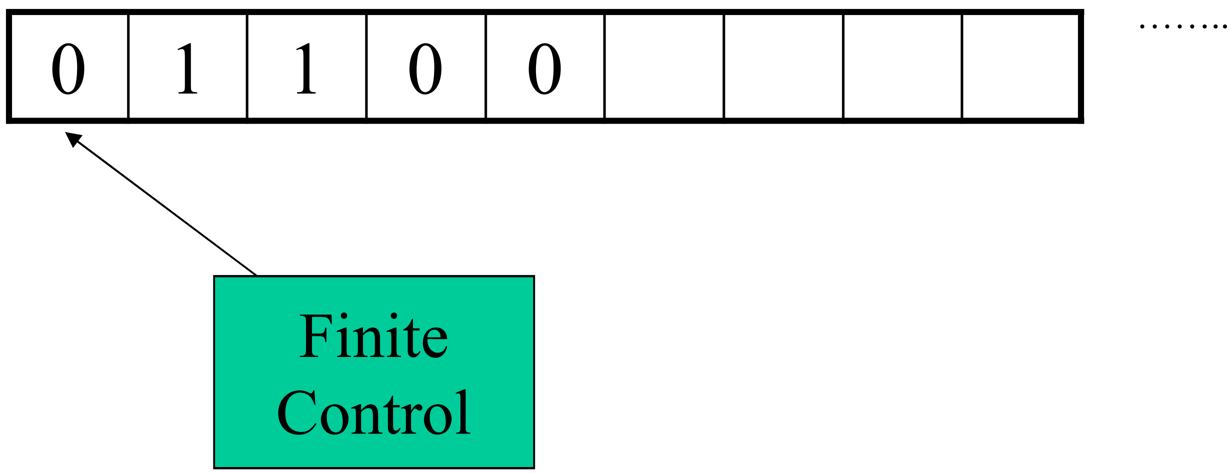
Deterministic Finite State Automata (DFA)



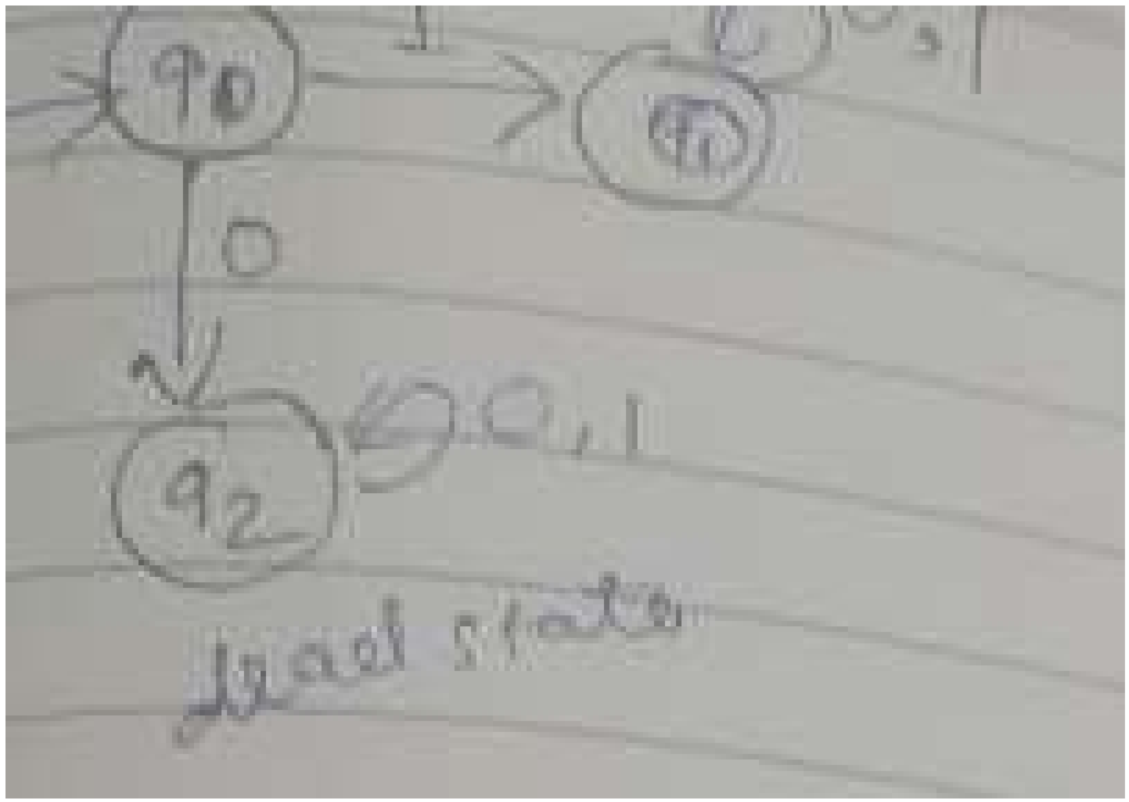
- One-way, infinite tape, broken into cells
- One-way, read-only tape head.
- Finite control, i.e.,
  - finite number of states, and
  - transition rules between them, i.e.,
  - a program, containing the position of the read head, current symbol being scanned, and the current "state."
- A string is placed on the tape, read head is positioned at the left end, and the DFA will read the string one symbol at a time until all symbols have been read. The DFA will then either *accept* or *reject* the string.

- {Q, sigma∑, del, q0, f}

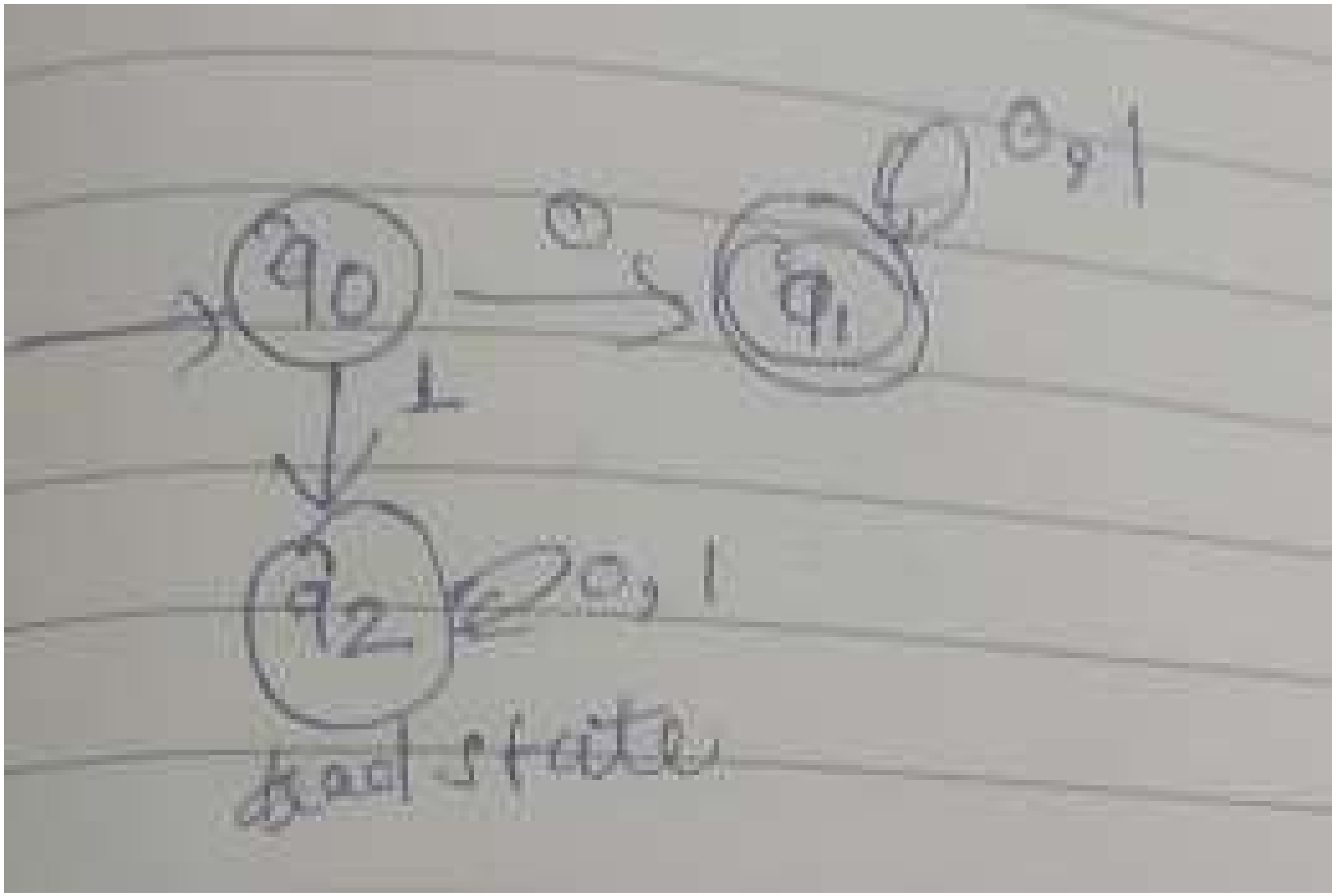
DFA	NFA
Epsilon transition are absent	Epsilon transition can be present

- Sigma={0,1}
- Starts with'1'
- L={1, 10,11,100,110,101,111,…….}

- Example #1:: Construct DFA which accepts those strings that starts with '1'over *Sigma* = {0,1}



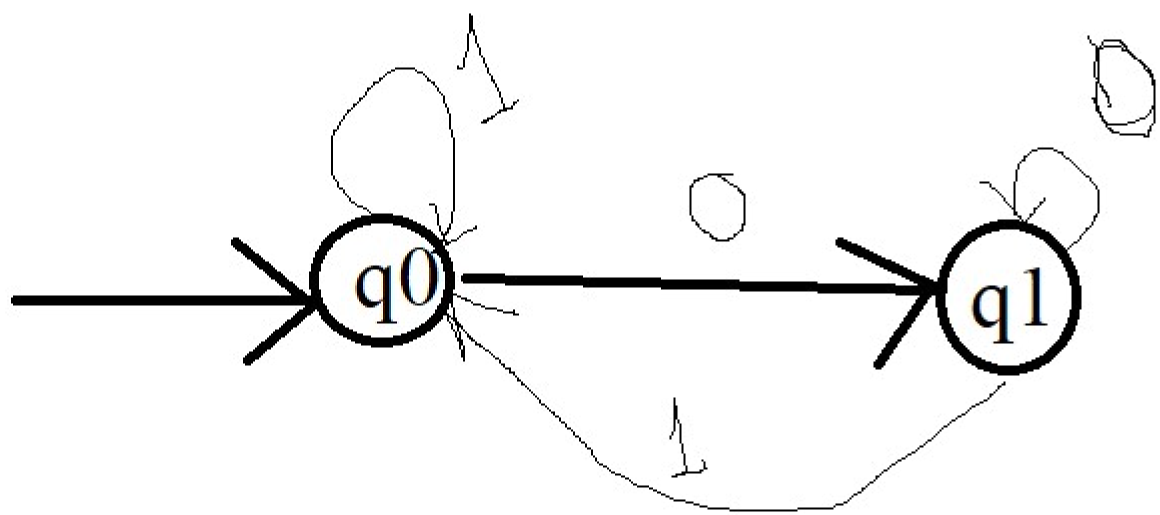
- Example #2:: Construct DFA which accepts those strings that starts with ‘0’over *Sigma* = {0,1}



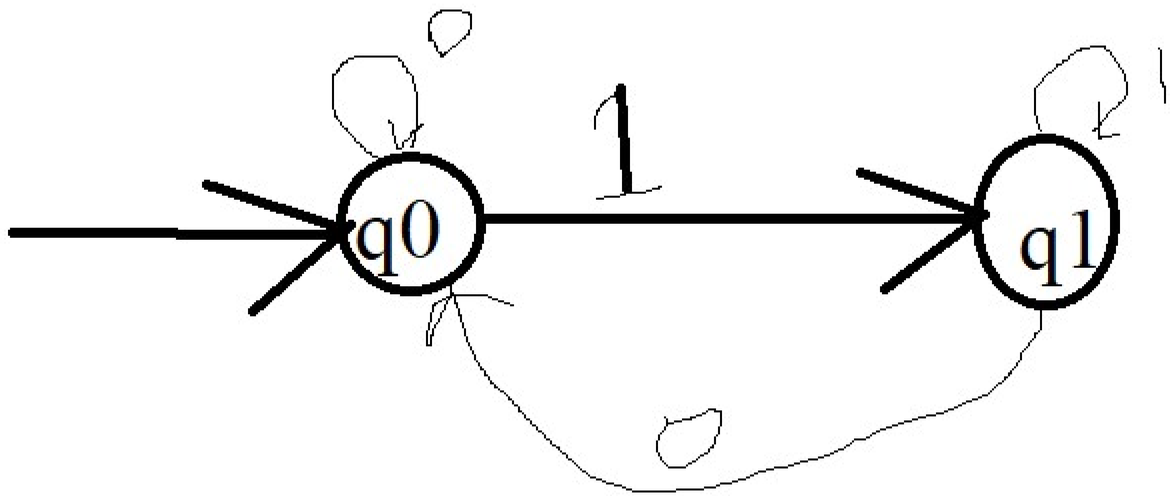
:: Construct DFA which accepts those strings that ends with ‘0’over *Sigma* = {0,1}

