2

FINITE STATE MACHINE

2.1 AUTOMATA

An automaton is a mathematical model of a system with discrete inputs and discrete outputs. The system is a control unit, which can be any one of a finite number of internal states and which can change states in some defined manner. The Fig. 2.1 shows a general representation of an automaton. The input is a string over a given alphabet written on an input file; which the automata can read but cannot change. The input file is divided into set of cells where each cell contains one symbol at a time and also read the input file left to right one symbol at a time. It also detects the end of the input string by using specific symbol (i.e. write end marker). The output is also a string of some form. It may have a temporary storage device, consisting of an unlimited number of cells, each cell containing a single symbol from an alphabet (cell can contain same or different symbols).

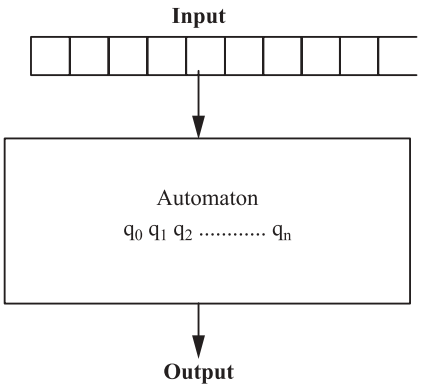


Fig. 2.1

An automaton is used to operate discrete timeframe. At any given time, the system is in some internal state and the input is scanning a particular symbol on the input file. The internal state of the system at the next time step is determined by the next state or transition system. This transition system gives the next state in terms of the current state, the current input symbol and the information currently in the temporary storage. During the transition from one state to next state the output may be produced or the information in the temporary storage changed depending upon the designed machine. The transition from one state to another state is called **move**.

2.2 DETERMINISTIC FINITE AUTOMATON (DFA)

A DFA consists of following three things

1. A finite set of states, one of which is designed as the initial state, called the starting state and some (may be none) of which are designed as final states.
2. An input alphabet Σ.
3. A transition system (i.e. transition graph, transition table or transition function) that tells for each state and each letter of the input alphabet, the state to which to go next.

or Mathematically a DFA is given by $M = (Q, \Sigma, \delta, q_0, F)$

1. Q is a finite non empty set of states.
2. Σ is a finite non empty set of input symbols.

3. δ is a transition system and $\delta \in Q \times \Sigma \rightarrow Q$
4. q_0 is an initial state and $q_0 \in Q$
5. F is a set of accepting states (or final states) and $F \subseteq Q$

The above definition describes the composition of a DFA but not how it works. It works by being presented with an input string of letters that it reads letter by letter starting at the leftmost letter. Beginning at the starting state, the letters determine a sequence of states. The sequence of states ends when the last input letter has been read and if the last state is an accepting (or final) state then that string is *accepted* by a DFA otherwise rejected.

2.2.1 TRANSITION SYSTEM

A transition system can be represented in any one of the following three ways:

- Transition graph
- Transition table
- Transition function.

(1) Transition graph: A transition graph is a finite directed labeled graph in which each vertex (or node) represents a state and the directed edges indicate the transition of the state and the edges are labelled with the input alphabet.

A transition graph is shown in Fig 2.2. In the Fig 2.2 the initial state is represented by a ring (or circle) with an arrow pointing towards it, the final state by two concentric rings (or circles) and the other states are represented by just a ring (or circle). The edges are labeled by input.

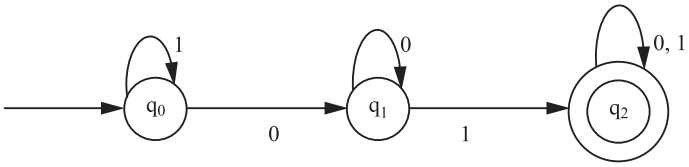


Fig. 2.2

For example, in the Fig 2.2 state q_0 is an initial state, q_2 is a final state and q_1 is another state, and the transitions performed by these states are follows

From state q_0 and input 0, go to the next state q_1

From state q_0 and input 1 go to the next state q_0

From state q_1 and input 0, go to the next state q_1

From state q_1 and input 1, go to the next state q_2

From state q_2 and input 0, go to the next state q_2

From state q_2 and input 1, go to the next state q_2

(2) Transition Table: A DFA may be conveniently represented by a *transition table*. The transition table is a tabular representation of a machine. For example, the tabular representation of Fig. 2.2 is shown in table 1.

Input alphabets

δ : Q \ Σ	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
(q_2)	q_2	q_2

Table 1

The table represents the transition graph of δ i.e. to find the value of present state q and input alphabet x we have to look at the row labeled q and the column labeled x. The initial state is marked by an $\rightarrow$ and the final state is marked by a single ring (or circle).

(3) Transition function: A DFA may also be represented by a transition function δ. The transition function of Fig 2.2 can be represented as follows:

$\delta(q_0, 0) = q_1$

$\delta(q_0, 1) = q_0$

$\delta(q_1, 0) = q_1$

$\delta(q_1, 1) = q_2$

$\delta(q_2, 0) = q_2$

$\delta(q_2, 1) = q_2$

Here the first argument of transition function δ represents the present (current) state and second argument represents the input letter and the right hand side represents the next state.

For example.

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

q_0 is a present state,

0 is an input alphabet

and q_1 is a next state

Example 1: *Consider the DFA machine M = ($\{q_0, q_1, q_2\}$, {0,1}, δ, q_0, $\{q_2\}$) where a transition system δ is defined in Fig. 2.2.*

This represents a mathematical model of DFA machine where the tuples values are

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

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

δ is shown in Fig 2.2

q_0 is an initial state

q_2 is a final state

Language of a DFA

To each DFA machine M we associate a language $L(M) \subseteq \Sigma^*$. To see whether a word $w \in L(M)$, we put a marker in the initial state and after reading a symbol move forward the marker along the edge marked with this symbol. If we reach the accepting state at the end of the word w, then $w \in L(M)$, otherwise $w \notin L(M)$.

In the above example we have $0 \notin L(M)$, $101 \in L(M)$ and $1101 \in L(M)$. Indeed we have, L(M) = {w|w contains the substring 01}.

To be more precise we give a *formal definition of L(M)*. First we define the entered transition function $\delta \in Q \times \Sigma^* \rightarrow Q$. Intuitively, $\delta(q, w)=q'$ if starting from state q we end up in state q′ when reading the word w formally, δ is defined by following rules:

Rule 1 $\delta\{q, \in\} = q$

Rule 2 $\delta\{q, xw\} = \delta\{\delta(q, x), w\}$

Here xw stands for a non empty word whose first symbol is x and the rest is w i.e. x is a prefix of word w. sometimes w may be empty.

For example

If xw = 010 then this entails that

x = 0 and w=10.

i.e. $xw = 0$ entails $x = 0$ and $w = \in$

As an example we calculate

$\delta(q_0, 101) = q_2$ (Using Fig. 2.2)

Using δ we may now define formally

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

Hence we have that $101 \in L(M)$ because

$\delta(q_0, 101) = q_2$ and $q_2 \in F$ as shown below

$\delta(q_0, 101) = \delta(\delta(q_0, 1),01)$ by rule 2

$= \delta(q_0, 01)$

$= \delta(\delta(q_0, 0)1)$ by rule 2

$= \delta(q_1, 1)$

$= \delta(\delta(q_1, 1), \in)$ by rule 2

$= \delta(q_2, \in)$

$= q_2$ by rule 1

Acceptability of a language (or string) by a finite Automaton

A language (string) w is accepted by a finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, w) = q$ for some $q \in F$.

Example 2: *Consider the following DFA machine $M = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_1\})$ where a transition graph is shown in Fig. 2.3.*

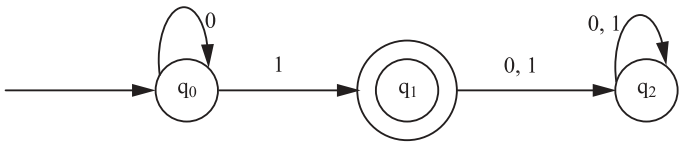


Fig. 2.3

Check if the string 00001 is accepted by DFA. Also find the sequence of states.

Solution: The given language is $\delta(q_0, 00001)$

$= \delta(\delta(q_0, 0), 0001)$ by rule 2

$= \delta(q_0, 0001)$ from transition graph i.e. $\delta(q_0, 0) = q_0$

$= \delta(\delta(q_0, 0), 001)$ by rule 2

$= \delta((q_0, 001)$

$= \delta(\delta(q_0, 0), 01)$ by rule 2

$= \delta(q_0, 01)$

$= \delta(\delta(q_0, 0), 1)$ by rule 2

$= \delta(q_0, 1)$

$= \delta(\delta(q_0, 1), \in)$ by rule 2

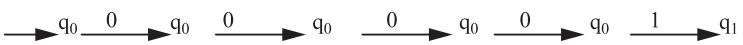
$= \delta((q_1, \in)$

$= q_1$ by rule 1

$= \boldsymbol{q_1 \in F}$

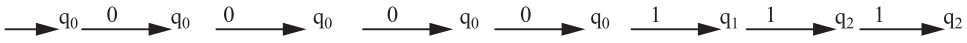
Hence the string is accepted by DFA machine.

The sequence of states corresponding to the input symbol is given by



Example 3: *Find the sequence of states corresponding to the input symbol 0000111 for the transition digraph shown in Fig. 2.3.*

Solution: The sequence of states of input symbol 0000111 corresponding to the given transition digraph is given below:



Since q_2 is not a final states, so the language 0000111 is not accepted by a DFA machine

Note: This DFA machine accepts all set of strings (or words) of the form $\{0^n1 | n \geq 0$ or $0^*1\}$.

Example 4: *Consider the following DFA machine $M = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}, \{a, b\}, \delta, q_0, \{q_5\})$ where a transition graph is shown in Fig. 2.4.*

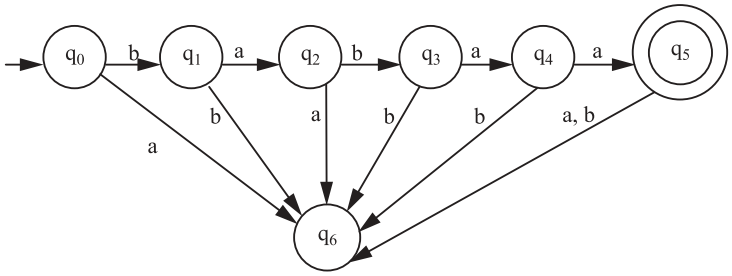


Fig. 2.4

Check if the string babaa is accepted by DFA. Also find the sequence of states.

Solution: The given language is $\delta(q_0, babaa)$

$= \delta(\delta(q_0, b), abaa)$ by rule 2

$= \delta(q_1, abaa)$ from transition graph i.e. $\delta(q_0, b) = q_1$

$= \delta(\delta(q_1, a), baa)$ by rule 2

$= \delta((q_2, baa)$

$= \delta(\delta(q_2, b), aa)$ by rule 2

$= \delta(q_3, aa)$

$= \delta(\delta(q_3, a), a)$ by rule 2

$= \delta(q_4, a)$

$= \delta(\delta(q_4, a), \in)$ by rule 2

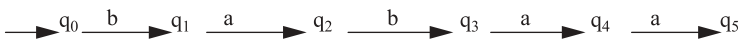
$= \delta((q_5, \in)$

$= q_5$ by rule 1

$= \boldsymbol{q_5 \in F}$

Hence the string is accepted by DFA machine.

The sequence of states corresponding to the input symbol is given by



Note: This machine accepts only one string i.e. babaa, no other string is accepted by this DFA machine because if you change the string (or word), the sequence of state will reach on non final state.

Some basic concept to design a DFA

To design a DFA machine we have to find our own logic but here we try to give some examples to develop our logic for DFA machine. We know that the heart of DFA machine is transition system. It means when we are able to design a transition system for any problem, we are able to design a DFA machine.

Example 5: *Find a DFA machine that accepts all the language (or strings) of a's including ∈ over input alphabet ∑={a}*

Solution: The language contains only **a** of any length including ∈ i.e. the set of language looks as {∈, a , aa, aaa, aaaa, ……..…..}. To design a DFA machine we use the following steps

Step 1: Let us consider an initial state say q_0

Step 2: The language contains empty string (i.e.∈), so the initial and final state must be the same states.

Step 3: Since the input symbol is **a,** then on initial state q_0 (or present state q_0) and on input symbol **a** the state will not change i.e. it will remain q_0, which is a final state and transition diagram is as shown in Fig. 2.5

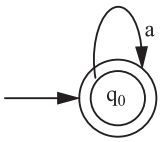


Fig. 2.5

This transition digraph contain all five tuples value of a DFA machine i.e. (Q,Σ,δ,q_0, F).

So a DFA machine is $(\{q_0\}, \{a\}, \delta, q_0, \{q_0\})$ where δ is shown in Fig. 2.5.

Example 6: *Find a DFA machine that accepts all the language(or string) of a's and b's of any order and of any length including ∈ over input alphabet ∑={a, b}*

Solution: The language contains **a** and **b** of any order and of any length including ∈ i.e. the set of language looks like as follows : {∈, a ,b, aa, bb, ab, ba, aaa, abb, aab, bbb,baa, bba, aba,bab, ……}. To design a DFA machine we use the following steps

Step 1: Let us consider an initial state say q_0

Step 2: The language contains empty string (i.e.∈), so the initial and final states must be the same states.

Step 3: Since the input symbols are **a** and **b,** then on initial state q_0 (or present state q_0) and on input symbol **a** and **b** the state will not change i.e. it will remain q_0, which is a final state and transition digraph is shown in Fig. 2.6.

Fig. 2.6

This transition digraph contains all five tuples value of a DFA machine i.e (Q,Σ,δ,q_0,F).

So a DFA machine is $(\{q_0, \{a, b\}, \delta,q_0,\{q_0\})$ where δ is shown in Fig. 2.6.

Example 7: *Find a DFA machine that accepts all language (or strings) that have exactly one 0 over input alphabet ∑={0,1}*

Solution: The language contains exactly one letter of 0 and the other letters may be only 1's of any order and of any length either before 0 or after 0 or before and after both, including ∈. So the set of language looks as {0, 10, 01, 110, 011, 101,……..}. To design a DFA machine we use the following steps

Step 1: Let us consider an initial state say q_0.

Step 2: Since the first input symbol is either 0 or 1, then on initial state q_0 (or present state q_0) and on input symbol 0 consider the next state i.e. q_1 and on initial state q_0 (or present state q_0) and on input symbol 1 the state will not change.

State 3: On state q_1 the input symbol may be any one i.e. either 0 or 1. If input symbol is 1 then the next state will not change i.e. it will remain q_1 which is a final state.

Step 4: On final state q_1 if input symbol is 0 the string will not be accepted by DFA as the string to be accepted contains one 0 only.

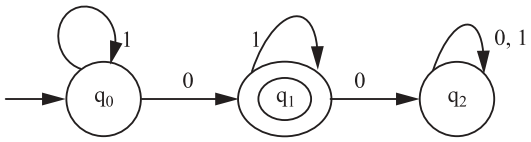


Fig. 2.7

The transition digraph is shown in Fig. 2.7.

This transition digraph contain all five tuples value of a DFA machine i.e. (Q,Σ,δ,q_0,F).

So a DFA machine is $(\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0,\{q_1\})$ where δ is shown in Fig. 2.7.

Example 8: *Find a DFA machine that accept all strings beginning with 0 over input alphabet Σ={0,1}*

Solution: The language begins with letter 0 and the other letters may be of any order of 0's and 1's and of any length including $\in$. So the set of language look likes as follows {0, 00, 01, 000, 011,010,001, 0000,0001,……..}. To design a DFA machine we use the following steps:

Step 1: Let us consider an initial state say q_0

Step 2: Since the first input symbol is 0 then on initial state q_0 (or present state q_0) and on input symbol 0 consider the next state i.e. q_1.

State 3: On state q_1 the input symbol may be any one i.e. either 0 or 1. If input symbol is either 0 or 1 the next state will not change i.e. it will remain q_1. It is a final state.

Step 4: On initial state q_0 if input symbol is 1 the string will be rejected by DFA. The transition digraph is shown in Fig. 2.8.

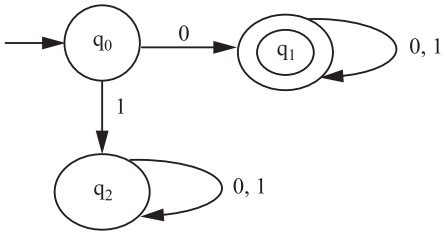


Fig. 2.8

This transition digraph contain all five tuples value of a DFA machine i.e(Q,Σ,δ,q_0,F).

So a DFA machine is $(\{q_0, q_1, q_2\}\ \{0,1\}, \delta,q_0,\{q_1\})$ where δ is shown in Fig. 2.8.

Example 9: *Find a DFA machine for the language L = { 0w0 : w∈{0,1}* }*

Solution: The language contains first and last letter as 0 and intermediate letters may be of any order of 0′s and 1′s and of any length including $\in$ i.e. the set of language looks as {00, 000, 010, 0000, 0010, 0100,0110,…..}. To design a DFA machine we use the following steps

Step 1: Let us consider an initial state say q_0

Step 2: Since the first input symbol is 0 then on initial state q_0 (or present state q_0) and input symbol 0 consider the next state as q_1.

State 3: On state q_1 the input symbol may be any one i.e. either 0 or 1 of any order and of any length. If input symbol is 0 then the next state is q_2 which is a final state and if input symbol is 1 the next state is q_1 because 1 may be of any length.

Step 4: On state q_2 if input symbol is 0 then the next state will not change i.e. it will remain q_2 but if input symbol is 1 then the next state is q_1

Step 5: On initial state q_0 if input symbol is 1 the string will not be accepted by DFA.

The transition digraph is shown in Fig.2.9. This transition digraph contain all five tuples value of a DFA machine i.e (Q,∑,δ,q_0,F).So a DFA machine is ({q_0, q_1, q_2,q_3} {0,1}, δ,q_0,{q_3}) where δ is shown in Fig. 2.9.

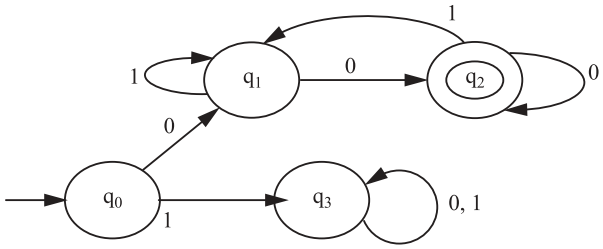


Fig. 2.9

Example 10: *Find a DFA machine that accept all strings (or language) having different first and last letters over input alphabet ∑={a,b}*

Solution: The DFA machine contains all strings whose first letter is different from the last letter i.e. if first letter is **a** then the last letter must be **b** or if first letter is **b** then the last letter must be **a**. In this language (or strings) intermediate letters may be in any order and of any lengths. The transition digraph is shown in Fig.2.10. In this transition diagram if the read/ write head will read first letter the state will change and if the next symbol is similar to first symbol the state will not change but if it is different, the state will change as shown in Fig. 2.10.

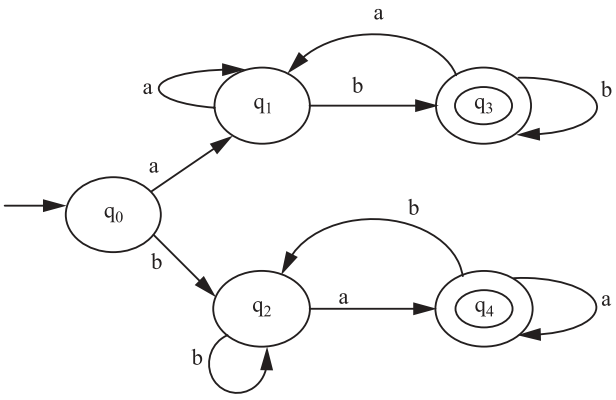


Fig. 2.10

Hence the DFA machine is ({q_0, q_1, q_2, q_3, q_4}, {a, b}, δ, q_0, {q_3,q_4})})

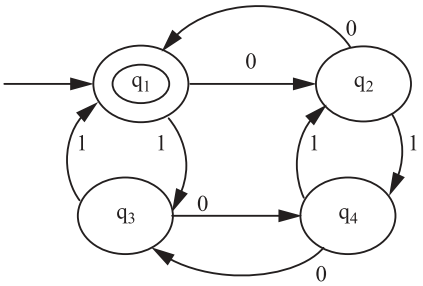


Fig. 2.11

Example 11: *Find a DFA machine that accepts an even number of either 0's or 1's or both 0's and 1's over input symbol ∑={0,1}.*
[U.P.T.U., B.Tech., 2001-02]

Solution: The heart part δ (i.e. transition system) of DFA machine is shown in Fig. 2.11

Hence a DFA machine is ({q_1, q_2, q_3, q_4}, {0, 1}, δ, q_1, {q_1})

Example 12: *Find a DFA machine that accepts the language which has a sub-string 0101 over input alphabet Σ = {0, 1}*

Solution: The heart part δ (i.e. transition system) of DFA machine is shown in Fig.2.12.

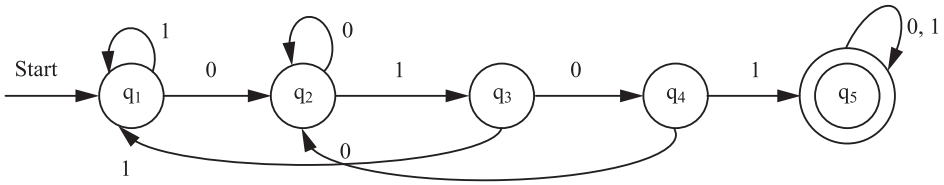


Fig. 2.12

Hence a DFA machine is $(\{q_1, q_2, q_3, q_4, q_5\}, \{0, 1\}, \delta, q_1, \{q_5\})$

Example 13: *Find a DFA machine that accepts the language which has either odd number of 0's or even number of 1's but not both together over input alphabet Σ = {0, 1}*

Solution: The heart part (i.e. transition system) of DFA machine is shown in Fig.2.13.

