Theory of Computation

Shukla Banik

Siksha 'O' Anushandhan University

shuklabanik@soa.ac.in

October 9, 2020

Preliminaries

Symbol : Symbol is the smallest unit of a language.
Example: {a, b, c, 0, 1, 2, ...}

Alphabet : An alphabet is any finite set of symbols.
Example: $\Sigma = \{a, b, c\}$ is an alphabet set where 'a', 'b', and 'c' are symbols.

String: A string is a finite sequence of symbols taken from Σ.
Example: 'cab' is a valid string on the alphabet set $\Sigma = \{a, b, c\}$

Length of a String : It is the number of symbols present in a string. (Denoted by |S|).
Examples:
If S='cabad', |S|= 5
If |S|= 0, it is called an empty string (Denoted by λ or ϵ)
Q:: Create strings of length 2 over $\Sigma = \{a, b\}$

Preliminaries

Language :

A language is a collection of strings . It can be finite or infinite.

Example:
If the language takes all possible strings of length 2 over $\Sigma = \{a, b\}$, then L = { ab, bb, ba, bb}

Q1: Construct a language that takes all possible strings of length 3 over $\Sigma = \{a, b\}$, then L= ?

Q2: Construct a language that takes all possible strings starting with a and ending with a over $\Sigma = \{a, b\}$, then L= ?
Q3: Construct a language that takes all possible strings containing at least one a over $\Sigma = \{a, b\}$, then L= ?

Preliminaries

Q1: Construct a language that takes all possible strings of length 3 over $\Sigma = \{a, b\}$, then L= ?
L= {aaa, aab, aba, abb, baa, bab, bba, bbb}

Q2: Construct a language that takes all possible strings starting with a and ending with a over $\Sigma = \{a, b\}$, then L= ?

L= {aa, aaa, aba, abba, abaa, ababa, ...}

Q3: Construct a language that takes all possible strings containing at least one a over $\Sigma = \{a, b\}$, then L= ?

L= {a, aa, aaa, aba, ab, aba, abba, abbba, ...}

Preliminaries

Power of Σ:

Σ = {a, b}
Σ^0= {set of all strings with length '0'} = $\{\epsilon\}$
Σ^1= {set of all strings with length '1'} = {a, b}
Σ^2= {set of all strings with length '2'} = {aa, ab, ba, bb}
Σ^3= {set of all strings with length '3'} = {aaa, aab, aba, abb, baa, bab, bba, bbb}

.

.

.

Σ^* = {set of all possible strings of all possible lengths over Σ } = $(a+b)^*$
= infinite language
Σ^* is called the Kleene star.

Preliminaries

Kleene star:
The Kleene star, Σ^*, is a unary operator on a set of symbols or strings, Σ, that gives the infinite set of all possible strings of all possible lengths over Σ including ϵ

Representation:

$\Sigma^* = \Sigma^0$ U Σ^1 U Σ^2U ..... where Σ^p is the set of all possible strings of length p.

Example:

If $\Sigma = \{$a, b$\}$,
$\Sigma^* = \{\epsilon$, a, b, aa, ab, ba, bb, ...$\}$

Preliminaries

Kleene Positive Closure / Plus

The set Σ^+, is the infinite set of all possible strings of all possible lengths over Σ excluding ϵ

Representation:

$\Sigma^+ = \Sigma^1$ U Σ^2 U Σ^3 U .....
$\Sigma^+ = \Sigma^*$ - ϵ

Example:

If $\Sigma = \{a, b\}$,
$\Sigma^+ = \{a, b, aa, ab, ba, bb, ...\}$

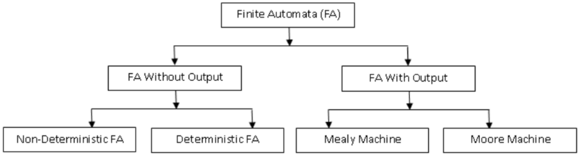
Introduction to Finite Automata

An automaton is an algorithm or program that automatically recognizes if a particular string belongs to the language or not, by checking the grammar of the string. In other words, an automaton is an abstract computing device (or machine).

$$\begin{tikzcd}
\text{Grammar} \arrow[r, "\text{(Generates)}"] & \text{Language} & \text{Automata} \arrow[l, "\text{(Recognizes)}"']
\end{tikzcd}$$

There are different varieties of such abstract machines (also called models of computation) which can be defined mathematically as: Finite Automaton (FA), PushDown Automaton (PDA), Linear Bounded Automaton (LBA), and Turing Machine (TM).