- Sigma{0,1}
- Ends with ‘0’
- L={0,10, 00, 100,110,010,000….}

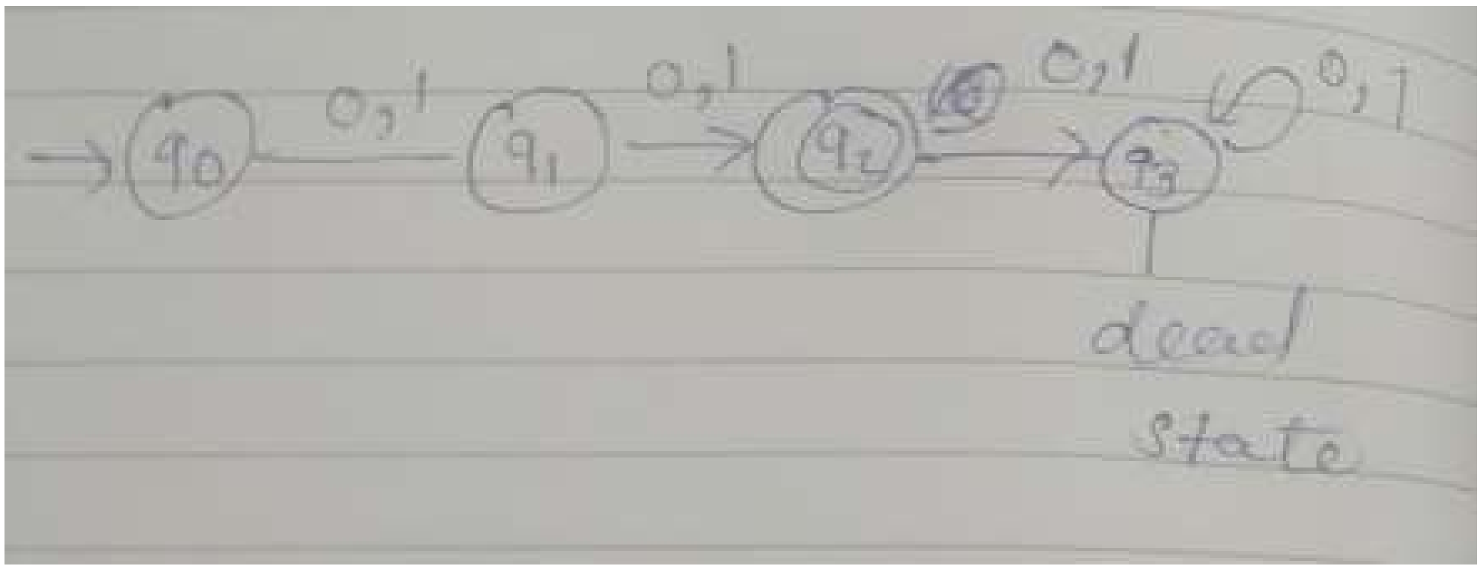
- Example #3:: Construct DFA which accepts those strings that ends with ‘0’over *Sigma* = {0,1}



- Example #4:: Construct DFA which accepts those strings that ends with ‘1’over *Sigma* = {0,1}



- Example #5:: Construct DFA over *Sigma* = {0,1} which accepts strings of length 2.
- L={}

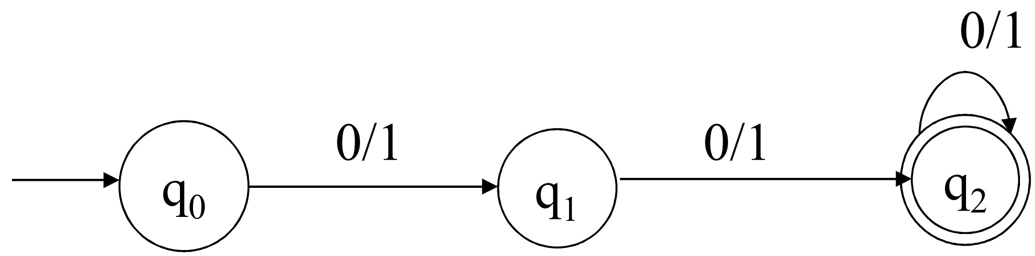


- Example #6:: Construct DFA over *Sigma* = {0,1} which accepts strings of length atleast 2(length>=2).

OR

- Give a DFA M such that:

L(M) = {x | x is a string of 0's and 1's and |x| >= 2}

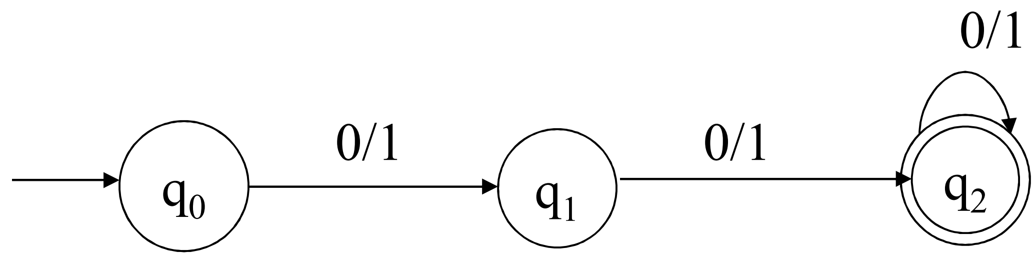


- Example #6:: Construct DFA over *Sigma* = {0,1} which accepts strings of length atleast 2(length>=2).

OR

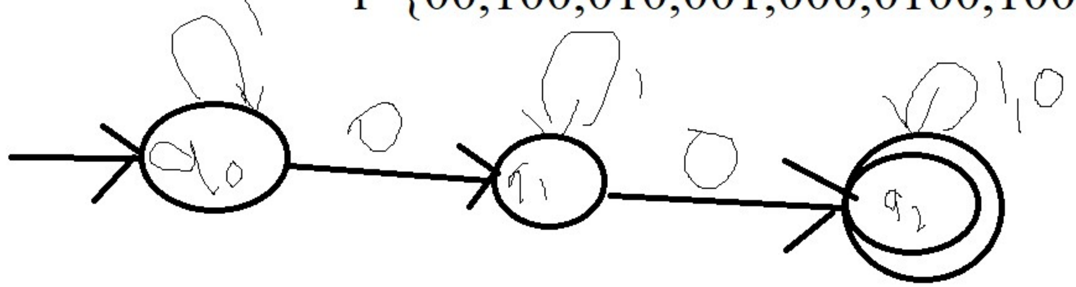
- Give a DFA M such that:

L(M) = {x | x is a string of 0's and 1's and |x| >= 2}



Example #6:: Construct DFA over *Sigma* = {0,1} which accepts strings in which no of ‘0’ s should be atleast 2(length>=2).

l={00,100,010,001,000,0100,1000....} nO OF '0'S SHOULI BE ATLEAST 2



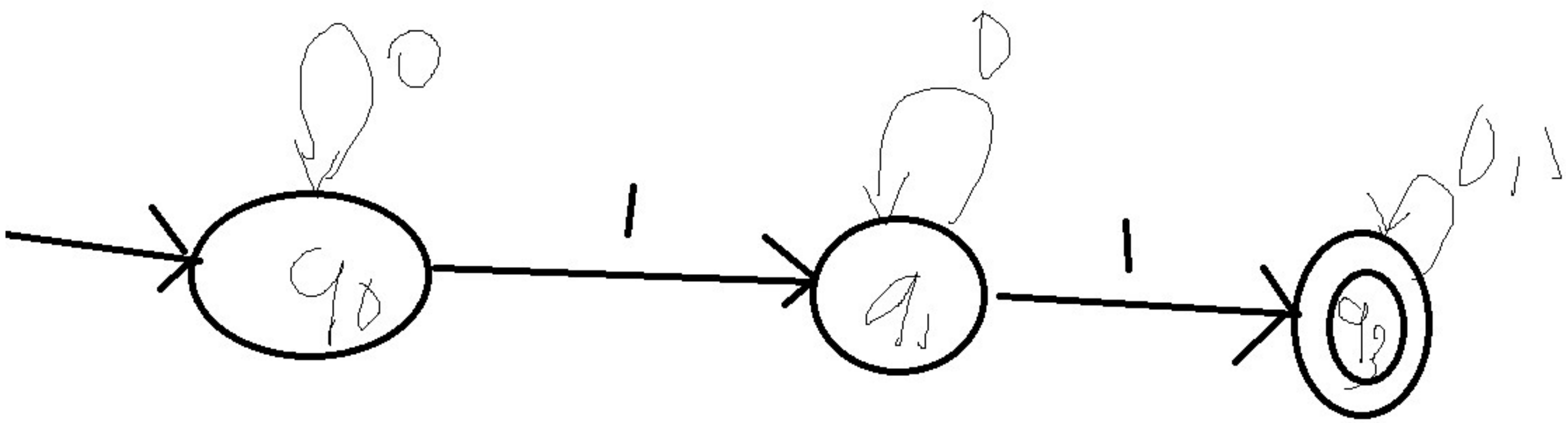
1101-iNAVLID

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

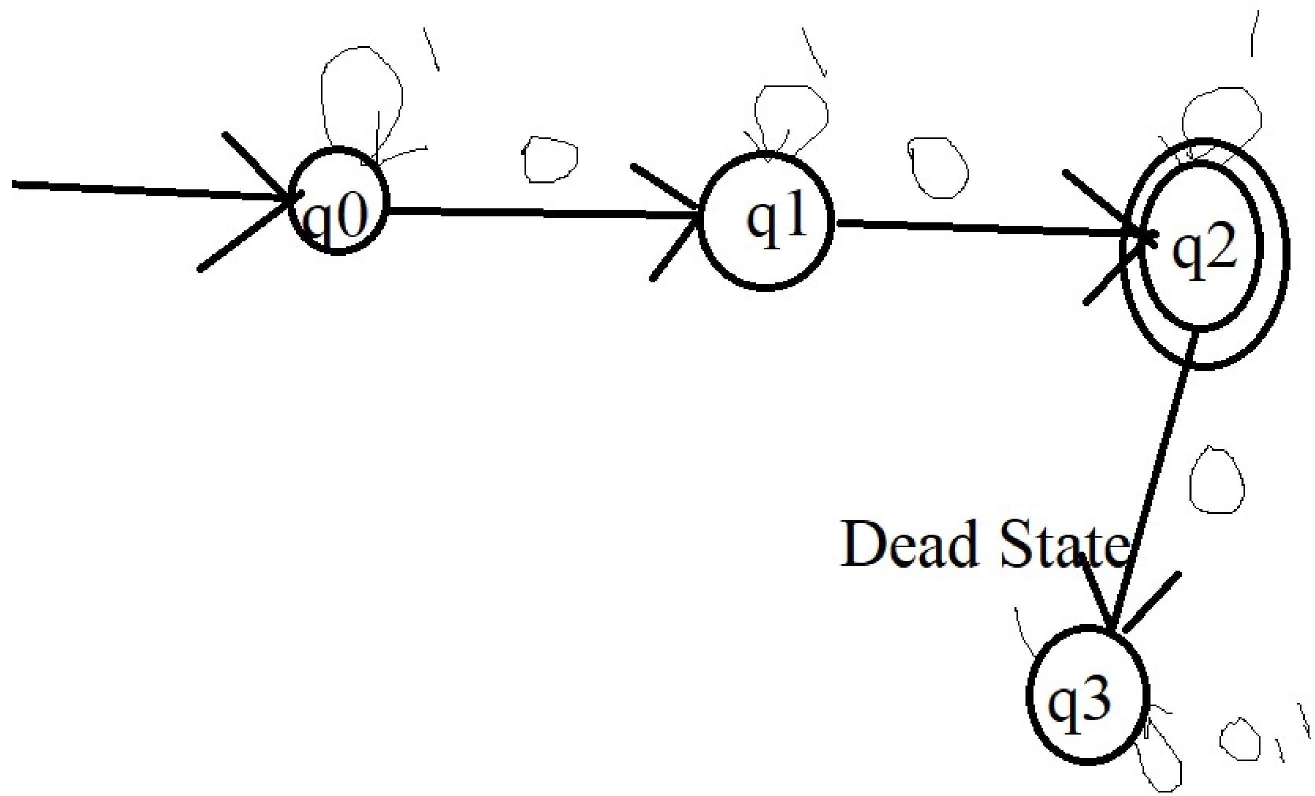
q1 is non final state so string'1101' will be rejected by this FA

Example #6:: Construct DFA over *Sigma* = {0,1} which accepts strings in which no of ‘1’ s should be atleast 2(length>=2).

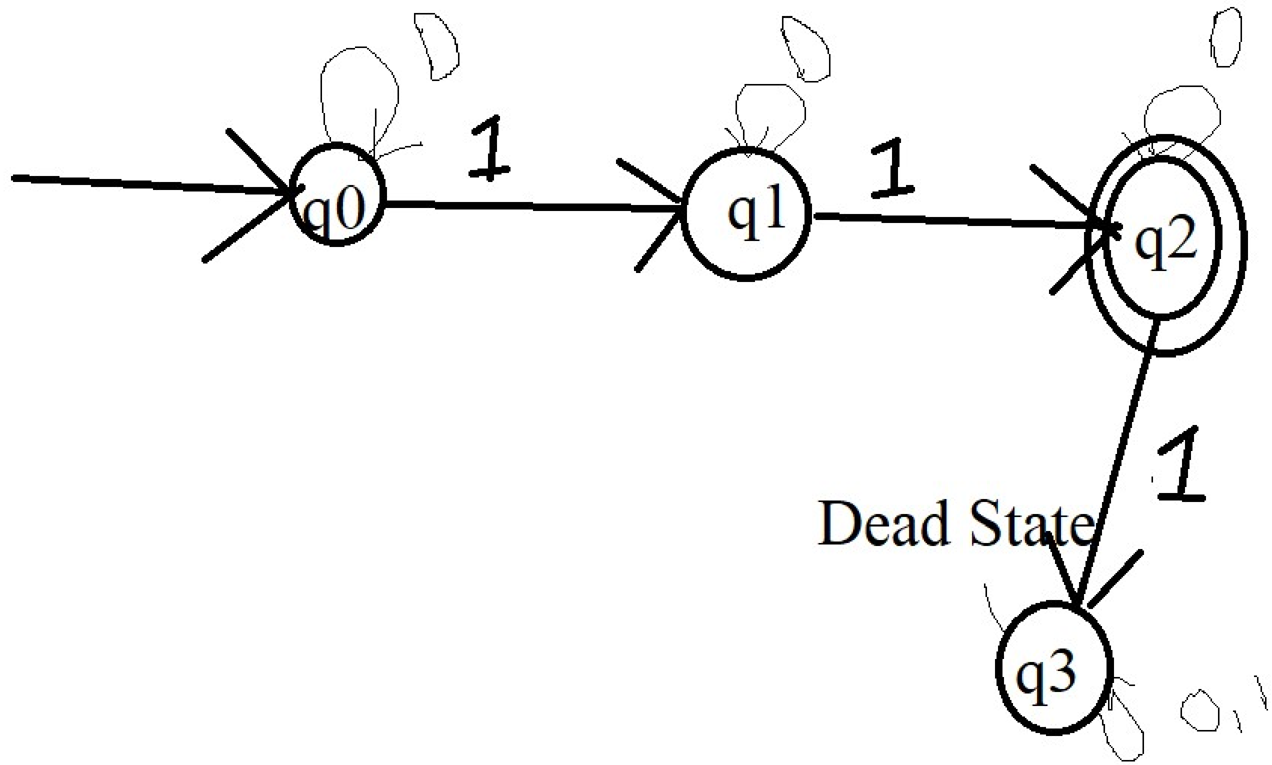
L={11,111,011,110,101,0111,1101,1110....}
No of '1's should be atleast 2.



