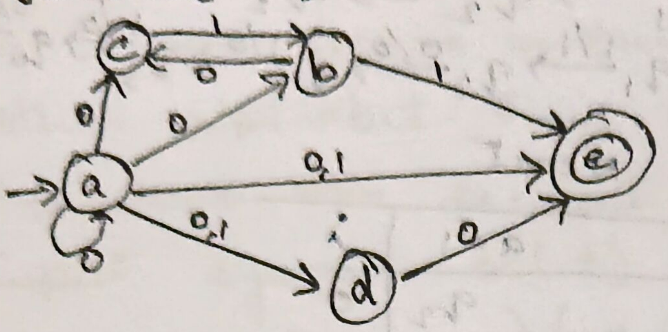


Moore machine graph

How to identify Melay or Moore using the graph?

Tutorial - 3

1. Find its equivalent DFA for the NDFA below -



2. Find the equivalent Melay machine for the following Moore machine -

Present state	Next state a=0	Next state a=1	output
→ a	d	b	1
b	a	d	0
c	c	c	0
d	b	a	1

3. Let us consider the following Melay machine, find its equivalent Moore machine -

Present state	a=0 Next state	a=0 O/P	a=1 Next state	a=1 O/P
→ a	d	0	b	1
b	a	1	d	0
c	c	1	c	0
d	b	0	a	1

4. The one' complement of an input bit string is a string that has 1 whenever there was a 0, and a 0 whenever there was a 1; for example the one's complement of 001 is 110. Construct a Melay machine that computes the one's complement.

5. Construct a nondeterministic finite automaton accepting {ab, ba}, and use it to find a deterministic automaton accepting the same set.

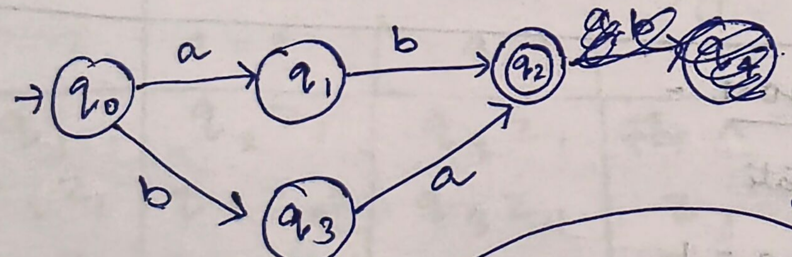
6. Construct a NDFA accepting {ab, ba} and use it to find DFA.

1.

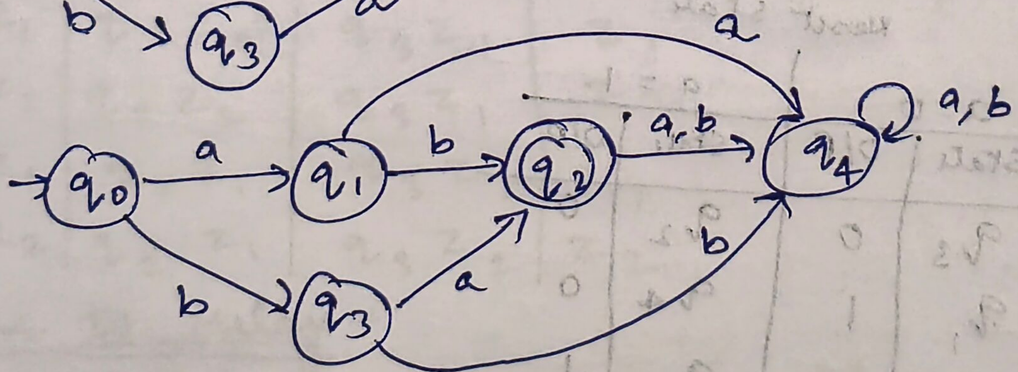
	0	1
→{a}	{b, a, d, e, c}	{d, e}
{b}	{c}	{e}
{c}	ϕ	{b}
{d}	{e}	ϕ
{e}	ϕ	ϕ

	0	1
→{a}	{abcde}	{d,e}
{abcde}	{abcde}	{bde}
{de}	{e}	ϕ
{bde}	{c,e}	{e}
{e}	ϕ	ϕ
{c,e}	ϕ	{b}
{b}	{c}	{e}
{c}	ϕ	{b}
ϕ	ϕ	ϕ

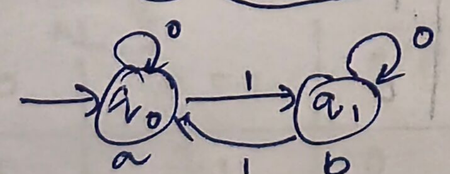
5.