Introduction to Finite Automata



Introduction to Finite Automata

An automaton with a finite number of states is called a Finite Automaton (FA)
The formal definition says that a finite automaton is a list of 5 objects: set of states, input alphabet, rules for moving, start state, and accept states. In mathematical language, a list of five elements is often called a 5-tuple.

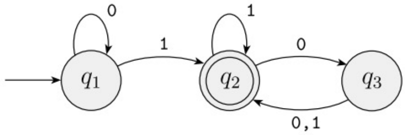
Formal Definition of a Finite Automaton

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

1. Q is a finite set called the **states**,
2. Σ is a finite set called the **alphabet**,
3. $\delta : Q \times \Sigma \to Q$ is the **transition function**,
4. $q_0 \in Q$ is the **start state**, and
5. $F \subseteq Q$ is the set of **accept states**.

Introduction to Finite Automata

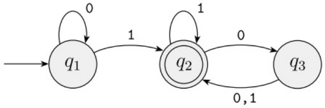
In beginning to describe the mathematical theory of finite automata, we do so in the abstract, without reference to any particular application. The following figure depicts a finite automaton called M1.



From the state diagram we observe that:

- 3 states represented by circles and labelled as q1, q2 and q3.
- Inputs are 0 and 1.
- The start state q1 is represented as an arrow pointing from nowhere.
- Accept state or final state is represented by a double circle and labelled as q2.
- Arrows from one state to another are called transitions, which represents the rules to move from one state to another.

Introduction to Finite Automata


$$\begin{tikzcd} {} \arrow[r] & q_1 \arrow[loop above, "0"] \arrow[r, bend left, "1"] & q_2 \arrow[loop above, "1"] \arrow[r, bend left, "0"] & q_3 \arrow[l, bend left, "{0,1}"] \end{tikzcd}$$

We can describe the automaton shown above formally by writing $M_1 = (Q, \Sigma, \delta, q_1, F)$, where

1. $Q = \{q_1, q_2, q_3\}$,
2. $\Sigma = \{0, 1\}$,
3. δ is described as

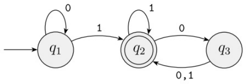
States	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

4. q_1 is the start state, and
5. $F = \{q_2\}$

Introduction to Finite Automata

If A is the set of all strings that machine M accepts, we say that A is the language of machine M and write $L(M) = A$. We say that M recognizes A or that M accepts A.

Consider the input string 1101 into the machine M_1 shown below, the processing proceeds as follows:



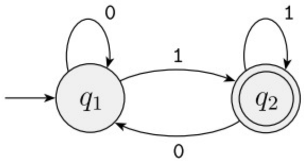
→ Start in state q1.
→ Read 1, follow transition from q1 to q2.
→ Read 1, follow transition from q2 to q2.
→ Read 0, follow transition from q2 to q3.
→ Read 1, follow transition from q3 to q2.
→ Accept because M1 is in an accept state q2 at the end of the input.

Introduction to Finite Automata

From the above, we can analyze that, M_1 accepts any string that ends with a 1, as it goes to its accept state q2 whenever it reads the symbol 1. In addition, it accepts any string that ends with an even number of 0's following the last 1. Hence, we can conclude that, if $A = \{w \mid w$ contains at least one 1 and an even number of 0's follow the last 1$\}$. Then $L(M_1) = A$, or equivalently, $M1$ recognizes A.