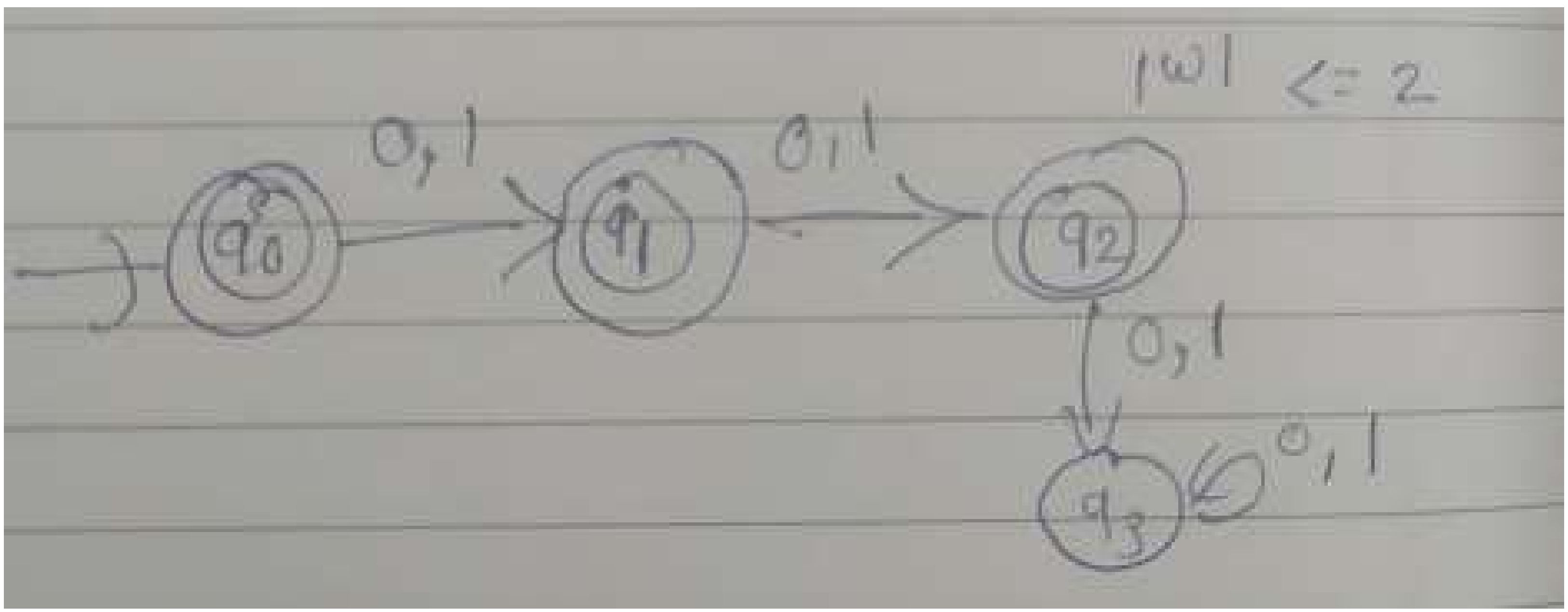
- Example #11:: Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 0's are 2.
- L={00,100,010,1010,}



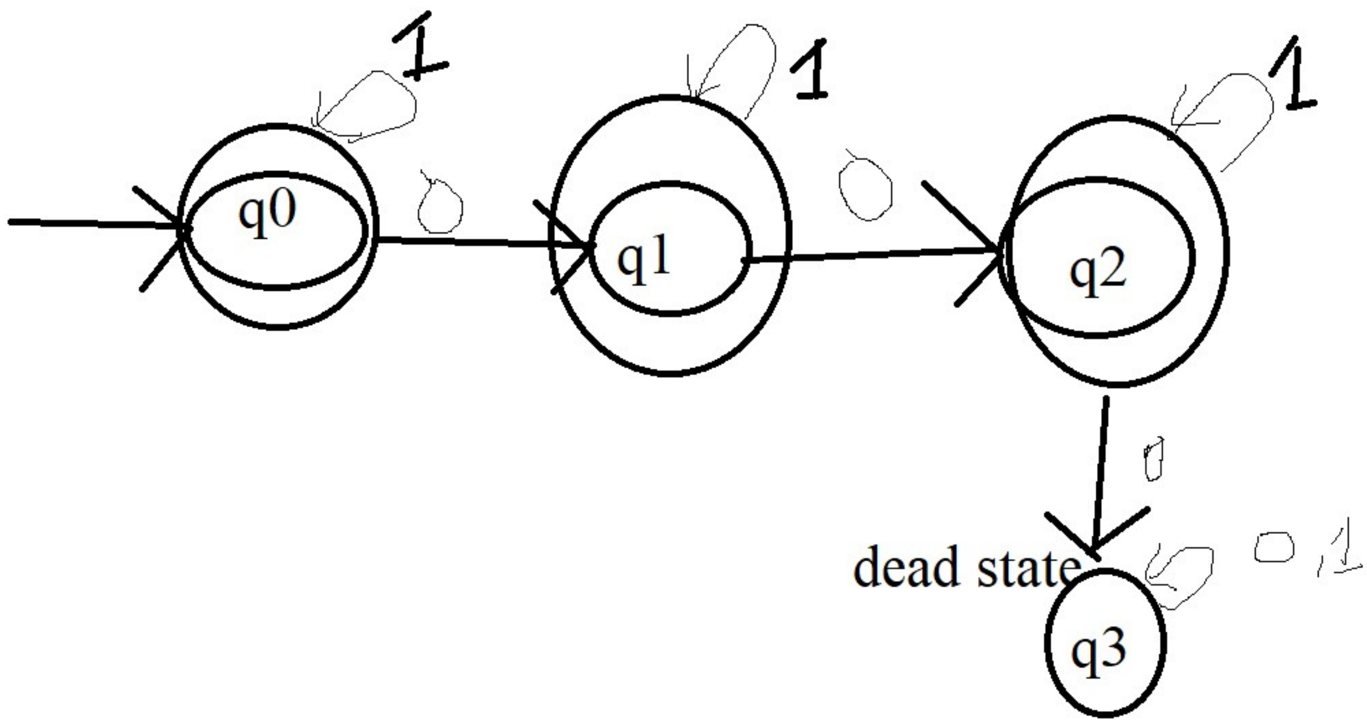
- Example #12Construct DFA over *Sigma* = {0,1} which accepts strings in which number of ‘1’s are 2. L={11,011,101,0110…..}



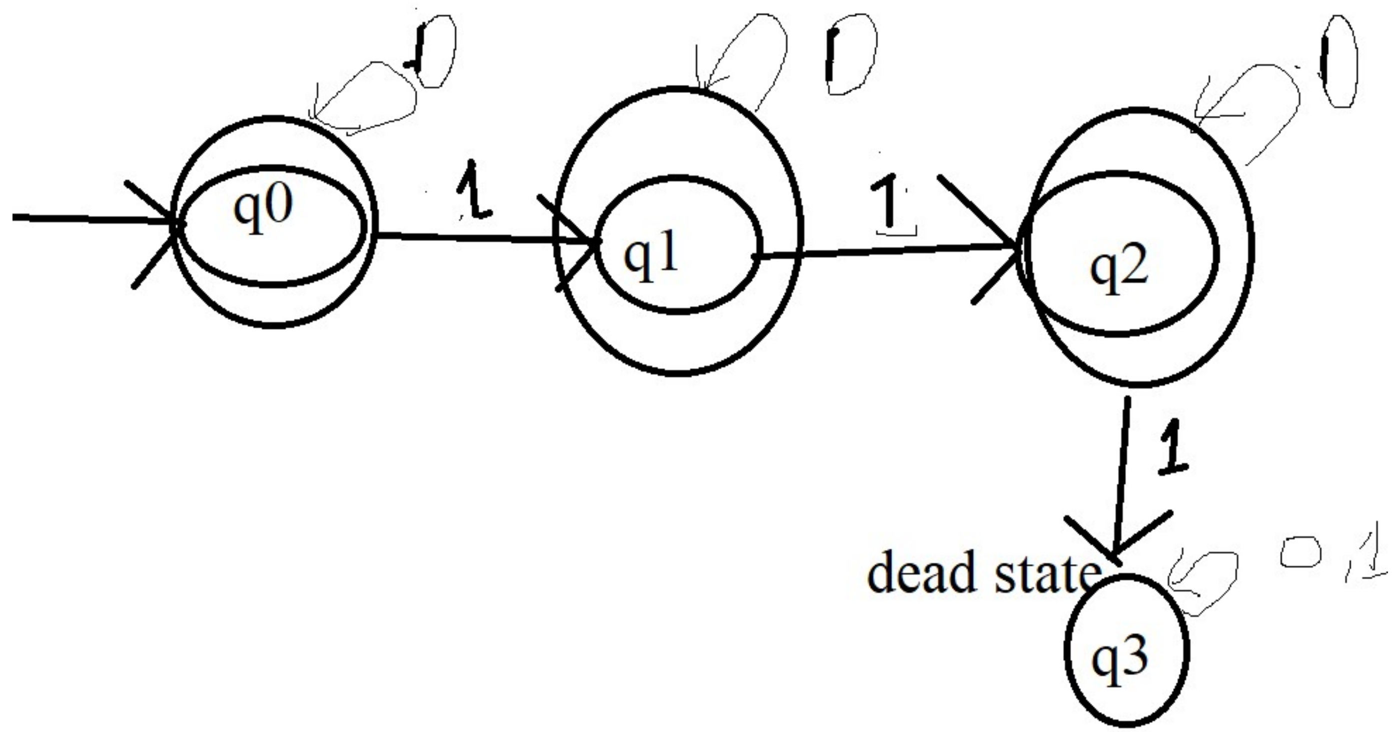
- Example #7:: Construct DFA over *Sigma* = {0,1} which accepts strings of length atmost2 (<=2).
- L={}



- Example #13:: Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 0's is atmost2 (<=2).
- L={1,11,111,0,00,010,….}

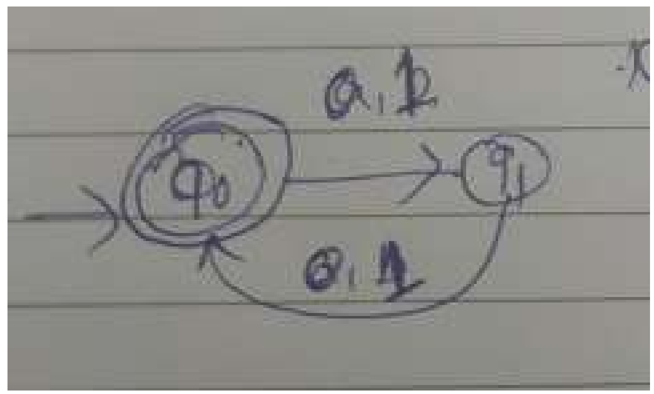


- Example #13:: Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 1's is atmost2 (<=2).
- L={0,00,000,1,11,101,….}