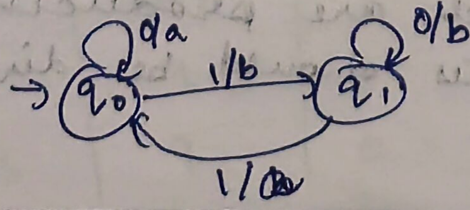
→ NDFA



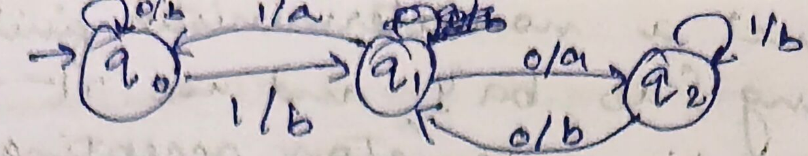
2. Moore



Mealy

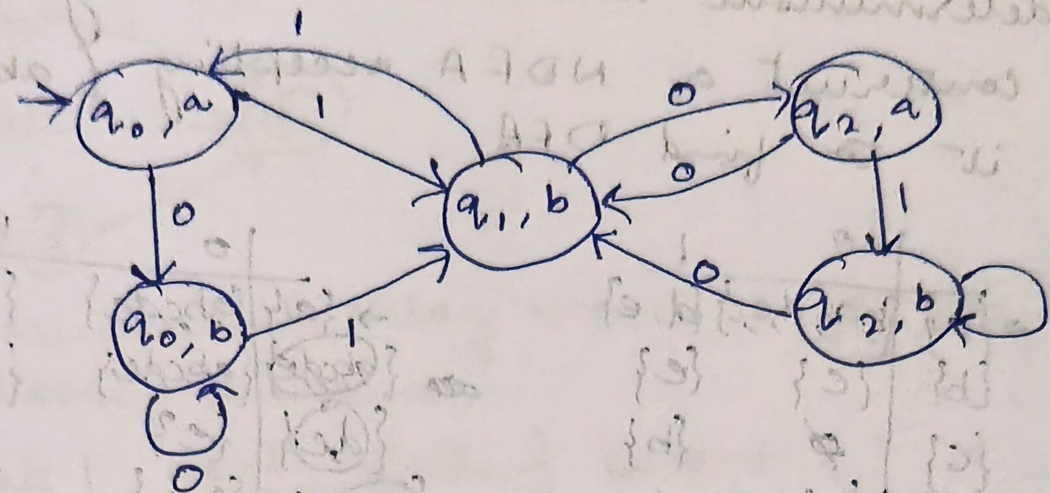


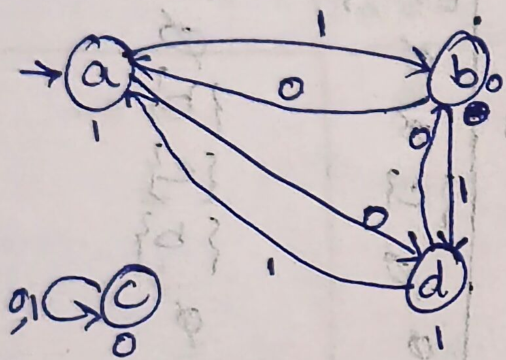
Mealy



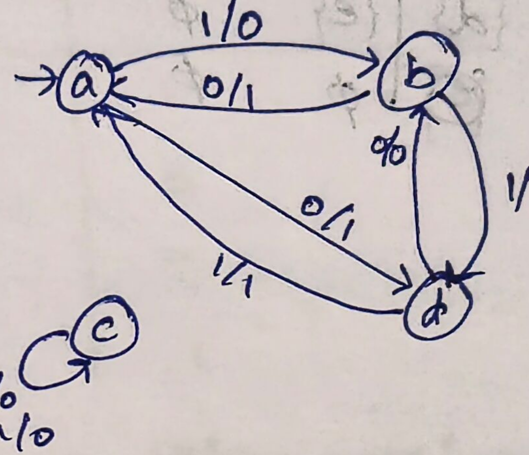
↓

Moore





→



Convert Melay into Moore -

P.S.	Next state a=0 State	O/P	a = 1 State	O/P
q_1	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

Find the states which are producing different outputs. → These states should be divided.

q_1

$q_2 \to q_{2}0$, $q_{2}1$

q_3

$q_4 \to q_{4}0$, $q_{4}1$

Present state	Next state $a=0$	$a=1$	O/P
$\rightarrow q_1$	q_3	$q_{2}0$	~~0~~ 1
$q_{2}0$	q_1	$q_{4}0$	0
$q_{2}1$	q_1	$q_{4}0$	1
q_3	$q_{2}1$	q_1	0
$q_{4}0$	$q_{4}1$	q_3	0
$q_{4}1$	$q_{4}1$	q_3	1

P.S.	Next state $a=0$ S	O/P	$a=1$ S	O/P
$\rightarrow q_1$	q_2	z_1	q_3	z_1
q_2	q_2	z_2	q_3	z_1
q_3	q_2	z_1	q_3	z_2

q_1

$q_2 \to q_2 z_1$, $q_2 z_2$

$q_3 \to q_3 z_1$, $q_3 z_2$

P.S.	Next state $a=0$	$a=1$	O/P
$\rightarrow q_1$	$q_2 z_1$	$q_3 z_1$	$\wedge$
$q_2 z_1$	$q_2 z_2$	$q_3 z_1$	z_1
$q_2 z_2$	$q_2 z_2$	$q_3 z_1$	z_2
$q_3 z_1$	$q_2 z_1$	$q_3 z_2$	z_1
$q_3 z_2$	$q_2 z_1$	$q_3 z_2$	z_2

Moore to Melay -

P.S.	N.S. $a=0$	$a=1$	O/P
$\rightarrow q_0$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

$3 \mid q_3 \mid 0 \quad \mid q_0 \mid 0$

Minimization of FSA -

1. Multiple solⁿ for given problem

→ Equivalence method - Two states q_1 & q_2 are said to be equivalent ($q_1 \equiv q_2$) if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are in final state or both of them are in non-final state $\forall\ x \in \Sigma^*$.

Two state q_1 & q_2 are said to be k-equivalent if both $\delta(q_1, x)$ & $\delta(q_2, x)$ are in final state or both of them are in non-final state ~~of length~~ $\forall$ string x of length $< k$.

Steps for minimization -

1. Find π_0 i.e. set of final state & non-final state.
2. Construct π_1 from π_0.
3. Construct π_{k+1} from π_k.

State	0	1
$\to q_0$	q_1	q_5
q_1	q_6	q_2
q_2	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

$\pi_0 = \{q_2\}\{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}$

$\pi_1 = \cancel{\{q_2\}\{q_0, q_1, q_3, q_5, q_7\}\{q_4, q_6\}}$

1-equivalent

q_0 & q_1 $\quad (q_0 q_1) \begin{cases} 0 & (q_1, q_6) \\ 1 & (q_5 q_2) \end{cases}$

Check q_0 is 1-equivalent with q_1.

~~& states are said to~~

ek final state hai, ek non-final state hai.