Introduction to Finite Automata

State diagram of the two-state finite automaton M_2

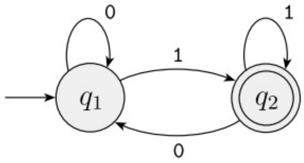


In the formal description, M_2 is $(\{q_1, q_2\}, \{0,1\}, \delta, q_1, \{q_2\})$. The transition function δ is

States	0	1
q_1	q_1	q_2
q_2	q_1	q_2

Introduction to Finite Automata

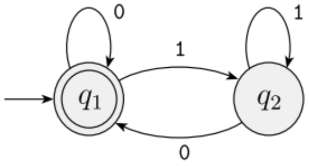
State diagram of the two-state finite automaton M_2


$$\begin{tikzcd}
\arrow[r] & q_1 \arrow[loop above, "0"] \arrow[r, bend left, "1"] & q_2 \arrow[loop above, "1"] \arrow[l, bend left, "0"]
\end{tikzcd}$$

On the sample string 1101, the machine M_2 starts in its start state q_1 and proceeds first to state q_2 after reading the first 1, and then to states q_2, q_1, and q_2 after reading 1, 0, and 1 respectively. Finally the string is accepted because q_2 is an accept state. But string 110 leaves M_2 in state q_1, so it is rejected. After trying a few more examples, you will find that M_2 accepts all strings that end in a 1. Thus $L(M_2) = \{w|w \text{ ends in a } 1\}$.

Introduction to Finite Automata

State diagram of the two-state finite automaton M_3



Machine M_3 is similar to M_2 except for the location of the accept state. As the start state is also an accept state, M_3 accepts the empty string ϵ, and comes to an end; so if the start state is an accept state, ϵ is accepted. In addition to the empty string, this machine accepts any string ending with a 0. Hence, $M_3 = \{w \mid w \text{ is the empty string } \epsilon \text{ or ends in a } 0\}$.

Formal Definition of Computation

Here we will discuss about computations in finite automata. We already have an informal idea of the way it computes, and we now formalize it mathematically.

Let $M = (Q, \Sigma, \delta, q_0, F)$, be a finite automaton and let $w = w_1 w_2 ... w_n$ be a string where each w_i is a member of the alphabet Σ. Then M accepts w if a sequence of states $r_0, r_1, ..., r_n$ in Q exists with the following three conditions:

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, ..., n-1$, and
3. $r_n \in F$.

Condition 1 says that the machine starts in the start state. Condition 2 says that the machine goes from state to state according to the transition function. Condition 3 says that the machine accepts its input if it ends up in an accept state. We say that M recognizes language A if $A = \{w \mid M \text{ accepts } w\}$. A language is called a regular language if some finite automaton recognizes it.

Designing Finite Automata

Example 1: Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length 1 i.e. $\mid w \mid= 1$

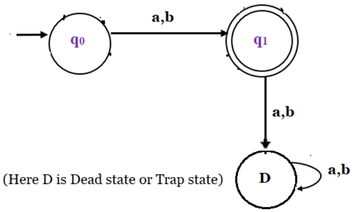
Explanation:

1. We have to create DFA which will accept string of unit length on alphabet $\{a, b\}$.
2. You just take 2 states such that 1 length string can be accepted. The second state will be final state.
3. But if we came up with a string of length 2, it should not be accepted in our DFA.
4. So we have to attach one more state to take care of string of length greater than 1 called $'D'$.
5. State $'D'$ is called dead state.

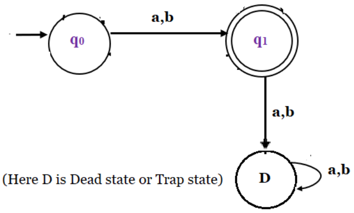
Designing Finite Automata

Example 1: Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length 1 i.e. $\mid w \mid = 1$

DFA can be described as $(\{q_0, q_1, D\}, \{a, b\}, \delta, q_0, \{q_1\})$.
$L =$\{All the strings of length 1\}
$L = \{a, b\}$



Designing Finite Automata



(Here D is Dead state or Trap state)

The transition table is:

States	a	b
q_0	q_1	q_1
q_1	D	D
D	D	D

The transition function is:

$\delta(q_0, a) = (q_1)$, $\delta(q_0, b) = (q_1)$
$\delta(q_1, a) = (D)$, $\delta(q_1, b) = (D)$