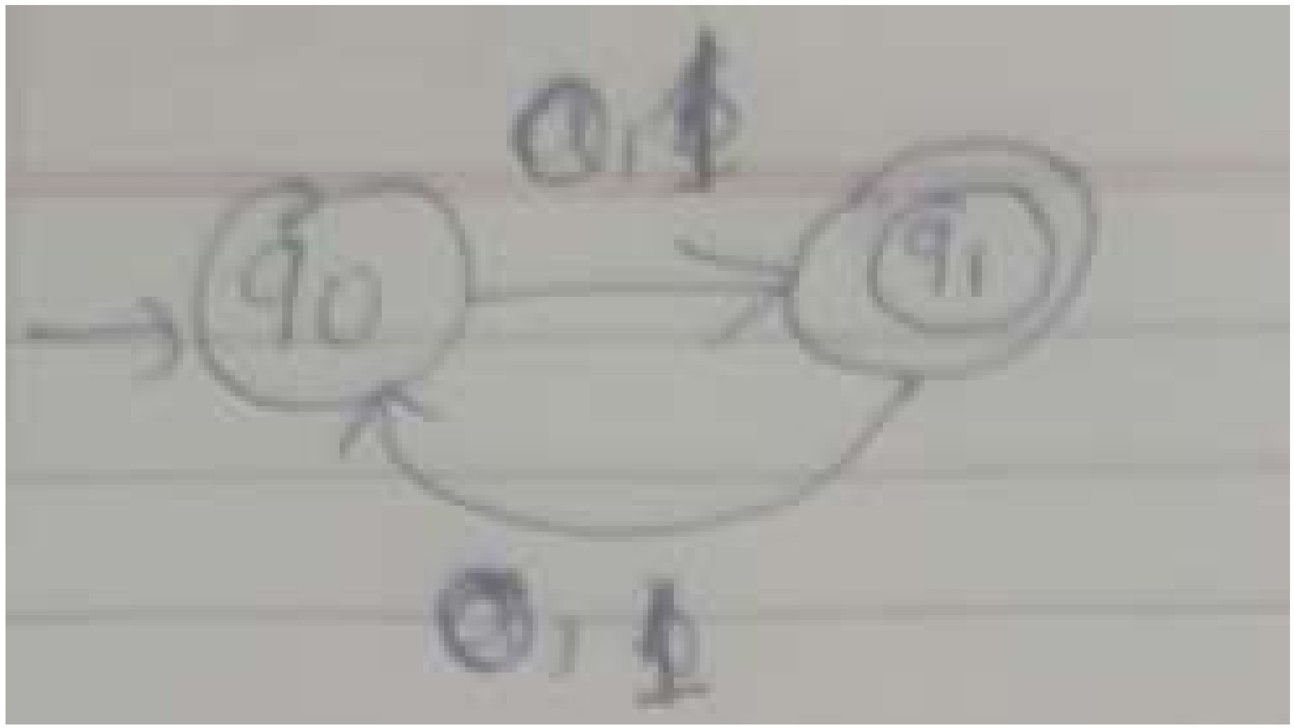


- Example #13:: Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 1's is atmost2 (<=2).
- L={0,00,000,1,11,101,….}

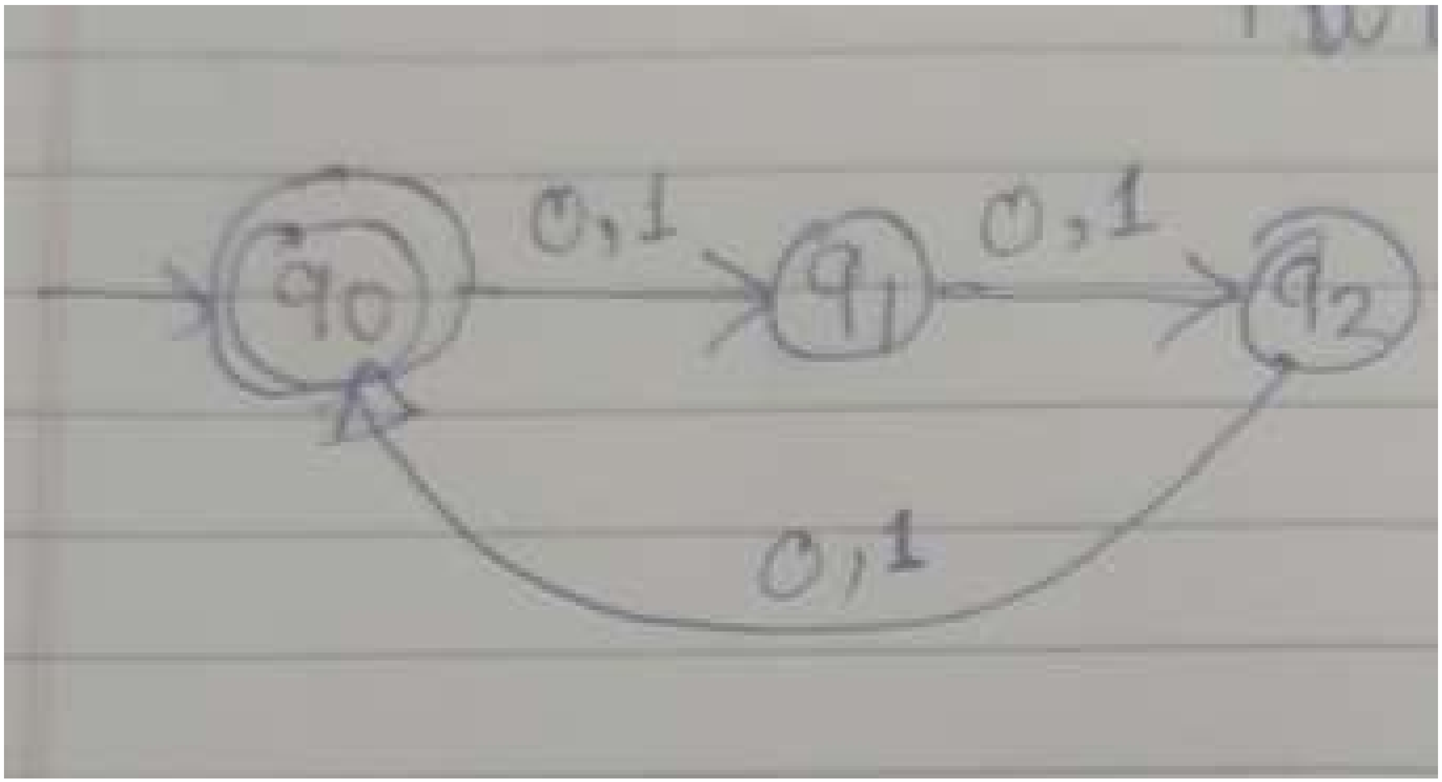
- Example #8:: Construct DFA over *Sigma* = {0,1} which accepts strings of even length or length of the string is divisible by2.
- L={}



- Example #9:: Construct DFA over *Sigma* = {0,1} which accepts strings of odd length.
- L={1,0,01,10,…..}

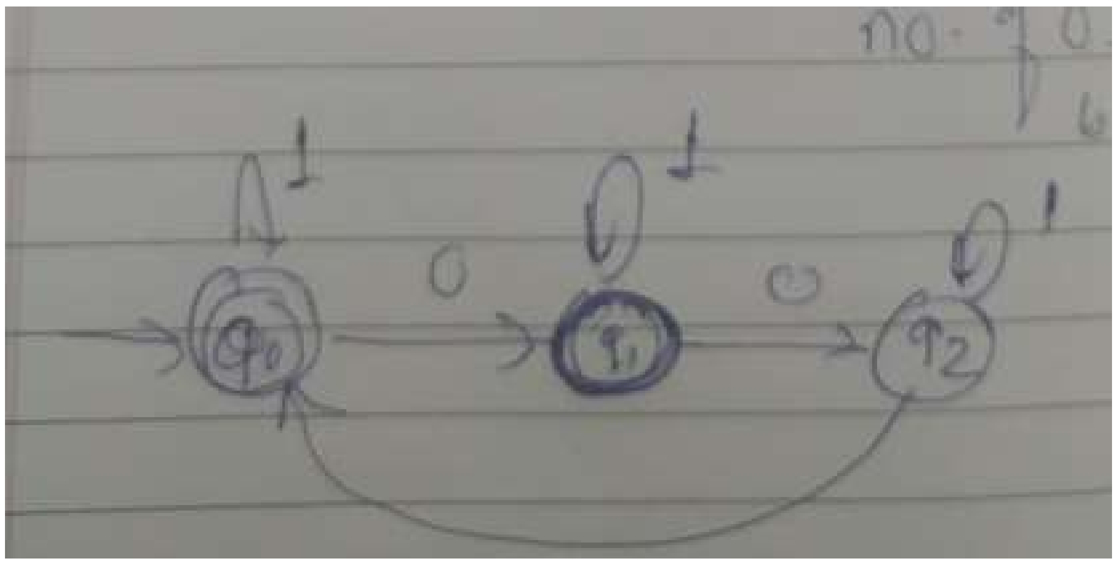


- Example #8:: Construct DFA over *Sigma* = {0,1} which accepts strings of even length or length of the string is divisible by 3.
- L={}

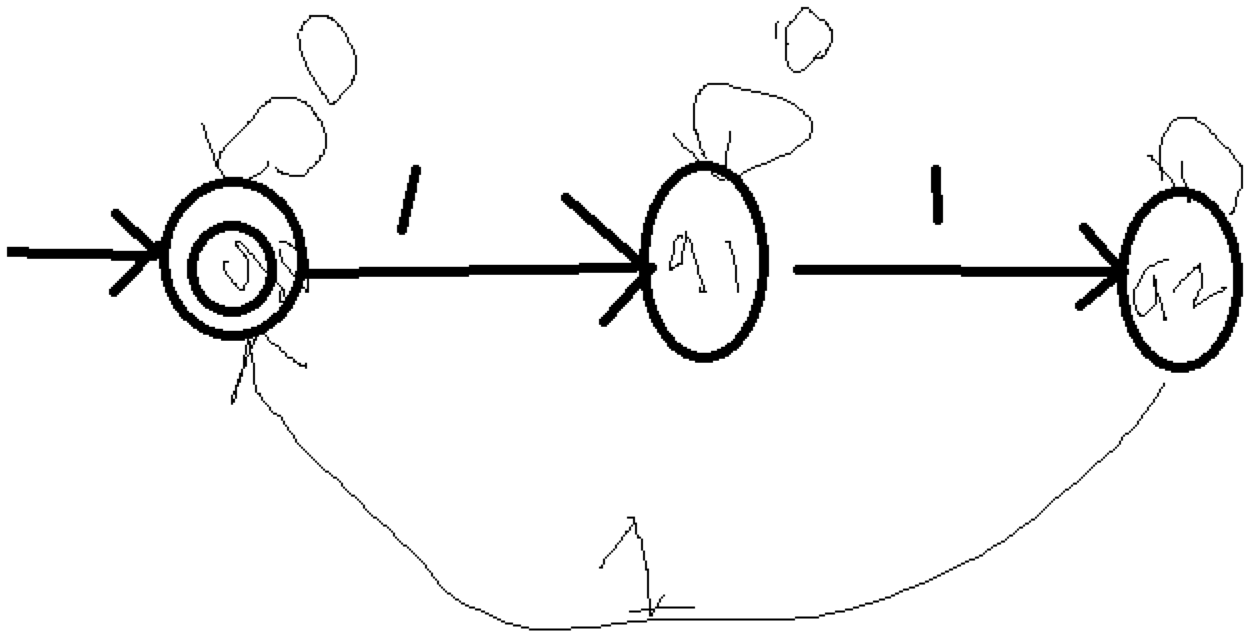


Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 0's are divisible by 3.

L={null, 000,1,11,111,1000,11000,….}

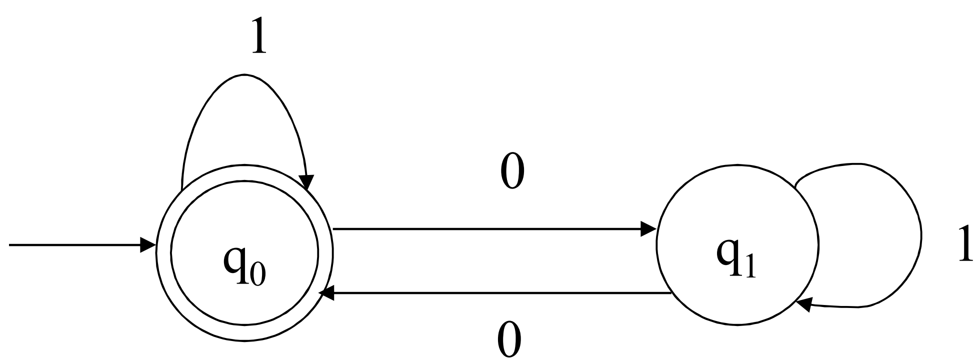


• Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 1's are divisible by 3.



- The finite control can be described by a <u>transition diagram</u> or <u>table</u>:

- Example #9:: Construct DFA which accepts those strings that contain an even number of 0’sover *Sigma* = {0,1}



