Elements of Finite state system: 4 main elements:

↳ States which define the behavior and may produce actions
↳ State transitions, which are movement from one state to another.
↳ Rule or conditions which must be met to allow a state transition
↳ Input Events, which are either externally or internally generated. These may possibly trigger the rules & lead to state transitions.

Not:

(1.) State:
↳ A state is a complete set of properties, transmitted by an object to an observer via one or more channels.
– Any change in the nature or quantity of such properties in a state is detected by an observer & thus a transmission of information occurs.

(i) Start State: An initial state or condition of a finite state Machine.

(ii) Accepting State: If a FSM finishes an i/p string & is in an accepting state, the string is accepted or considered to be valid.

(iii) Next State: ↳ The state immediately following current state;

iv) Universal State: A state in an alternating Turing M/c, from which M/c only accepts all possible moves leading to acceptance.

v) Existential State: ↳ A State in a nondeterministic T.M. from which the M/c accepts any move that leads to acceptance, is the existential state.

vi) Dead/Trap State: → A nonfinite state of a FSM, whose transitions on every input symbol terminates on itself.

Transition:
↳ "Transition is the act of passing from one place/state to the next."

Scanned by CamScanner

↳ A change from one place or state or subject to another

Transitions are represented in the following ways

↳ state diagram or Transition diagram.

↳ State Transition Table.

↳ Transition functions.

↳ state diagram :-

A diag. consisting of circles to represent states & directed line segments to represent transitions b/w states, is called a state diagram.

Start of the process

final / Accepting / Absorption state (end of process)

Transition.

↳ For a FSM, a state diagram is a directed graph where,

a) Each edge is a transition between two states.

↳ For DFA, NFA and Moore M/C, i/p is labeled on each edge.

↳ For a mealy M/C, i/p & o/ps r labelled on each edge.

b) Each vertex is a state

↳ for a Moore M/C, o/p is signified for each state

Scanned by CamScanner

another.

directed i/p

↳ State Transition Table :-

A state Transition table can be described, in general, as :- = any F.A.

→ Tabular representation of transitions that take two arguments & return a value.

→ Rows correspond to states & columns correspond to inputs.

→ Entries correspond to next states.

→ the start state is marked with an arrow (→).

→ the accepting states are marked with a star. (*).

State Transition Table format.

States \ I/Ps	Present I/Ps		--- ---	
	a1	a2	--- ---	an
S_0				
S_1				

S_n				

↳ A Finite Automaton has-

a.) a finite set of states, one of which is designed as the initial state or start state and some of which are designated as final states.

b.) An alphabet Σ of possible input symbols.

c) a finite set of transitions that informs each state and each symbol of the i/p alphabet, about next state.

Scanned by CamScanner

Finite Automata :-

According formal definition,

"A finite automation is a list of five objects: set of states, input alphabet, rules for moving, start & accept states."

⇒ Deterministic Finite Automata (DFA) :-

"A DFA is a finite state machine where for each pair of states and input symbol; there is a unique next state

Elements of DFA :-

The DFA exhibits 5 characteristics:

a.) A finite set of states Q.

b.) an Alphabet Σ of possible i/p symbols.

c.) a transition function δ such that $\delta(x,1)=y$ where $x,y \in Q$ and $1 \in \Sigma$.

d.) the initial state $q_0 \in Q$.

e.) the set of final states (F), where $F \subseteq Q$.

"The term deterministic refers to the fact that on each input, there is one and only one state to which the automaton can transit from its current state".

Ordered Quintuple Specification of DFA :-

Formally, a DFA, M is a five-tuple:

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

where,

a) Q is a finite set of states of finite automata.

b) Σ is a finite set of input symbols called "alphabet",

c.) δ: $Q \times$

d.) $q_0 \in Q$

e.) $F \subseteq$

If from

State

writter

wher

Thus,

Des

T

→

Scanned by CamScanner

Regular Language :- for R.L.; (Finite). (DFA/NFA).

c.) $\delta: Q \times \Sigma \to Q$, is the transition function,

d.) $q_0 \in Q$ is the start state.

e) $F \subseteq Q$ is the set of accept states.