Designing Finite Automata

Example 2:
Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length 2 i.e. $\mid w \mid = 2$

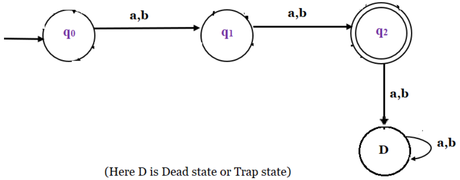
Explanation:

1. We have to create DFA which will accept string of length 2 on alphabet $\{a, b\}$.
2. To take 2 length string we need 3 states.
3. The third state will be final state.
4. But if we came up with a string of length 3, it should not be accepted in our DFA.
5. So we have to attach one more state to take care of string length greater than 2 called 'D'.
6. State 'D' is called dead state.

A finite automaton that accepts the strings of length exactly n can be constructed with $n + 2$ number of states.

Designing Finite Automata

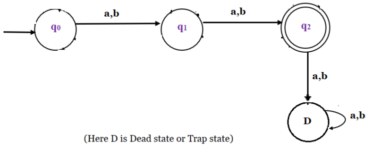
Example 2:
Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length 2 i.e. $| w |= 2$ is shown below where,
DFA can be described as $(\{q_0, q_1, q_2, D\}, \{a, b\}, \delta, q_0, \{q_2\})$.
$L = \{$All the strings of length 2$\}$
$L = \{aa, ab, ba, bb\}$



(Here D is Dead state or Trap state)

Designing Finite Automata

Example 2:



(Here D is Dead state or Trap state)

The transition table is:

States	a	b
q_0	q_1	q_1
q_1	q_2	q_2
q_2	D	D
D	D	D

The transition function is:

$\delta(q_0, a) = (q_1)$, $\delta(q_0, b) = (q_1)$
$\delta(q_1, a) = (q_2)$, $\delta(q_1, b) = (q_2)$
$\delta(q_2, a) = (D)$, $\delta(q_2, b) = (D)$
$\delta(D, a) = (D)$, $\delta(D, b) = (D)$

Designing Finite Automata

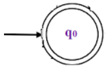
Example 3:
Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $\mid w \mid \leq 2$

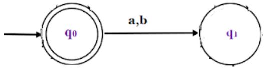
Explanation:

1. We have to create DFA which will accept string of length at most 2 i.e. of length 0, 1 and 2 on alphabet $\{a, b\}$. Hence, the language content $= \{\in, a, b, ab, ba, bb, aa\}$.
2. To accept 2 length string we need 3 states.
3. The first state accepts strings of length 0, the second state accepts the strings of length 1 and the third state accepts the strings of length 2. So all the three states will be the final states.
4. But if we came up with a string of length 3, it should not be accepted in our DFA. ?
5. So we have to attach one more state to take care of string length greater than 2 called 'q_3'.
6. State 'q_3' is called dead state.

Designing Finite Automata

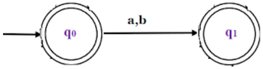
Example 3: Construction of a DFA,that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $\mid w \mid \leq 2$ is shown below where,
Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $\mid w \mid \leq 2$ is shown below where,
DFA can be described as $(\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_2\})$.
$L = \{$All the strings of length at most 2$\}$
$L = \{\epsilon, a, b, aa, ab, ba, bb\}$

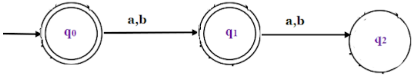




Designing Finite Automata

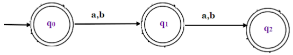
Example 3: Construction of a DFA,that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $| w | \leq 2$ is shown below where,
Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $| w | \leq 2$ is shown below where,
DFA can be described as $(\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_2\})$.
$L = \{$All the strings of length at most 2$\}$
$L = \{\epsilon, a, b, aa, ab, ba, bb\}$