L={null,
1,11,111,00,0000,010….}

1 0 0 1 1

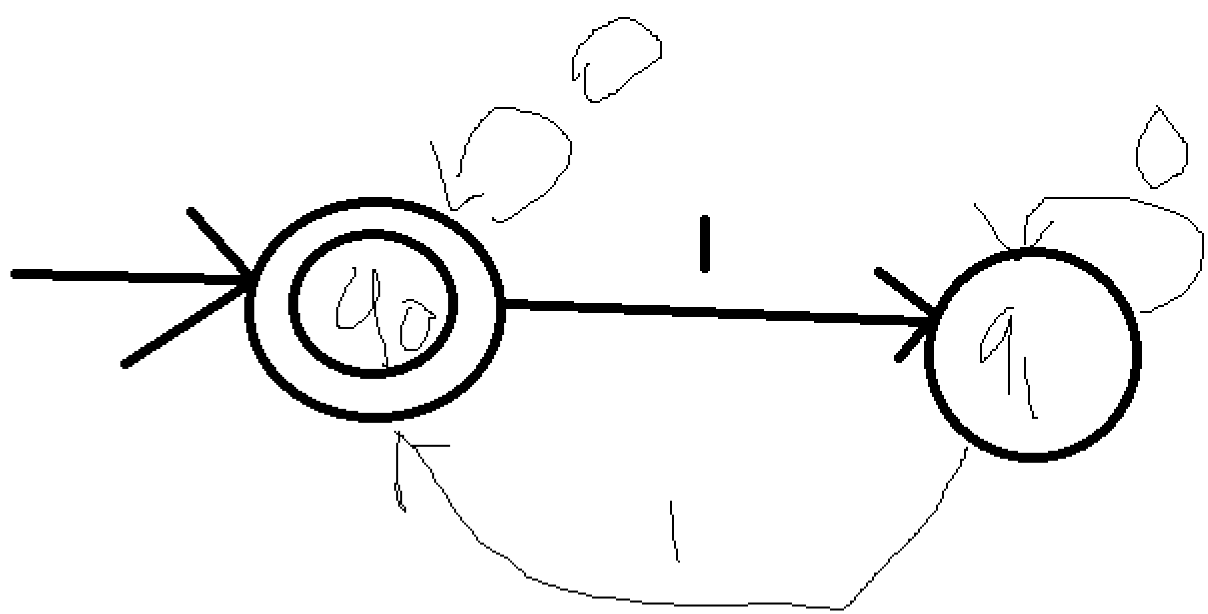
q_0 q_0 q_1 q_0 q_0 q_0

- }

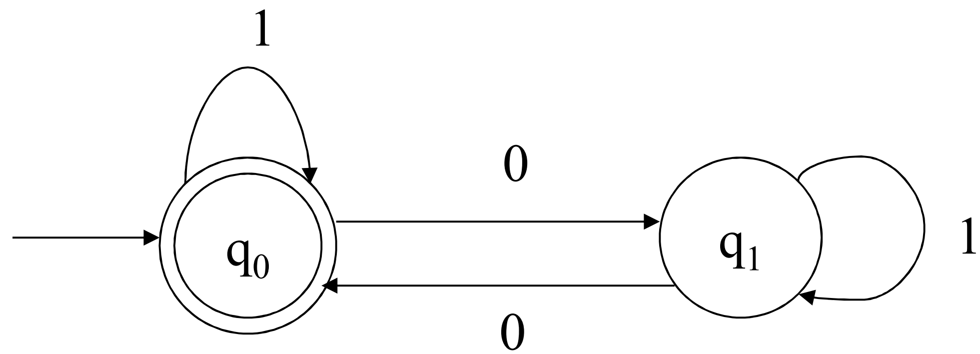
L = {all strings with zero or more 0’s}

- Note, the DFA must reject all other strings

- Example #9:: Construct DFA which accepts those strings that contain an even number of 1's, over *Sigma* = {0,1}
- {null, 0,00,11,…..}



- Revisit example #9: Construct DFA which accepts those strings that contain an even number of 0's, including the *null* string, over *Sigma* = {0,1}



No.of states

$Q = \{q_0, q_1\}$

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

Start state is q_0

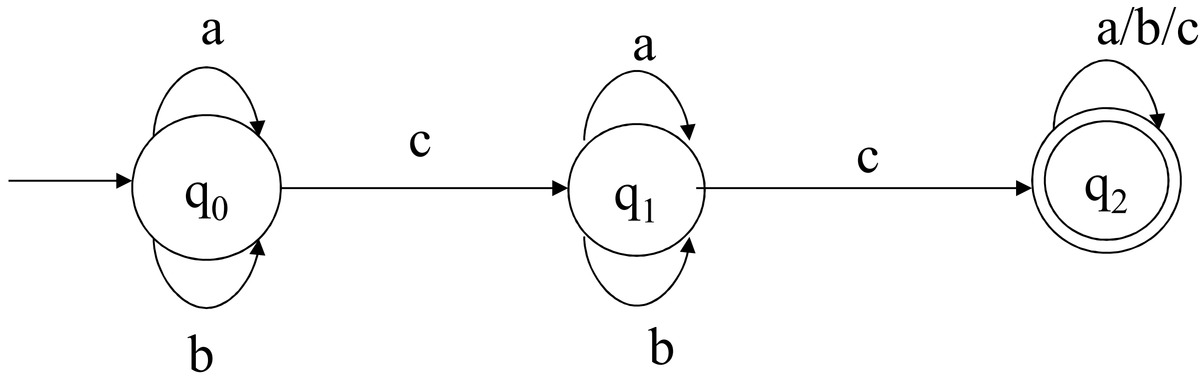
$F = \{q_0\}$

Transition Table

δ:

States \ Σ	0	1
q_0	q_1	q_0
q_1	q_0	q_1

- Example #10: Construct DFA which accepts those strings that contain at least two c's over sigma$\sum = \{a,b,c\}$



Q={q0,q1,q2}

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

Q0={q0}

F={q2}

	a	c	c	c	b	<u>accepted</u>
q_0	q_0	q_1	q_2	q_2	q_2	
	a	a	c			<u>rejected</u>
q_0	q_0	q_0	q_1			

- Accepts those strings that contain <u>at least</u> two *c*'s

- Revisit example #10:

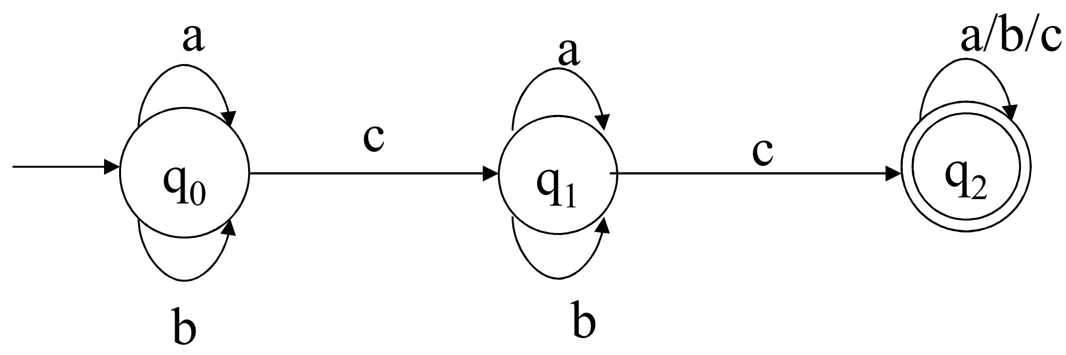
Transition diagram of DFA

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

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

Start state is q_0

$F = \{q_2\}$



δ:

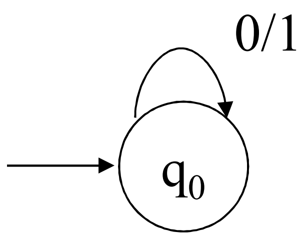
$Q \backslash \Sigma$	a	b	c
q_0	q_0	q_0	q_1
q_1	q_1	q_1	q_2
q_2	q_2	q_2	q_2

Transition table of DFA

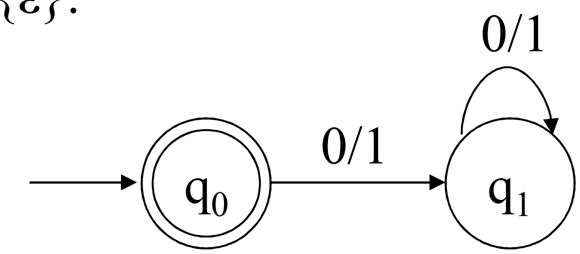
- Since δ is a function, at each step M has exactly one option.
- It follows that for a given string, there is exactly one computation.

- Let $\Sigma = \{0, 1\}$. Give DFAs for $\{\}$, $\{\varepsilon\}$, Σ^*, and Σ^+.

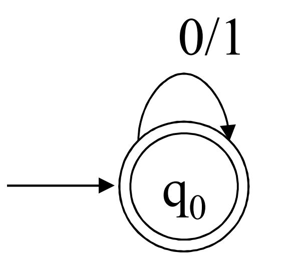
For $\{\}$:



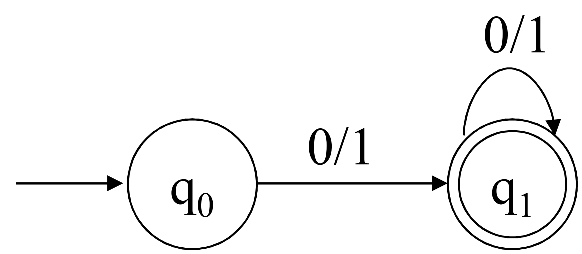
For $\{\varepsilon\}$:



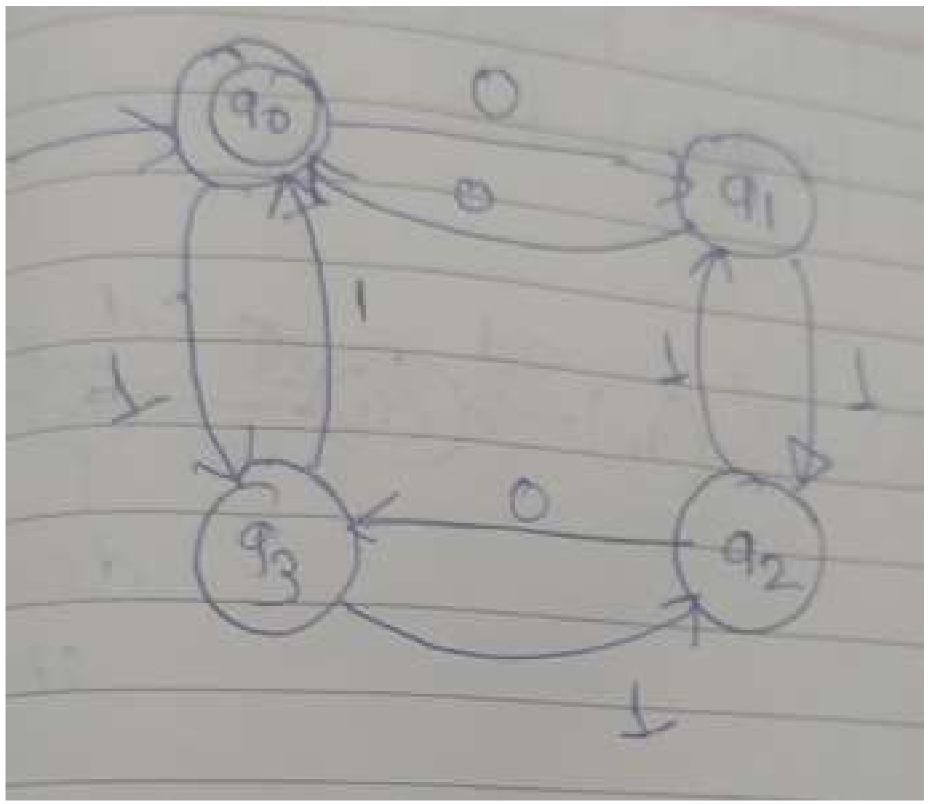
For Σ^*:



For Σ^+:



- Example #13:: Construct DFA over *Sigma* = {0,1} which accepts strings in which number of 0's and number of 1's are even
- L={}



Acceptability of a string by a FA

2.5 ACCEPTABILITY OF A STRING BY A FINITE AUTOMATON

Definition 2.4 A string x is accepted by a finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$. This is basically the acceptability of a string by the final state.

NOTE: A final state is also called an accepting state.

Acceptability of a string by a FA

States	Inputs 0	Inputs 1
$\rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

SOLUTION

$$\begin{aligned}
\delta(q_0, 110101) &= \delta(q_1, 10101) \\
&= \delta(q_0, 0101) \\
&= \delta(q_2, 101) \\
&= \delta(q_3, 01) \\
&= \delta(q_1, 1) \\
&= \delta(q_0, \Lambda) = q_0
\end{aligned}$$

Hence,

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

The symbol $\downarrow$ indicates the current input symbol being processed by the machine

NFA

Definition 2.5 A nondeterministic finite automaton (NDFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

(i) Q is a finite nonempty set of states;
(ii) Σ is a finite nonempty set of inputs;
(iii) δ is the transition function mapping from $Q \times \Sigma$ into 2^Q which is the power set of Q, the set of all subsets of Q;
(iv) $q_0 \in Q$ is the initial state; and
(v) $F \subseteq Q$ is the set of final states.

We note that the difference between the deterministic and nondeterministic automata is only in δ. For deterministic automaton (DFA), the outcome is a state, i.e. an ... automaton the outcome is a subset of Q.

NFA	DFA
Epsilon transition	No Epsilon transition

$\delta'([q_1 \ldots q_k], a)$.

NOTE: We write δ' as δ itself when there is no ambiguity. We also mark the initial state with $\rightarrow$ and final state with circle in the state table.

Table 2.2 State Table for Example 2.6

State/Σ	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_1	q_0, q_1

EXAMPLE 2.6 Construct a deterministic automaton equivalent to $M = (\{q_0, q_1\}, \{0, 1\}, \delta, q_0, \{q_0\})$. δ is given by its state table (Table 2.2).

SOLUTION For the deterministic automaton M_1,

(i) the states are subsets of $\{q_0, q_1\}$, i.e. $\emptyset$, $[q_0]$, $[q_0, q_1]$, $[q_1]$;
(ii) $[q_0]$ is the initial state;
(iii) $[q_0]$ and $[q_0, q_1]$ are the final states as these are the only states containing q_0; and
(iv) δ is defined by the state table given by Table 2.3.