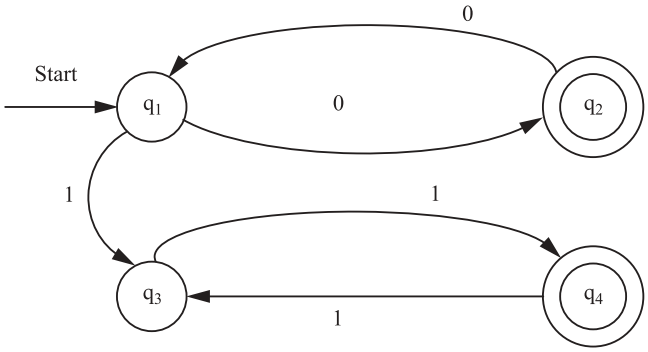


Fig. 2.13

Hence a DFA machine is $(\{q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_1, \{q_2, q_4\})$

Example 14: *Find a DFA machine that accepts the language which has both 01 and 10 as substring over input alphabet Σ = {0, 1}*

Solution: The heart part (i.e. transition system) of DFA machine is shown in Fig. 2.14

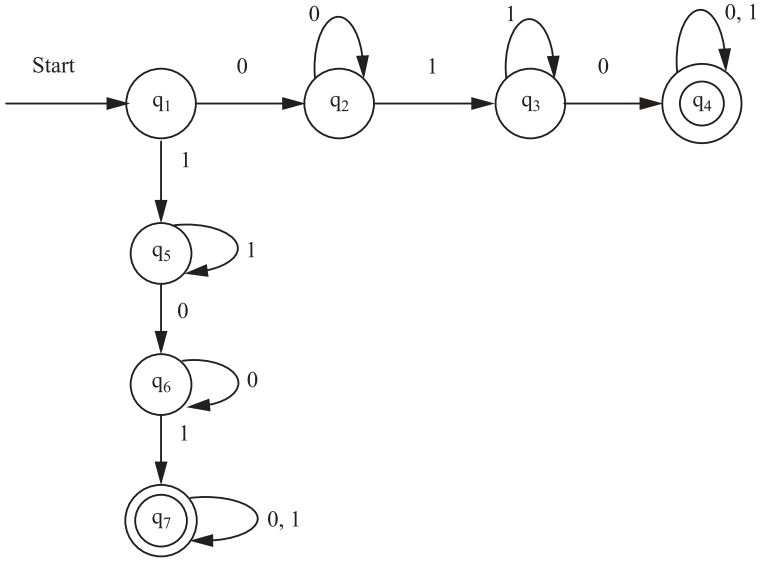


Fig. 2.14

Hence a DFA machine is $(\{q_1, q_2, q_3, q_4, q_5, q_6, q_7\}, \{0, 1\}, \delta, q_1, \{q_4, q_7\})$

Example 15: *Find a DFA machine that accepts the language which has neither 00 nor 11 as a substring over input alphabet $\Sigma = \{0, 1\}$*

Solution: The heart part (i.e. transition system) of DFA machine is shown in Fig. 2.15.

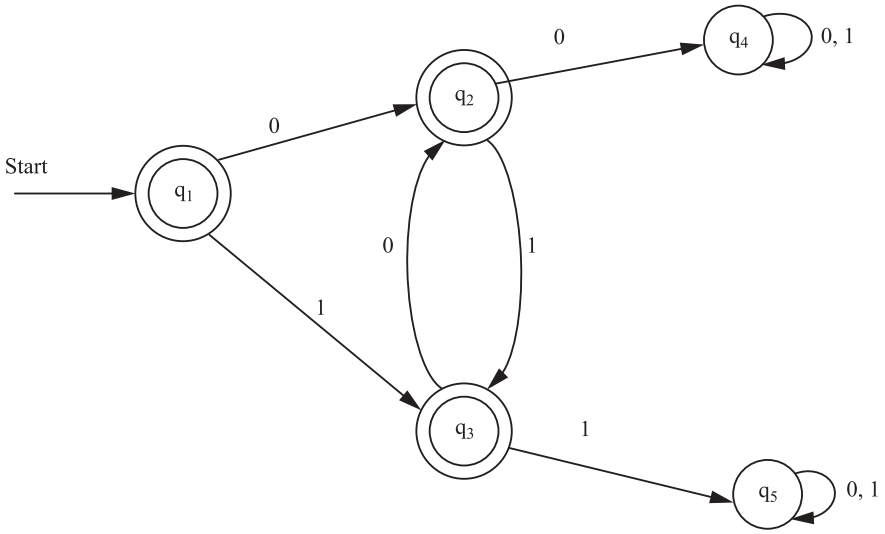


Fig. 2.15

Hence a DFA machine is $(\{q_1, q_2, q_3, q_4, q_5\}, \{0, 1\}, \delta, q_1, \{q_2, q_3\})$

Example 16: *Design a DFA machine that accept all strings with no more than five a's over input symbol $\Sigma = \{a, b\}$.*

Solution: The transition diagram of DFA machine δ is shown in Fig. 2.16.

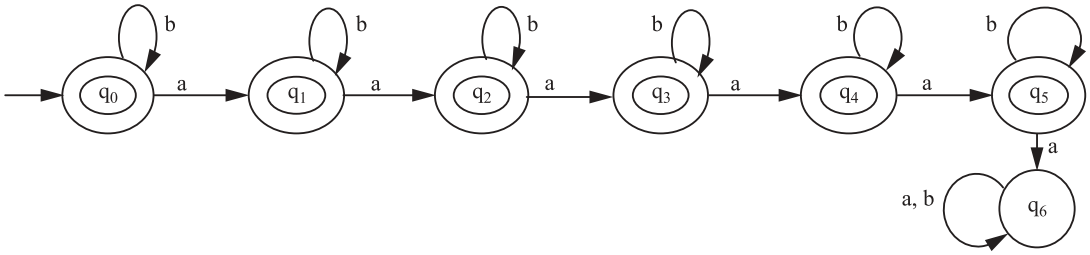


Fig. 2.16

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_2, q_3, q_4, q_5\})$

Example 17: *Design a DFA machine that accept all strings with at most three b's over input symbol $\Sigma = \{a, b\}$.*

Solution: The transition system of DFA machine δ is shown Fig. 2.17.

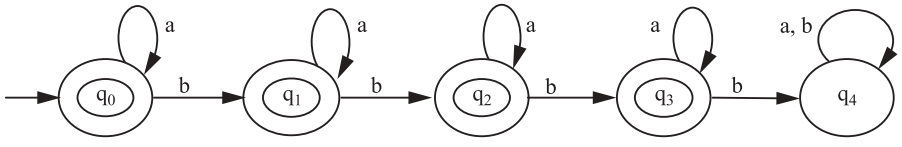


Fig. 2.17

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_2, q_3\})$

Example 18: *Design a DFA machine that accepts all language (or strings) where the number of 1's is multiple of 5 over the alphabet Σ = {0, 1}:*

Solution: The transition system of DFA machine δ is shown Fig. 2.18.

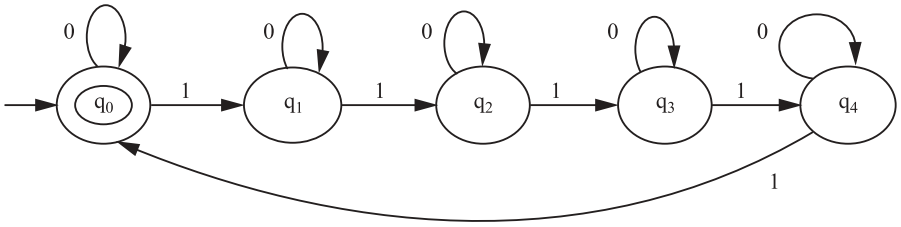


Fig. 2.18

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_0, \{q_0\})$

Example 19: *Consider the following automata(DFA) M.*

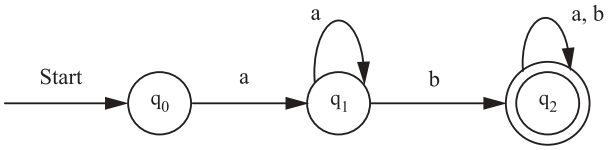


Fig. 2.19

Obtain a DFA which accepts the complement of the language accepted by M.

Solution: A DFA machine which accepts the complement of the above DFA machine is shown below: The complement of a DFA machine can be obtained by converting all final states to non-final states and all non-final states to final states.

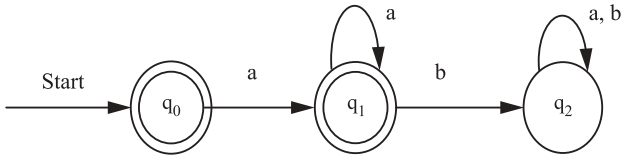


Fig. 2.20

Hence a DFA machine is $(\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_0, q_1\})$

Example 20: *Construct a DFA recognizing the following language (or strings) :*

$\{ a^n b^m \mid$ *n is divisible by 3 and m is divisible by 2 or n - m* $\geq 1 \}$

Solution: The transition system of DFA machine δ is shown below

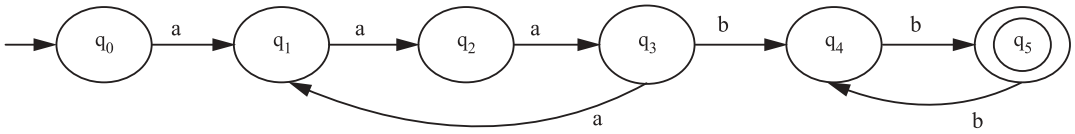


Fig. 2.21

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \delta, q_0, \{q_5\})$

Example 21: *Design a DFA machine that accept the all language (or strings) which are divisible by 3 over the input alphabet Σ = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}*

Solution: The transition table of a DFA machine δ is shown below

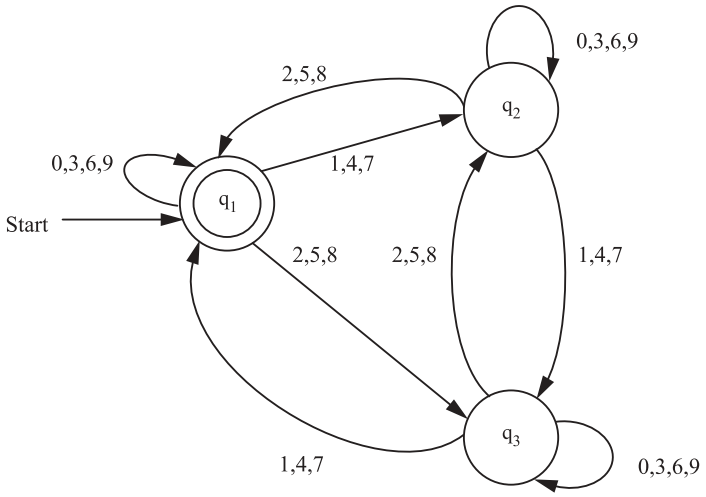


Fig. 2.22

Hence a DFA machine is ({q_1, q_2, q_3}, {0, 1, 2,}, {4, 5, 6, 7, 8, 9,} δ, q_1, {q_1})

Example 22: *Design a DFA machine that accept the all language (or strings) which are divisible by 3 over the input alphabet ∑ = {0, 1}*

Solution: The transition table of a DFA machine δ is shown below

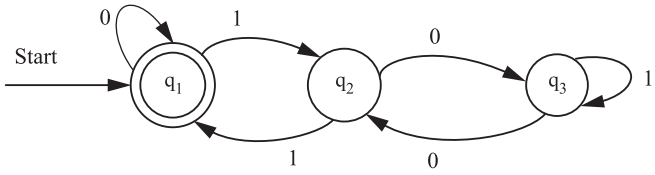


Fig. 2.23

Hence a DFA machine is ({q_1, q_2, q_3}, {0, 1}, δ, q_1, {q_1})

Example 23: *Design a DFA for a vending machine which accepts only 5-rupee coins and delivers tea for each such coins.*

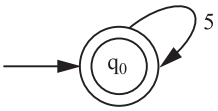


Fig. 2.24

Solution: The heart part (δ) of DFA machine is shown in Fig. 2.24.
Hence a DFA machine is ({q_0}, {5}, δ, q_0, {q_0})

Example 24: *Design a DFA machine which does not accept any other coins other than rupee 5 over input alphabet* $\Sigma = \{1, 2, 5\}$

Solution: The heart part(δ) of DFA machine is shown in Fig. 2.25.

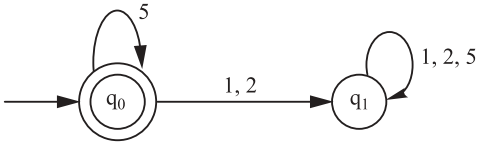


Fig. 2.25

Hence a DFA machine is ({q_0, q_1}, {1, 2, 5}, δ, q_0, {q_0})

Example 25: *Design a DFA nmachine that only accepts any sequence consisting of at least 3 one-rupee coins over input alphabet* $\Sigma = \{1, 2, 5\}$

Solution: The heart part (δ) of DFA machine is shown in Fig. 2.26.

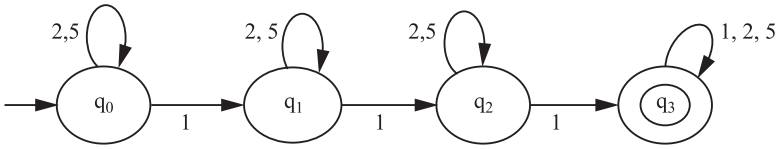


Fig. 2.26

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3\}, \{1, 2, 5\}, \delta, q_0, \{q_3\})$

Example 26: *Design a DFA machine that accepts a sequence of coins in which there is at least one sub-sequence of three consecutive 1's over input alphabet* $\Sigma = \{1, 2, 5\}$

Solution: The heart part (δ) of DFA machine is shown in Fig. 2.27.

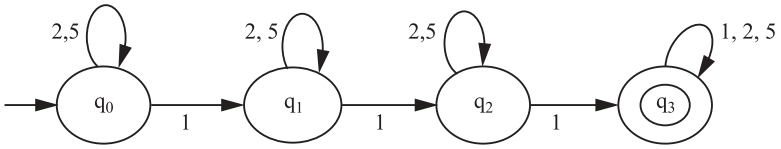


Fig. 2.27

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3\}, \{1, 2, 5\}, \delta, q_0, \{q_3\})$

Example 27: *Design a DFA machine that accepts triple a's or triple b's over input alphabet* $\Sigma = \{a, b\}$

Solution: The heart part (δ) of DFA machine is shown in Fig. 2.28.

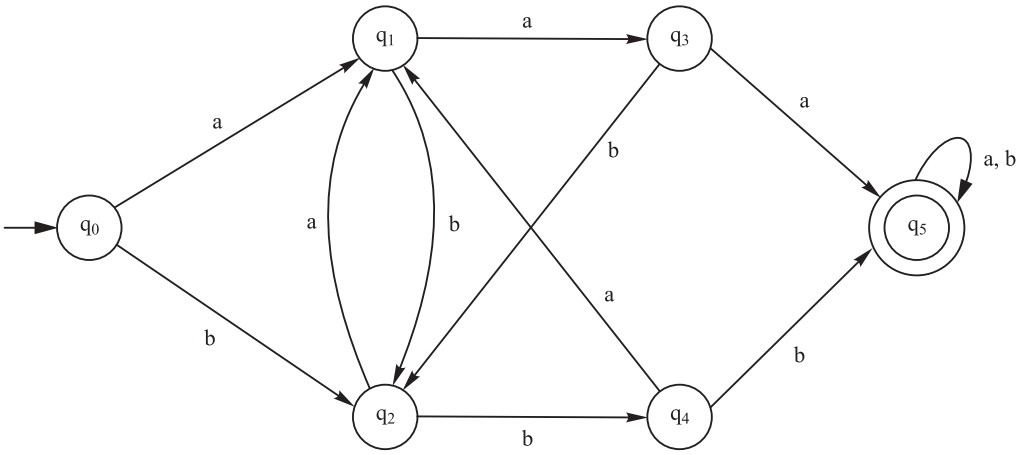


Fig. 2.28

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \delta, q_0, \{q_5\})$

Example 28: *Design a DFA machine that accept the language* $\in + b + ab + bb$ *over input alphabet* $\Sigma = \{a, b\}$

Solution: The heart part (δ) of a DFA machine is shown in Fig. 2.29.

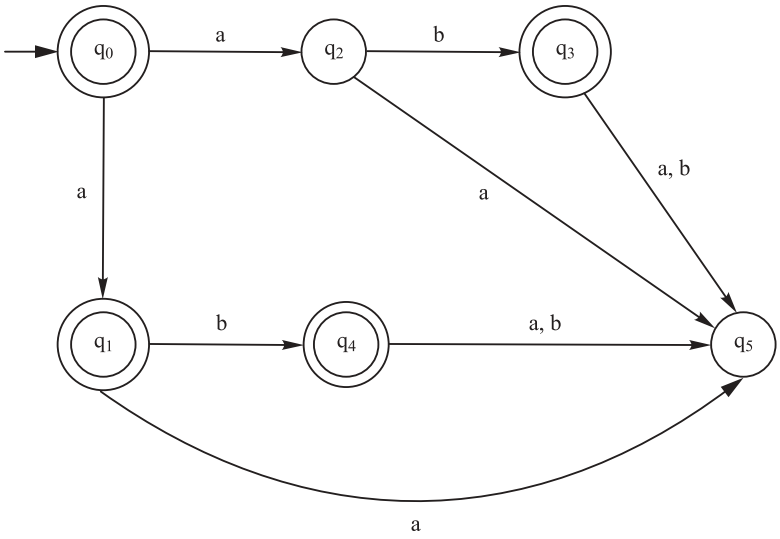


Fig. 2.29

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_3, q_4\})$

Example 29: *Design a DFA machine for the language L = {w ∈ {a, b} : length (w) ≥ 2 and w neither ends in aa nor bb}.*

Solution: The heart part (δ) of a DFA machine is shown in Fig. 2.30.

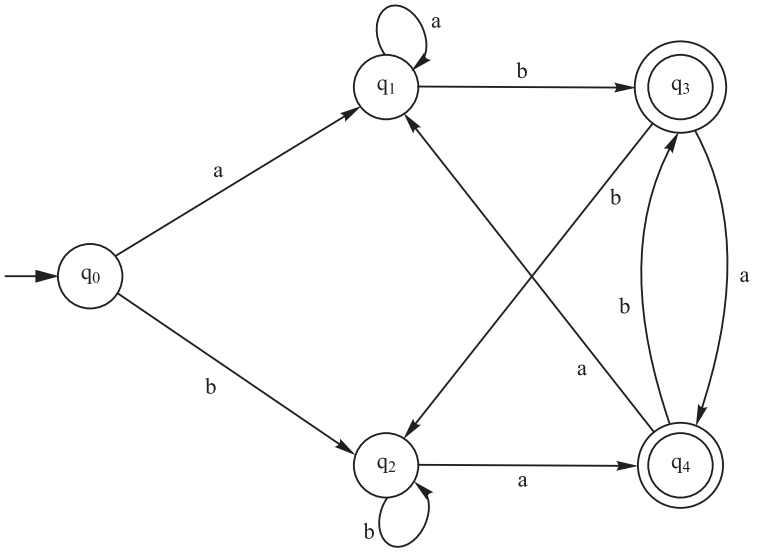


Fig. 2.30

Hence a DFA machine is $(\{q_0, q_1, q_2, q_3, q_4\}, \{a, b\}, \delta, q_0, \{q_3, q_4\})$

2.3 NON-DETERMINISTIC FINITE AUTOMATA (NFA)

A non-deterministic finite automata (NFA) consists of a variable that can only take on a finite number of different states, a read head with which the input tape will be read from left to right, a transition system that forms the control of the automaton, an initial state and one or several final states.

OR (Mathematically)

A non-deterministic finite automaton (NFA) M consists of 5-tuples.

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

where

Q is a finite non-empty set of states,

Σ is a finite alphabet, the input alphabet,

δ is the transition system which represents a mapping between states with input alphabet to power set of states i.e $Q \times \Sigma \rightarrow 2^Q$ (2^Q is power set of Q)

q_0 ($q_0 \in Q$) is the initial state and

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

Note: The only difference between the DFA and NFA is in the transition system δ. For DFA, the outcome is a state which is an element of Q; for NFA, the outcome is a subset of Q. (it may consist of one or more elements of Q).

Example 30: *Consider the following NFA machine $M = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_1\})$ where a transition graph is shown in Fig. 2.31.*

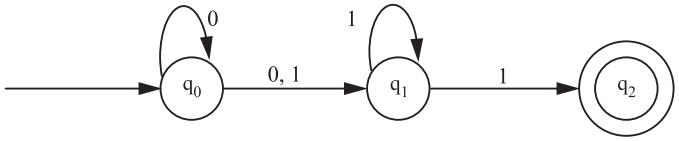


Fig. 2.31

Check if the string 00001 is accepted by NFA. Also find the sequence of states.

Note: Since the present state q_0 contains the next states q_0 and q_1 on input symbol 0, the transition system is a transition system of NFA and the transition diagram is a NFA.

Solution: The given language is $\delta(q_0, 00001)$

$= \delta(\delta(q_0, 0), 0001)$ by rule 2

$= \delta(\{q_0, q_1\}, 0001)$ from transition graph i.e. $\delta(q_0, 0) = \{q_0, q_1\}$

$= \delta(\delta(\{q_0, q_1\}\ 0), 001)$ by rule 2

$= \delta((\delta(q_0, 0) \cup \delta(q_1, 0)), 001)$

$= \delta(\{\{q_0, q_1\} \cup \phi\}, 001)$

$= \delta(\{q_0, q_1\}, 001)$

$= \delta(\delta(\{q_0, q_1\}\ 0), 01)$ by rule 2

$= \delta((\delta(q_0, 0) \cup \delta(q_1, 0)), 01)$

$= \delta(\{\{q_0, q_1\} \cup \phi\}, 01)$

$= \delta(\{q_0, q_1\}, 01)$

$= \delta(\delta(\{q_0, q_1\}\ 0), 1)$ by rule 2

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

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

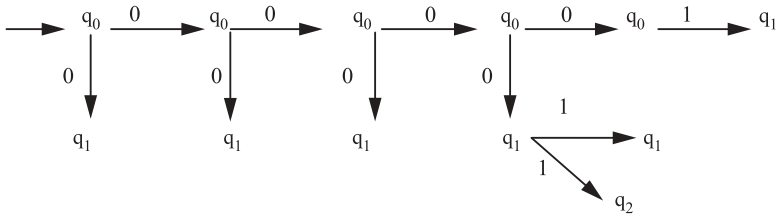
$= \delta(\{q_0, q_1\}, 1)$

$= \delta(\delta(\{q_0, q_1\}\ 1), \in)$ by rule 2

$= \delta((\delta(q_{0,} 1) \cup \delta(q_{1,} 1)), \in)$
$= \delta(\{q_1\} \cup \{q_{1,} q_2\}, \in)$
$= \delta(\{\{q_1, q_2\}, \in)$
$= \delta(\{q_{1,} q_2\}, \in)$
$= \delta(q_{1,} \in) \cup \delta(q_{2,} \in)$
$= q_1 \cup q_2$
$= \{q_1, q_2\}$

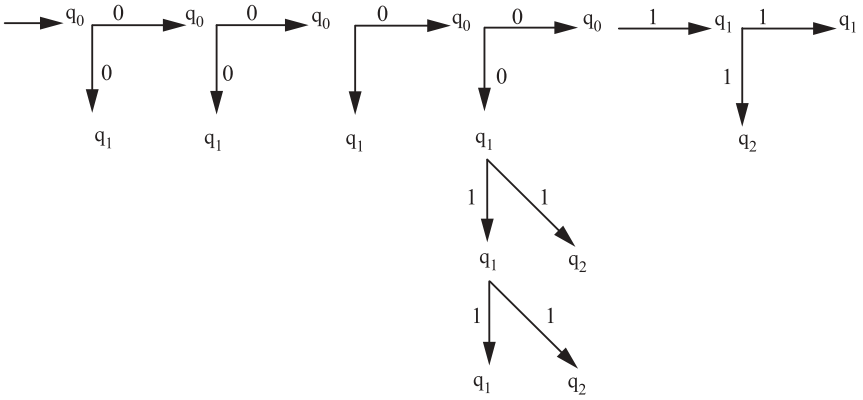
This set of states contain the final state q_2. Hence the string is accepted by this NFA machine.