$\therefore$ q_0 & q_1 are not 1-equivalent.

$(q_0 q_3) \begin{cases} 0 & (q_1, q_2) \\ 1 & (q_5 q_6) \end{cases} \rightarrow$ Not equivalent

$(q_0 q_4) \begin{cases} 0 & (q_1 q_7) \\ 1 & (q_5 q_5) \end{cases}$ ✓ $\qquad (q_0 q_5) \begin{cases} 0 & (q_1 q_2) \\ 1 & \end{cases}$ ×

$(q_0 q_6) \begin{cases} 0 & (q_1 q_6) \\ 1 & (q_5 q_4) \end{cases}$ ✓ $\qquad (q_0 q_7) \begin{cases} 0 & (q_1 q_6) \\ 1 & (q_5 q_2) \end{cases}$ ×

$\cancel{\pi_1 = \{q_2\}\{q_0 q_1 q_3 q_5 q_7\}\{q_4 q_6\}}$

$\pi_1 = \{q_2\}\{q_0 q_4 q_6\}\{q_1 q_3 q_5 q_7\}$

$= \{q_2\}\{q_0 q_4 q_6\}\{q_1\}\{q_3 q_5 q_7\}$

$(q_1 q_3) \begin{cases} 0 & (q_6 q_2) \\ 1 & \end{cases}$ × $\qquad (q_1 q_5) \begin{cases} 0 & (q_6 q_2) \\ 1 & \end{cases}$ ×

$(q_1 q_7) \begin{cases} 0 & (q_1 q_6) \\ 1 & (q_5 q_2) \end{cases}$ × $\qquad (q_3 q_5) \begin{cases} 0 & (q_2 q_2) \\ 1 & (q_6 q_6) \end{cases}$ ✓

$(q_3 q_7) \begin{cases} 0 & (q_2 q_6) \\ 1 & \end{cases}$ ×

$\pi_1 = \{q_2\}\{q_0 q_4 q_6\}\{q_1, q_7\}\{q_3 q_5\} \cancel{\{q_7\}}$

Now, π_2

$(q_0 q_4) \begin{cases} \xrightarrow{0} (q_1, q_7) \\ \xrightarrow{1} (q_5 q_5) \end{cases}$ ✓ $(q_0 q_6) \begin{cases} \xrightarrow{0} (q_1 q_6) \\ \xrightarrow{1} \end{cases}$ ×

~~$\pi_2 = \{q_2\}\{q_0\}\{q_4 q_6\}\{q_1\}\{q_3 q_5\}\{q_7\}$~~

$\pi_2 = \{q_2\}\{q_0 q_4\}\{q_6\}\{q_1 q_7\}\{q_3 q_5\}$

$(q_1 q_7) \begin{cases} \xrightarrow{0} (q_6 q_6) \\ \xrightarrow{1} (q_2 q_2) \end{cases}$ ✓ $(q_3 q_5) \begin{cases} \xrightarrow{0} (q_2 q_2) \\ \xrightarrow{1} (q_6 q_6) \end{cases}$ ✓

Now, π_3

$(q_0 q_4) \begin{cases} \xrightarrow{0} (q_1 q_7) \\ \xrightarrow{1} (q_5 q_5) \end{cases}$ ✓ $(q_1 q_7) \begin{cases} \xrightarrow{0} (q_6 q_6) \\ \xrightarrow{1} (q_2 q_2) \end{cases}$ ✓

$(q_3 q_5) \begin{cases} \xrightarrow{0} (q_2 q_2) \\ \xrightarrow{1} (q_6 q_6) \end{cases}$ ✓

~~Not~~ No further partition is possible.

$\pi_3 = \pi_2$.

No. of states = 5.

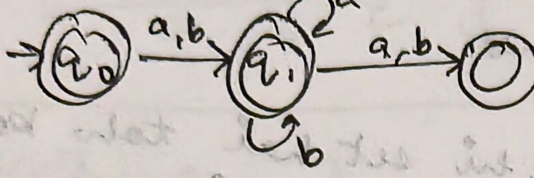
States	0	1
$\rightarrow q_0 q_4$	$q_1 q_7$	q_5
$q_1 q_7$	q_6	q_2
q_2	q_0	q_2
$q_3 q_5$	q_2	q_6
q_6	q_6	q_4

~~For~~ Tutorial - 4 -

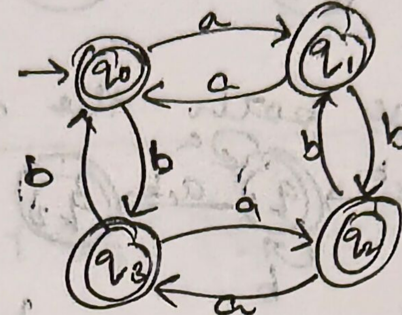
1. Construct an FA accepting all strings over $\{0, 1\}$ ending with either 010 or 0010.
2. If $L_1 = \{s_1 \mid \in (a, b)^*\}$ and $L_2 = \{s_2 \in (a, b)^+\}$ draw FA for $L = L_1 L_2$.

3. Draw FA for all strings over $\{0,1\}$ consisting 010 or 111 as substring.

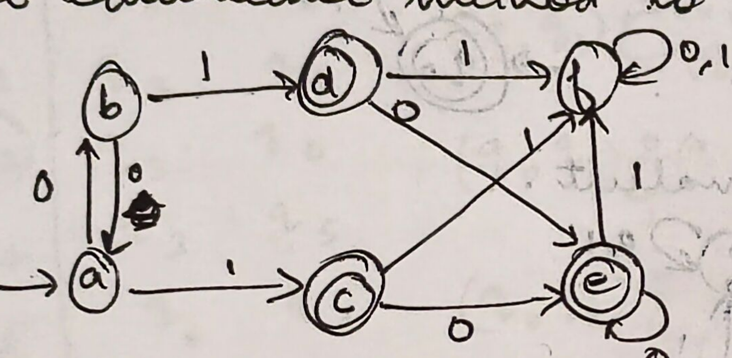
4. Prove or disprove whether following FA M_1 & M_2 are equivalent.