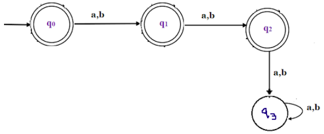




Designing Finite Automata

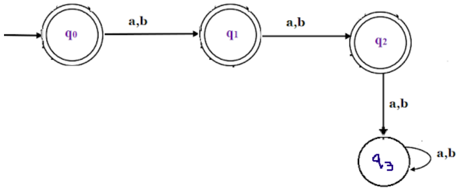
Example 3: Construction of a DFA,that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $\mid w \mid \leq 2$ is shown below where, Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length at most 2 i.e. $\mid w \mid \leq 2$ is shown below where,
DFA can be described as $(\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_2\})$.
$L = \{$All the strings of length at most 2$\}$
$L = \{\epsilon, a, b, aa, ab, ba, bb\}$





Designing Finite Automata

Example 3:



The transition table is:

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

The transition function is:

$$\delta(q_0, a) = (q_1)\ ,\ \delta(q_0, b) = (q_1)$$
$$\delta(q_1, a) = (q_2)\ ,\ \delta(q_1, b) = (q_2)$$
$$\delta(q_2, a) = (q_3)\ ,\ \delta(q_2, b) = (q_3)$$
$$\delta(q_3, a) = (q_3)\ ,\ \delta(q_3, b) = (q_3)$$

Designing Finite Automata

Example 4:
Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length at least 2 i.e. $\mid w \mid \geq 2$

Explanation:

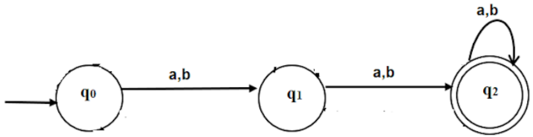
1. We have to create DFA which will accept string of length at least 2 i.e. of length ≥ 2 on alphabet $\{a, b\}$.
2. To take 2 length string we need 3 states.
3. The third state will be the final state.
4. If input comes over the third state q_2 then it will go to q_2 itself to accept strings greater than 2.
5. Now if string < 2 will not reach final state, so will not be accepted.

A finite automaton that accepts the strings of length at least n can be constructed with $n + 1$ number of states.

Designing Finite Automata

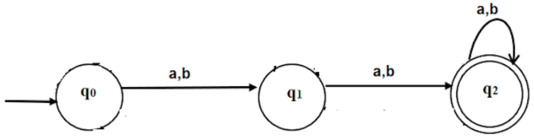
Example 4: Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length at least 2 i.e. $\mid w \mid \leq 2$ is shown below where,

DFA can be described as $(\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$.
$L = \{\text{All the strings of length at least 2}\}$
$L = \{aa, ab, ba, bb, aaa, \ldots, bbb, \ldots\}$



Designing Finite Automata

Example 4:



The transition table is:

States	a	b
q_0	q_1	q_1
q_1	q_2	q_2
q_2	q_2	q_2

The transition function is:

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

Designing Finite Automata

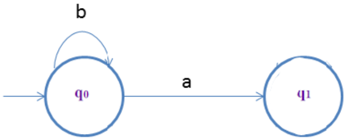
Example 5:
Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ which ends with 'a',

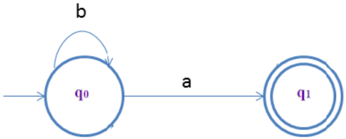
Explanation:

1. We have to create DFA that accepts set of strings which ends with 'a'.
2. We will first start accepting 'b' on start state itself and then on 'a' we will go to final state.
3. And on final state we do not care for 'a''s so we have to put a self-loop.
4. On final state if 'b' comes then we will redirect it start state so that any string which does not end with 'a' should not get accepted.

Designing Finite Automata

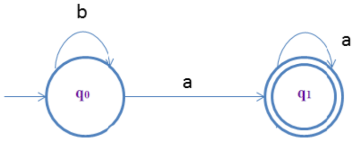
Example 5: Construction of a DFA,that accepts set of strings over $\Sigma = \{a, b\}$ which ends with 'a' is shown below where, DFA can be described as $(\{q_0, q_1\}, \{a, b\}, \delta, q_0, \{q_1\})$.
$L = \{$All the strings which ends with 'a'$\}$
$L = \{aa, ba, aba, abba, baba, \ldots, aaabbba, \ldots\}$