The sequence of states corresponding to the input symbol is given by



Example 31: *Find the sequence of states corresponding to the input symbol 000011 for the transition digraph shown in Fig. 2.31.*

Solution: The sequence of states of input symbol 000011 corresponding to the given transition digraph is given below



The highest length from starting symbol is 6 and the states are of 6 length from initial state q_0 are $\{q_1, q_2\}$. Since in the states $\{q_1, q_2\}$, q_2 is a final state, so the language 000011 is accepted by a NFA machine

Example 32: *Design a NFA machine that accept all strings ending with aa over input symbol Σ = {a, b}.*

Solution: The transition diagram of NFA machine δ is shown in Fig.2.32.

Hence the NFA machine

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

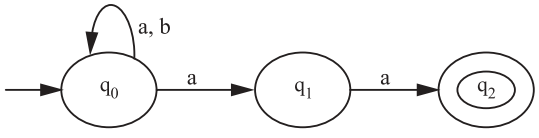


Fig. 2.32

Example 33: *Design a NFA machine that accept all strings beginning with **ab** and ending with **bb** over input symbol Σ = {a, b}.*

Solution: The transition diagram of NFA machine δ is shown in Fig. 2.33.

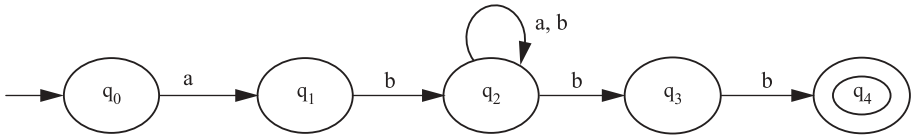


Fig. 2.33

Hence NFA machine is $(\{q_0, q_1, q_2, q_3, q_4\}, \{a, b\}, \delta, q_0, \{q_4\})$

Example 34: *Design a NFA machine that accept all strings with at least two **a's** over input symbol Σ = {a, b}.*

Solution: The transition diagram of NFA machine δ is shown in Fig. 2.34.

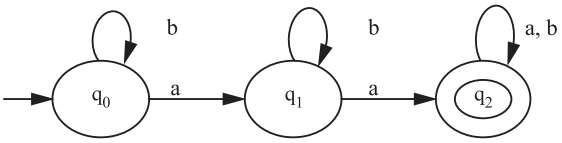


Fig. 2.34

Hence NFA machine is $(\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$

Example 35: *Design a NFA machine that accept all strings with exactly two **b's** and at least one **a** over input symbol Σ = {a, b}.*

Solution: The transition diagram of NFA machine δ is shown in Fig. 2.35.

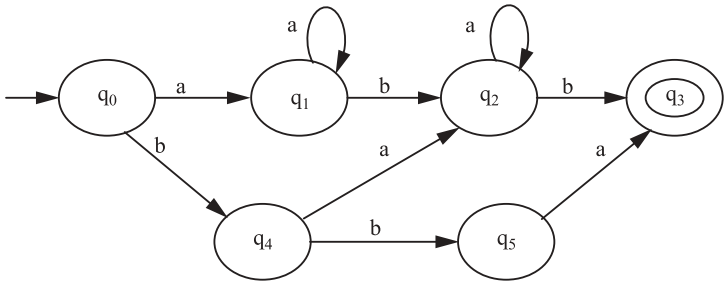


Fig. 2.35

Hence NFA machine is $(\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \delta, q_0, \{q_3\})$

2.4 EQUIVALENCE OF NFA AND DFA [U.P.T.U., B.Tech., 2001-2002]

For every non-deterministic finite automata (NFA) there exists an equivalent deterministic automata (DFA). The equivalence is defined in terms of language acceptance. Since a NFA is nothing but a finite automata in which zero, one or more transition on an input symbol is permitted. We can always construct a DFA which will simulate all the moves of NFA on particular input symbol in parallel to get a DFA in which there will be exactly one transition input symbol. Hence it will be a DFA equivalent to NFA.

Since the DFA equivalent to NFA is to simulate the moves of NFA in parallel, every state of DFA will be a combination of one or more states of NFA. Hence every state of DFA will be represented by some subset of set of states of NFA and therefore transformation of NFA to DFA is normally called as **subset construction**. Therefore if subset has **n** sates then the number of states of DFA will be equivalent

to 2^n, with initial state corresponding to the subset $\{q_0\}$. Therefore transformation of NFA to DFA involves, finding all possible subsets of set of states of NFA and then considering each subset to be state of DFA and finding transition from it on every input symbol. But out of all states of DFA obtained in this way, if any state is not reachable from initial state on any possible input sequence, then such a state does not play any role to decide the language to be accepted by a DFA. Such states are eliminated from the set of states. This can be done as follows

Let M = (Q, Σ, δ, q_0, F) be a NFA accepting language L. We construct a DFA M′ that can also accept the same language L. Then a DFA M′ is defined as follows

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

where

Q' = 2^Q (Power set of Q or all subset of Q) and all states of Q′ are denoted by $[q_1, q_2, q_3 \ldots\ldots, q_n]$ where $q_1, q_2, q_3 \ldots\ldots, q_n \in Q$

Σ = Similar to NFA

q_0' = $[q_0]$

F′ = Set of all subsets of Q′ containing an element of F(i.e. final state of NFA)

δ' (Transition system): To define transition system, use the following steps.

Step 1: Draw an initial state (or vertex) with label $[q_0]$ of transition diagram δ'

Step 2: Take this state (i.e.$[q_0]$) and identify all next states from Q′ on all different given input symbols.

Let us consider the next states as $[q_i, q_j, \ldots\ldots, q_k]$ on input symbol $0 \in \Sigma$ from state $[q_0]$

Create a state labelled $[q_i, q_j, \ldots\ldots, q_k]$ and add to transition digraph δ' an edge from $[q_0]$ to $[q_i, q_j, \ldots\ldots, q_k]$ with label 0.

Step 3: Take any state of step 2 (or previously defined state) and find new next state (i.e. it does not exist already) from Q′ on all different given input symbols .

Let us consider the next state as $[q_i, q_j, \ldots\ldots, q_n]$ on input symbol $0 \in \Sigma$ from state $[q_i, q_j, \ldots, q_k]$

Compute $\delta'(q_i, 0)$, $\delta'(q_j, 0), \ldots\ldots, \delta'(q_k, 0)$ and then form a union of all δ', yielding the set of states $[q_i, q_j, \ldots\ldots, q_n]$

Create a state with label $[q_i, q_j, \ldots\ldots, q_n]$ and add to transition digraph δ' an edge from $[q_i, q_j, \ldots\ldots, q_k]$ to $[q_i, q_j, \ldots\ldots, q_n]$ labeled 0.

Step 4: Repeat the step 3 until no new states are generated

Step 5: Draw all states as a final states of δ' if it contain the final state of NFA.

Example 36: *Find a deterministic finite automata equivalent to M ($\{q_0, q_1, q_2\}$,{0,1},δ, q_0,$\{q_2\}$) where δ is defined in Fig.2.36.*

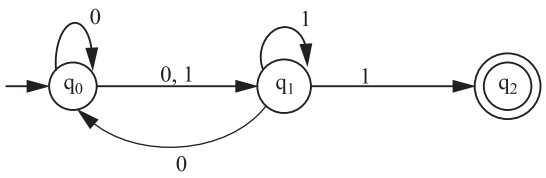


Fig. 2.36

Solution: The given NFA, M = ($\{q_0, q_1, q_2\}$, {0,1},δ, q_0, $\{q_2\}$)) then its equivalent DFA M′ be defined by

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

where

$Q' = 2^Q$ (Power set Q)

i.e.$\{\phi, [q_0],[q_1],[q_2],[q_0, q_1],[q_0,q_2],[q_1, q_2],[q_0,q_1, q_2]\}$

These are the total number of states of DFA

Σ = Similar to NFA i.e. {0,1}

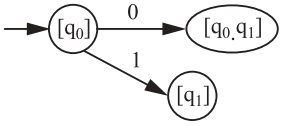
q_0' = $[q_0]$. This is the initial state of DFA

F' = $\{[q_2], [q_0,q_2],[q_1, q_2],[q_0,q_1, q_2]\}$. These are the final states of DFA [$\subseteq Q'$ containing q_2]

δ' = The transition digraph is determined as follows:

By step 1: The initial state is $[q_0]$, so transition diagram for it is shown here:

By **step 2:** The next states from $[q_0]$ are $[q_0,q_1]$ and $[q_1]$ on giving input symbol 0 and 1 respectively. Then add these new states (i.e. $[q_0,q_1]$ and $[q_1]$) in transition digraph from state $[q_0]$ with labels 0 and 1 respectively. So the new transition digraph is given here



By step 3: The next states from $[q_1]$ are $[q_0]$ and $[q_1, q_2]$ on input symbol 0 and 1 respectively, and from $[q_0,q_1]$ are $[q_0,q_1,]$ and $[q_1, q_2]$ on giving input symbol 0 and 1 respectively as explained below.

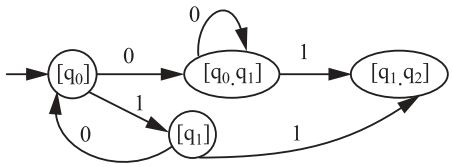
$\delta'(q_1,0) = \{q_0\}$,

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

Now, $\delta'([q_0,q_1], 0) = \delta'(q_0,0) \cup \delta'(q_1,0) = \{q_0,q_1\} \cup \{q_0\} = \{q_0,q_1\}$

and, $\delta'([q_0,q_1], 1) = \delta'(q_0,1) \cup \delta'(q_1,1) = \{q_1\} \cup \{q_1, q_2\} = \{q_1,q_2\}$

Since the new state is $\{q_1,q_2\}$.Then add this new state (i.e. $[q_1,q_2]$) in transition digraph from state $[q_0,q_1]$ with label 1. So the new transition digraph is given below

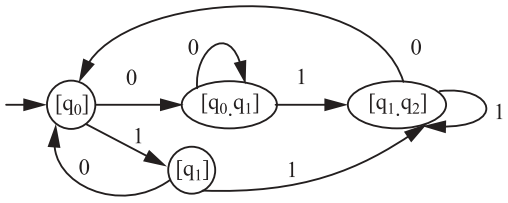


By step 4: The next state from state $[q_1,q_2]$ are $[q_0]$ and $[q_1, q_2]$ on input symbol 0 and 1 respectively as explained below.

$\delta'(q_1,0) \cup \delta'(q_2,0) = [q_0] \cup \phi = [q_0]$

and $\delta'(q_1,1) \cup \delta'(q_2,1) = \{q_1,q_2\} \cup \phi = \{q_1,q_2\}$

Hence the transition digraph is given below:



Since no new state is added in this computation so there is no need to do any more computation on any states

By step 5: Draw all states as a final state if it contain the final state of NFA (i.e. q_2). Hence the final transition diagram is shown in Fig. 2.37.

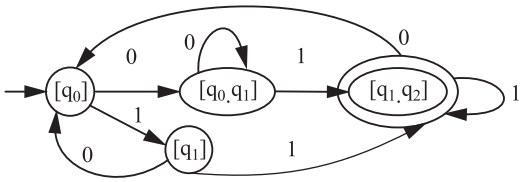


Fig. 2.37

or this can be represented by transition table shown in Table 2.2

Table 2.2

States	Input symbol	
	0	1
→[q_0]	[q_0,q_1]	[q_1]
[q_1]	[q_0]	[q_1, q_2]
[q_0,q_1]	[q_0,q_1]	[q_1, q_2]
([q_1, q_2])	[q_0]	[q_1,q_2]

The DFA equivalent to NFA is $M' = (\{[q_0],[q_1,],[q_0,q_1], [q_1,q_2]\},\{0,1\},\delta', [q_0], \{[q_1,q_2]\})$ where δ' is shown in Fig. 2.37 or Table 2.2.

Example 37: *Find a deterministic finite automata equivalent to $M=(\{q_0,q_1, q_2\},\{0,1\},\delta, q_0,\{q_1\})$ where δ is defined in Fig. 2.38.*

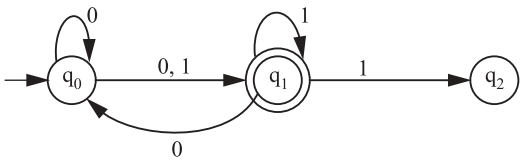


Fig. 2.38

Solution: The given NFA $M = (\{q_0,q_1, q_2\}, \{0,1\},\delta, q_0, \{q_1\}))$ then its equivalent DFA M' is defined by

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

where $Q' = 2^Q$ (Power set Q)

i.e.$\{\phi, [q_0],[q_1],[q_2],[q_0, q_1],[q_0,q_2], [q_1, q_2],[q_0,q_1, q_2]\}$

These are the total number of states of DFA

Σ = Similar to NFA i.e. {0,1}

q_0' = $[q_0]$. This is the initial state of DFA

F' = $\{[q_1], [q_0,q_1],[q_1, q_2],[q_0,q_1, q_2]\}$. These are the final states of DFA

δ' = The transition digraph is similar to the transition digraph of Example 29 except the final digraph. Since the final state is q_1, so make all states as a final state which contains the state q_1. Hence the transition digraph of DFA is shown in Fig. 2.39

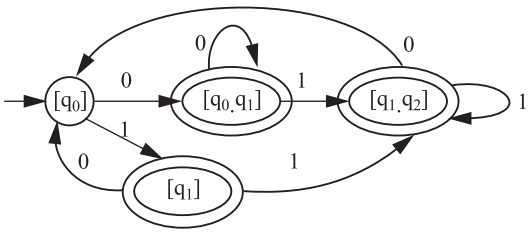


Fig. 2.39

or this can be represented by transition table shown in Table 2.3

Table 2.3

States	Input symbol	
	0	1
→ $[q_0]$	$[q_0,q_1]$	$[q_1]$
$[q_1]$	$[q_0]$	$[q_1, q_2]$
$[q_0,q_1]$	$[q_0,q_1]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_0]$	$[q_1,q_2]$

The DFA equivalent to NFA is $M' = (\{[q_0],[q_1],[q_0,q_1], [q_1,q_2]\},\{0,1\},\delta', [q_0],\{ [q_1],[q_0,q_1], [q_1,q_2]\})$ where δ' is shown in Fig. 2.39 or Table 2.3.

Example 38: *Find a deterministic finite automata equivalent to $M = (\{q_0,q_1, q_2\},\{0,1\},\delta,q_0,\{q_2\})$ where δ is defined in Fig. 2.40.*

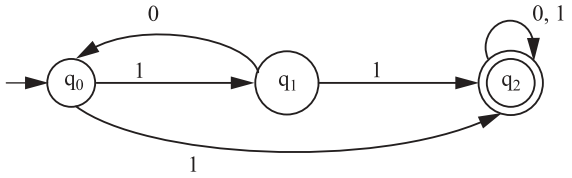


Fig. 2.40

Solution: The given NFA $M = (\{q_0,q_1, q_2\}, \{0,1\},\delta, q_0, \{q_2\}))$, then its equivalent DFA M' is defined by

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

where

$Q' = 2^Q$ (Power set Q)

i.e. $\{\phi, [q_0],[q_1][q_2],[q_0,q_1],[q_0,q_2],[q_1,q_2],[q_0,q_1, q_2]\}$

These are the total number of states of DFA

Σ = Similar to NFA i.e. {0,1}

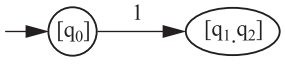
q_0' = $[q_0]$. This is the initial state of DFA

F' = $\{[q_2], [q_0,q_2],[q_1, q_2],[q_0,q_1,q_2]\}$. These are the final states of DFA

δ' = The transition digraph which is determined as follows

By step 1, the initial state is $[q_0]$, so transition diagram for it is as shown here

By step 2, the next states from $[q_0]$ are ϕ and $[q_1,q_2]$ on giving input symbol 0 and 1 respectively. Then add this new state (i.e. $[q_1, q_2]$) in transition digraph from state $[q_0]$ with label 1. So the new transition digraph is given here.

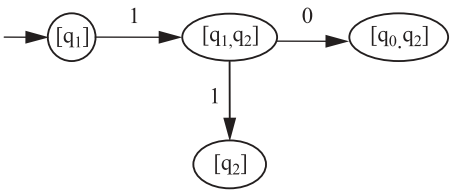


By step 3, the next states from $[q_1, q_2]$ are $[q_0, q_2]$ and $[q_2]$ on giving input symbol 0 and 1 respectively as explained below.

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

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

Since the new states are $[q_0, q_2]$ and $[q_2]$, we add these new states (i.e. $[q_0,q_2]$ and $[q_2]$) in transition digraph from state $[q_1,q_2]$ with labels 0 and 1respectively. The new transition digraph is as shown below:



By step 4, the next state from state $[q_2]$ are $[q_2]$ and $[q_2]$ on giving input symbol 0 and 1 and from $[q_0,q_2]$ are $[q_2]$ and $[q_1,q_2]$ on applying symbol 0 and 1 respectively as explained below.

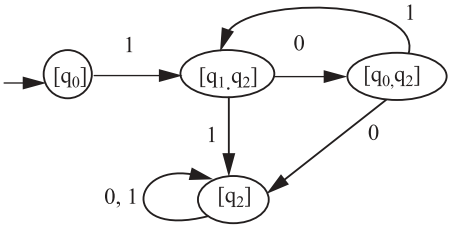
$$\delta'(q_2,0) = \{q_2\},$$

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

$$\delta'(\{q_0,q_2\}, 0) = \delta'(q_0, 0) \cup \delta'(q_2,0) = \phi \cup \{q_2\} = \{q_2\}$$

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

Since no new state is obtained in this computation so no need to do any more computation on any states. Hence the transition diagram is shown below:



By step 5, draw all states as a final state if it contains the final state of NFA (i.e. q_2). Hence the final transition digraph is shown in Fig.2.41.

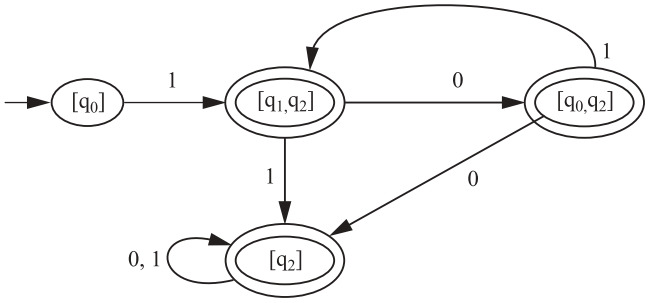


Fig. 2.41

or this can be represented by transition table shown in Table 2.4.

Table 2.4

States	Input symbol	
	0	1
→$[q_0]$	ϕ	$[q_1,q_2]$
(*)$[q_1, q_2]$	$[q_0, q_2]$	$[q_2]$
(*)$[q_2]$	$[q_2]$	$[q_2]$
(*)$[q_0, q_2]$	$[q_2]$	$[q_1, q_2]$

The DFA equivalent to NFA is $M' = (\{[q_0],[q_2],[q_0, q_2],[q_1,q_2]\},\{0,1\},\delta', [q_0], \{[q_2],[q_0,q_2],[q_1,q_2]\})$ where δ' is shown in Fig. 2.41 or Table 2.4.

Example 39: *Find out Deterministic Finite Automata of a machine M = ($\{q_0, q_1, q_2\}$,{a, b}, δ, q_0, $\{q_2\}$) for the table given below* **[U.P.T.U., MCA, 2002-2003]**

Table 2.5

States	a	b
q_0	q_0, q_1	q_0
q_1	q_0	q_1
(*)q_2	–	q_0, q_1

Solution: The given NFA, $M = (\{q_0, q_1, q_2\}, \{a,b\},\delta, q_0, \{q_2\}))$ then its equivalent DFA, M′ is defined by

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

where

$Q' = 2^Q$ (Power set Q)

i.e. $\{\phi, [q_0],[q_1][q_2],[q_0, q_1],[q_0,q_2],[q_1,q_2],[q_0,q_1,q_2]\}$

These are the total number of states of DFA

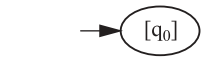
Σ = Similar to NFA i.e. {a, b}

$q_0' = [q_0]$. This is the initial state of DFA

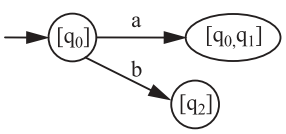
$F' = \{[q_2], [q_0,q_2],[q_1,q_2],[q_0,q_1,q_2]\}$. These are the final states of DFA

δ' = The transition digraph is determind as follows

By step 1, the initial state is $[q_0]$, so transition diagram for it is shown here



By step 2, the next states from $[q_0]$ are $[q_0,q_1]$ and $[q_2]$ on giving input symbol **a** and **b** respectively. Then add these new states (i.e. $[q_0,q_1]$ and $[q_2]$) in transition digraph from state $[q_0]$ with labels **a** and **b** respectively. So the new transition digraph is as given here



By step 3, the next states from $[q_2]$ are ϕ and $[q_0, q_1]$ on given input symbol **a** and **b** respectively, and from $[q_0,q_1]$ are $[q_0,q_1]$ and $[q_1, q_2]$ on giving input symbol **a** and **b** respectively as explained below.

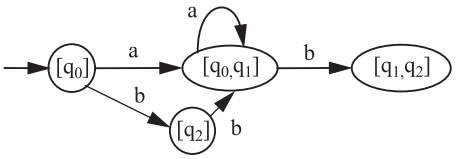
$\delta'(q_2,a) = \{\phi\}$,

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

Now $\delta'([q_0,q_1], a) = \delta'(q_0,a) \cup \delta'(q_1,a) = \{q_0,q_1\} \cup \{q_0\} = \{q_0,q_1\}$

and $\delta'([q_0,q_1], b) = \delta'(q_0,b) \cup \delta'(q_1,b) = \{q_2\} \cup \{q_1\} = \{q_1,q_2\}$

Since the new state is $\{q_1,q_2\}$.Then add this new state (i.e. $[q_1,q_2]$) in transition digraph from state $[q_0,q_1]$ with label b. So the new transition digraph is given below

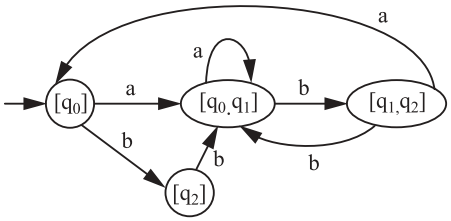


By step 4, the next state from state $[q_1,q_2]$ are $[q_0]$ and $[q_0, q_1]$ on input symbol **a** and **b** respectively as explained below.