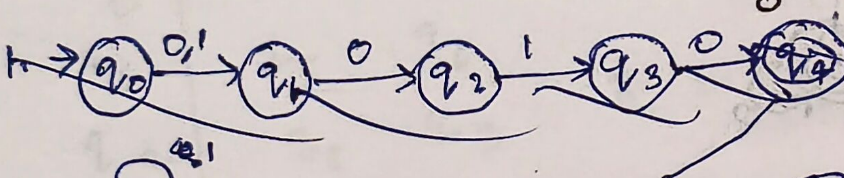


M_1

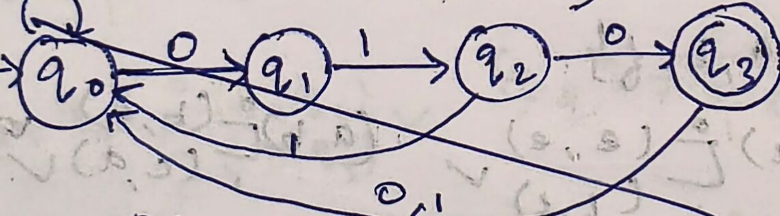


5. Use equivalence method to minimize the given FA

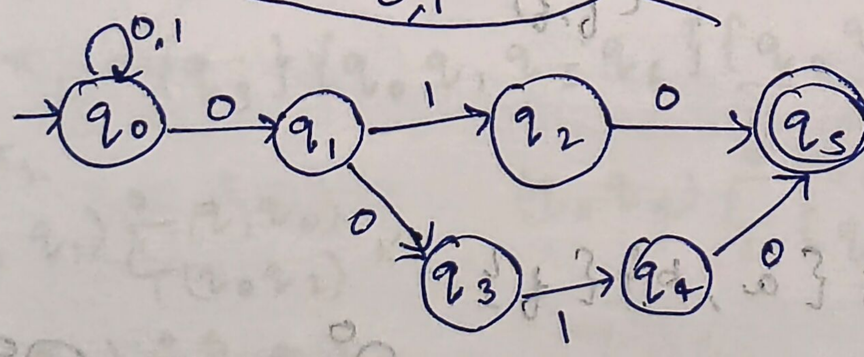




1.



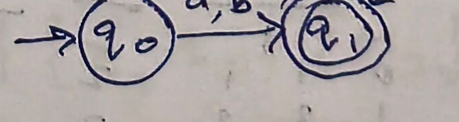
1.

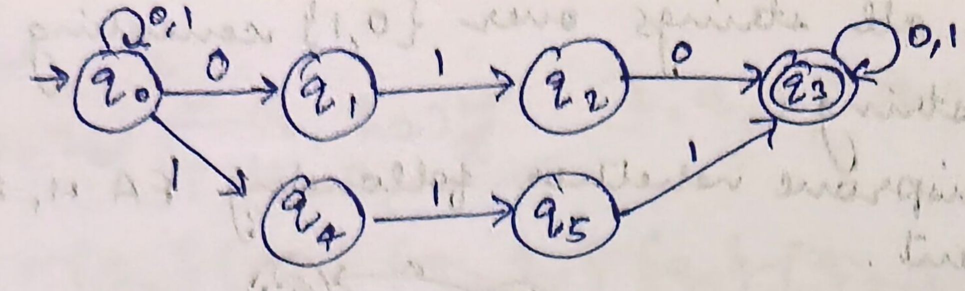


2. $L_1 = \{\phi, a, b, aa \ldots\}$

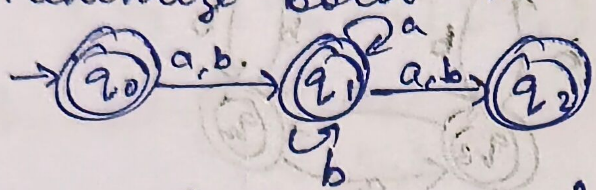
$L_2 = \{a, b, aa, \ldots\}$

$L_1 L_2 = L_2 = (a+b)^+$

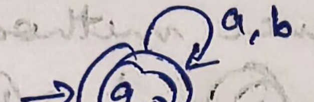


3. 

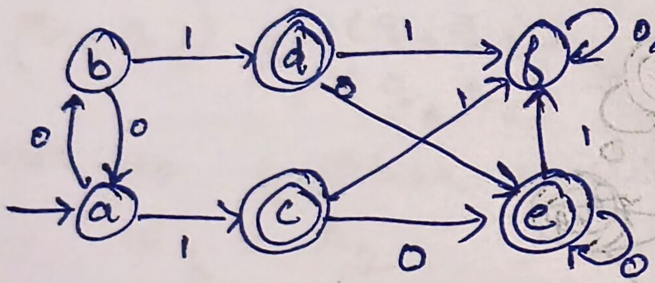
4. Minimize both the machines.



$\pi_0 = \{q_0, q_1, q_2\}$ → Ek hi set hai toh koi division nahi hoga.

Minimized machine : 

Machines are equivalent.

5. 