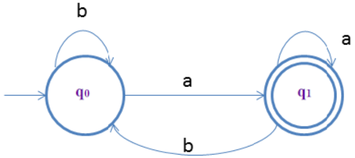




Designing Finite Automata

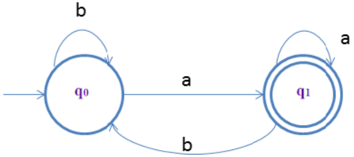
Example 5: Construction of a DFA,that accepts set of strings over $\Sigma = \{a, b\}$ which ends with 'a' is shown below where,
DFA can be described as $(\{q_0, q_1\}, \{a, b\}, \delta, q_0, \{q_1\})$.
$L = \{$All the strings which ends with 'a'$\}$
$L = \{aa, ba, aba, abba, baba, \ldots, aaabbba, \ldots\}$





Designing Finite Automata

Example 5:



The transition table is:

States	a	b
$\rightarrow q_0$	q_1	q_0
q_1*	q_1	q_0

The transition function is:

$$\delta(q_0, a) = (q_1)\ ,\ \delta(q_0, b) = (q_0)$$
$$\delta(q_1, a) = (q_1)\ ,\ \delta(q_1, b) = (q_0)$$

Designing Finite Automata

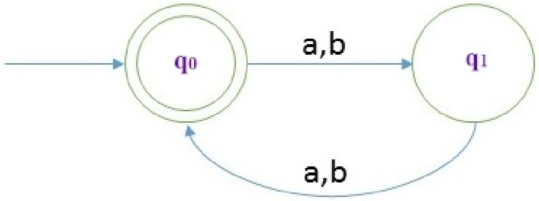
Example 6:
Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of even length i.e. $\mid w \mid mod 2 = 0$

$L = ?$

Designing Finite Automata

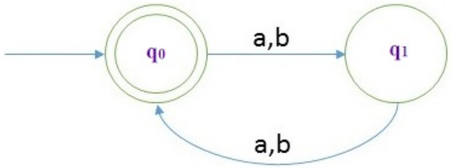
Example 6: Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of even length i.e. $\mid w \mid mod 2 = 0$ is shown below where,

DFA can be described as $(\{q_0, q_1\}, \{a, b\}, \delta, q_0, \{q_0\})$.
$L = \{\text{All the strings of even length}\}$
$L = \{\epsilon, aa, ab, ba, bb, aaab, \ldots, abbbba, \ldots\}$



Designing Finite Automata

Example 6:



The transition table is:

States	a	b
$\rightarrow q_0 *$	q_1	q_1
q_1	q_0	q_0

The transition function is:

$$\delta(q_0, a) = (q_1)\ ,\ \delta(q_0, b) = (q_1)$$
$$\delta(q_1, a) = (q_0)\ ,\ \delta(q_1, b) = (q_0)$$

Designing Finite Automata

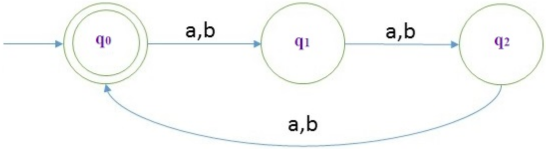
Example 7:
Construct a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length divisible by 3 i.e. $\mid w \mid \bmod 3 = 0$

$L = ?$

Designing Finite Automata

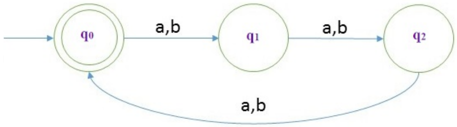
Example 7: Construction of a DFA, that accepts set of strings over $\Sigma = \{a, b\}$ of length divisible by 3 i.e. $\mid w \mid$ mod $3 = 0$ is shown below where,

DFA can be described as $(\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_0\})$.
$L = \{$All the strings of length divisible by 3$\}$
$L = \{\epsilon, aaa, aba, baa, bbb, \ldots, abbbba, \ldots\}$



Designing Finite Automata

Example 7:



The transition table is:

States	a	b
q_0	q_1	q_1
q_1	q_2	q_2
q_2	q_0	q_0

The transition function is:

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