If from a state P, the exists a transition going to state 'q' on an input symbol 'a', then this is written as:

$$\boxed{\delta(P, a) = q}$$

Where, δ – is a func. whose domain is a set of ordered pair(P,a)

P – a state

a – input symbol.

Thus, 'δ' → defines a mapping whose domain will be a set of ordered pairs of the form (P, a) & whose range will be a set of states. i.e

$$\delta: Q \times \Sigma \longrightarrow Q.$$

Description of a DFA:

The transitions of DFA can be represented using transition diagram or table.

→ Extended transition function for DFA:

for DFA, $M = (Q, \Sigma, \delta, q_0, F)$ the function 'δ' is extended as –

$$\delta: Q \times \Sigma^* \longrightarrow Q$$

and is defined recursively as follows :-

a) For any state q of Q

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

This means that DFA stays in the same state, q when it reads an empty string at q.

Scanned by CamScanner

b) for any state q of Q, any string $x \in \Sigma^*$ with a as the last symbol of 'x' and $a \in \Sigma$.

$$\delta(q, xa) = \delta(\delta(q, x), a).$$

Language accepted by DFA's

The lang. accepted by a DFA, $M = (Q, \Sigma, \delta, q_0, F)$ is the set of all strings on Σ accepted by M. i.e;

$$L(M) = \{W \in \Sigma^* \mid \delta(q_0, W) \in F\}$$

A lang. is said to be rejected by DFA if

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

$$L(M) = \{W \in \Sigma^* \mid \delta(q_0, W) \notin F\}$$

* A machine may accept several strings but it recognises only one language.

Design of DFA's

The basic design strategy for DFA is as follows :-

a) Understand the language properties for which the DFA has to be designed.

b) Determine the state set required.

c) Identify the initial, accepting and Dead state of DFA.

d) For each state, decide on the transition to be made for each character of the input string.

e) obtain the transition table and diagram for DFA.

f) Test the DFA obtained on short strings.

Scanned by CamScanner

Regular Language :- For R.L., it is (finite). (DFA/NFA).

Ex.- To design a DFA that accepts set of all strings that contain 0's or 1's and end in "00".

Sol^{n}:- We are required to design a DFA for the regular expr^{n}

$r = (0+1)^{*}00$ ---

In other words, the DFA should be designed to accept the language of r;

i.e. $L(M) = \{00, 100, 1100, 0000, \text{------}\}$

where, M is a DFA.

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

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

q_0 = initial state

q_2 = final state and

$\delta : Q \times \Sigma \longrightarrow Q$ is given by

Transition diagram :-

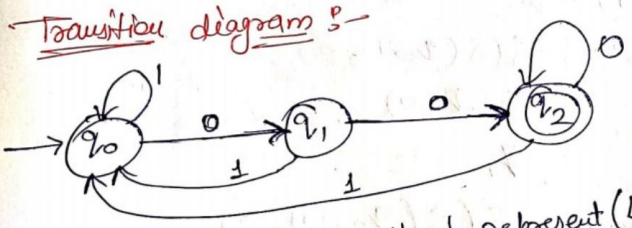


Fig:- State Transition to represent $(L(M) = \{w \in \Sigma^{*} \mid w \text{ ends with } 00\}$

Transition Table :-

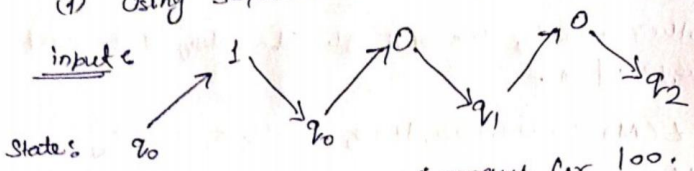
Q	Σ	Present I/Ps	
		0	1
→q0		q1	q0
q1		q2	q0
*q2		q2	q0

Scanned by CamScanner

DFA action for the input string

1.) To show that the string 100 is accepted by DFA:

(i) Using Sequence state diagram:-



Sequence State diagram for 100.