$\pi_0 = \{c, d, e\}\{a, b, f\}$

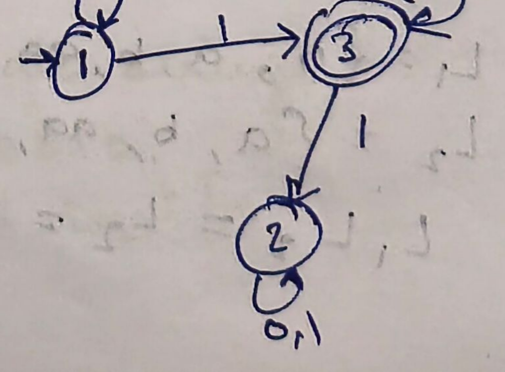
$(c,d) \begin{cases} 0: (e, e) \checkmark \\ 1: (f, f) \end{cases}$ $(c,e) \begin{cases} 0: (e, e) \\ 1: (f, f) \checkmark \end{cases}$ $(a,b) \begin{cases} 0: (b, a) \\ 1: (c, d) \checkmark \end{cases}$

$(a,f) \begin{cases} 0: (b, f) \\ 1: (c, f) \end{cases} \times$

$\pi_1 = \{c, d, e\}\{a, b\}\{f\}$

$\pi_2 = \underbrace{\{a, b\}}_{1}\ \underbrace{\{f\}}_{2}\ \underbrace{\{c, d, e\}}_{3}$

	a	b	f	c	d	e
0	1	1	2	3	3	3
1	3	3	2	2	2	2



b				
c	x	x		
d	x	x		
e	x	x		
	a	b	c	d

	b, a
0	a b
1	d c

---

state	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
q_3	q_3	q_0
q_4	q_3	q_5
q_5	q_6	q_4
q_6	q_5	q_6
q_7	q_6	q_3

$\pi_0 = \{q_3\}\{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}$

$(q_0 q_1) \begin{cases} 0: (q_1, q_0) \\ 1: (q_0 q_2) \end{cases}$ ✓ $\quad (q_0 q_2) \begin{cases} 0: (q_1, q_3) \\ 1 \end{cases}$ ✗

$(q_0 q_4) \begin{cases} 0: (q_1, q_3) \\ 1 \end{cases}$ ✗ $\quad (q_0 q_5) \begin{cases} 0: (q_1, q_6) \\ 5: (q_0 q_4) \end{cases}$ ✓

$(q_0 q_6) \begin{cases} 0: (q_1, q_5) \\ 1: (q_0 q_6) \end{cases}$ ✓ $\quad (q_0 q_7) \begin{cases} 0: (q_1, q_6) \\ 1: (q_0 q_3) \end{cases}$ ✗

$\pi_1 = \{q_3\}\{q_0 q_1 q_5 q_6\}\{q_2 q_4 q_7\}$

$(q_2 q_4) \begin{cases} 0: (q_3 q_3) \\ 1: (q_1 q_5) \end{cases}$ ✓ $\quad (q_2 q_7) \begin{cases} 0: (q_3 q_6) \\ 1: (q_1 q_3) \end{cases}$ ✗

$\pi_1 = \{q_3\}\{q_0 q_1 q_5 q_6\}\{q_2 q_4\}\{q_7\}$

Now, π_2

$(q_0 q_1) \begin{cases} 0: (q_1 q_0) \\ 1: (q_0 q_2) \end{cases}$ ✗ $\quad (q_0 q_5) \begin{cases} 0: (q_1, q_6) \\ 1: (q_0 q_4) \end{cases}$ ✗

$(q_0 q_6) \begin{cases} 0: (q_1 q_5) \\ 1: (q_0 q_6) \end{cases}$ ✓

$\pi_2 = \{q_3\}\{q_0 q_6\}\{q_1 q_5\}\{q_2 q_4\}\{q_7\}$

Now, π_3

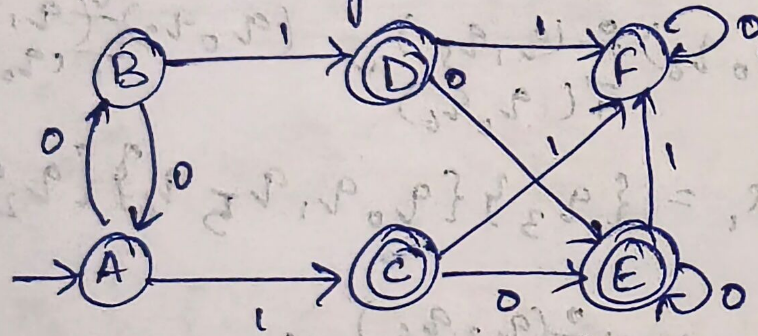
$(q_1 q_5) \begin{cases} 0: (q_0 q_6) \\ 1: (q_2 q_4) \end{cases}$ ✓ $\quad (q_2 q_4) \begin{cases} 0: (q_3 q_3) \\ 1: (q_1 q_5) \end{cases}$ ✓ $\quad (q_0 q_6) \begin{cases} 0: (q_1 q_5) \\ 1: (q_0 q_6) \end{cases}$ ✓

$\pi_3 = \pi_2$

Minimization with tabulation -

Steps -

1. Draw a table for all pair of states (P, Q).
2. Mark all pair cell where P is element of F and Q $\notin$ F.
3. If there are any unmarked pair cell in the table such that $\delta(P, x)$ and $\delta(Q, x)$ is marked then (P, Q) should also be marked.
4. Repeat this ~~process~~ step until no more marking is process.
5. Combine all the unmarked pair cells and make them a single state in minimized DFA.



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

$(A, B) \begin{cases} \xrightarrow{0} (B, A) \\ \xrightarrow{1} (C, D) \end{cases}$

$(C, D) \begin{cases} \xrightarrow{0} (E, E) \\ \xrightarrow{1} (F, F) \end{cases}$

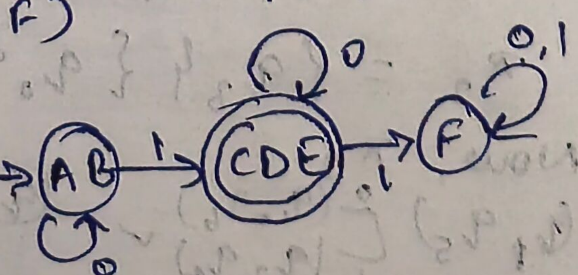
$(C, E) \begin{cases} \xrightarrow{0} (E, E) \\ \xrightarrow{1} (F, F) \end{cases}$