$\delta'([q_1, q_2], a) = \delta'(q_1,a) \cup \delta'(q_2,a) = [q_0] \cup \phi = [q_0]$

and $\delta'([q_1, q_2], b) = \delta'(q_1,b) \cup \delta'(q_2,b) = \{q_1\} \cup \{q_0, q_1\} = \{q_0, q_1\}$

Hence the transition digraph is given below



Since no new state is added in this computation so no need to do any more computation on any states.

By step 5 draw all states as final states if these contain the final state of NFA (i.e. q_2). Hence the final transition diagram is shown in Fig. 2.42.

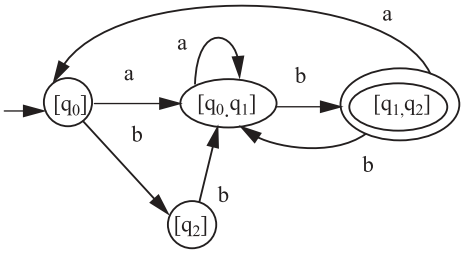


Fig. 2.42

Hence the DFA equivalent to NFA is $M' = (\{[q_0],[q_1],[q_0, q_1],[q_1,q_2]\},\{a, b\},\delta', [q_0], \{[q_1,q_2]\})$ where δ' is shown in Fig. 2.42.

Example 40: *Construct a deterministic Finite Automata equivalent to*

$M = (\{q_0,q_1,q_2,q_3\},\{a, b\}, \delta, q_0, \{q_3\})$ *where* δ *is given as follows*

Table 2.6

States	a	b
q_0	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
(q_3)	–	q_2

[U.P.T.U., MCA, 2001-2002]

Solution: The given NFA, $M = (\{q_0,q_1,q_2,q_3\},\{a, b\}, \delta, q_0, \{q_3\})$ then its equivalent DFA, M' be defined by

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

where

$Q' = 2^Q$ (Power set Q)

i.e. $\{\phi, [q_0],[q_1],[q_2],[q_3],[q_0, q_1],[q_0,q_2],[q_0,q_3],[q_1, q_2],[q_1, q_3],[q_2,q_3],[q_0,q_1,q_2],[q_0,q_1,q_3],[q_0,q_2,q_3][q_1,q_2,q_3], [q_0,q_1,q_2,q_3]\}$

These are the total number of states of DFA

Σ = Similar to NFA i.e. {a, b}

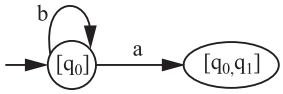
$q_0' = [q_0]$. This is the initial state of DFA

$F' = \{[q_3], [q_0,q_3],[q_1,q_3],[q_2,q_3],[q_0,q_1,q_3],[q_0,q_2,q_3],[q_1,q_2,q_3],[q_0,q_1,q_2,q_3]\}$. These are the final states of DFA

δ' = The transition digraph is determined follows

By step 1, the initial state is $[q_0]$, so transition diagram for it is shown here:

By step 2, the next states from $[q_0]$ are $[q_0,q_1]$ and $[q_0]$ on giving input symbol a and b respectively. Then add new state (i.e. $[q_0,q_1]$) in transition digraph from state $[q_0]$ with label a. So the new transition digraph is given below:

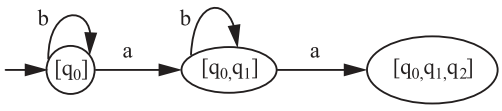


By step 3, the next states from state $[q_0, q_1]$ are $[q_0,q_1,q_2,]$ and $[q_0,q_1]$ on giving input symbol a and b respectively as explained below.

$$\delta'([q_0,q_1], a) = \delta'(q_0,a) \cup \delta'(q_1,a) = \{q_0,q_1\} \cup \{q_2\} = \{ q_0,q_1,q_2\}$$

and $\delta'([q_0,q_1], b) = \delta'(q_0,b) \cup \delta'(q_1,b) = \{q_0\} \cup \{q_1\} = \{q_0, q_1\}$

Since the new state is $\{ q_0,q_1,q_2\}$.Then add this new state (i.e. $[q_0,q_1,q_2]$) in transition digraph from state $[q_0,q_1]$ with label a. So the new transition digraph is given below

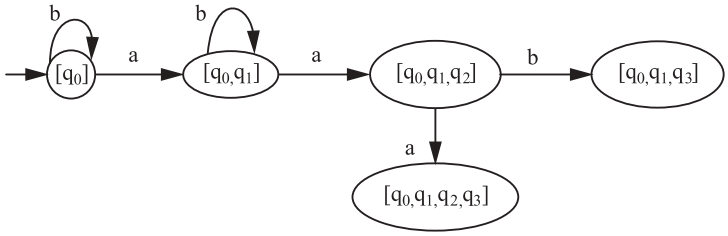


By step 4, the next states from state $[q_0,q_1,q_2]$ are $[q_0,q_1,q_2,q_3]$ and $[q_0, q_1, q_3]$ on input symbol a and b respectively as explained below:

$\delta'([q_0,q_1,q_2], a) = \delta'(q_0,a) \cup \delta'(q_1,a) \cup \delta'(q_2,a) = [q_0,q_1] \cup [q_2] \cup [q_3] = [q_0,q_1,q_2,q_3]$

and $\delta'([q_0,q_1,q_2], b) = \delta'(q_0,b) \cup \delta'(q_1,b) \cup \delta'(q_2,b) = [q_0] \cup [q_1] \cup [q_3] = [q_0,q_1,q_3]$

Hence the transition digraph is given below:



Again applying step 4 the next states from state $[q_0,q_1,q_3]$ are $[q_0,q_1,q_2]$ and $[q_0, q_1, q_2]$ on input symbol a and b respectively and from state $[q_0,q_1,q_2,q_3]$ are $[q_0,q_1,q_2,q_3]$ and $[q_0,q_1,q_2,q_3]$ as explained below.

$\delta'([q_0,q_1,q_3], a) = \delta'(q_0,a) \cup \delta'(q_1,a) \cup \delta'(q_3,a) = [q_0,q_1] \cup [q_2] \cup \phi = [q_0,q_1,q_2]$

$\delta'([q_0,q_1,q_3], b) = \delta'(q_0,b) \cup \delta'(q_1,b) \cup \delta'(q_3,b) = [q_0] \cup [q_1] \cup [q_2] = [q_0,q_1,q_2]$

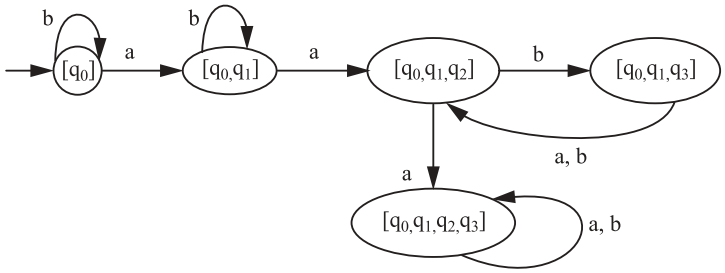
$\delta'([q_0,q_1,q_2,q_3], a) = \delta'(q_0,a) \cup \delta'(q_1,a) \cup \delta'(q_2,a) \cup \delta'(q_3,a) = [q_0,q_1] \cup [q_2] \cup [q_3] \cup \phi$

$= [q_0,q_1,q_2,q_3]$

$\delta'([q_0,q_1,q_2,q_3], b) = \delta'(q_0,b) \cup \delta'(q_1,b) \cup \delta'(q_2,b) \cup \delta'(q_2,b)$

$= [q_0] \cup [q_1] \cup [q_3] \cup [q_2] = [q_0,q_1,q_2,q_3]$

Hence the transition digraph is given below:



Since no new state is added in this computation so no need to do any more computation on any states

By step 5 draw all states as a final state if it contain the final state of NFA (i.e. q_3). Hence the final transition diagram is shown in Fig. 2.43.

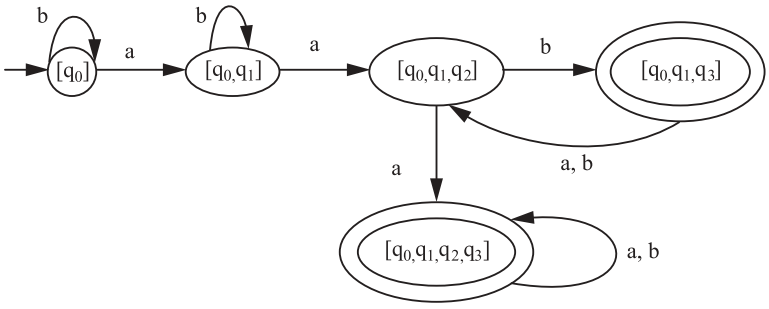


Fig. 2.43

Hence the DFA equivalent to NFA is $M' = (\{[q_0],[q_0, q_1],[q_0,q_1,q_2,],[q_0,q_1,q_3], [q_0,q_1,q_2,q_3]\},\{a, b\},\delta', [q_0], \{[q_0,q_1,q_3], [q_0,q_1,q_2,q_3]\})$ where δ' is shown in Fig. 2.43.

2.5 NFA WITH $\in$-MOVES

If a nondeterministic finite automata (i.e. zero, one or more transitions) is modified to allow transitions with at least one edge without input symbols(i.e. $\in$), then we get a NFA with $\in$-move, because the transition without input symbol is called as $\in$-transition.

Or mathematically a non-deterministic finite automaton (NFA) M with $\in$- moves is defined by

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

Q is a finite non-empty set of states,

Σ is a finite alphabet, the input alphabet,

δ is the transition system which represents a mapping between states with input alphabet or $\in$ to power set of states

i.e $Q \times \{\Sigma \cup \in\} \rightarrow 2^Q$ (2^Q is power set of Q)

q_0 ($q_0 \in Q$) is the initial state and

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

Example 41: *Consider the NFA given by following digraph :*

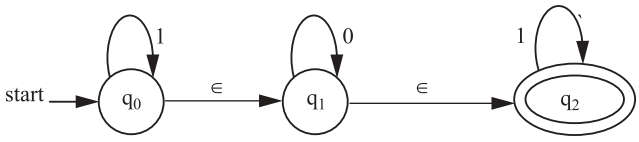


Fig. 2.44

This is a NFA with $\in$-moves, because it is possible to make a transition from state q_0 to q_1 without consuming any of the input symbols, similarly from state q_1 to q_2 also without consuming any of the input symbols, being a non deterministic with $\in$-moves.

Acceptance of String by NFA

A string x in Σ^* will be accepted by NFA with $\in$-moves, if there exists at least one path corresponding to x, which starts with an initial state and ends with one of the final states, but since path may be formed by $\in$-transition as well as non $\in$-transition; to find out whether x is accepted or not by the NFA with $\in$-moves, we require to define *a function $\in$-closure(q)*, where q is the state of automata.

The *function $\in$-closure(q)* is defined as follows :

$\in$-closure(q) = Set of all states of the automata which can be reached from q on a path labeled by $\in$.

For example in the NFA with $\in$-moves in Fig.2.44

$\in$-closure(q_0) = $\{q_0, q_1, q_2\}$

$\in$-closure(q_1) = $\{q_1, q_2\}$

$\in$-closure(q_2) = $\{q_2\}$

The $\in$-closure (q) will never be an empty set, because q is always reachable from itself without consuming any input symbol, i.e. on path labeled by $\in$, hence q will always be there in $\in$-closure(q).

If P be a set of states then $\in$-closure function can be extended to find out $\in$-closure(p) as follows :

$\in$-closure(p) = $\cup_{\text{for every q in p}} \in$-closure(q)

2.6 EQUIVALENCE OF NFA WITH ∈-MOVES TO NFA WITHOUT ∈-MOVES

For every NFA with ∈-moves there exists an equivalent NFA without ∈-moves, accepting the same language. That is we have to find a NFA without ∈-moves equivalent to a NFA with ∈-moves or we have to remove ∈-transition from a given NFA without changing its behavior, but simply removing the ∈-transitions from a given NFA with ∈-moves will change the language accepted by an NFA. Hence for every ∈-transition, to be eliminated you have to add some non ∈-transitions as substitute so as keep the language accepted by the NFA the same. Therefore transformation of NFA with ∈-moves to NFA without ∈-moves involve finding out the non ∈-transitions to be added to the NFA for every ∈-transition to be eliminated.

Use the following steps to convert the NFA with ∈-moves to NFA without ∈-move.

Let us consider that we want to remove ∈-moves that changes state q_1 to state q_2

Step1: Find all the edges starting from state q_2

Step 2: Duplicate all these edges starting from state q_1 without changing the edge labels.

Step 3: If state q_1 is an initial state make state q_2 also as initial state.

Step 4: If state q_2 is a final state make state q_1 also as final state.

Example 42: *We want to convert the following NFA with ∈-moves into NFA without ∈-moves.*

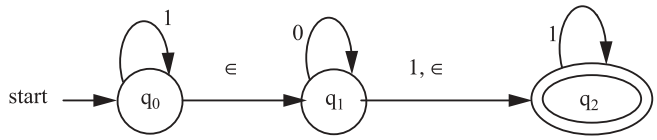


Fig. 2.45

Solution: The ∈-move is given between state q_0 and q_1 and q_1 and q_2 .

So, we first remove ∈-moves from state q_0 to state q_1

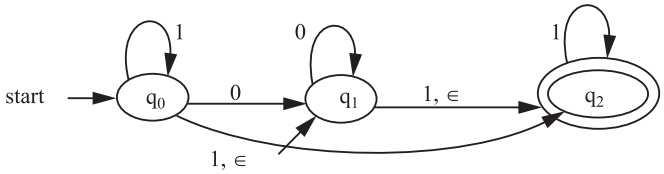
By step 1: The starting edges from state q_1 are 0,1 and ∈

By step 2: Duplicate all these edges (i.e. 0,1,∈) starting from state q_0 without changing the edge labels.

By step 3: The state q_0 is an initial state, make state q_1 also as initial state.

By step 4: The state q_1 is not a final state, so state q_0 shall not be a final state.

The transition digraph looks like as shown in Fig below



Now remove ∈-moves from state q_0 to state q_2

By step 1: The starting edge from state q_2 is 1

By step 2: Duplicate this edge (i.e. 1) starting from state q_0 without changing the edge label.

By step 3: The state q_0 is an initial state, make state q_2 also as initial state.

By step 4: The state q_2 is a final state, so state q_0 also a final state.

The transition digraph looks like as shown in Fig. 2.46 (a) below

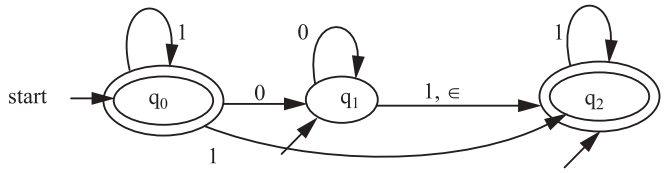


Fig. 2.46 (a)

Now remove ∈-moves from state q_1 to state q_2

By step 1: The starting edge from state q_2 is 1

By step 2: Duplicate this edge (i.e. 1) starting from state q_1 without changing the edge label.

By step 3: The state q_1 is an initial state, make state q_2 also as initial state which is already initial state, so no need to do any updation.

By step 4: The state q_2 is a final state, so state q_1 also a final state.

The transition digraph looks like as shown in Fig. 2.47.

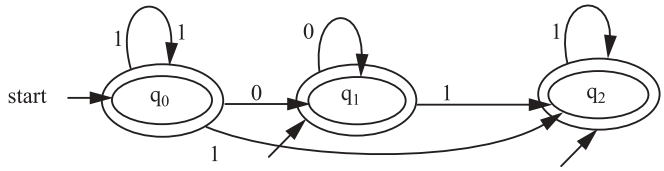


Fig. 2.47

This transition digraph does not contain ∈-moves, so this is the equivalent NFA to the given transition digraph without ∈-moves.

Example 43: *Convert the following NFA to DFA.* **[U.P.T.U., MCA 2002-2003]**

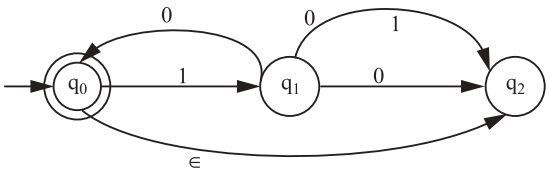


Fig. 2.48

Solution: This transition digraph contains ∈-moves, so the transition digraph without ∈-move is shown as in Fig. 2.49 using above steps.

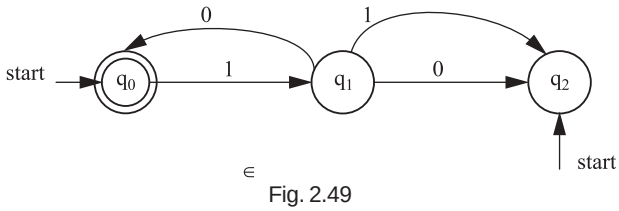


Fig. 2.49

Now using the procedure to convert NFA to DFA, we will get the transition digraph of an equivalent DFA machine which is shown in Fig. 2.50.

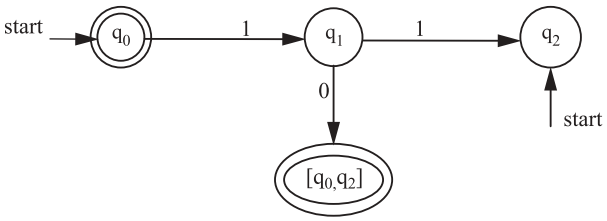


Fig. 2.50

2.7 MINIMIZATION / OPTIMIZATION OF A DFA

Minimization/optimization of a deterministic finite automata refers to detecting those states of a DFA whose presence or absence in a DFA does not affect the language accepted by the automata, hence these states can be eliminated from the automata without affecting the language accepted by the automata. Such states are *Unreachable States*, *Dead States* and *Non Distinguishable* (or equivalent) States.

Unreachable States

These are those states of a DFA, which are not reachable from the initial state of DFA on any possible input sequence.

Dead States

A dead state is that non-final state of a DFA, whose transitions on every input symbol terminates on itself i.e. q is a dead state if q is non final state and $\delta(q, a) = q$ for every a in Σ.

Equivalent or Non-distinguishable States

These are those states of DFA for which there exist no distinguishing string. Hence they can not be distinguished from one another and we can replace all of them by one equivalent state.

We can find equivalent states by the following definitions.

Definition 1: Two states q_1 and q_2 are equivalent if the transition system of both states on same input symbols (i.e. $\delta(q_1, a)$ and $\delta(q_2, a)$) are exactly equal.

Definition 2: Two states q_1 and q_2 are equivalent if the transition system of both states on similar input symbols (i.e. $\delta(q_1, w)$ and $\delta(q_2, w)$) are both final states or both of them are non-final states for all $w \in \Sigma^*$.

Definition 3: Two states q_1 and q_2 are n-equivalent ($n \geq 0$) if the transition system of both states on similar input symbols (i.e. $\delta(q_1, w)$ and $\delta(q_2, w)$) are final states or both of them are non-final states for all string w of length n or less $\in \Sigma^*$.

0-equivalent : 1. Any two final states are 0-equivalent states.

or **2.** Any two non-final states are also 0-equivalent states.

1-equivalent: Two states q_1 and q_2 are 1-equivalent if the transition system of both states on same input symbols (i.e. $\delta(q_1, a)$ and $\delta(q_2, a)$) are exactly equal.

2-equivalent: Two states q_1 and q_2 are 2-equivalent if the transition system of both states on same input symbols (i.e. $\delta(q_1, aa)$ and $\delta(q_2, aa)$ or $\delta(q_1, ab)$ and $\delta(q_2, ab)$ or $\delta(q_1, ba)$ and $\delta(q_2, ba)$ or $\delta(q_1, bb)$ and $\delta(q_2, bb)$) are exactly equal.

2.8 CONSTRUCTION OF MINIMUM AUTOMATON

Use the following steps to construct a minimum automaton

Step 1: Construct π_0 by definition of 0-equivalent i.e.

$$\pi_0 = (Q_1^0, Q_2^0)$$

where Q_1^0 = The set of all final states

Q_2^0 = The set of all non-final states (i.e. $Q - Q_1^0$)

Step 2: Construct π_1 from π_0 by definition of 1-equivalent

Step 3: Construct π_{n+1} from π_n by definition of n-equivalent

Step 4: Construct π_k for k = 1,2 ,3,…… until $\pi_k = \pi_{k+1}$ by definition of 1-equivalent

Step 5: Convert all states of step 4 into equivalence classes. The equivalence classes are obtained by replacing a state q by [q].

Example 44: *Construct the minimum state automaton equivalent to the transition diagram given in Fig.2.51:*

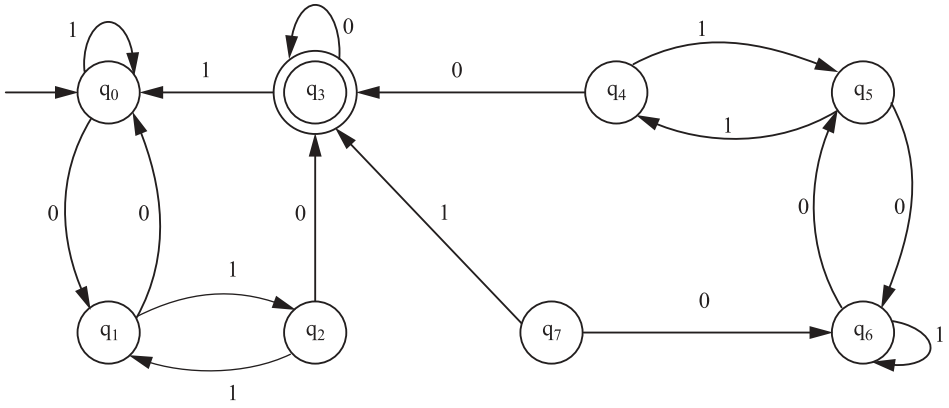


Fig. 2.51

Solution: Transition table is a given below:

Table 2.7

State	Σ	
	0	1
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

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

$Q_1^0 = \{q_3\}$

$Q_2^0 = Q - Q_1^0 = \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}$

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

Formation of π_1 is achieved by partitioning the subsets of π_0.

$Q_1^0 = \{q_3\}$ can not be partitioned further. So $Q_1^1 = \{q_3\}$

$Q_2^0 = \{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}$ may be partitioned further which can be verified as given below:

[A] Consider q_0 with every other state coming after (succeeding) it i.e. $q_1, q_2, q_4, q_5, q_6, q_7$ successively as follows.

q_0 and q_1: Entries corresponding to q_0 and q_1 under 0-col. are $q_1 \in Q_2^0$ and $q_0 \in Q_2^0$ i.e. same equivalence class Q_2^0.