Since, we encounter the end of the input and we are in the final state, we say that string is accepted by machine M. Thus, 100 is in L(M).

(ii) Using extended transition function:-

Consider,

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

$$\delta(q_0, 10) = \delta(\delta(q_0, 1), 0)$$
$$= \delta(q_0, 0)$$
$$= q_1$$

$$\delta(q_0, 100) = \delta(\delta(q_0, 10), 0)$$
$$= \delta(q_1, 0)$$
$$= q_2$$

Since, after scanning the entire string, we reach at the final state q_2, the given string 100 is thus accepted by the DFA M. Thus, 100 is in L(M).

(iii) Using vdash function ($\vdash$):

Scanned by CamScanner

DFA:

$(q_0 100) = \vdash M(q_0, 00)$

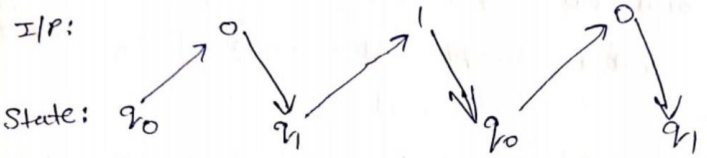
$\vdash M(q_1, 0)$

$\vdash M(q_2, e)$ → means 'reached end of i/p'.

Since, we reached the end of input (e) and we r in final state q_2, hence 100 is accepted by M. Thus, 100 is in L(M).

2) To show that the string 010 is rejected by DFA:—

(i) Using state (sequence. state) diagram:



Since, we encounter the end of i/p & q_1 is not the final state, we say that the string 010 is rejected by the m/c.

(ii) Using extended transition functions:

$\delta(q_0, 0) = q_1$

$\delta(q_0, 01) = \delta(\delta(q_0, 0), 1)$

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

$\delta(q_0, 010) = \delta(\delta(q_0, 01), 0)$

$= \delta(q_0, 0)$

$= q_1$

Since, after scanning the entire string, we did not reach the final state q_2, hence the string 010 is rejected by DFA 'M'.

=

Scanned by CamScanner

Q.1.) To design a DFA that accepts a set of even numbers of a's.

2.) Design a DFA that accepts odd number of 1's.

3.) Design a DFA that
→ starts with 0 and has odd number of 0's.
→ starts with 1 and has even number of 1's.

4.) Design a DFA that accepts even number of a's & b's.

5.) Design a DFA to accept odd number of a's and b's.

6.) Design a DFA to accept odd number of a's and even number of b's.

7.) Design a DFA that contains set of all strings ending with 3 consecutive zeros, over the $\Sigma = \{0,1\}$.

8.) Design a DFA, to accept the language $L = \{awa | w \in (a+$ over $\Sigma = \{a,b\}$,

9.) Design a DFA to accept the set of all strings of a starting with string 'ab'.

10.) Design a DFA, over $\Sigma = \{0,1\}$, that contains set of strings of 0's & 1's. except those containing substring '110'.

11.) Design a DFA over $\Sigma = \{0,1\}$, that contains set of 0's except those containing substring 001.

* Design a DFA accepted must
1.) w = 'a'
i.)
iii)
II)
III

Scanned by CamScanner

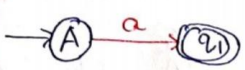
Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must start with w.

1.) $w = 'a'$

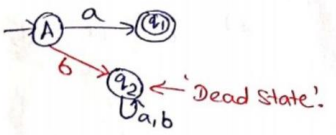
Sol: Possible Language;

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

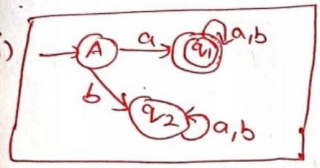
i.)