Table 2.3 State Table of M_1

States/Σ	0	1
$\emptyset$	$\emptyset$	$\emptyset$
$[q_0]$	$[q_0]$	$[q_1]$

SOLUTION T

where

We start the
we construc
columns. A
we termina

EXAMI
$M = ((\{$

Transition Table of DFA

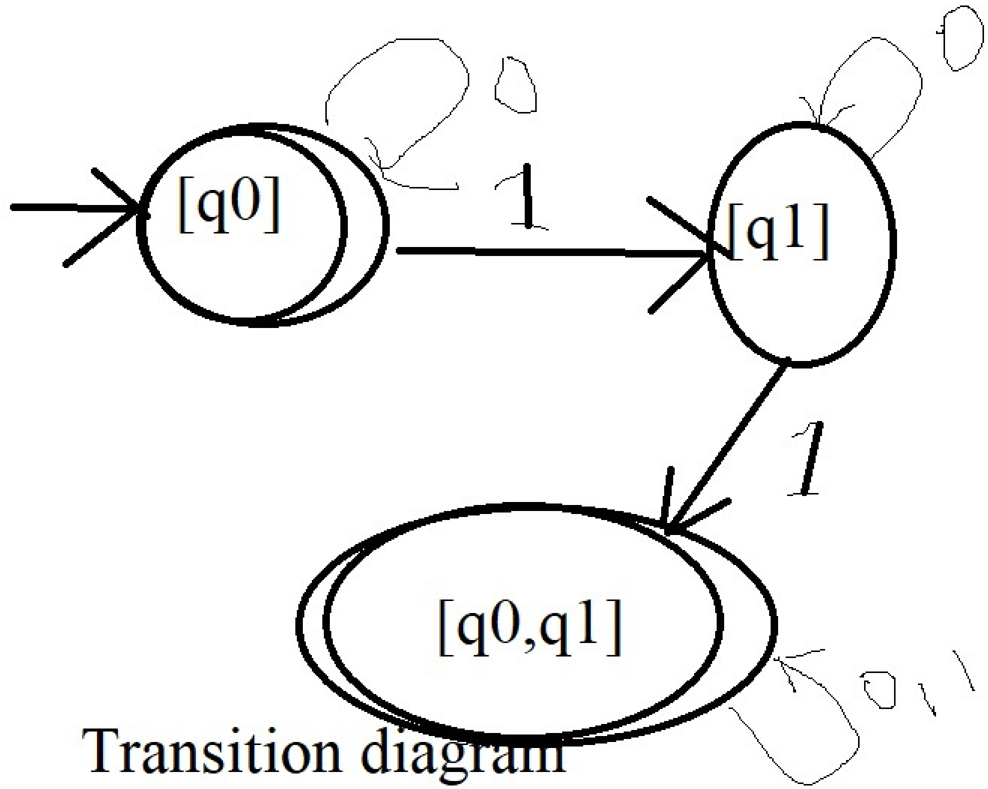
	0	1
[q0]	[q0]	[q1]
[q1]	[q1]	[q0,q1]
[q0,q1]	[q0,q1]	[q0,q1]

Transition table of DFA

State\ ∑	0	1
[q0]	[q0]	[q1]
[q1]	[q1]	[q0,q1]
[q0.q1]	[q0.q1]	[qo,q1]

- As after [q0,q1] no new state is generated we can say that conversion is completed.Now next step is to mark the final state of DFA.
- Initial state of NFA becomes the initial state of DFA.
- Final state of DFA will be those **states** in which the final state of NFA appears.

Transition diagram of DFA



Transition diagram
of DFA

EXAMPLE 2.7 Find a deterministic acceptor equivalent to

$$M = (\underbrace{\{q_0, q_1, q_2\}}_{Q}, \underset{\Sigma}{\{a, b\}}, \delta, \underset{Q_0}{q_0}, \underset{F}{\{q_2\}})$$

δ is given in Table 2.4.

Table 2.4 State Table for Example 2.7

States/Σ	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
q_2		q_0, q_1

Transition Tables of DFA

Q	a	b
[q0]	[q0,q1]	[q2]
[q0,q1]	[q0,q1]	[q1,q2]
[q2]		[q0,q1]
[q1,q2]	[q0]	[q0,q1]

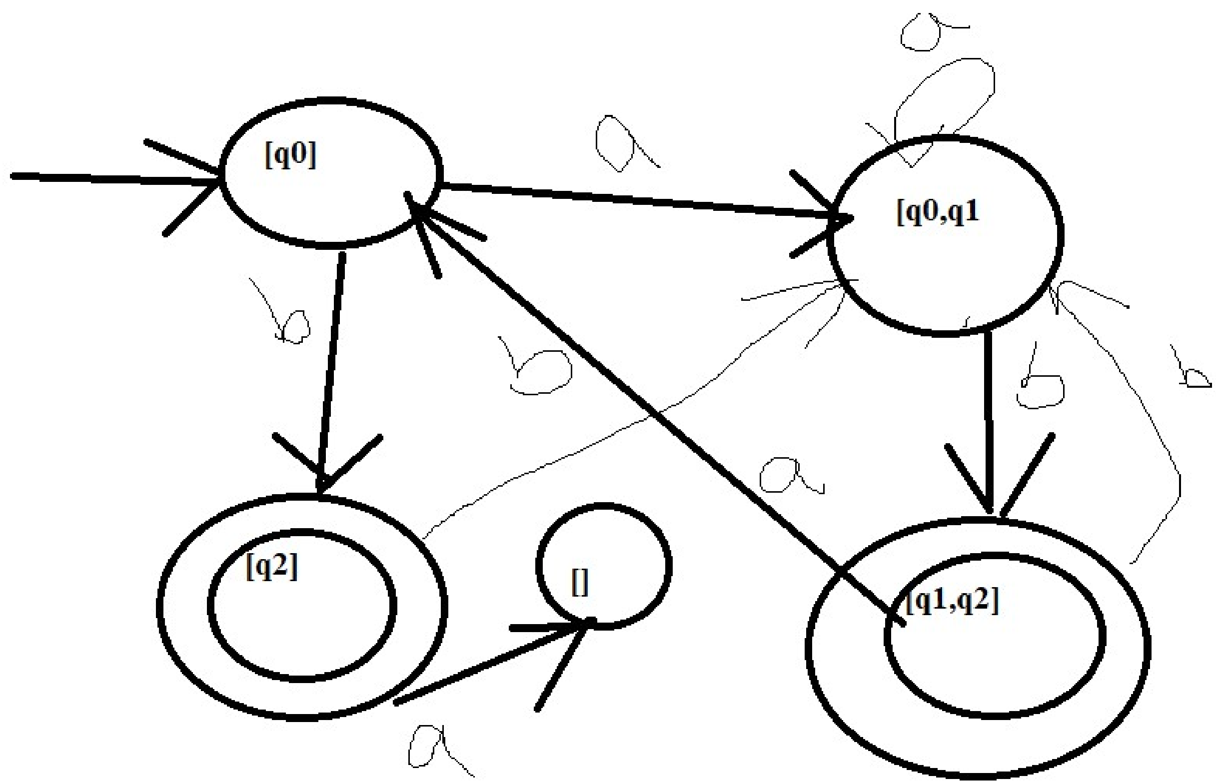
Transition table of DFA

States\	a	b
[q0]	[q0,q1]	[q2]
[q0,q1]	[q0,q1]	[q1,q2]
[q2]	[]	[q0,q1]
[q1,q2]	[q0]	[q0,q1]

Q\∑	a	b
[q0]	[q0,q1]	[q2]
[q0,q1]	[q0,q1]	[q1,q2]
[q2]	[ ]	[q0,q1]
[q1,q2]	[q0]	[q0,q1]

Transition table of DFA

- Final state of DFA will be those **states** in which the final state of NFA appears.



7. The transition table of a nondeterministic finite automaton M is given in Table 2.25. Construct a deterministic finite automaton equivalent to M.

Table 2.25 Transition Table for Exercise 2.7

State	0	1	2
$\rightarrow q_0$	q_1q_4	q_4	q_2q_3
q_1		q_4	
q_2			q_2q_3
q_3		q_4	
q_4			

final states

Transition table of DFA(HW Solution Q7

States\∑	0	1	2
[qo]	[q1,q4]	[q4]	[q2,q3]
[q1,q4]	[]	[q4]	[]
[q4]	[]	[]	[]
[q2,q3]	[]	[q4]	[q2,q3]

HW

9. $M = (\{q_1, q_2, q_3\}, \{0, 1\}, \delta, q_1, \{q_3\})$ is a nondeterministic finite automaton, where δ is given by

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

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

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

Construct an equivalent DFA.

HW Solution Q9

Transition Table of NFA

States\sigma	0	1
q1	q2,q3	q1
q2	q1,q2	
q3	q2	q1,q2

Transition Table of DFA

States\sigma	0	1
[q1]	[q2,q3]	[q1]
[q2,q3]	[q1,q2]	[q1,q2]
[q1,q2]	[q1,q2,q3]	[q1]
[q1,q2,q3]	[q1,q2,q3]	[q1,q2]

Conversion process will stop, when all the states in column 2 and 3 will appear in column 1.

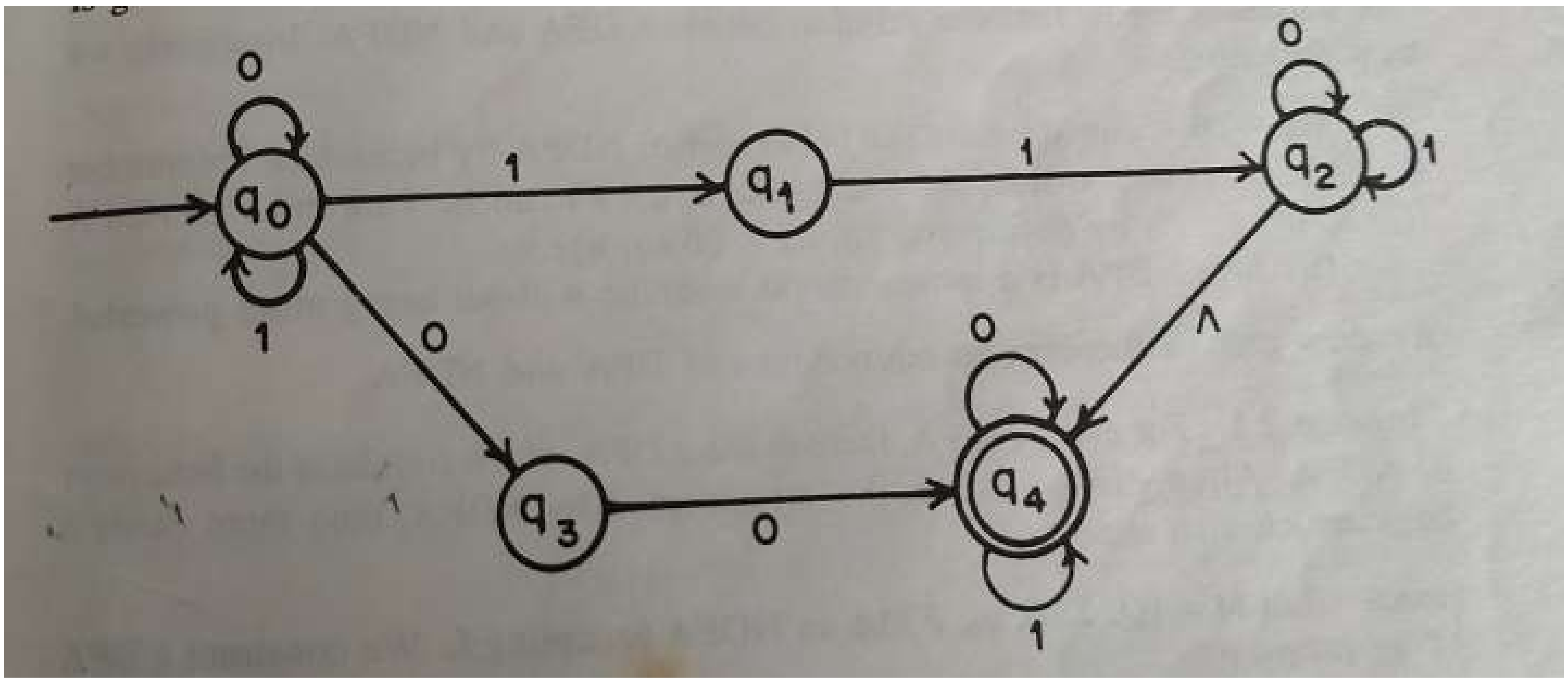
Solution

Transition Table of DFA

Table A2.6 State Table for Exercise 2.9

States	0	1
$[q_1]$	$[q_2, q_3]$	$[q_1]$
$[q_2, q_3]$	$[q_1, q_2]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_1, q_2, q_3]$	$[q_1]$
$[q_1, q_2, q_3]$	$[q_1, q_2, q_3]$	$[q_1, q_2]$

Convert the following NFA into DFA-HW

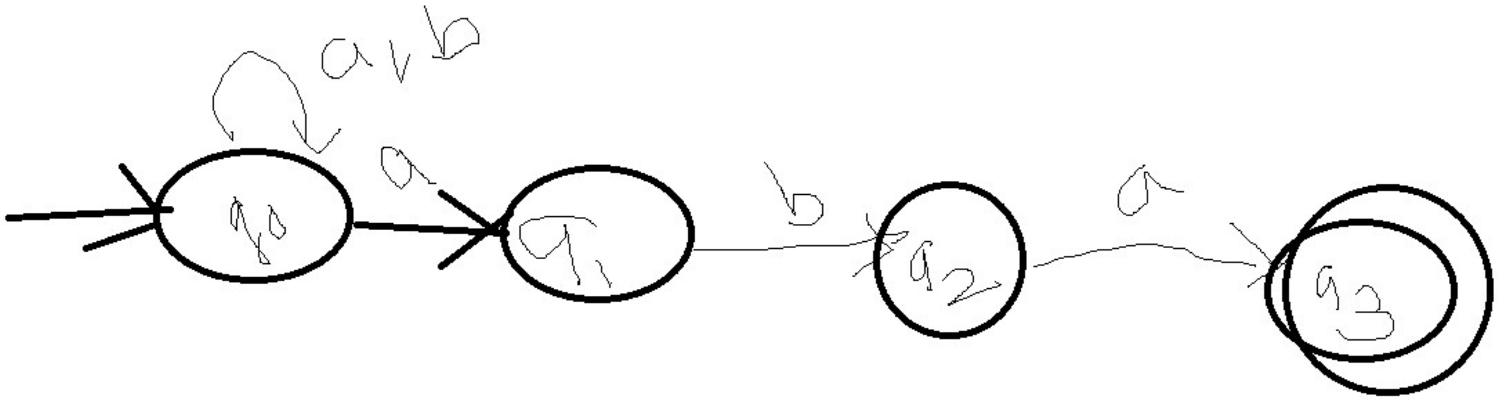


Q. Construct a nondeterministic finite automaton accepting the set of all strings over{a,b} **ending in aba**. Use it to construct a DFA accepting the same set of all strings