Entries corresponding to q_0 and q_1 under 1-col. are $q_0 \in Q_2^0$ and $q_2 \in Q_2^0$ i.e. same equivalence class. So q_0 is 1-equivalent to q_1.

q_0 and q_2: Entries corresponding to q_0 and q_2 under 0-col. are $q_1 \in Q_2^0$ and $q_3 \in Q_1^0$ i.e. different equivalence classes Q_1^0 and Q_2^0. Now there is no need to see q_0 and q_2 in 1-column.

q_0 and q_4: Entries corresponding to q_0 and q_4 under 0-col are $q_1 \in Q_2^0$ and $q_3 \in Q_1^0$ i.e. different equivalence classes Q_2^0 and Q_1^0. Now there is no need to see q_0. and q_4 in 1-col.

q_0 and q_5: Entries corresponding to q_0 and q_5 under 0-col. are $q_1 \in Q_2^0$ and $q_6 \in Q_2^0$ i.e. same equivalence class Q_2^0.

Entries corresponding to q_0 and q_5 under 1-col. are $q_0 \in Q_2^0$ and $q_4 \in Q_2^0$ i.e. same equivalence class. So q_0 is 1-equivalent to q_5.

q_0 and q_6: Entries corresponding to q_0 and q_6 under 0-col are $q_1 \in Q_2^0$ and $q_5 \in Q_2^0$ i.e. same equivalence class Q_2^0.

Entries corresponding to q_0 and q_6 under 1-col. are $q_0 \in Q_2^0$ and $q_6 \in Q_2^0$ i.e. same equivalence classes Q_2^0. So q_0 is 1-equivalence to q_6.

q_0 and q_7: Entries corresponding to q_0 and q_7 under 0-col. are $q_1 \in Q_2^0$ and $q_6 \in Q_2^0$ i.e. same equivalence class Q_2^0.

Entries corresponding to q_0 and q_7 under 1-col. are $q_0 \in Q_2^0$ and $q_3 \in Q_1^0$ i.e. different equivalence classes. So q_0 is not 1-equivalent to q_7.

Thus q_0 is 1-equivalent to q_1, q_5 and q_6.

So $Q_2^1 = \{q_0, q_1, q_5, q_6\}$ Remaining elements are $\{q_2, q_4, q_7\}$

[B] Consider q_2 with q_4, and q_7 as q_1, q_5 and q_6 have been included in Q_2^1.

q_2 and q_4: Entries corresponding to q_2 and q_4 under 0-col are $q_3 \in Q_1^0$ and $q_3 \in Q_1^0$ i.e. same equivalence class Q_1^0.

Entries corresponding to q_2 and q_4 under 1-col are $q_1 \in Q_2^0$ and $q_5 \in Q_2^0$ i.e. same equivalence class. So q_2 is 1-equivalent to q_4.

q_2 and q_7: Entries corresponding to q_2 and q_7 under 0-col are $q_3 \in Q_1^0$ and $q_6 \in Q_2^0$ different equivalence classes.

No need to consider – 1-col now.

So q_2 is not 1-equivalent to q_7.

Therefore $Q_3^1 = \{q_2, q_4\}$. The only element left over is q_7. So $Q_4^1 = \{q_7\}$

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

Q_1^1 Q_2^1 Q_3^1 Q_4^1

To find π_2:

Formation of π_2 is achieved by partitioning the subsets of π_1.

$Q_1^2 = \{q_3\}$ as q_3 can not be partitioned further. We shall partition other subsets.

Partitioning of $\{q_0, q_1, q_5, q_6\} = Q$

[A] Consider q_0 with q_1, q_5 and q_6

q_0 and q_1: Entries corresponding to q_0 and q_1 under 0-col are $q_1 \in Q_2^1$ and $q_0 \in Q_2^1$ i.e. same class.

Entries corresponding to q_0 and q_1 under 1-col are $q_0 \in Q_2^1$ and $q_2 \in Q_3^1$ different equivalence classes. So q_0 is not 2-equivalent to q_1.

q_0 and q_5: Entries corresponding to q_0 and q_5 under 0-col are $q_1 \in Q_2^1$ and $q_6 \in Q_2^1$ i.e. same class.

Entries corresponding to q_0 and q_5 under 1-col are $q_0 \in Q_2^1$ and $q_4 \in Q_3^1$ different classes. So q_0 is not equivalent to q_5.

q_0 and q_6: Entries corresponding to q_0 and q_6 under 0-col are $q_1 \in Q_2^1$ and $q_5 \in Q_2^1$ same class.

Entries corresponding to q_0 and q_6 under 1-col are $q_0 \in Q_2^1$ and $q_6 \in Q_2^1$ same class. So q_0 is 2-equivalent to q_6.

$$Q_2^2 = \{q_0, q_6\}$$

Left over elements are q_1 and q_5. So we see if they are 2-equivalent or not. We can see that q_1 is 2-equivalent to q_5. So $Q_3^2 = \{q_1, q_5\}$

Again q_2 and q_4 remain out of which q_2 is 2-equivalent with q_4.

So, $Q_4^2 = \{q_1, q_4\}$

$Q_5^2 = \{q_7\}$

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

To find π_3: q_0 is 3-equivalent to q_6 $Q_2^3 = \{q_0, q_6\}$

$Q_1^3 = \{q_3\}$ q_1 is 3-equivalent to q_5 $Q_3^3 = \{q_1, q_5\}$

q_2 is 3-equivalent to q_4 $Q_4^3 = \{q_2, q_4\}$

$Q_5^3 = \{q_7\}$

So π_3 is same as π_2.

So π_2 gives the equivalence classes.

Minimum state automaton is $M' = (Q', \{0, 1\}, \delta', q'_0, F')$

where $Q' = \{\{q_3\}, \{q_0, q_6\}, \{q_1, q_5\}, \{q_2, q_4\}, \{q_7\}\}$

$q'_0 = \{q_0, q_6\}$, $F' = \{q_3\}$ and δ' is given below:

Table 2.8

State	Σ	
	0	1
$[q_0, q_6]$	$[q_1, q_5]$	$[q_0, q_6]$
$[q_1, q_5]$	$[q_0, q_6]$	$[q_2, q_4]$
$[q_2, q_4]$	$[q_3]$	$[q_1, q_5]$
$[q_3]$	$[q_3]$	$[q_0, q_6]$
$[q_7]$	$[q_0, q_6]$	$[q_3]$

Note: In fourth row of 1-column, we can not write $[q_0]$ as this is no more a state now. It is $[q_0, q_6]$. Similarly we cannot write $[q_6]$ in fifth row of 0-column.

This represents a transition function of a transition system. This can also be represented in the form of transition digraph, which is shown in Fig. 2.52.

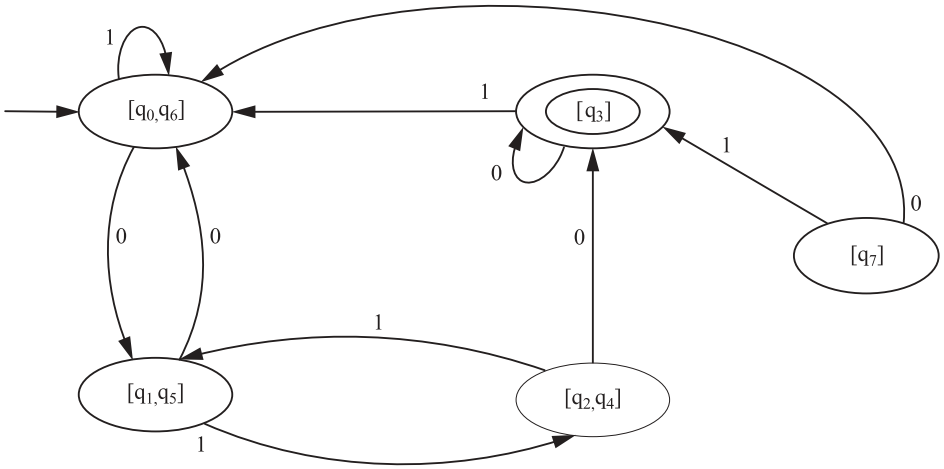


Fig. 2.52

Example 45: *Construct a minimum state automaton equivalent to the finite automaton given in Fig.2.53:*

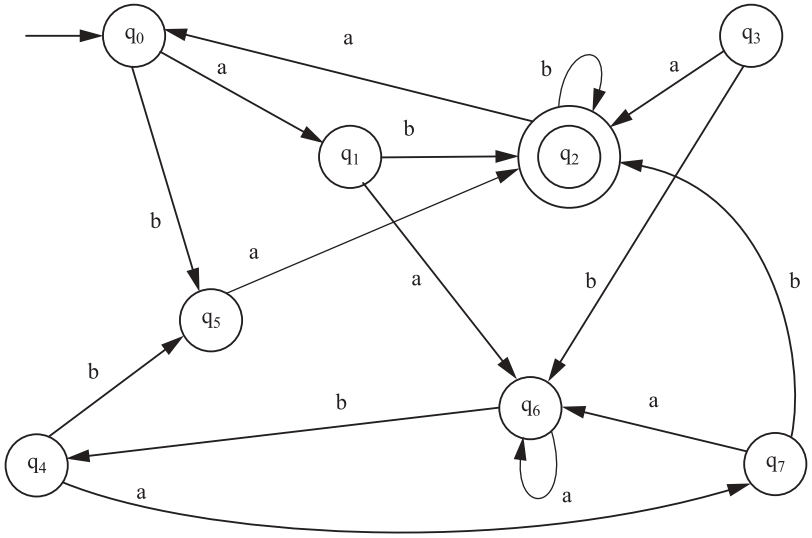


Fig. 2.53

Solution: Transition Table shall be as follows:

Table 2.9

State	Σ	
	a	b
→ 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

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

$Q_1^0 = \{q_2\}$, $Q_2^0 = Q - Q_1^0 = \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}$

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

Determination of π_1, by further partitioning of subsets of π_0.

Set $\{q_2\}$cannot be partitioned further so $Q_1^1 = \{q_2\}$ is a subset in π_1

Partitioning of $Q_2^0 = \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}$ to find π_1.

A – Consider q_0 with every other $q_{i,}$ i = 1, 3, 4, 5, 6, 7 succeeding it separately

(i) **q_0 and q_1:** Entries under a-column corresponding to q_0 and q_1 are q_1 and q_6 both of which belong Q^0_2.

Entries under b-column corresponding to q_0 and q_1 are q_5 and q_2. These both $\notin Q_1^0$ and also $\notin Q_2^0$

$$\begin{cases} \text{one of them } q_5 \in Q_2^0 \\ \text{and the other } q_2 \in Q_1^0 \end{cases}$$

So q_0 and q_1 are not 1-equivalent [Had they both belonged to either Q_1^0 or Q_2^0, then q_0 and q_1, would have been 1-equivalent.

(ii) **q_0 and q_3:** Entries corresponding to q_0 and q_3 under a-col are $\dfrac{q_1 \in Q_2^0 \text{ and } q_2 \in Q_1^0}{\text{Different sets}}$

Entries corresponding to q_0 and q_3 under b-col are $\dfrac{q_5 \in Q_2^0 \text{ and } q_6 \in Q_2^0}{\text{Same set}}$

(iii) **q_0 and q_4:** Entries corresponding to q_0 and q_4 under a-col are $q_1 \in Q_2^0$ and $q_7 \in Q_2^0$

Entries corresponding to q_0 and q_4 under b-col are $q_5 \in Q_2^0$ and $q_5 \in Q_2^0$ } same set

(iv) **q_0 and q_5:** Entries corresponding to q_0 and q_5 under a-col are $\dfrac{q_1 \in Q_2^0 \text{ and } q_2 \in Q_1^0}{\text{Different sets}}$

Entries corresponding to q_0 and q_5 under b-col are $\dfrac{q_5 \in Q_2^0 \text{ and } q_6 \in Q_2^0}{\text{Same sets}}$

(v) **q_0 and q_6:** Entries corresponding to q_0 and q_6 under a-col are $q_1 \in Q_2^0$ and $q_5 \in Q_2^0$

Entries corresponding to q_0 and q_6 under b-col are $q_5 \in Q_2^0$ and $q_4 \in Q_2^0$ } same set

(vi) **q_0 and q_7:** Entries corresponding to q_0 and q_7 under a-col are $q_1 \in \dfrac{Q_2^0 \text{ and } q_6 \in Q_2^0}{\text{same sets}}$

Entries corresponding to q_0 and q_7 under b-col are $\dfrac{q_5 \in Q_2 \text{ and } q_2 \in Q_1}{\text{Different sets}}$

Therefore $Q^1{}_2 = \{q_0, q_4, q_6\}$ is a subset in π_1

Remaining elements are $\{q_1, q_3, q_5, q_7\}$, q_0 is not 1-equivalent to q_3, q_5 and q_7.

B Now consider q_1 with each of the other q_i (i = 3, 5, 7) successding it i.e. q_3, q_5, q_7 as q_4 and q_6 have been taken in Q'_2 as in **A.**

(i) **q_1 and q_3:** Entries corresponding to q_1 and q_3 under a-col are $\dfrac{q_6 \in Q_2{}^0 \text{ and } q_2 \in Q_1{}^0}{\text{Different sets}}$

Entries corresponding to q_1 and q_3 under b-col are $\dfrac{q_2 \in Q_1{}^0 \text{ and } q_6 \in Q_2{}^0}{\text{Different sets}}$

(ii) **q_1 and q_5:** Entries corresponding to q_1 and q_5 under a-col are $q_6 \in Q_2{}^0$ and $q_2 \in Q_1{}^0$
Entries corresponding to q_1 and q_5 under b-col are $q_2 \in Q_1{}^0$ and $q_6 \in Q_2{}^0$ } Different sets

(iii) **q_1 and q_7:** Entries corresponding to q_1 and q_7 under a-col are $q_6 \in Q_2{}^0$ and $q_6 \in Q_2{}^0$
Entries corresponding to q_1 and q_7 under b-col are $q_2 \in Q_1{}^0$ and $q_2 \in Q_1{}^0$ } same set

So q_1 is 1-equivalent to q_7. Therefore $Q'_3 = \{q_1, q_7\}$ is also a subset in π_1. Now only q_3 and q_5 are left to be considered in $Q_2{}^0$.

[C] Consider q_3 with q_5 the only left over value

q_3 and q_5: Entries corresponding to q_3 and q_5 under a-col are $q_2 \in Q_1{}^0$ and $q_2 \in Q_1{}^0$
Entries corresponding to q_3 and q_5 under b-col are $q_6 \in Q_2{}^0$ and $q_6 \in Q_2{}^0$

So q_3 is 1-equivalent to q_5

Therefore, **$Q'_4 = \{q_3, q_5\}$**

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

Now we have to find π_2 by further partitionining subsets of π_1.

$Q_1{}^2 = \{q_2\}$ is in π_2 as it can not be partitioned further

[D] Consider q_0 with q_4 and q_6

(i) **q_0 and q_4**: Entries Corresponding q_0 and q_4 under a-col are q_1 and q_7 lying in same equivalence class in π_1.

Entries Corresponding q_0 and q_4 under b-col are q_5 and q_5 lying in same equivalence class in π_1.

So, q_0 and q_4 are 2-equivalent.

Therefore $\{q_0, q_4\}$ is a subset of π_2 or $Q_2{}^2 = \{q_0, q_4\}$

(ii) **q_0 and q_6:** Entries Corresponding q_0 and q_6 under a-col are q_1 and q_6 lying in different equivalence class in π_1.

Entries Corresponding q_0 and q_6 under b-col are q_5 and q_4 also lying in different equivalence class in π_1.

So q_0 and q_6 are not 2-equivalent.

Therefore $\{q_0, q_4, q_6\}$ in π_1 is partitioned into 2 classes $\{q_0, q_4\}$ and $\{q_6\}$ in π_2.

q_1 and q_7 are 2-equivlent i.e. $Q_3{}^2 = \{q_1, q_7\}$

q_3 and q_5 are 2-equivlent i.e. $Q_4{}^2 = \{q_3, q_5\}$

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

Again q_0 and q_4 are 2-equivalent

q_1 and q_7 are 2-equivalent

q_3 and q_5 are 2-equivalent

Hence $\pi_3 = \{\{q_2\}, \{q_0, q_4\}, \{q_6\}, \{q_1, q_7\}, \{q_3, q_5\}\}$

Then $\pi_2 = \pi_3$

π_2 gives the equivalence classes and the minimum state automaton is

$M' = \{Q', \{a, b\}, \delta', q'_0, F')$

where $Q' = \{[q_2], [q_0, q_4], [q_6], [q_1, q_7], [q_3, q_5]\}$

$q'_0 = [q_0, q_4]$ as q_0 the initial state has been combined will q_4.

$F' = [q_2]$ and δ' is given below.

Table 2.10

State	Σ	
	a	b
$\rightarrow [q_0, q_4]$	$[q_1, q_7]$	$[q_3, q_5]$
$[q_1, q_7]$	$[q_6]$	$[q_2]$
$[q_2]$	$[q_0, q_4]$	$[q_2]$
$[q_3, q_5]$	$[q_2]$	$[q_6]$
$[q_6]$	$[q_6]$	$[q_0, q_4]$

Note: We can not write $[q_5]$ in first row of b-column because it does not remain a state now. Now the state is $[q_3, q_5]$ which is written there. Similarly we cannot write $[q_0]$ in third row of a-column and $[q_4]$ in fifth row of b-column in place of $[q_0, q_4]$ and $[q_0, q_4]$ respectively.

Although corresponding to q_6 under b-col in original table the state is q_4, but q_4 is now not a state. It has been merged with q_0. So the state shall become $\{q_0, q_4\}$

Its transition diagram shown in Fig. 2.54

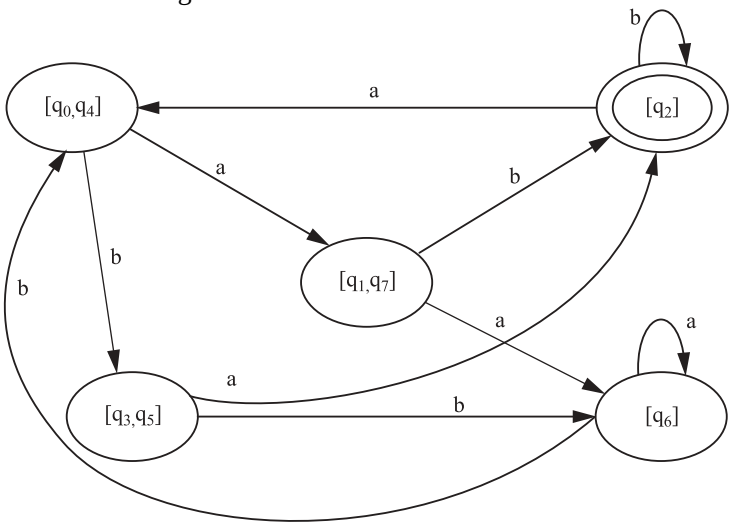


Fig. 2.54

2.9 EQUIVALENCE OF THE FINITE AUTOMATA

Definition

Two finite automata over Σ are equivalent if they accept the same strings over Σ. If they are **not equivalent**, then there must be some string w over Σ, such that one automaton reaches a final state on application of w (i.e. recognizes or accepts w), while the other automaton reaches a non final state on application of w (i.e. does not recognize or accept w).

Comparison Method

To test equivalence of two Deterministic Finite Automata (DFA).

M_1 and M_2 are two FAs over Σ (represented by diagram or otherwise) each consisting of n input symbols say a_1, a_2, …… a_n i.e., $\Sigma = \{a_1, a_2, \ldots\ldots, a_n\}$

Construct a comparison table having (n + 1) columns. Second, third, …… columns shall correspond to first, second, …… input paths.

First row } First entry in the first column shall consist of the pair of initial states of M_1 and M_2 e.g. $q^1{}_0$ and $q^2{}_0$ ($q_0{}^1 \in M_1$ and $q_0{}^2 \in M_2$, both being initial states).

The first entry in second column (a_1 –col) shall consist of a_1-paths i.e. that pair of states which are reached on application of input a_1 to the initial states $q^1{}_0$ and $q_0{}^2$.

The first entry in third column (a_2 –col) shall consist of a_2-paths i.e. that pair of states which are reached on application of input a_2 to the initial states $q^1{}_0$ and $q_0{}^2$.

This procedure is continued to the last column

Then we check all the pairs of states in this first row under each column by the method given below.

Check: If any of the pair under any column is such that one element of the pair denotes the final state of M_1 (or M_2) and the other element of the same pair denotes the non final state of M_2 (or M_1), the construction is terminated and it is concluded that M_1 and M_2 are not equivalent.

If such a condition is not observed in the first row, we proceed to construct the second row as follows:

Construction of second and Succeeding rows. } The pairs of states in first row under second, third, ……..columns (i.e. input columns) that are different from the first entry in the first column are entered as succeeding entries under first column.

Corresponding to each such entry under first column, we find a_1-paths, a_2-paths, ……..as explained earlier.

If any pair of entries (states) in any row under input column is different from all the entries under first column upto **that row**, then this pair is also entered as a succeeding entry under first column.

The check as explained earlier is applied at the end of each row to decide whether to terminate the further construction (M1 and M2 being declared not equivalent) or to continue the construction of the table further.

In the latter case the construction is, continued upto that row in which no pair of states is different from all the pairs of states under first column upto that row and in this case we say that the two automata are equivalent.

Example 46: *Test the equivalence of the following two deterministic finite automata M_1 and M_2 over Σ = {a, b}.*

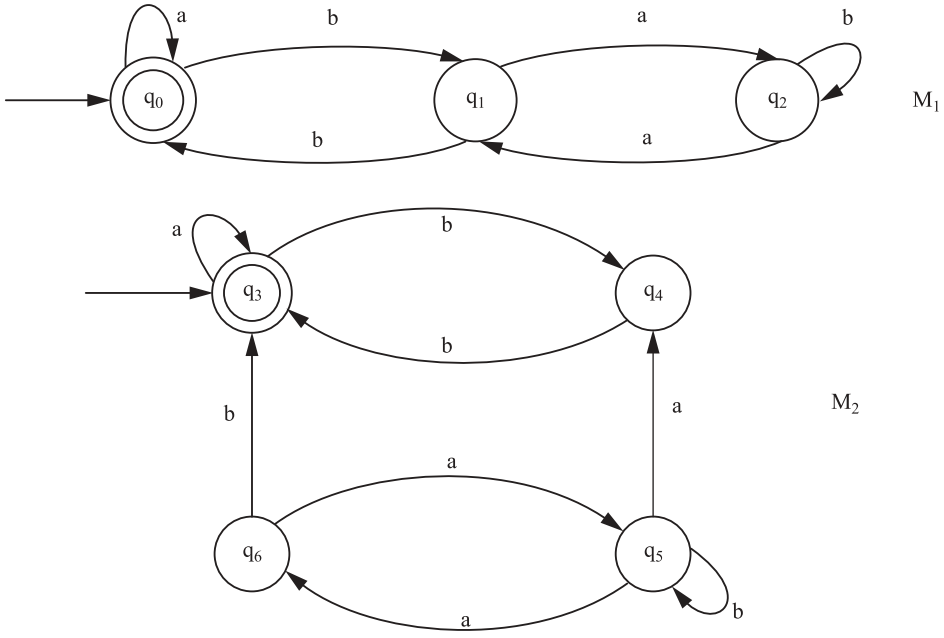


Fig. 2.55

Solution: Comparison table shall consist of 3 columns in all as number of input symbols is 2. Input symbols are a and b. Pair of initial states to be kept as first entry of first row under first column is (q_0, q_3).