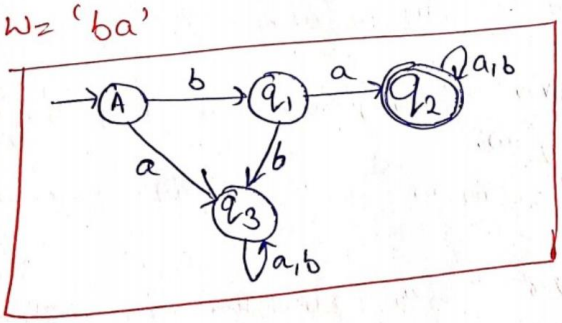
ii.)



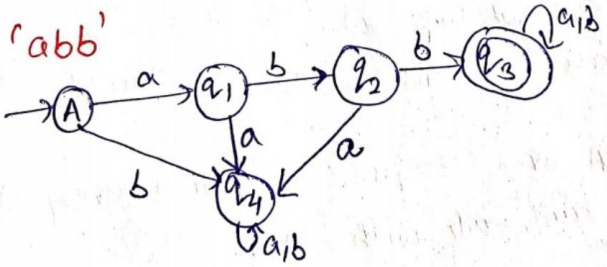
iii)



II) $w = 'ba'$



III $w = 'abb'$



* NOTE:- (n-2) states in MDFA.

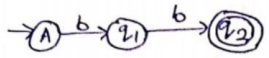
Scanned by CamScanner

Design a MDFA over $\Sigma=\{a,b\}$ such that every string accepted must ends with a substring ω.

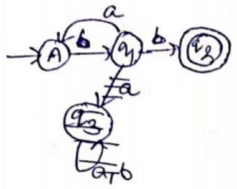
i) ω = 'bb'

$L=\{bb, abb, aabb, bbb, ----\}$

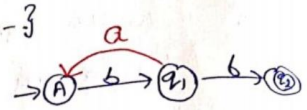
1.)



2.)

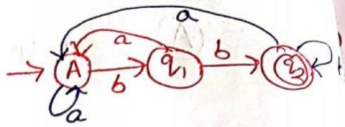


3.)



No Need of dead state

4.)



PRACTICE PROBLEMS

TRY IT YOURSELF!

(1.) Design a DFA over $\Sigma=\{a,b\}$ such that every string accepted must start with a substring ω.

i) ω = ba ii) ω = abb

(2.) Design a DFA over $\Sigma=\{a,b\}$ such that every string accepted ending with a substring ω.

i) ω = bb ii) ω = ab iii) ω = bab

(3.) Design a DFA over $\Sigma=\{a,b\}$ such that every string accepted contains a substring ω.

i) ω = aa ii) ω = ba iii) ω = bb

(4.) Design a DFA over $\Sigma=\{a,b\}$ such that every string accepted must starts and ends with 'a'.

Scanned by CamScanner

5.) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must start and ends with same symbol.

6) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must start with 'a' ending with 'b' and vice-versa

7.) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must start with 'aa' or 'bb'.

8.) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must ends with 'aa' or 'bb'.

9.) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must contains a substring 'aa' or 'bb'.

10.) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must

a) $|w| = 2$ b) $|w| \geq 2$ c.) $|w| \leq 2$

d.) $|w|_a = 2$ e.) $|w|_a \geq 2$ f.) $|w|_b \leq 2$.

11.) Design a DFA over $\Sigma = \{a, b\}$ such that every string accepted must

a.) $|w| = 2 \pmod 3$ b.) $|w| = 3 \pmod 4$

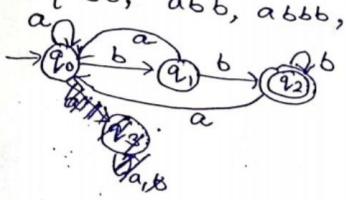
c.) $|w| = 1 \pmod 5$ d.) $|w|_b = 2 \pmod 3$

e.) $|w|_a = 3 \pmod 4$ f.) $|w|_b = 1 \pmod 5$

Scanned by CamScanner

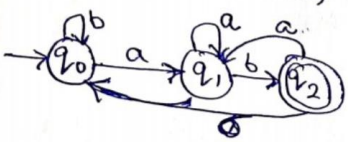
i.) ending with bb.