$(E, D) \begin{cases} \xrightarrow{0} (E, E) \\ \xrightarrow{1} (F, F) \end{cases}$

$(A, F) \begin{cases} \xrightarrow{0} (B, F) \\ \xrightarrow{1} (C, F) \end{cases}$

$(B, F) \begin{cases} \xrightarrow{0} (A, F) \\ \xrightarrow{1} \end{cases}$

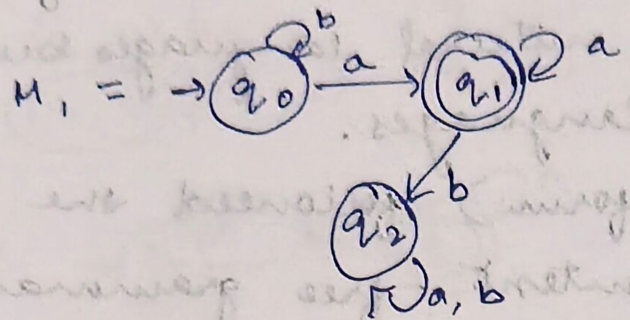
{A, B} {C, D, E} {F}

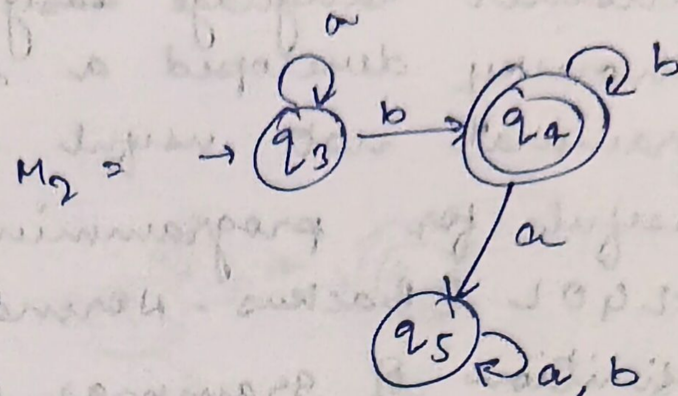


Operation on regular languages -

$L_1 \; L_2$

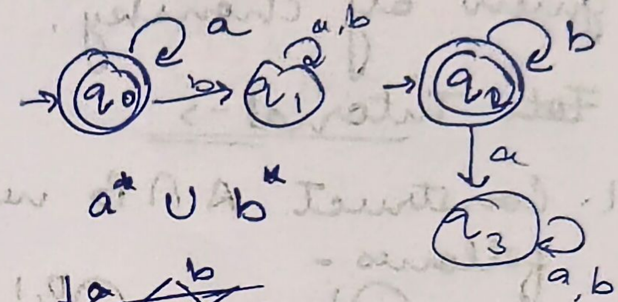
$L_1 \cup L_2 \qquad L_1 \cap L_2 \qquad \overline{L_1}$



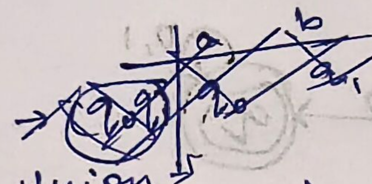


$M_1 \cup M_2 = a^+ \cup b^+$

States	a	b
$\to(q_0 q_3)$	$(q_1 q_3)$	$(q_0 q_4)$
$(q_0 q_4)$	(q_1, q_5)	$(q_0 q_4)$
$(q_0 q_5)$	(q_1, q_5)	$(q_0 q_5)$
(q_1, q_2)	(q_1, q_3)	$(q_2 q_4)$
(q_1, q_4)		
(q_1, q_5)		
$(q_2 q_3)$		
$(q_2 q_4)$		
$(q_2 q_5)$		



$a^* \cup b^*$



union →

	a	b
$\to(q_0 q_2)$	$(q_0 q_3)$	(q_1, q_2)
$(q_0 q_3)$	$(q_0 q_3)$	(q_1, q_3)
(q_1, q_2)	(q_1, q_3)	(q_1, q_2)
(q_1, q_3)	(q_1, q_3)	(q_1, q_3)

Complement -

	a	b
$(q_0 q_3)$		
(q_1, q_2)		

Intersection →

$a^* \cap b^*$

	a	b
$\to(q_0 q_2)$		
$(q_0 q_3)$		
(q_1, q_2)		
(q_1, q_3)		

Formal languages -

Mathematicians were trying in early 1950s to define definition of valid sentences and they also wanted to give a formal structure of any language. The objective was to develop a mathematical model

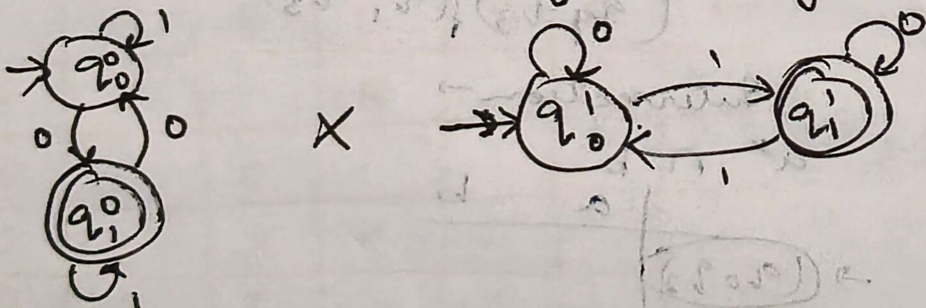
that helps the translation of one language to another language ~~easy~~ easily. In 1956 Noam Chomsky developed a mathematical model of grammar not useful for natural languages but useful for programming languages.
ALGOL (Backus - Normal form) followed the definition of grammar (context - free grammar) given by Chomsky.

~~Tut~~ Tutorial-5.