(A) Initial state q_0 of M_1 changes to q_0 on receiving input a
Initial state q_3 of M_2 changes to q_3 on receiving input a
} Hence entry in first row corresponding to (q_0, q_3) under a-col is (q_0, q_3)

Initial state q_0 of M_1 changes to q_1 on receiving input b
Initial state q_3 of M_2 changes to q_4 on receiving input b
} and under b-col is (q_1, q_4)

Check: No pair of entry in this row contain one final state of M_1 (or M_2) and one non-final state of M_2 (or M_1).

So proceed further.

Now in first row the entry different from those under col-1 upto row-1 is (q_1, q_4)

So the second entry under col-1 becomes (q_1, q_4) in second row.

To find corresponding entries under a and b cols in second row, we proved as in (A) above.

State q_1 of M_1 changes to q_2 on receiving input a
State q_4 of M_2 changes to q_5 on receiving input a
} Hence entry in row-2 corresponding to (q_1, q_4) under a-col is (q_2, q_5)

State q_1 of M_1 changes to q_0 on receiving input b
State q_4 of M_2 changes to q_3 on receiving input b
} under b-col is (q_0, q_3)

Check : It allows to proceed further.

In second row the entry different from all the entries under col-1 upto second row is (q_2, q_5).

So the third entry under col-1 becomes (q_2, q_5) in third row.

To find corresponding entries in third row under a-col and b-col. (corresponding to entry (q_2, q_5) under 1-col) we again proceed as in (A) above.

State q_2 of M_1 changes to q_1 on receiving input a
State q_5 of M_2 changes to q_6 on receiving input a
} Hence row-3 entry corresponding to (q_2, q_5)under a-col is (q_1, q_6)

State q_2 of M_1 changes to q_2 on receiving input b
State q_5 of M_2 changes to q_5 on receiving input b
} under b-col is (q_2, q_5)

Check : allows us to proceed further.

In the third row, the entry different from all the entries under col-1 upto third row is (q_1, q_6). So the fourth entry under col-1 becomes (q_1,q_6) in fourth row.

To find corresponding entries in fourth row (corresponding to entry (q_1, q_6) under col-1) we again proceed as in (A.)

State q_1 of M_1 changes to q_2 after receiving input a
State q_6 of M_2 changes to q_5 after receiving input a
} Hence row-4 entry corresponding to (q_1, q_6)under a-col is (q_2, q_5)

State q_1 of M_1 changes to q_0 after receiving input b
State q_6 of M_2 changes to q_3 after receiving input b
} under b-col is (q_0, q_3)

Check : It allows to proceed further

In the fourth row there is no entry different form all the entries under col-1 upto 4^{th} row. Therefore we stop here and conclude that M_1 is equivalent to M_2.

Table 2.11 Comparison Table

col – 1 (q, q')	a-col (q_a, q'_a)	b-col (q_b, q'_b)
(q_0, q_3)	(q_0, q_3)	(q_1, q_4)
(q_1, q_4)	(q_2, q_5)	(q_0, q_3)
(q_2, q_5)	(q_1, q_6)	(q_2, q_5)
(q_1, q_6)	(q_2, q_5)	(q_0, q_3)

Note: As we do not have a pair (q, q'), where q is a finial state and q' is a non final state (or vice versa) at every row, we proceed until all the elements in the second and third column are also in the first column. Therefore M_1 and M_2 are equivalent.

2.10 FINITE AUTOMATA WITH OUTPUT

2.10.1 Introduction

A finite automaton with output is defined in two different ways: First, a machine whose outputs are corresponding to their each state. Such type of machine is called *Moore machine*. Second, machines whose outputs are corresponding to their input and state both. Such type of machine is called *Mealy machine*.

If P(t) be the output of these machines then the output function of a Moore machine is defined as follows

$$P(t) = \lambda (q(t))$$

The output function of a Mealy machine is defined as follows

$$P(t) = \lambda (q(t) . w(t))$$

where q(t) is a present state, w(t) is a present input symbol and λ is an output function.
These two machines can be defined mathematically as follows.

2.10.2 Moore Machine

A Moore machine M consists of 6 tuples.

$$M = (Q, \Sigma, \Delta , \delta, \lambda, q_0) \text{ where}$$

1. Q is a finite non-empty set of states.
2. Σ is an non-empty set of input symbols
3. Δ is an output alphabet
4. δ is an transition system and $\delta \in Q \times \Sigma \rightarrow Q$
5. λ is an output function and $\lambda \in Q \rightarrow \Delta$
6. q_0 is a initial state and $q_0 \in Q$.

Graphical representation of a Moore Machine is given below.

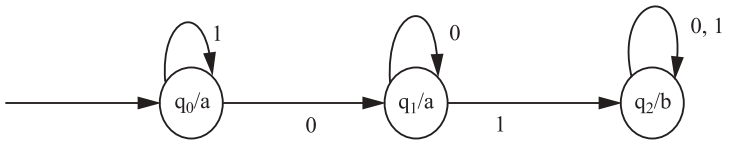


Fig. 2.56

Here the outputs are a and b corresponding to the non-final and final states respectively.
Tabular representation of the same Moore Machine is given below:

Table 2.12

δ / Σ / States(Q)	Input alphabet		Outputs
	0	1	λ
q_0	q_1	q_0	a
q_1	q_1	q_2	a
q_2	q_2	q_2	b

Note: If the length of input string is n then the length of output string is n + 1.

2.10.3 Mealy Machine

A Mealy machine M consists of 6 tuples.

$$M = (Q, \Sigma, \Delta, \delta, \lambda, q_0) \text{ where}$$

1. Q is a finite non-empty set of states.

2. Σ is a non-empty set of input symbols
3. Δ is an output alphabet
4. δ is a transition system and $\lambda \in Q \times \Sigma \rightarrow Q$
5. λ is an output function and $\lambda \in Q \times \Sigma \rightarrow \Delta$
6. q_0 is a initial state and $q_0 \in Q$.

Graphical representation of a Mealy Machine is given below:

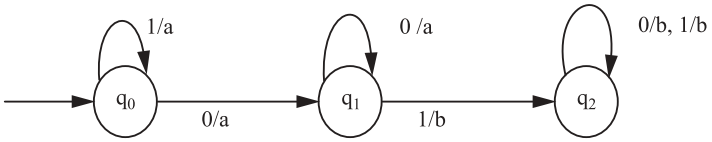


Fig. 2.57

Here the outputs are a and b corresponding to the non-final and final states respectively.

Tabular representation of the same Mealy Machine is as follows:

Table 2.13

δ / Σ / States (Q)	Input alphabet 0	Outputs (λ)	Input alphabet 1	Outputs (λ)
q_0	q_1	a	q_0	a
q_1	q_1	a	q_2	a
q_2	q_2	b	q_2	b

Note : If the length of input string is n then the length of output string is also n.

Example 47: *Design a Mealy machine to produce a 1's complement of any binary string.*

Solutioin: The transition digraph for a Mealy machine is shown below:

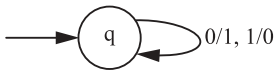


Fig. 2.58

2.10.4 Equivalence of Moore machine and Mealy machine

These two machines are equivalent to each other if they are producing the same output for any given input string. Even though if they are not equal, then we can generate equivalent Moore machine from Mealy machine or vice versa.

2.10.4.1 Conversion from Moore machine to Mealy machine

Theorem: If $M_0 = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ is a Moore machine then there exists an equivalent Mealy machine M_e.

Proof: Let $M_0 = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ be a Moore machine then we can define its equivalent Mealy machine $M_e = (Q', \Sigma', \Delta', \delta', \lambda', q_0')$ where

$Q' = Q$,

$\Sigma' = \Sigma$,

$\Delta' = \Delta$,

$q_0' = q_0$

and δ' and λ', are defined as follows

Let us consider a transition system of a Moore machine in the form given in Fig. (a).

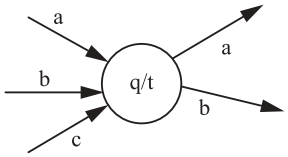


Fig. (a)

This Moore machine contains some incoming and outgoing edges on state q with the output t.

Then the a transition system of a Mealy machine equivalent to this Moore machine is as given in Fig. (b).

It is obtained by assigning output of a state q to all incoming edges on state q, which is equivalent to a Mealy machine.

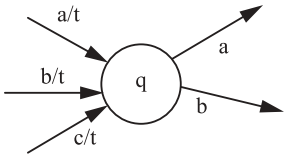


Fig. (b)

Repeat this procedure of all states of Moore machine and the resultant machine is called a Mealy machine.

Example 48: *Convert the following Moore machine into its equivalent Mealy machine.*

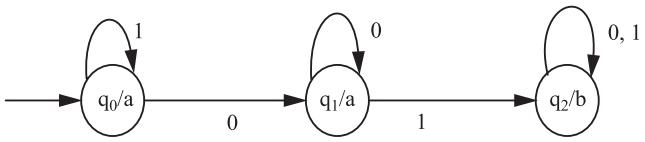


Fig. 2.59

Solution: In the given Moore machine, three states q_0, q_1 and q_2 are given whose corresponding outputs are a, a and b respectively. The incoming edge on state q_0 is 1 so 1 becomes as 1/a. Similarly the incoming edges on state q_1 are 0 from state q_0 and 0 from state q_1, so both these zero, become 0/a and 0/a. Now the incoming edges on state q_2 are 1 from state q_1, 0 from state q_2 and 1 from state q_2, so these 1,0 and 1 becomes 1/b, 0/b and 1/b. All these are shown below:

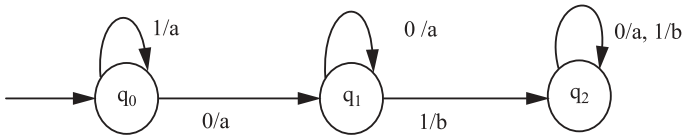


Fig. 2.60

Alternatively a tabular method for the above mentioned problem is as follows.

A tabular representation of the above Moore Machine is shown below:

Table 2.14

δ / Σ / States (Q)	Input alphabet		Outputs
	0	1	(λ)
q_0	q_1	q_0	a
q_1	q_1	q_2	a
q_2	q_2	q_2	b

The outputs of the states q_0, q_1 and q_2 are a, a and b respectively. In a Mealy machine the next state is similar to a Moore machine for all states and inputs. So, assign output of a Mealy machine for all inputs corresponding to their next state. For example, the next state of a state q_0 is q_1 on input symbol 0, so the output of state q_0 on input symbol o is a because a is an output of state q_1 in Moore machine.

A Mealy machine equivalent to given Moore machine is shown below:

Table 2.15

δ / Σ / States (Q)	Input alphabet 0	Outputs (λ)	Input alphabet 1	Outputs (λ)
q_0	q_1	a	q_0	a
q_1	q_1	a	q_2	b
q_2	q_2	b	q_2	b

Example 49: *Convert the following Moore machine into its equivalent Mealy machine.*

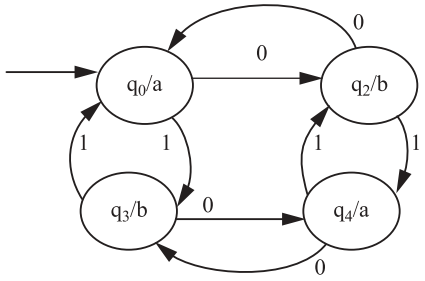


Fig. 2.61

Solution: A Mealy machine equivalent to given Moore machine is shown below

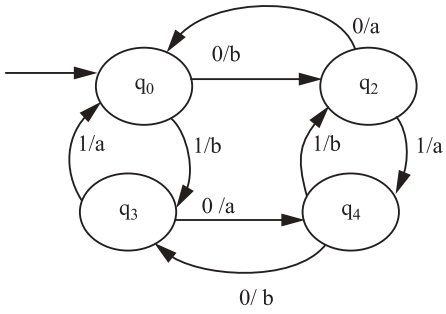


Fig. 2.62

2.10.4.2 Conversion from Mealy machine to Moore machine

Theorem: If $M_e = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ is a Mealy machine then there exists an equivalent Moore machine M_o.

Proof: Let $M_e = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$ be a Mealy machine then we can define its equivalent Moore machine $M_o = (Q', \Sigma', \Delta', \delta', \lambda', q_0)$ where

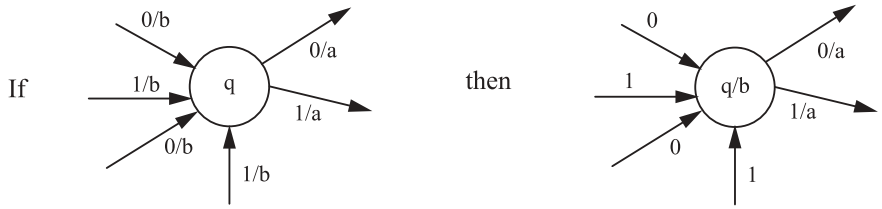
$\Sigma' = \Sigma$,

$\Delta' = \Delta$,

$q_0' = q_0$

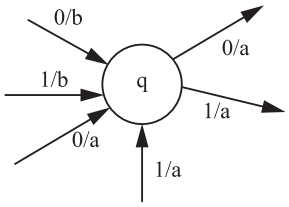
and δ' and λ' are defined as follows

If all the incoming edges of a state have same output then assign that output to a particular state. For example

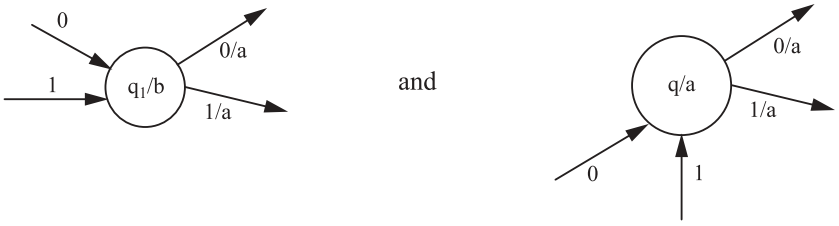


If the incoming edges of a state have different outputs then factorize that state corresponding to the different outputs. For example

A Mealy machine with different outputs on their incoming edges to state q is given below:



Then the transition system of a Moore machine equivalent to this Mealy machine is



and Q' is the total number of states of Moore machine

Repeat this procedure for all incoming edges of all states of Mealy machine. The resultant machine is called a Moore machine.

Example 50: *Convert the following Mealy machine into its equivalent Moore machine*

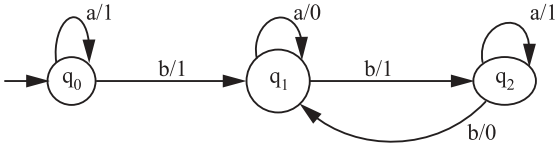


Fig. 2.63

Solution: In the given Mealy machine, three states q_0, q_1 and q_2 are given. On state q_0 there is only one output symbol (= 1) corresponding to the incoming edge Similarly on state q_2 there is only one output

symbol (= 1) corresponding to the two incoming edges. But on state q_1, there are two output symbols (= 0 and 1) corresponding to their two incoming edges. So, we break the state q_1 into two parts corresponding to their outputs according to procedure explained above. Let us consider these two states are q_{10} and q_{11} corresponding to the output 0 and 1 respectively. All these are shown in below:

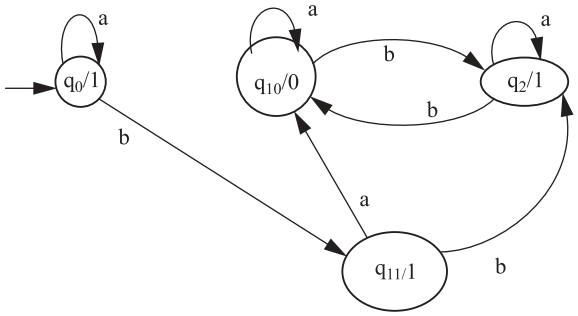


Fig. 2.64

Alternatively a tabular method for the above mentioned problem is described below:
A tabular representation of the above Mealy Machine is described below.

Table 2.16

δ / Σ / States (Q)	Input alphabet a	Outputs (λ)	Input alphabet b	Outputs (λ)
q_0	q_0	1	q_1	1
q_1	q_1	0	q_2	1
q_2	q_2	1	q_1	0

A Moore machine equivalent to given Mealy machine is shown below:

Table 2.17

δ / Σ / States (Q)	Input alphabet a	Input alphabet b	Outputs λ
q_0	q_0	q_{11}	1
q_{10}	q_{10}	q_2	0
q_{11}	q_{10}	q_2	1
q_2	q_2	q_{10}	1

Example 51: *Convert the following Mealy machine into its equivalent Moore machine.*

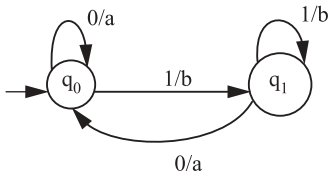


Fig. 2.65

Solution: A Moore machine equivalent to given Mealy machine is shown below

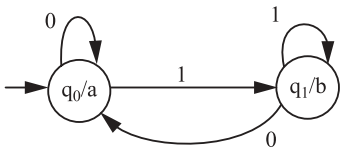


Fig. 2.66

Alternatively a tabular method for the above mentioned problem is also given below:

A tabular representation of the above Mealy Machine is shown below:

Table 2.18

δ / Σ / States (Q)	Input alphabet a	Outputs (λ)	Input alphabet b	Outputs (λ)
q_0	q_0	a	q_1	b
q_1	q_0	0	q_1	1

A Moore machine equivalent to given Mealy machine is shown below:

Table 2.19

δ / Σ / States (Q)	Input alphabet a	Input alphabet b	Outputs λ
q_0	q_0	q_1	a
q_1	q_0	q_1	b

2.11 TWO-WAY FINITE AUTOMATA

Two way finite automata is a finite automata in which read/ write head of a transition system will move in both direction (i.e. left to right or right to left) but in one direction at a time. It is defined in the form of deterministic and non-deterministic both.

2.11.1 Two-Way Deterministic Finite Automaton (2DFA)

A mathematical model of a 2DFA machine M is given by

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

1. Q is a finite non-empty set of states.
2. Σ is a finite non-empty set of input symbols
3. δ is a transition system and $\delta \in Q \times \Sigma \to Q \times \{L, R\}$
4. q_0 is an initial state and $q_0 \in Q$
5. F is a set of accepting states (or final states) and $F \subseteq Q$

2.11.2 Two-Way Non-Deterministic Finite Automaton (2NDFA) [U.P.T.U., B.Tech., 2002-03]

The difference between 2DFA and 2NDFA lies in definition of transition system s.

A mathematical model of a 2NDFA machine M is given by

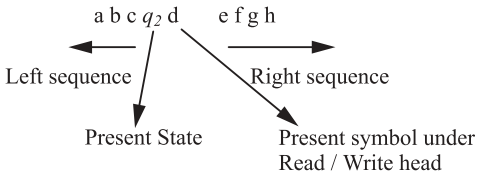
$M = (Q, \Sigma, \delta, q_0, F)$ where

1. Q is a finite non empty set of states.
2. Σ is a finite non empty set of input symbols
3. δ is a transition system and $\delta \in Q \times \Sigma \to 2^Q \times \{L, R\}$
4. q_0 is an initial state and $q_0 \in Q$
5. F is a set of accepting states (or final states) and $F \subseteq Q$

Instantaneous Descriptions (ID) using move relation

An Instantaneous Descriptions(ID) of Two-way Deterministic Finite Automaton is in the form of string xyz, where x is the sub-string of input string , y is the present state of Two-way Deterministic Finite Automaton M and z is also a sub-string of input string and first symbol of z contains the current symbol of Read / Write (R/W) head ,

Example 52: *Let us consider a string abcq$_2$defgh over input alphabet Σ = {a, b, c, d, e, f, g h} and let the present state be q$_2$ and the present symbol under read / write head is d. Then the instantaneous description of this string is graphically represented as follows:*



Moves in a Two-Way Deterministic Finite Automaton

Moves in a Two-way Deterministic Finite Automaton occur under the following two rules:

(i) If the transition function $\delta(q, a_i) = (q_1, R)$ in the input string

$$a_1\ a_2\ a_3 \ldots a_{i-1}\ a_i\ a_{i+1}\ a_{i+2} \ldots\ldots a_n$$

and the present state is q and the present symbol under read / write head is a_i. Then the instantaneous description before processing a_i is

$$a_1\ a_2\ a_3 \ldots a_{i-1}\ q\ a_i\ a_{i+1}\ a_{i+2} \ldots\ldots a_n$$

and after processing a_i is

$$a_1\ a_2\ a_3 \ldots a_{i-1}\ b\ q_1\ a_{i+1}\ a_{i+2} \ldots\ldots a_n$$

(ii) If the transition function $\delta(q, a_i) = (q_1, L)$ in the input string

$$a_1\ a_2\ a_3 \ldots a_{i-1} a_i\ a_{i+1}\ a_{i+2} \ldots\ldots a_n$$

and the present state is q and the present symbol under read / write head is a_i. Then the instantaneous description before processing a_i is

$$a_1\ a_2\ a_{3.....}\ a_{i-1}\ q\ a_i\ a_{i+1}\ a_{i+2}\ \ldots\ldots\ a_n$$

and after processing a_i is

$$a_1\ a_2\ a_{3.....}\ a_{i-2}\ q_1\ a_{i-1}\ b\ a_{i+1}\ a_{i+2}\ \ldots\ldots\ a_n$$

Acceptability of a language (or string) by a Two-Way Deterministic finite Automaton

A language (string) w is accepted by a two-way finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, w) = q$ for some $q \in F$.

Example 53: *Consider the following 2DFA machine M = ({q_0 , q_1 , q_2}, {0, 1}, δ, q_0 {q_1}) where a transition graph is shown below*

Check if the string 010110 is accepted by 2DFA. Also find the sequence of states.