Solution:

- Step1 : Construct transition diagram of NFA
- Step 2: Construct a Transition Table of NFA from Transition diag.
- Step3: convert this NFA transition table into DFA transition table.
- Step4: Construct a Transition diag. of DFA

L-{aba, aaba,baba, aaaba,bbaba,baaba,ababa.....}



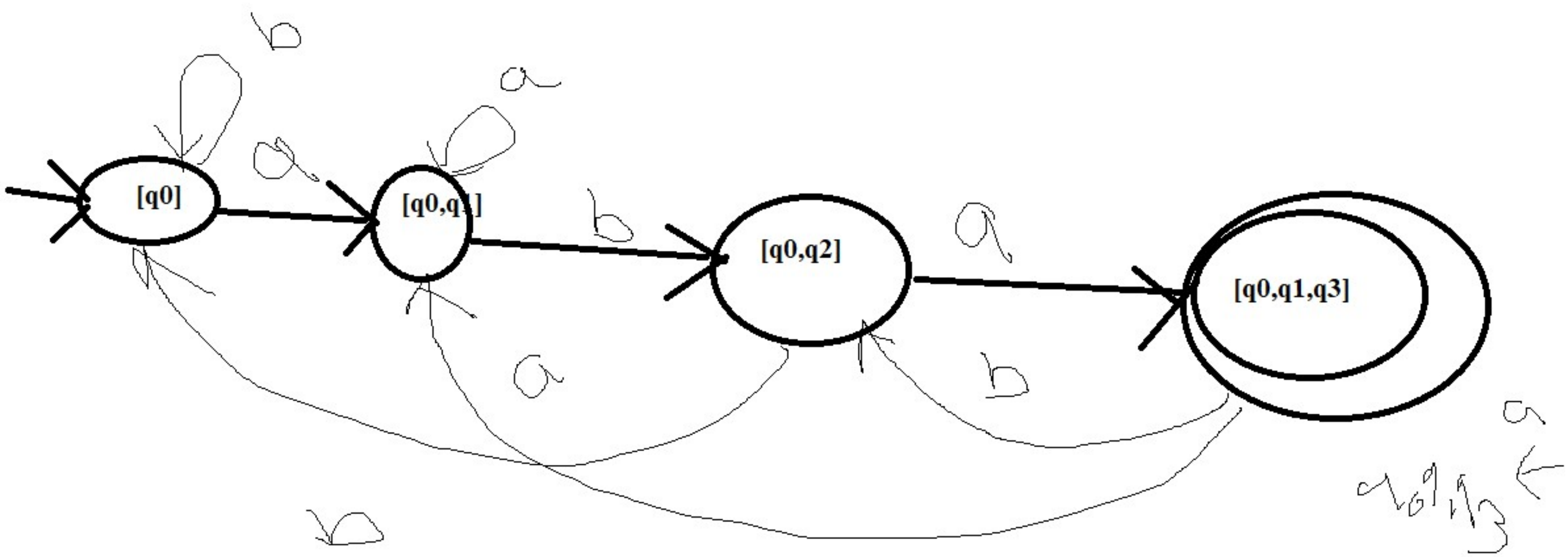
Transition table of NFA

Stattes	a	b
q0	q0,q1	q0
q1		q2
q2	q3	
q3		

Transition table of DFA

States\sigma	a	b
[q0]	[q0,q1]	[q0]
[q0,q1]	[q0,q1]	[q0,q2]
[q0,q2]	[q0,q1,q3]	[q0]
[q0,q1,q3]	[q0,q1]	[q0,q2]

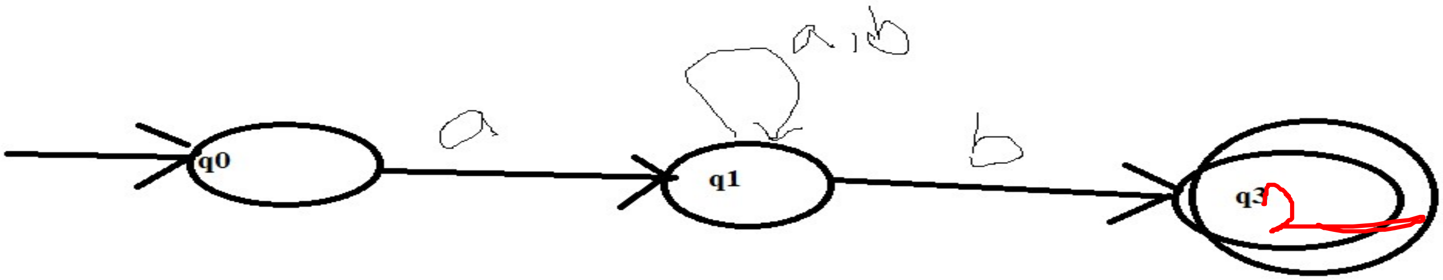
Transition diagram of DFA



Question :Construct a nondeterministic finite automaton accepting the set of all strings over{a,b} starting with ‘a’ ending with ‘b’. Use it to construct a DFA accepting the same set of all strings

L={ab,aab,abb,aaab….}

Transition diagram of NFA



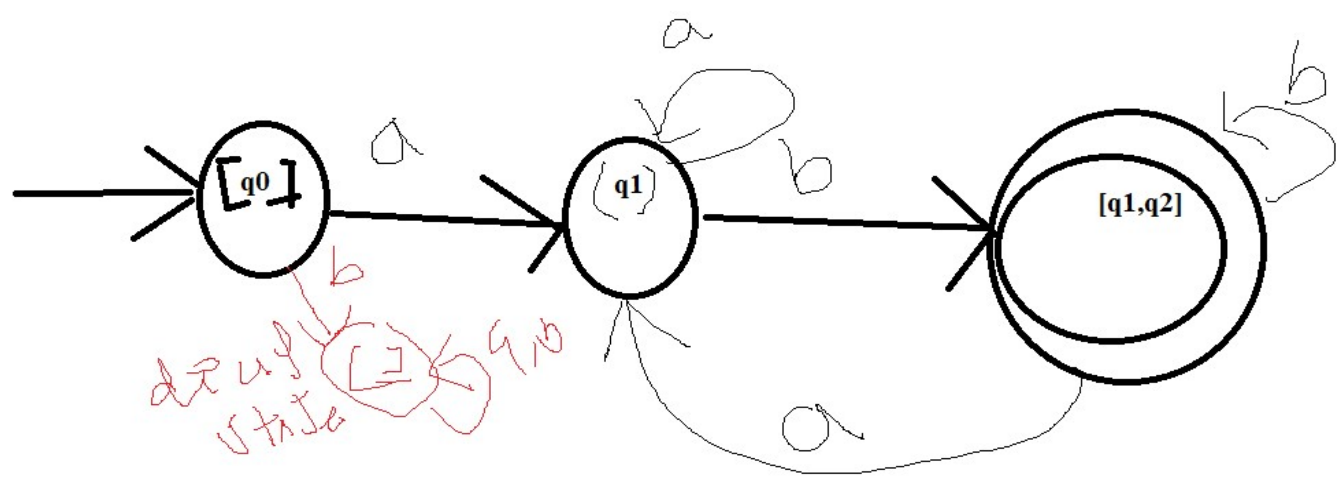
Transition table of NFA

	a	b
q0	q1	
q1	q1	q1,q2
q2		

	a	b
[q0]	[q1]	[]
[q1]	[q1]	[q1,q2]
[q1,q2]	[q1]	[q1,q2]
[]	[]	[]

Transition table of DFA

Transition diagram of DFA



aab=valid string

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

String 'aab' is accepted by this DFA

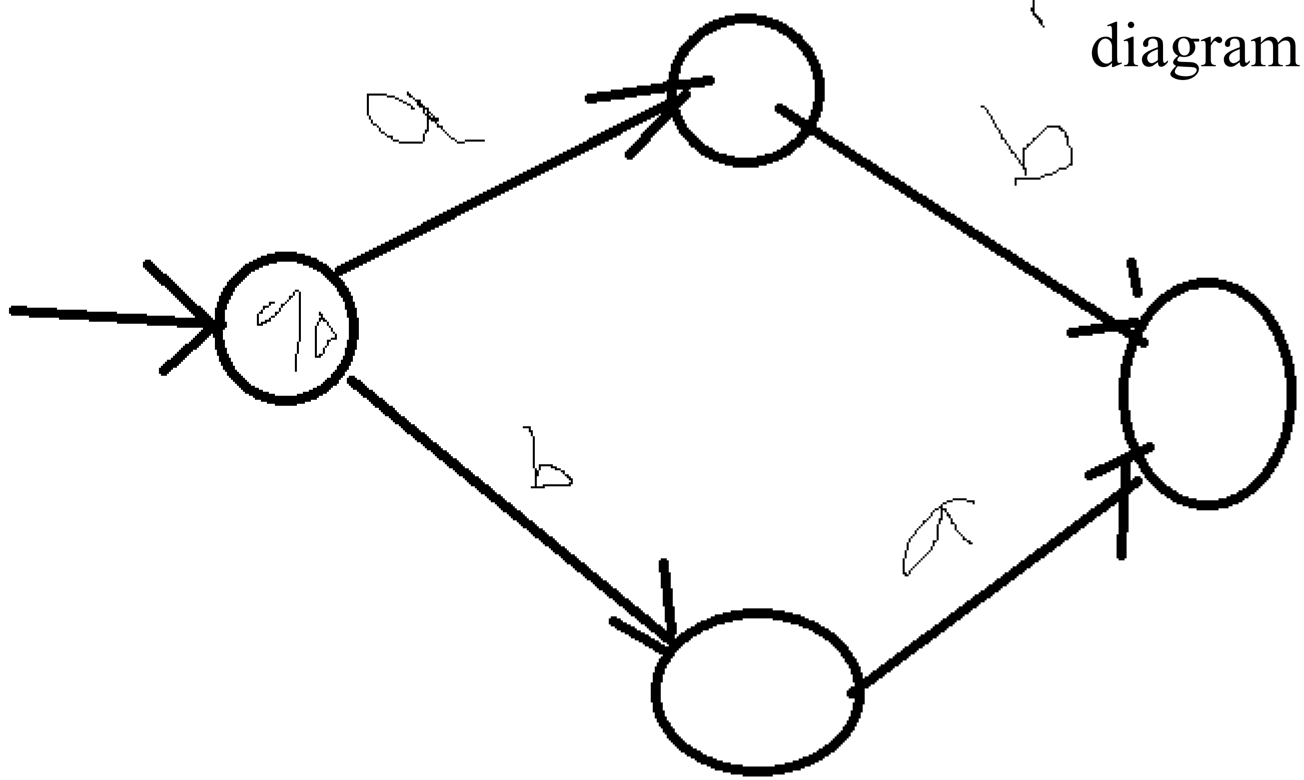
abba

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_1,q_2 \xrightarrow{b} q_1q_2 \xrightarrow{a} q_1$

String 'abba' is rejected by this DFA

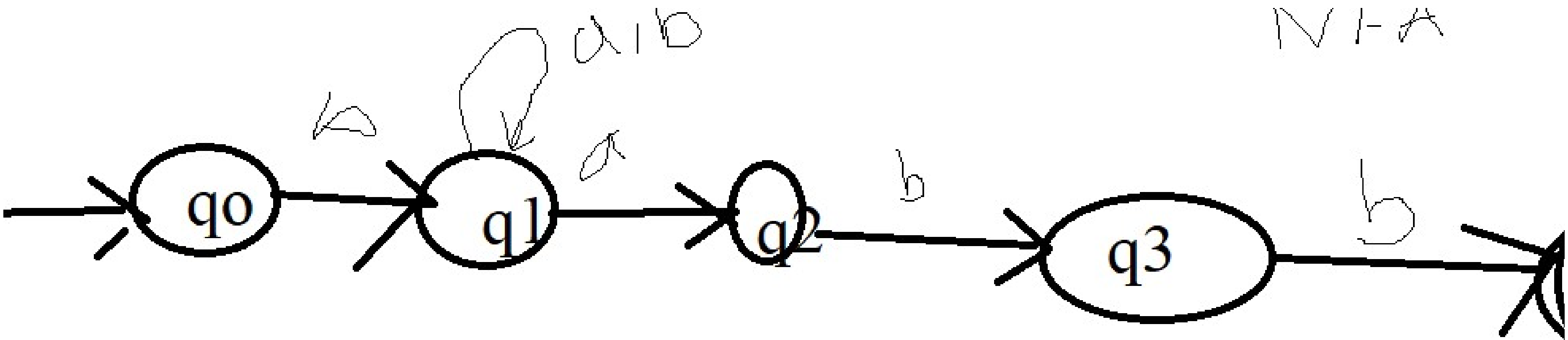
Question :Construct a nondeterministic finite automaton accepting {ab,ba}. Use it to construct a DFA accepting the same set of strings

Transition diagram of NFA



Question :Construct a nondeterministic finite automaton accepting the set of all strings over{a,b} starting with ‘b’ ending with ‘abb’. Use it to construct a DFA accepting the same set of all strings