$L = \{ bb, abb, abbb, bbb, babb, \ldots \}$



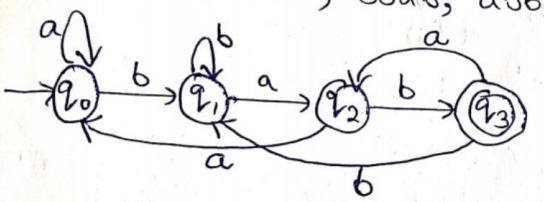
ii) ending with $w = ab$;

$L = \{ ab, aab, bab, aaab, bbab, baab, \ldots \}$



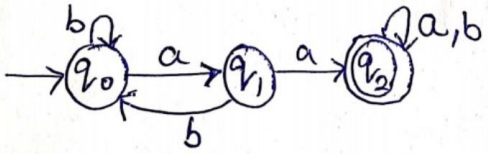
iii) ending with bab.

$L = \{ bab, abab, bbab, abbab, \ldots \}$



iv) Contains a substring $w = aa$.

$L = \{ aa, baa, aab, baab, aaa, \ldots \}$

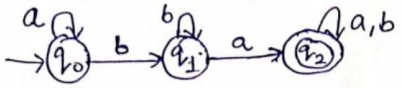


v) ~~Eol~~

Scanned by CamScanner

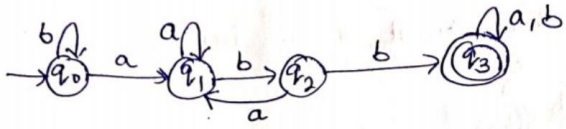
v) Containing substring $w = ba$.

$L = \{ ba, bba, aba, bab, aba, baa, abab, bbab, \ldots \}$



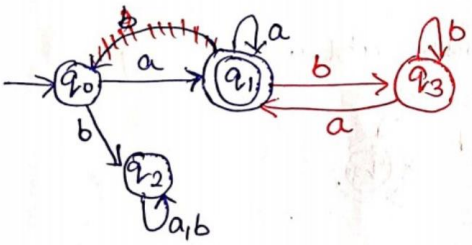
vi) Containing substring abb.

$L = \{ abb, babb, abba, abbb, \ldots \}$



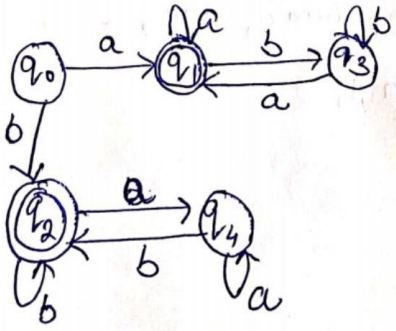
vii) Starts & ends with 'a'.

$L = \{ a, aa, aaa, aba, abba, \ldots \}$



viii) Start & end with same symbol $\Sigma = \{a, b\}$.

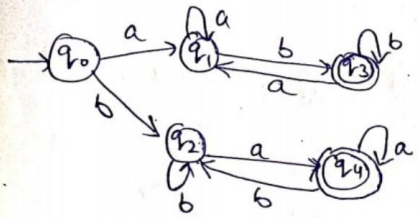
$L = \{ a, b, aba, bab, baab, babb, abaa, baabab, \ldots \}$



Scanned by CamScanner

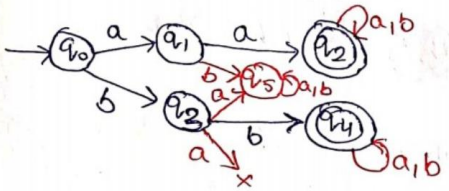
, bbab, --

x) Starts with 'a', ~~end~~ ending with 'b' and vice-versa.