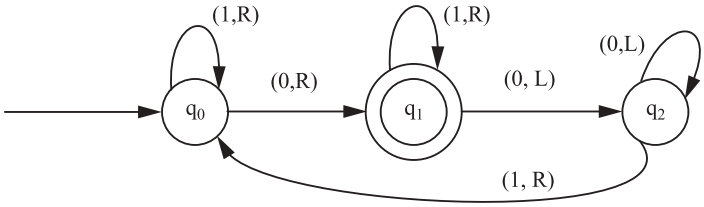


Fig. 2.67

Solution: The Instantaneous Descriptions (ID) of a given input and its moves are represented as follows:

$\boldsymbol{q_0}010110$

$\vdash 0\boldsymbol{q_1}10110$

$\vdash 01\boldsymbol{q_1}0110$

$\vdash 0\boldsymbol{q_2}10110$

$\vdash 01\boldsymbol{q_0}0110$

$\vdash 010\boldsymbol{q_1}110$

$\vdash 0101\ \boldsymbol{q_1}10$

$\vdash 01011\ \boldsymbol{q_1}0$

$\vdash 0101\ \boldsymbol{q_2}10$

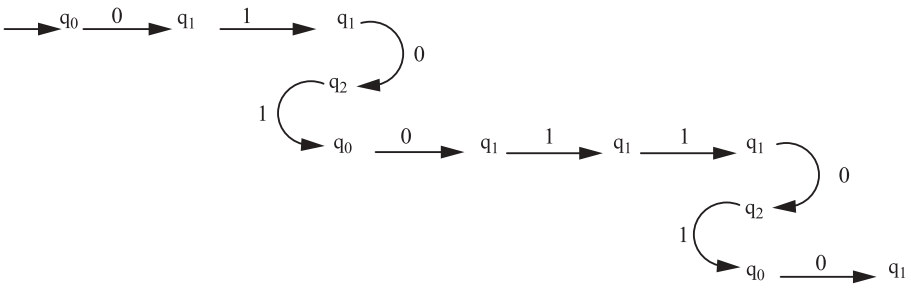
$\vdash 01011\ \boldsymbol{q_0}0$

$\vdash 010110\ \boldsymbol{q_1} \in$

$\vdash \boldsymbol{q_1 \in F}$

Hence the string is accepted by 2DFA machine.

The sequence of states corresponding to the input symbol is given by



Such type of sequence of states is called ***crossing sequence.***

EXERCISE

A. *Choose the most appropriate alternative in each of the following questions.*

1. The maximum number of states in a DFA which is equivalent to a NFA having n states are
(a) 2^n (b) n^2 (c) n (d) none of these

2. Two DFA's M_1 and M_2 are equivalent if:
(a) M_1 and M_2 have the same final states
(b) M_1 and M_2 have the same number of states
(c) M_1 and M_2 accept the same language i.e. L (M_1) = L (M_2)
(d) none of these

3. If in a finite automata $M = (Q, \Sigma, \delta, q_0, F)$, δ maps $Q \times \Sigma$ to 2^Q, then
(a) M is NFA (b) M is DFA
(c) M is NFA with ε moves (d) none of these

4. A DFA has
(a) only one final state
(b) more than one final states
(c) unique path to final state for a set of inputs
(d) all the above

5. The language generated by a DFA is
(a) context free (b) regular (c) context sensitive (d) all the above

6. A DFA can recognize
(a) Regular grammar only (b) CFG
(c) Any grammar (d) none of these

7. For a DFA, $M = \{S, \Sigma, \delta, q_0, F)$, the transition function δ is given by
(a) $\delta : S \times \Sigma \to \Sigma$ (b) $\delta : S \times S \to \Sigma$
(c) $\delta : S \times (\Sigma \cup \{A\}) \to Q$ (d) $\delta : S \times \Sigma \to S^+$

8. If L is a non-empty language such that | w| = n where $w \in L$, then DFA which accepts L shall have
(a) (n + 1) states (b) at least (n + 1) states
(c) atmost (n + 1) states (d) n states

9. For a NFA $M = (S, \Sigma, \delta, q_0, F)$, the transition function δ is given by
(a) $\delta : S \times \Sigma = 2^S$ (b) $\delta : S \times \Sigma \to S$
(c) $\delta : S \times (\Sigma \cup \{\in\}) \to 2^S$ (d) none of these

10. The automate in which output depends on the status of the machine only is
(a) Moore machine (b) Mealy machine
(c) Any automata (d) both (a) and (b)

11. The automaton in which output depends on states and input is called
(a) Moore machine (b) Mealy machine
(c) both (a) and (b) (d) none of these

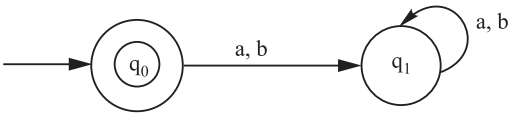
12. Mealy machine is an
(a) an automaton in which output depends on input only
(b) an automaton in which output depends on state only

(c) an automaton in which output depends on input and state both
(d) none of these

13. The main difference between Moore and Mealy machine is
(a) output of Moore machine depends on present state only
(b) output of Moore machine depends on present input only
(c) output of Moore machine depends on present state and input
(d) none of these

14. Can a DFA simulate NFA
(a) yes (b) no (c) sometimes (d) depends on DFA

15. The FA whose transition diagram is given below accepts



(a) all words over $\Sigma = (a, b)$ including $\in$ (b) only $\in$
(c) all words over $\Sigma = (a, b)$ such that a and b alternate
(d) all words over $\Sigma = (a, b)$ except $\in$

B. *Fill in the blanks.*

1. DFA stands for..... .
2. NFA stands for..... .
3. The transition system of a DFA machine is defined by..... .
4. The transition system of a NFA machine is defined by..... .
5. Every NFA machine is also recognized by..... .
6. A dead state is always.....state.
7. Two states are equivalent if..... .
8. A machine whose outputs depends upon only states is called ..... .
9. A machine whose outputs depends upon state and input both is called ..... machine.
10. In Moore machine, if length of input string is n then the length of ouoput string is ..... .
11. In Mealy machine, if length of input string is n then the length of output string is..... .
12. 2DFA stands for..... .
13. The transition system of a 2DFA machine is defined by ..... .
14. A final state of each machine is always..... of states.
15. DFA machine always recognizes ..... language.

C. *State whether the following statements are True (T) or False (F).*

1. A non-deterministic finite automata is more powerful than a deterministic finite automata.
2. Every infinite language is non-regular.
3. A DFA equivalent to a given NDFA must have less number of states than the number of states of the NDFA.
4. For a finite automata $\delta(q, x\,w) = \delta(q, w\,x)$.
5. For a finite automata $\delta(q, x\,w) = \delta\big((q, x), w\big)$.
6. For a finite automata $\delta(q, \lambda) = q$.

7. A Mealy machine has no terminal state.

8. A Moore machine generates no language as such.

9. Context sensitive grammer can be recognized by finite state automata.

10. A language is accepted by finite automata if and only if it is right linear.

11. The powers of DFA and NDFA are the same.

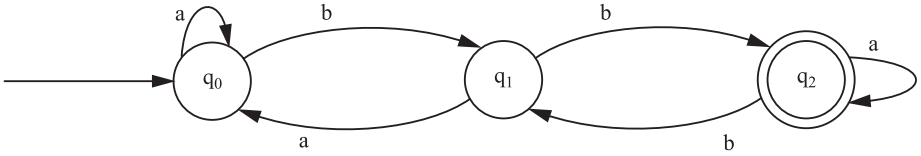
12. The languages accepted by finite automata are the languages denoted by regular expressions.

13. For every DFA there is a regular expression denoting the language.

14. Finite machines can not recognize palindromes.

D. *Write the answer of the following questions.*

1. Which of the following strings are accepted by the DFA machine M whose transition diagram is shown below.

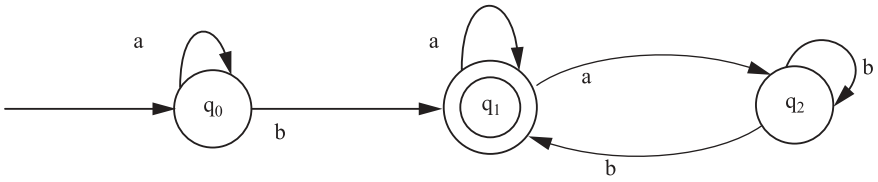


(a) aaab (b) abaab (c) aaaaba (d) ababbaa (e) bbbb
(f) bbbabba (g) aaaabbab (h) aaaaaa (i) bbbabb

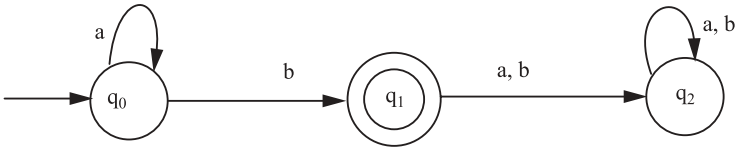
2. For the transiiotn diagram M given in Q.1, find the sequence of states for the following inputs
(a) aaaaabaaa (b) baaabb (c) bbbbbb (d) bbaaabb (e) aabbaabb

3. Design a DFA machine M for the following language
(a) L ={ ab^4wb^3 : w ∈ {a, b}*} (b) L = {awa : w ∈ {a, b}*}
(c) L = {bw_1abw_2 : w_1, w_2 ∈ {a, b}*} (d) L = {w : |w| mod 3 = 0, w ∈ {a, b}*}
(e) L = {w : |w| mod 3 ≠ 0, w ∈ {a, b}*} (f) L = {a^nb^m : n ≥ 1, m ≥ 2}
(g) L = {a^nb : n ≥ 0} ∪ {b^na: n ≥ 1} (h) L = {a^n : n ≥ 0, n ≠ 3}

4. Find all set of strings of DFA machine M whose transition diagram is shown in below:



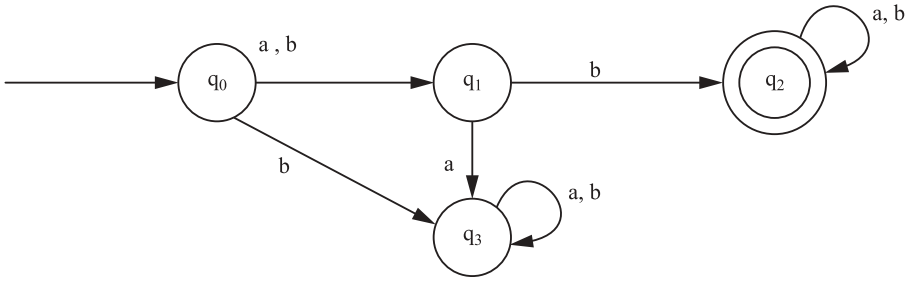
5. Find a language L accepted by the automaton in the figure given below. Also design a DFA that accepts L^2.



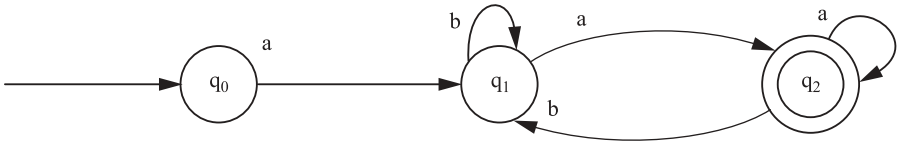
6. Design a DFA machine M over {a, b} that accepts the strings in which the left most symbol differ from right most one.

7. Design a DFA machine over input alphabet ∑ = {a,c,m,r,t } that accepts the strings mat, rat or cat.

8. Design a DFA with exactly six states that accepts the same language as the DFA shown below:



9. Find the language accepted by the automata represented below. If we change state q_2 in a nonfinal state and q_0 and q_1 in final states, then what language is accepted by this machine.



10. Find a DFA that accepts only the strings baa, ab and abb and no other string longer or shorter over input symbol $\Sigma = \{a, b\}$.

11. Find a DFA that accepts only those strings that have more than four letters over input symbol $\Sigma = \{a, b\}$.

12. Find a DFA that accepts only those strings that have fewer than four letters over input symbol $\Sigma = \{a, b\}$.

13. Find a DFA that accepts only those strings with exactly four letters over input symbol $\Sigma = \{a, b\}$.

14. Find a DFA that accepts only those strings that do not end with ba over input symbol $\Sigma = \{a, b\}$.

15. Find a DFA that accepts only those strings that begin and end with double letters over input symbol $\Sigma = \{a, b\}$.

16. Find a DFA that accepts only those strings that have even number of substrings ab over input symbol $\Sigma = \{a, b\}$.

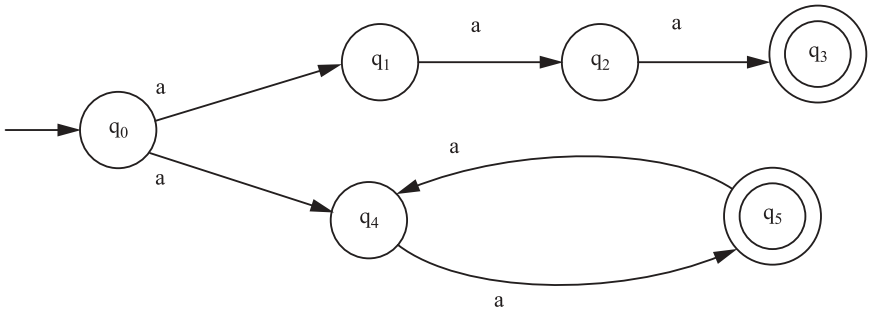
17. Find a DFA that accepts all strings that have even length that is not divisible by 6 over input symbol $\Sigma = \{a, b\}$.

18. Find an NFA that accepts the language $L = (aa^*(a + b))$.

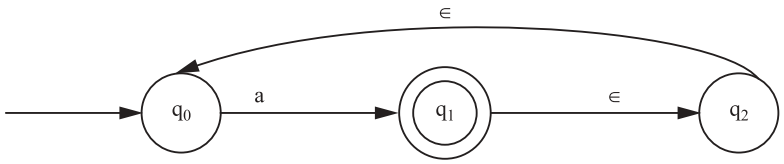
19. Find an NFA which accepts L(r) where, $r = 01^* + 01$.

20. Find an NFA that accepts the language $L = (ab^*aa + bba^*ab)$.

21. Find a DFA that accepts language defined by the NFA represented in the following diagram.



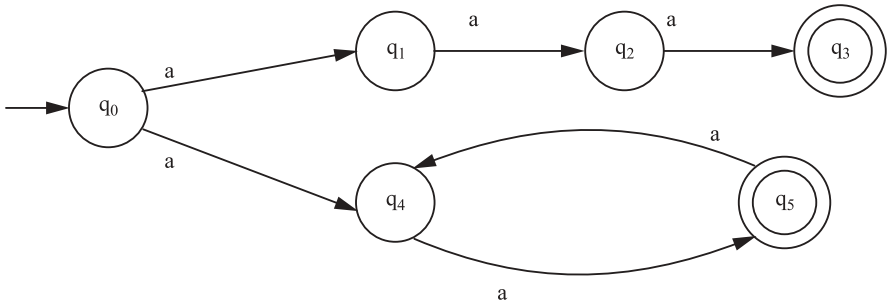
22. What is the complement of the language accepted by the NFA shown below.



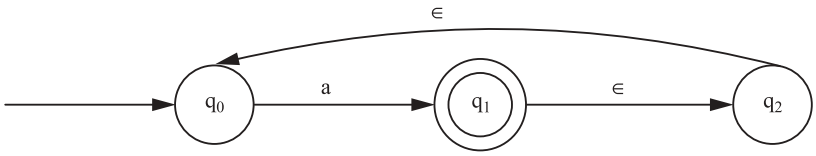
23. Let L be the language accepted by the NFA in figure of Q. 22. Find an NFA that accepts L U$\{a^5\}$.

24. Find an NFA that accepts $\{a\}^*$ and is such that if in its transition graph a single edge is removed (without any other changes), the resulting automaton accepts$\{a\}$.

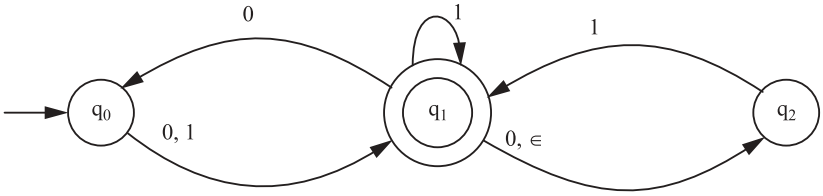
25. Let L be the language accepted by the NFA shown in the figure below. Find an NFA that accepts L$\{a^5\}$.



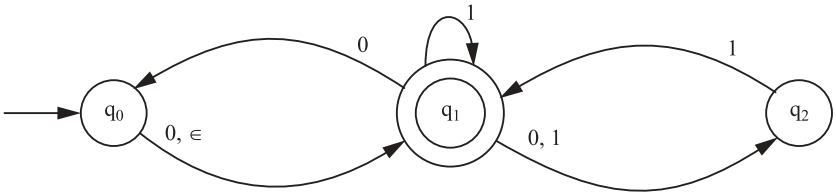
26. Convert the NFA in the figure to a DFA.



27. Convert the NFA into an equivalent DFA.



28. Convert the following NFA into an equivalent DFA.



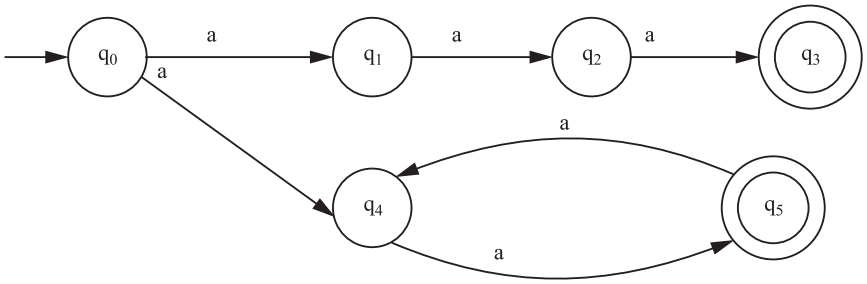
29. Find an NFA without $\in$-transitions and with a single final state that accepts the set $\{a\} \cup \{b^n : n \geq 1\}$.

30. Let L = $\{w \, |w| \in \{a, b\}^*\}$Construct a DFA that accepts strings which have odd number of a's and odd number of b's.

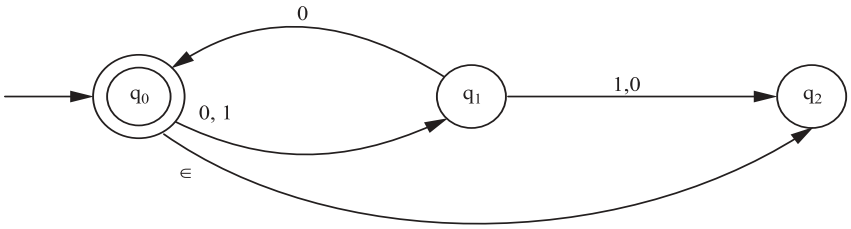
31. Construct a DFA which accepts strings which have sub-string *baab.*

32. Construct DFA that accepts all strings on {0, 1} except those containing the sub-string 001.

33. Find a DFA that accepts the language defined by the NFA in figure given below.



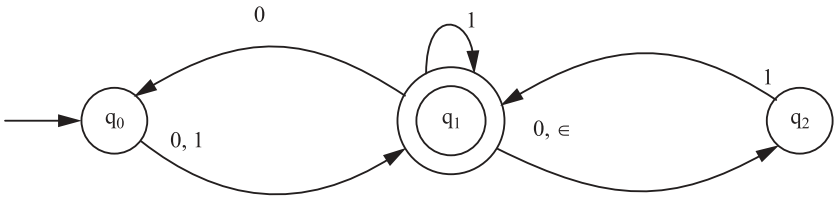
34. For the NFA in figure given below, find its equivalent DFA.



35. Design an NFA with no more than five states for the set
$\{abab^n : n \geq 0\} \cup \{aba^n : \geq 0\}$.

36. Construct an NFA with three states that accepts the language $\{ab, abc\}^*$.
$L = \{a^n : n \geq 0\} \cup \{b^n a : n \geq 1\}$.

37. Which of the strings 00, 01001, 10010, 000, 0000 are accepted by the following NFA?



38. Construct a DFA with reduced states equivalent to the regular expression 10 + (0 + 11)0*1.

39. Construct a DFA with reduced states equivalent to the regular expression (0 + 1)*(00 + 11) $(0 + 1)^*$

40. Construct DFA corresponding to the regular expression

(a) $(ab + a)^*(aa + b)$ (c) $a^* + (ab + a)^*$

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

41. Construct a finite automaton in reduced form accepting all strings over (0, 1) ending in 010 or 0010.

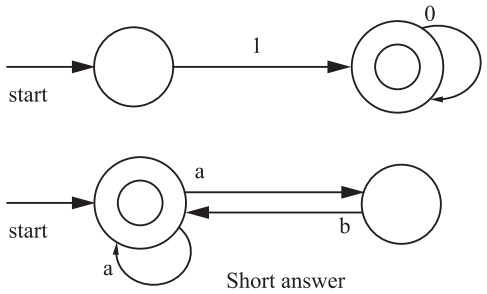
42. Find minimal DFA's for the following languages

(a) $L = \{a^n b^m : n \geq 2, m \geq 1\}$

(b) $L = \{a^n b : n \geq 0\} \cup \{b^n a : n \geq 1\}$

(c) $L = \{a^n : n \geq 0, n \neq 3\}$

43. Find the languages accepted by the finite automate with the following transition diagram:



Short answer

Hint: (a) All the strings of binary digits whose integer value is an integral power of 2.

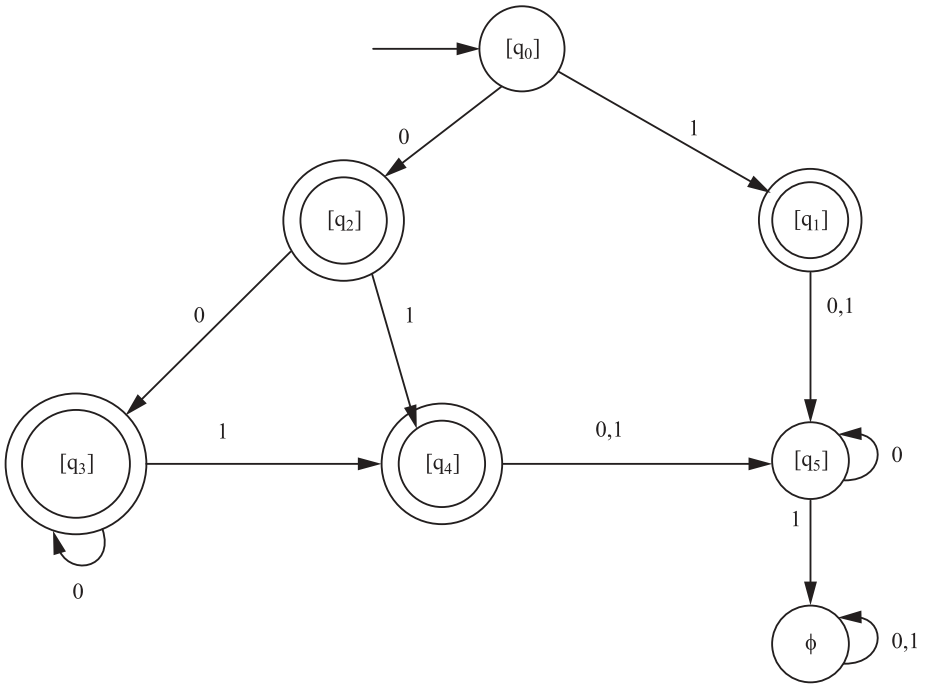
(b) All the strings of a's and b's having every occurrence of b preceded by at least one a.