L={babb,baabb, bbabb,bababb…..}



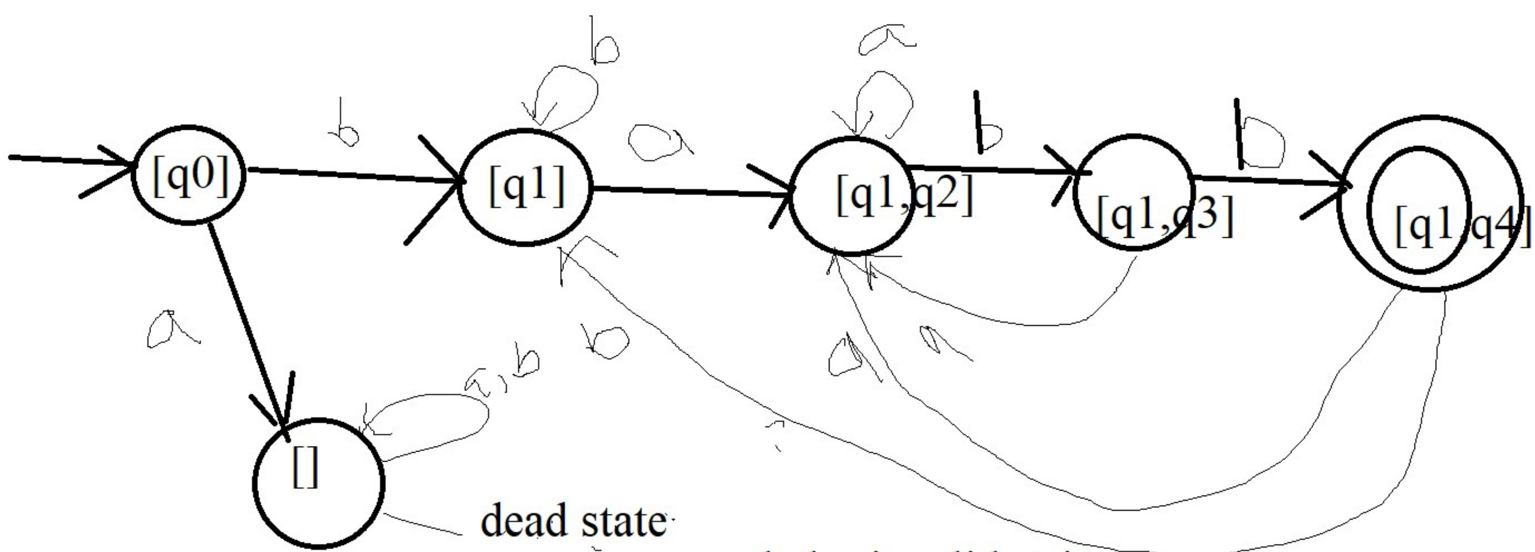
Transition Table of NFA

	a	b
q0		q1
q1	q1,q2	q1
q2		q3
q3		q4
q4		

Transition Table of DFA

	a	b
[q0]	[]	[q1]
[q1]	[q1,q2]	[q1]
[q1,q2]	[q1,q2]	[q1,q3]
[q1,q3]	[q1,q2]	[q1,q4]
[q1,q4]	[q1,q2]	[q1]
[]	[]	[]

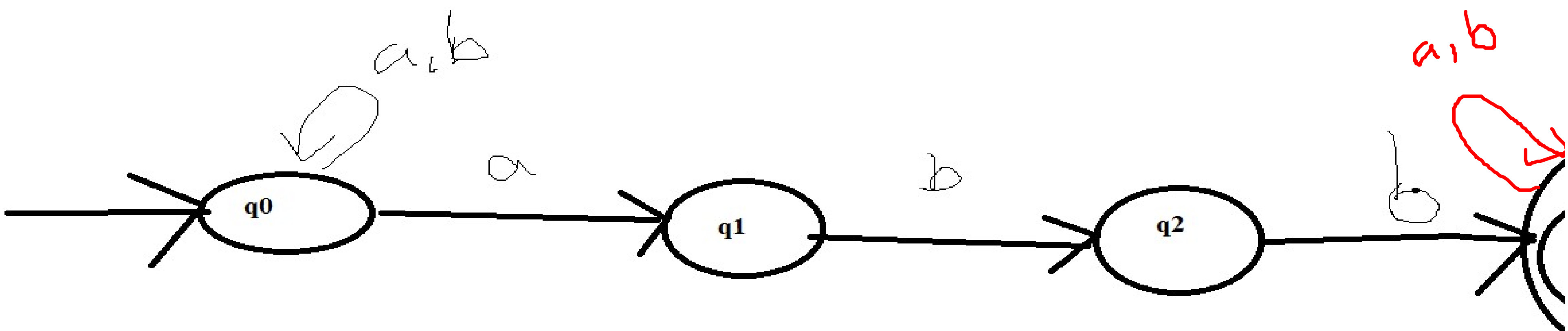
Transition diagram of DFA



baba-invalid string
q0->q1->q1,q2-> accepted as [q1,q2] is not
q1,q3-> q1, q2 a final state.

- Question: Construct a nondeterministic finite automaton accepting the set of all strings over{a,b} having 'abb' as a substring. Use it to construct a DFA accepting the same set of all strings L={ abb,aabb,abbb,aabbb,babba…..}

Transition diagram of NFA



	a	b
qo	q0,q1	q0
q1		q2
q2		q3
q3	q3	q3

	a	B
[q0]	[q0,q1]	[q0]
[q0,q1]	[q0,q1]	[q0,q2]
[q0,q2]	[q0,q1]	[q0,q3]
*[q0,q3]	[q0,q1,q3]	[q0,q3]
*[q0,q1,q3]	[q0,q1,q3]	[q0,q2,q3]
*[q0,q2,q3]	[q0,q1,q3]	[q0,q3]

- For minimization /optimization of DFA Please note
- We can merge two Non final non final states
- Or two final-final states **if** both these 2 states are having same transition on all the input symbols

Optimizing a DFA

	a	b
[q0]	[q0,q1]	[q0]
[qo,q1]	[q0,q1]	[q0,q2]
[q0,q2]	[q0,q1]	[q0,q3]
[q0,q3]	[q0,q1,q3]	[q0,q3]
[q0,q1,q3]	[q0,q1,q3]	[q0,q2,q3]
[q0,q2,q3]	[q0,q1,q3]	[q0,q3]

[q0,q3], and [q0,q2,q3] can be merged

After 1st optimization

	a	b
[q0]	[q0,q1]	[q0]
[qo,q1]	[q0,q1]	[q0,q2]
[q0,q2]	[q0,q1]	[q0,q3]
[q0,q3]	[q0,q1,q3]	[q0,q3]
[q0,q1,q3]	[q0,q1,q3]	[q0,q3]
[illegible]	[q0,q1,q3]	[q0,q3]

[q0,q3] and [q0,q1,q3] can be merged

After 2nd Optimization

	a	b
[q0]	[q0,q1]	[q0]
[qo,q1]	[q0,q1]	[q0,q2]
[q0,q2]	[q0,q1]	[q0,q3]
[q0,q3]	[q0,q[illegible]]	[q0,q3]

Final Transition diagram of DFA after optimization

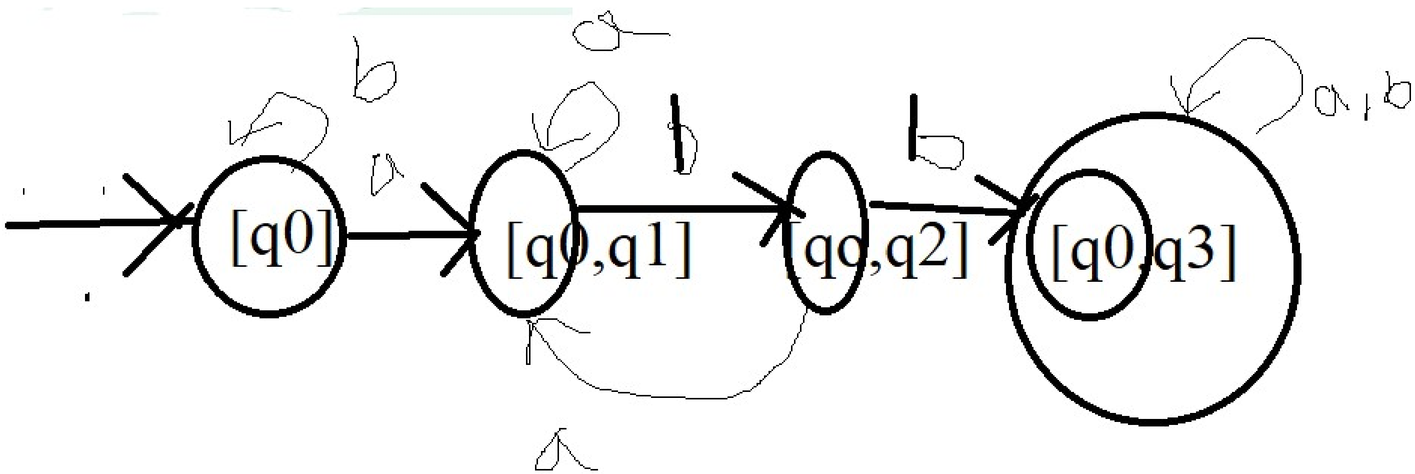
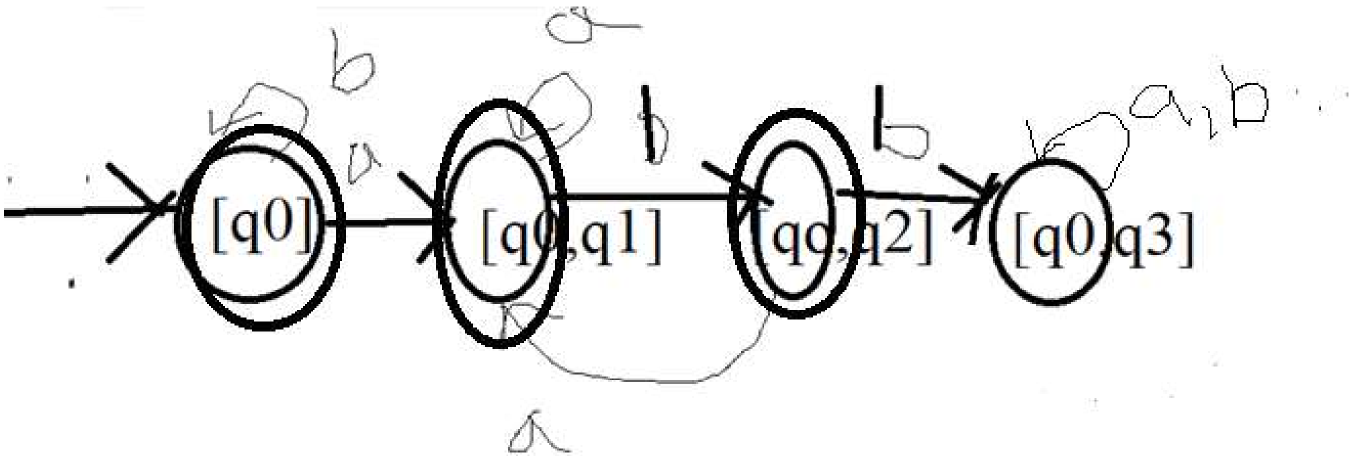


abb-valid-q0→q0q1→[q0,q2]--b--→[q0,q3]----accepted
bab-invalid==q0→[q0]-→[q0,q1]→[q0,q2]----rejected

DFA for a language which does not contain 'abb' as a substring



aba-valid string

$q_0 \xrightarrow{a} [q0,q1] \xrightarrow{b} [q0,q2] \xrightarrow{a} [q0,q1]$

accepted as [q0,q1] is a final state

abb-invalid

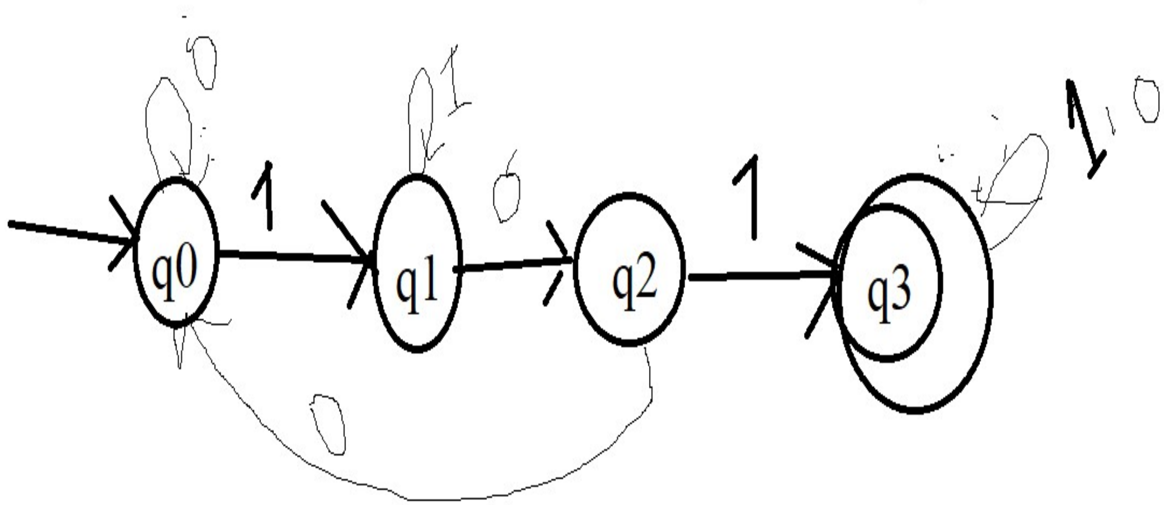
$q_0 \rightarrow [q0,q1] \xrightarrow{b} [q0q2] \xrightarrow{b} [q0q3]$

string 'abb' is rejected as [q0,q3] is non final state.

- For minimization of DFA Please note
- We can merge
- Non final non final states
- And final-final states **if** both these 2 states are going to the same state on same input syambol.

- Question: Construct a deterministic finite automaton accepting the set of all strings over{a,b}which contain '101' as a substring.

L={1101,0101,00101, 1101, 100101, 10101, 1010}



Question: Construct a deterministic finite automaton accepting the set of all strings over{a,b}which does not contain ‘101’ as a substring.