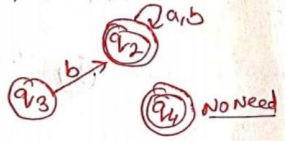


) starts with either 'aa' or 'bb'.

L = {aa, bb, aab, bba, aabb, aaba, bbba,

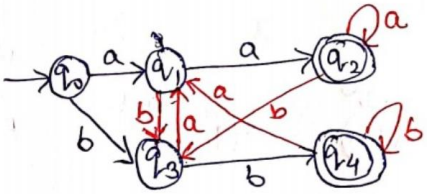


or



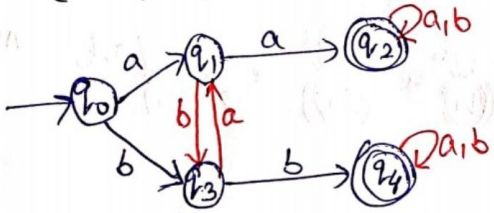
xi) ending with either 'aa' or 'bb'.

L = { aa, bb, baa, bbb, aaa, abb, babb, ---- }

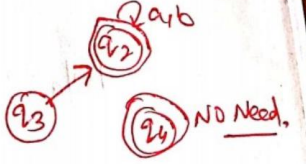


ab, xii) contains substring either 'aa' or 'bb'.

L = { aa, bb, baab, aaa, bbb, abbb, baaa, babb, ---}

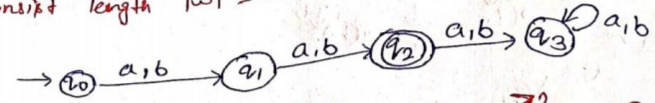


or

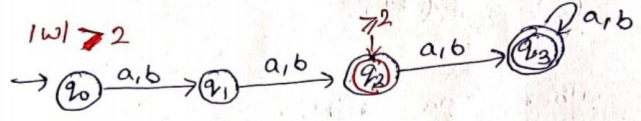


Scanned by CamScanner

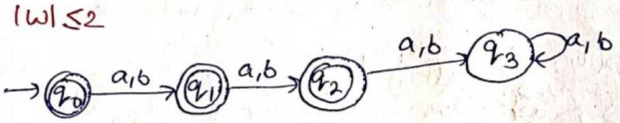
xiii) ~~Contain substring either 'aa' or 'bb'~~
Consist length $|w| = 2$.



xiv) $|w| \geq 2$



(xv) $|w| \leq 2$



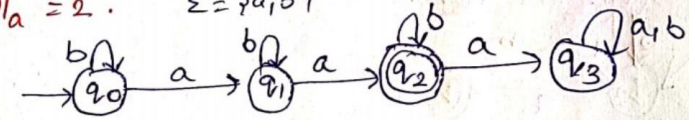
Generally, $|w| = n$; No. of states required?

↳ $n+2$

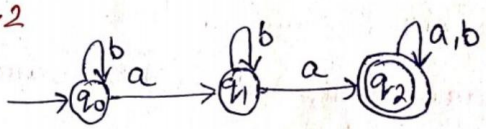
atleast ⟶ $n+1$

atmost ⟶ $n+2$

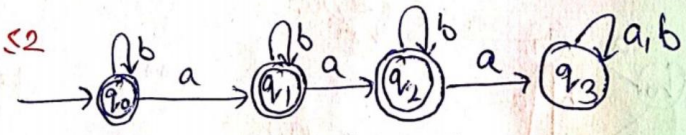
(xvi) $|w|_a = 2$. $\Sigma = \{a, b\}$?



$|w|_a \geq 2$ Atleast 2 a's.



$|w|_a \leq 2$ Atmost 2 b's.



*NOTE ⟶ F.A. doesnot hv memory.

Scanned by CamScanner

Design a DFA over $\Sigma = \{0, 1\}$, that contains set of strings of 0's & 1's which contain substring. 0101.

Construct a DFA to accept the set of all strings of the form $0^{2n}11$; $n \geq 0$.

Scanned by CamScanner