44. Prove or disprove the following connective. If $M = \{Q, \Sigma, \delta, q_0, F\}$ is a minimal DFA for a regular language L, then $M = (Q, \Sigma, \delta, q_0, Q\ F)$ is a minimal DFA for Σ.

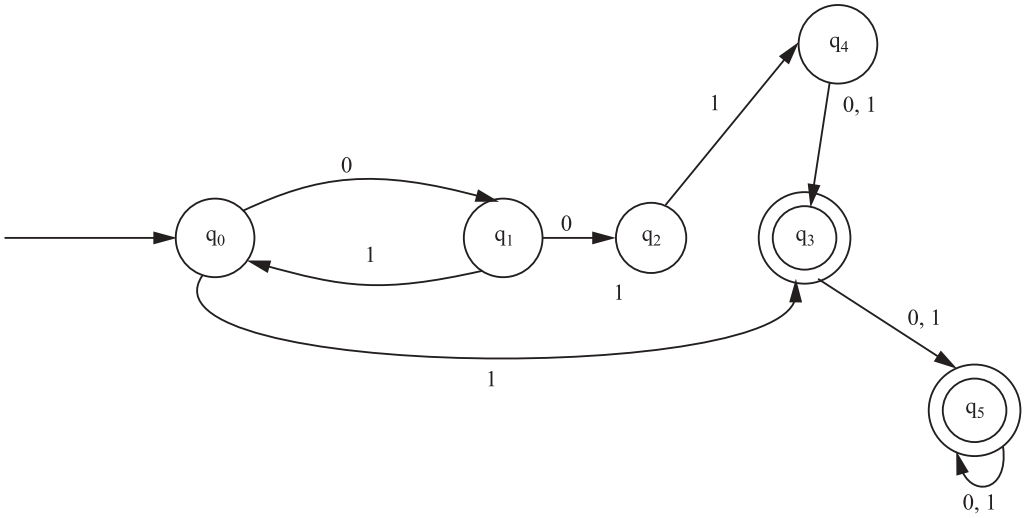
45. Write a computer program that produces a minimal DFA for any given DFA.

46. Show that indistinguishability is an equivalence relation but that distinguishing ability is not.

47. Minimize the number of states in the DFA figure:



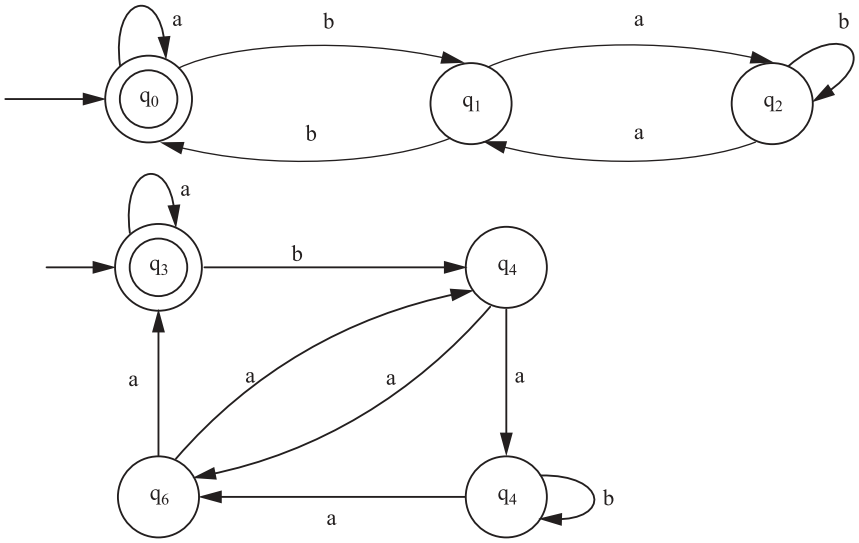
48. Minimize the states in the DFA depicted in the following diagram:



49. Show that if L is a non empty language such that any w in L has length at least n, then any DFA accepting L must have at least n + 1 states.

50. Prove the following: if the states q_a and q_b are indistinguishable, and if q_a and q_c are distinguishable, then q_b and q_c must be distinguishable.

51. Verify if the following two automata are equivalent or not.



[**Hint:** Proceeding as in the previous example the comparison table comes out to be as follows:

Comparison Table

col – 1 (q, q')	a-col (q_a, q'_a)	b-col (q_b, q'_b)
(q_0, q_3)	(q_0, q_3)	(q_1, q_4)
(q_1, q_4)	(q_2, q_6)	(q_0, q_5)

Now in row-2 we have a pair (q_0, q_5) such that q_0 is final state of M_1 and q_5 is non-final state of M_2. We do not proceed and terminate the construction with the conclusion that M_1 and M_2 are not equivalent.

52. Construct a minimum automaton equivalent to a given automaton M whose transition table is given below:

State	Input symbol Σ	
	0	1
$\rightarrow q_0$	q_0	q_3
q_1	q_2	q_5
(q_2)	q_3	q_4
q_3	q_0	q_5
q_4	q_0	q_6
q_5	q_1	q_4
q_6	q_1	q_3

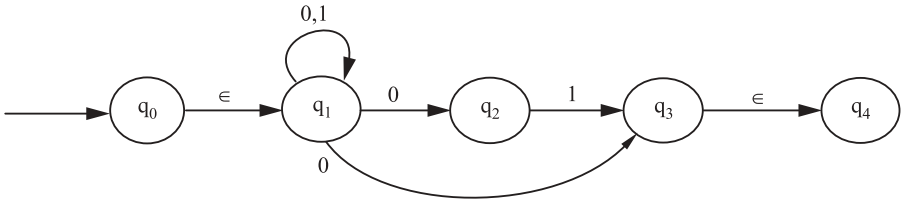
53. Draw Non-deterministic finite automata with specified number of states to accept the following languages:

(i) All strings that contain the sub string 0 | 01 (5 states). **[U.P.T.U., B.Tech., 2001-02]**

Hint. A non-deterministic finite automata M for all strings that contain the sub string 0|01 with 5 states is defined as follows

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

and transition system δ is shown below



Since the transition system δ contains 5 states, so

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

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

$q_0 = \{ q_0\}$ is a initial states.

$F = \{q_4\}$ is a final states.

Hence the NDFA machines M is

$M = (\{q_0, q_1, q_2 ,q_3,q_4\},\{0,1\}, \delta,\{q_0,\}, \{q_4 \})$

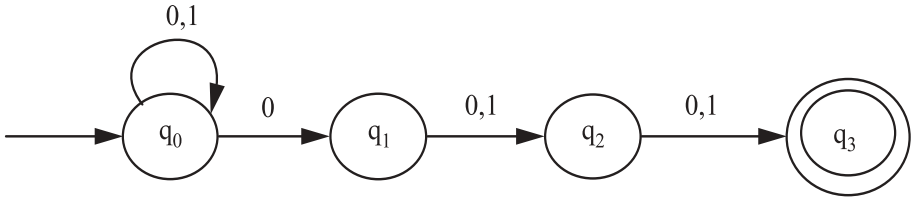
54. All strings that the third symbol from the right end is a "0" (4 states).

[U.P.T.U., B.Tech., 2001-02]

Hint. The non-deterministic finite automata M for all string such that the third symbol from the right end is a "0" over input alphabets {0,1} is define as follows

$M = \{Q, \Sigma, \delta, q_0, F\}$

and transition system δ is shown below



Since the transition system δ contains four states, so

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

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

$q_0 = \{ q_0\}$

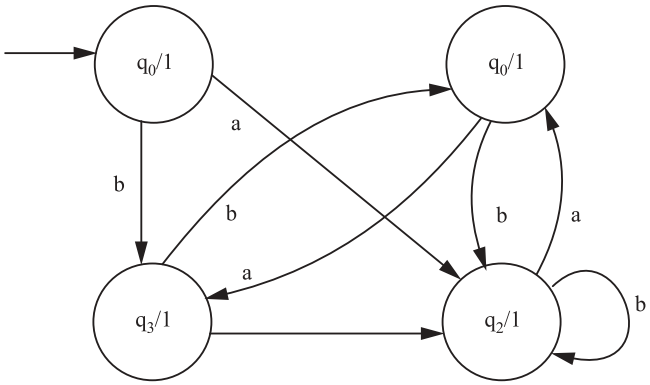
$F = \{q_3\}$

Hence the NDFA machines is

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

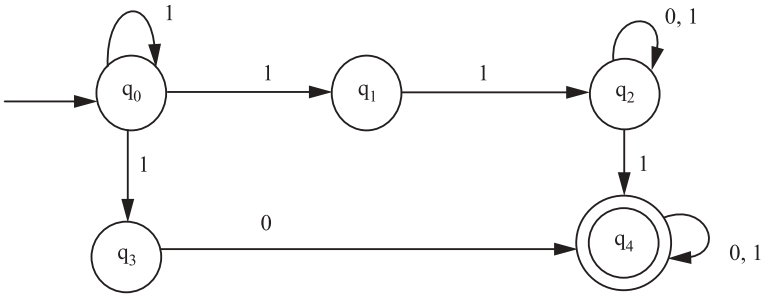
55. What is Moore Machine ? Change the given Moore machine into Mealy machine:

[U.P.T.U., B.Tech., 2002-03]



Hint. See article 2.10.2 on page 2.41 and conversion is similar to example 49 on page 2.44

56. Construct a DFA equivalent to the following NDFA: **[U.P.T.U., B.Tech., 2002-03]**



Hint. Similar to example 36 on page 2.18

57. Draw DFA for following over the set

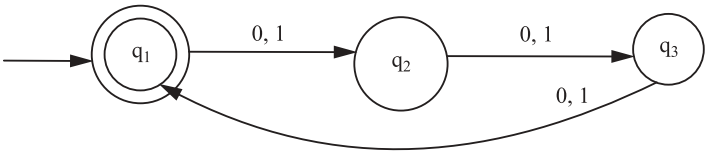
$\Sigma = \{0, 1\}$

(i) $L = \{w \mid\mid w \mid \bmod 3 = 0\}$

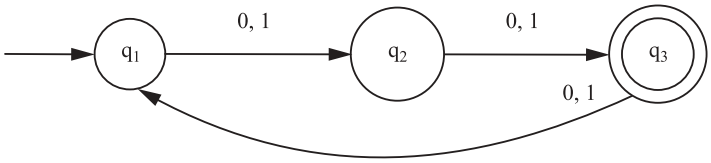
(ii) $L = \{w \mid\mid w \mid \bmod 3 > 1\}$

Note | w | represents the length of the string w. **[U.P.T.U., B.Tech., 2003-04]**

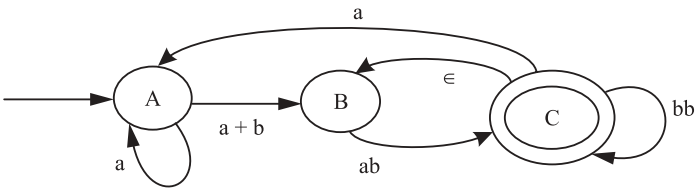
Hint. (i)



(ii)



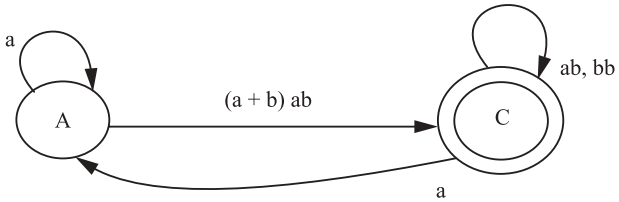
58. Consider the following generalized transition diagram: **[U.P.T.U., B.Tech., 2003-04]**



(i) Find an equivalent generalized transition graph with two states.

(ii) What is the language accepted by this machine?

Hint. (i) The transition diagram of two states is

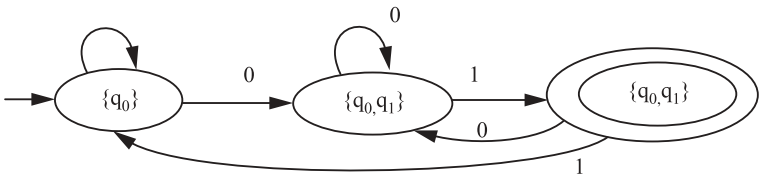


59. Convert the following NFA to equivalent DFA. **[U.P.T.U., B.Tech., 2004-05]**

	0	1
→ P	{q,}	{q,}
*q	{r}	{q,r}
R	{s}	{p}
*s	ϕ	{p}

Hint. Similar to example 39 on page 2.23

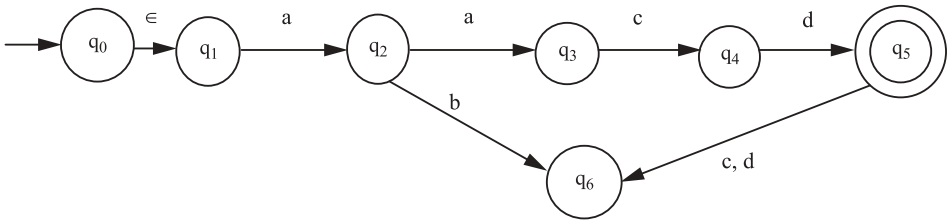
60. Construct DFA from the graph given below. **[U.P.T.U., B.Tech., 2004-05]**



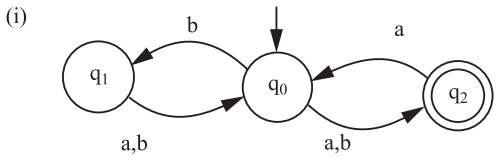
Hint. Similar to example 36 on page 2.18

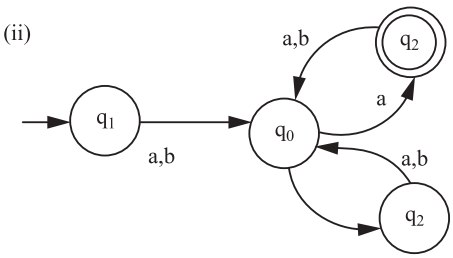
61. Design NFA to recognize the following sets of string abc, abd and aacd. Assume the alphabet is {a,b,c,d}. **[U.P.T.U., B.Tech., 2004-05]**

Hint. The heart part (i.e. transition system) of NFA machine is shown in below



62. Describe an English language accepted by following FAS. **[U.P.T.U., B.Tech., 2004-05]**





Hint. (i) By Ardens Theorem

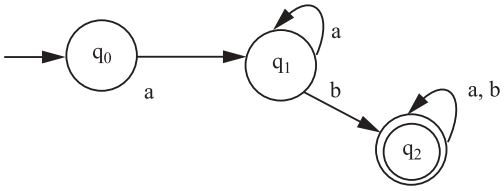
$q_2 = (b(a+b) + a(a+b))^*a$

So it generate "Set of all odd length string ending with a"

(ii) By Ardens Theorem $q_2 = (a+b)(b(a+b) + a(a+b))^*a$

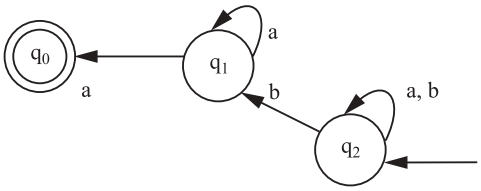
So it generate "Set of all even length string ending with a"

63. Consider the following automata (DFA) M. **[U.P.T.U., B.Tech., 2004-05]**



Obtain a DFA which accepts the complement of the language accepted by M.

Hint. The complement of a graph is shown below:



64. Differentiate between Moore machine and Mealy machine by taking a suitable example.

[U.P.T.U., B.Tech., 2004-05]

Hint. See article 2.10.2 and 2.10.3 on page no. 2.41.

65. Convert to the following NFA. **[U.P.T.U., B.Tech., 2004-05]**

	0	1
→ P	{q,q}	{p}
q	{r}	{r}
r	{s}	ϕ
s	{s}	{s}

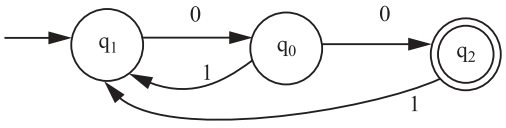
Hint. Similar to example 39 on page no. 2.23

66. Give a DFA's accepting the following languages over the alphabet { 0,1}.

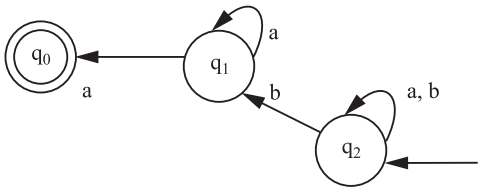
(i) The set of all string ending in 00.

(ii) The set of strings with 011 as a substring. **[U.P.T.U., B.Tech., 2004-05]**

Hint. (i) A DFA machine is shown below:



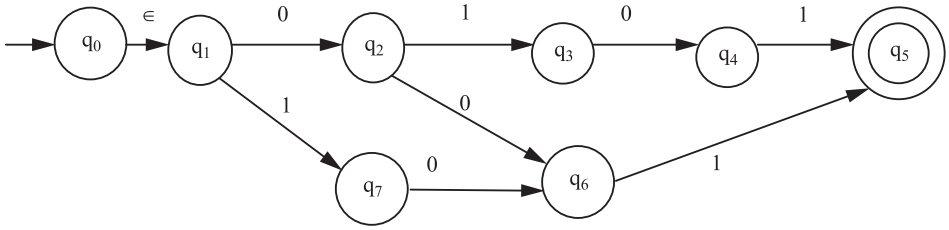
(ii) A DFA machine is shown below



67. Design a NFA to recognize the following set of strings. 0101, 101 and 011. Assume the alphabet is {0, 1}. Hence obtain the equivalent deterministic finite automata DFA.

[U.P.T.U., B.Tech., 2004-05]

Hint. The heart part (i.e. transition system) of NFA machine is shown in below:



Convert it into DFA, Similar to example 43 on page 2.2.29

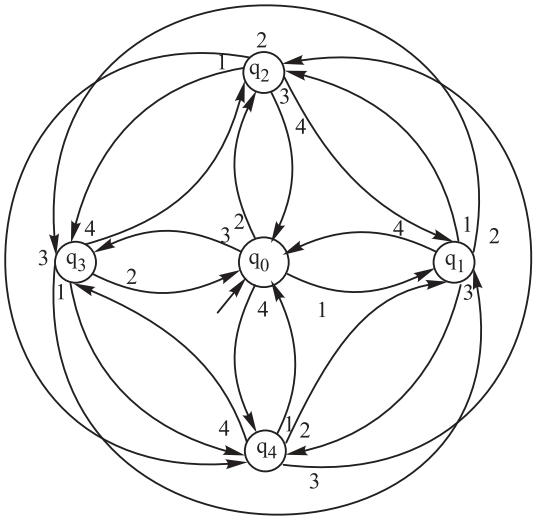
68. Define nondeterministic finite automata. How does it differ from deterministic finite automata?

[U.P.T.U., B.Tech., 2007-08]

69. (i) Design a DFA which accept that set of strings over alphabet $\Sigma = \{1, 2, 3, 4\}$ such that string when interpreted as decimal numbers, sum of their digits are divisible by 5.

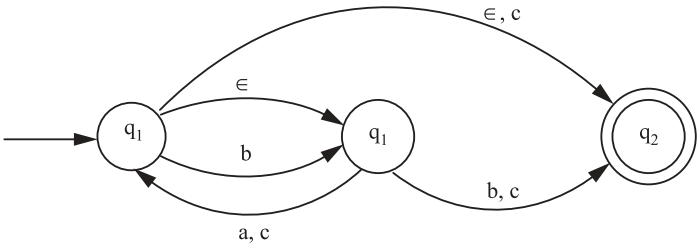
[U.P.T.U., B.Tech., 2007-08]

Hint:



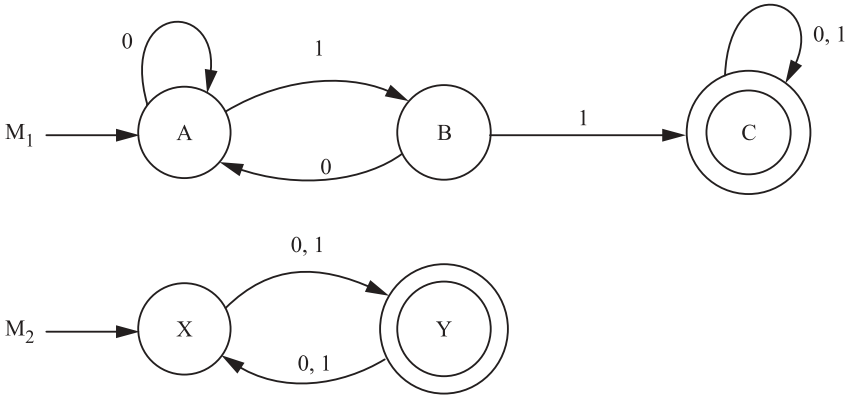
70. Convert following NFA to equivalent DFA and hence minimize the number of states in the DFA.

[U.P.T.U., B.Tech., 2007-08]



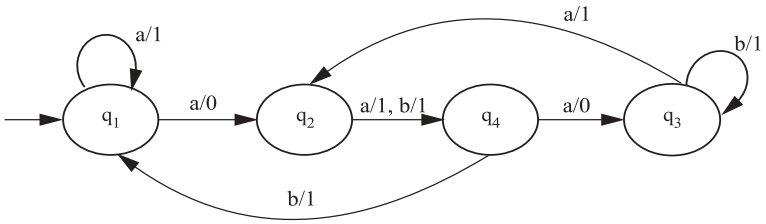
71. (i) Let M_1 and M_2 be the FA recognizing the languages L_1 and L_2 respectively.

[U.P.T.U., B.Tech., 2007-08]



Draw finite automata which recognize the language $L_2 - L_1$.

(ii) Transform the following Mealy machine into equivalent Moore machine.



(A) ANSWERS

1. (a)	**2.** (c)	**3.** (a)	**4.** (c)	**5.** (b)	**6.** (a)
7. (a)	**8.** (b)	**9.** (c)	**10.** (a)	**11.** (b)	**12.** (c)
13. (a)	**14.** (a)	**15.** (b)			

(B) ANSWERS

1. deterministic finite automaton
2. non-deterministic finite automaton
3. $Q \times \Sigma \to Q$
4. $Q \times \Sigma \to 2^Q$
5. DFA machine
6. non-final
7. $\delta(q_1, w) \equiv \delta(q_2, w)$
8. moore machine
9. mealy machine
10. n + 1

11. n
12. two-way deterministic finite automaton
13. $Q \times \Sigma \rightarrow Q \times (L, R)$
14. subset
15. regular

(C) ANSWERS

1. False	**2.** False	**3.** False	**4.** False	**5.** True	**6.** True
7. True	**8.** False	**9.** False	**10.** True	**11.** True	**12.** True
13. True	**14.** True				