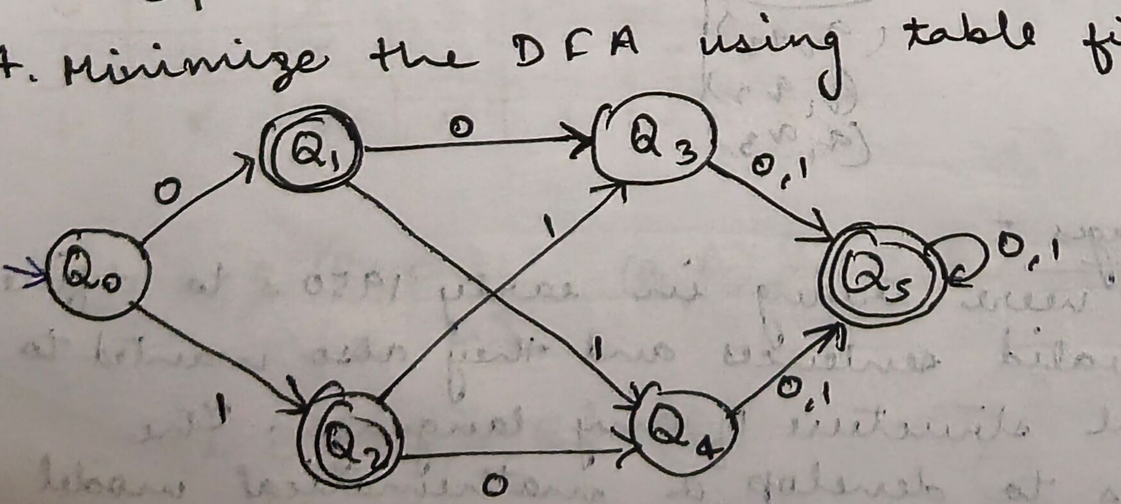
1. Construct $A \cap B$ where A and B is given as follows -

2. $L1$ = { starts with a and ends with b } and $L2$ = { starts with b and ends with a }. Then find ~~L~~ $L = L1 \cup L2$ or $L = L1 + L2$.

3. Find the product of 2 FA given below:



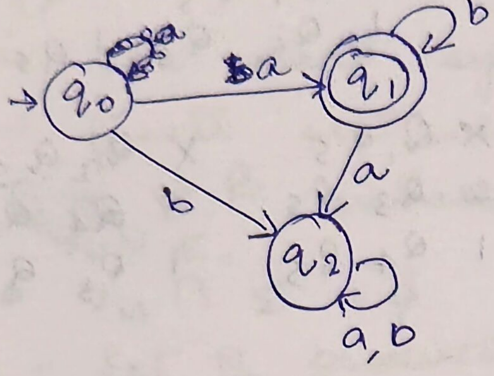
4. Minimize the DFA using table filling method.

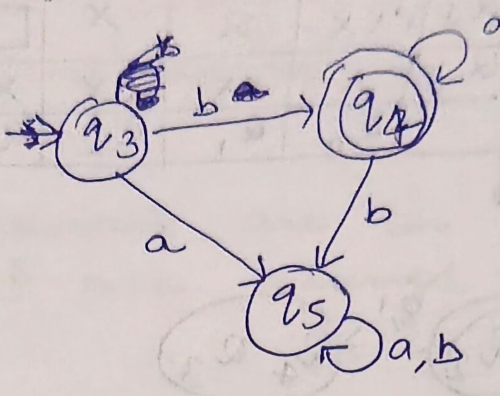


1.

States	0	1
→(p, r)	(q, r)	(p, s)
(p, s)	(q, s)	(p, s)
(q, r)	(q, r)	(q, s)
(q, s)	(q, s)	(q, s)

2.





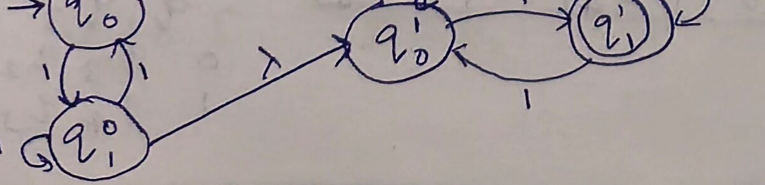
States	a	b
→(q_0, q_3)	(q_1, q_5)	(q_2, q_4)
(q_0, q_4)	(q_1, q_4)	(q_2, q_5)
(q_0, q_5)	(q_1, q_5)	(q_2, q_5)
(q_1, q_3)	(q_2, q_5)	(q_1, q_4)
(q_1, q_4)	(q_2, q_4)	(q_1, q_5)
(q_1, q_5)	(q_2, q_5)	(q_1, q_5)
(q_2, q_3)	(q_2, q_5)	(q_2, q_4)
(q_2, q_4)	(q_2, q_4)	(q_2, q_5)
(q_2, q_5)	(q_2, q_5)	(q_2, q_5)

$L_1 \cup L_2$: fs : Combinations containing fs(M(L_1)) or fs(M(L_2)) or both

$L_1 \cap L_2$: fs : Combinations containing fs(M(L_1)) and fs(M(L_2)) both.

$L_1 - L_2$ fs : Combinations containing fs(M(L_1)) and non fs(M(L_2)) both.

3.



4.

Q_1	X				
Q_2	X	$\boxed{}$			
Q_3	XXX	X	X		
Q_4	XXX	X	X	$\boxed{}$	
Q_5	X	XX	XX	X	X
	Q_0	Q_1	Q_2	Q_3	Q_4

	Q_1	Q_2
0	Q_3	Q_4
1	Q_4	Q_3

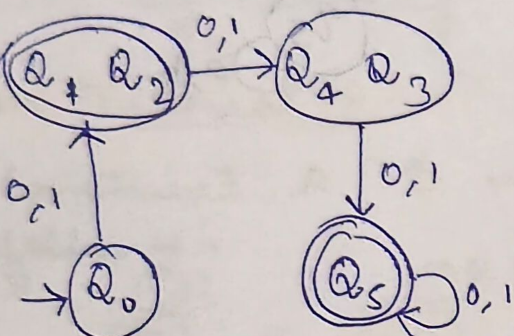
X	Q_0	Q_4
0	Q_1	Q_5
1	Q_2	Q_5

X	Q_0	Q_3
0	Q_1	Q_5
1	Q_2	Q_5

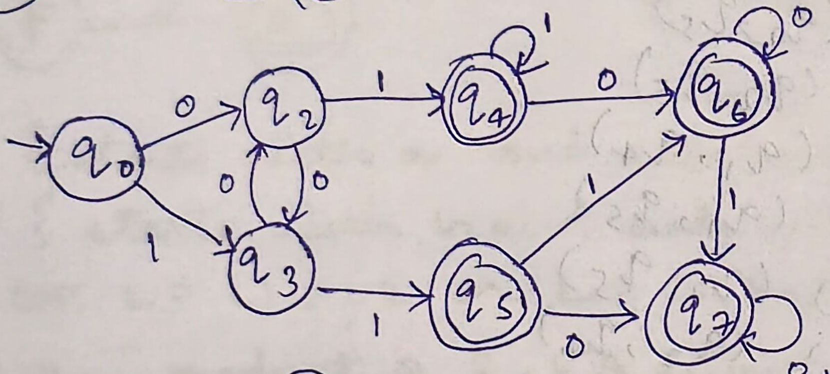
	Q_3	Q_4
0	Q_5	Q_5
1	Q_5	Q_5

X	Q_1	Q_5
0	Q_3	Q_5
1	Q_4	Q_5

X	Q_2	Q_5
0	Q_4	Q_5
1	Q_3	Q_5

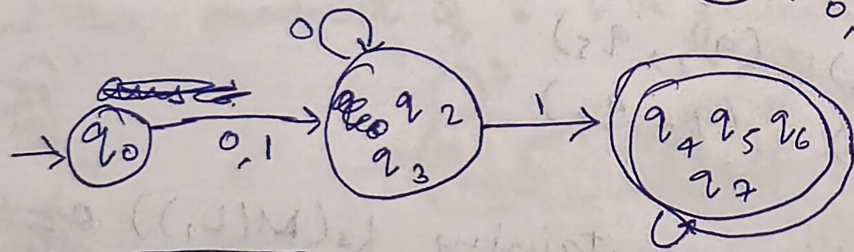


Q:



ans: ~~7~~ ~~8~~ cells empty hoye

ans:



q_2	XX					
q_3	XX					
q_4	X	X	X			
q_5	X	X	X			
q_6	X	X	X			
q_7	X	X	X			
	q_0	q_2	q_3	q_4	q_5	q_6

X	q_0	q_2
0	q_2	q_3
1	q_3	q_4

X	q_0	q_3
0	q_2	q_2
1	q_3	q_5

	q_2	q_3
0	q_3	q_2
1	q_4	q_5

Grammar - In structure, grammar is a 4 ~~by~~ tuple quantity of V_N, Σ, P, S where V_N is finite non-empty set of elements known as variable. Σ is finite non-empty set of elements known as terminal symbol.

S is the start symbol.

P → finite set whose elements are in form of $\alpha \to \beta$ where α & β are strings on ~~the~~ $V_N \cup \Sigma$.

Also $V_N \cap \Sigma = \phi$

Note - Set of production rule is the kernel of grammar and languages.

1. $S \to AB$ Reverse replacement is not permitted.
2. No inversion operation is allowed.

Derivation of language by grammar -

1. ~~In~~ If $\alpha \to \beta$ is a production then we can write it as $\alpha \underset{G}{\Rightarrow} \beta$. $\alpha \underset{G}{\to} \beta$.
2. If α & β are string on $V_N \cup \Sigma$ then we can say α derives β if $\alpha \xrightarrow[G]{*} \beta$
3. The language generated by G is denoted as $L(G)$ and it is defined as $\{ w \in \Sigma^* \mid S \xrightarrow[G]{*} w \}$.
4. If $S \xrightarrow[G]{*} \alpha$ then α is called sentential form.
5. Two grammar G_1 and G_2 are said to be equivalent if $L(G_1) = L(G_2)$

Q: $G = (\{S\}, \{0,1\}, \{S \to 0S1, S \to \wedge\}, S)$

$V_N = S$ $\quad \Sigma = 0,1 \quad P = \quad S \to 0S1$ — (1)

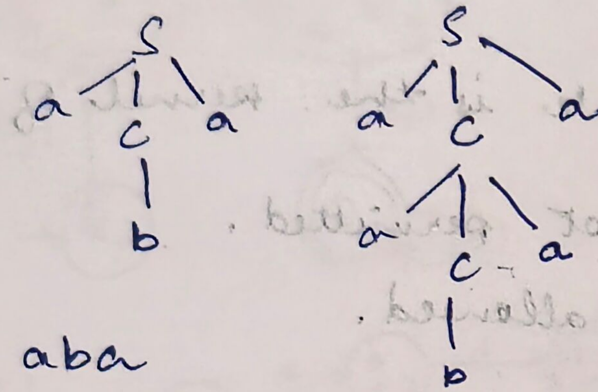
$S \to \wedge$ — (2)

$S \to 0S1$
$\to 01 = 01$

$S \to 0S1$
$\to 00S11$
$\to 0011 = 0011$
$\to 000S111 = 000111$

Q: $G = (\{S, C\}, \{a, b\}, P, S)$

$S \to aCa \quad C \to aCa \quad C \to b$



aba

aabaa

Q: $S \to aS \mid bS \mid a \mid b$ Find $L(G)$

Q: $L(G) = a^i b^n c^n \mid n \geq 1, i \geq 0$

Smallest string: bc

$S \to aS \quad S \to A / bSc \quad S \to bc$