Theory of Automata

Theory of automata is a theoretical branch of computer science and mathematical. It is the study of abstract machines and the computation problems that can be solved using these machines. The abstract machine is called the automata. The main motivation behind developing the automata theory was to develop methods to describe and analyze the dynamic behavior of discrete systems.
There are the basic terminologies that are important and frequently used in automata:

Symbols:

Symbols are an entity or individual objects, which can be any letter, alphabet or any picture.
Example:
1, a, b, #

Alphabets:

Alphabets are a finite set of symbols. It is denoted by ∑.
Examples:
∑ = {a, b}
∑ = {A, B, C, D}
∑ = {0, 1, 2}
∑ = {0, 1, ..., 5]
∑ = {#, β, Δ}

String:

It is a finite collection of symbols from the alphabet. The string is denoted by w.
Example 1:
If ∑ = {a, b}, various string that can be generated from ∑ are {ab, aa, aaa, bb, bbb, ba, aba.....}.
A string with zero occurrences of symbols is known as an empty string. It is represented by ε.
The number of symbols in a string w is called the length of a string. It is denoted by |w|.
Example 2:
w = 010
Number of Sting |w| = 3

Language:

A language is a collection of appropriate string. A language which is formed over Σ can be **Finite** or **Infinite**.
Example: 1
L1 = {Set of string of length 2}
= {aa, bb, ba, bb} **Finite Language**
Example: 2
L2 = {Set of all strings starts with 'a'}
= {a, aa, aaa, abb, abbb, ababb} **Infinite Language**

Features of Finite Automata

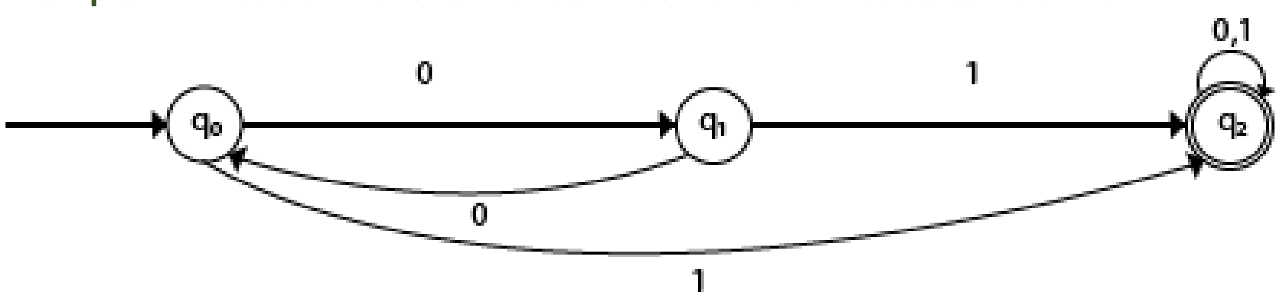
- **Input:** Set of symbols or characters provided to the machine.
- **Output:** Accept or reject based on the input pattern.
- **States of Automata:** The conditions or configurations of the machine.
- **State Relation:** The transitions between states.
- **Output Relation:** Based on the final state, the output decision is made.

❖ Formal Definition of Finite Automata/Transition Diagram/State diagram

A finite automaton can be defined as a tuple:
{ Q, Σ, q, F, δ }, where:

- Q: Finite set of states
- Σ: Set of input symbols
- q: Initial state
- F: Set of final states
- δ: Transition function

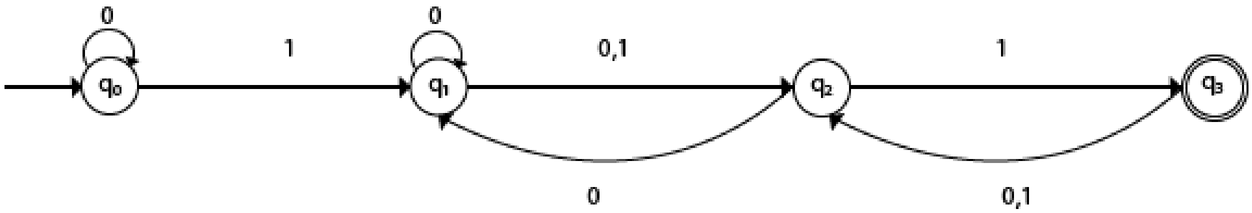
Example 1: Write down the formal definition of this finite state machine



- Finite set of states Q: q0, q1,q2
- Set of input symbols Σ: 0,1
- Initial state q: q0
- Set of final states F: q2
- Transition function δ: Q X Σ --> Q.

State	0	1
→q0	q1	q2
q1	q0	q2
*q2	q2	q2

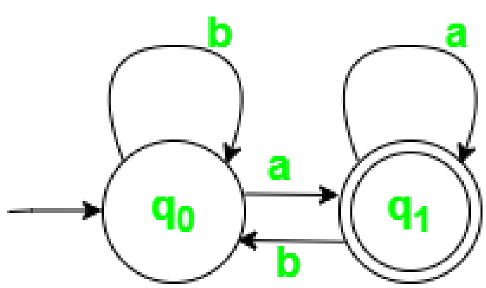
Example 2: Write down the formal definition of this finite state machine



- Finite set of states Q: q0, q1, q2, q3
- Set of input symbols Σ: 0,1
- Initial state q: q0
- Set of final states F: q3
- Transition function δ: Q X Σ --> Q

State	**0**	**1**
→q0	q0	q1
q1	q1, q2	q2
q2	q1	q3
*q3	q2	q2

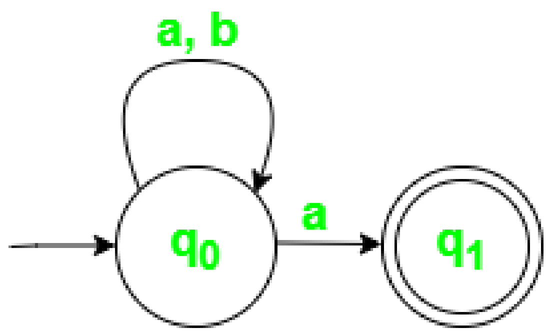
Example 3: Write down the formal definition of this finite state machine



Finite set of states Q = {q0, q1},
Set of input symbols Σ = {a, b},
Initial state q = q0
Set of final states F = {q1}
Transition function δ: Q X Σ --> Q.

State\Symbol	**a**	**b**
q0	q1	q0
*q1	q1	q0

Example 4: Write down the formal definition of this finite state machine



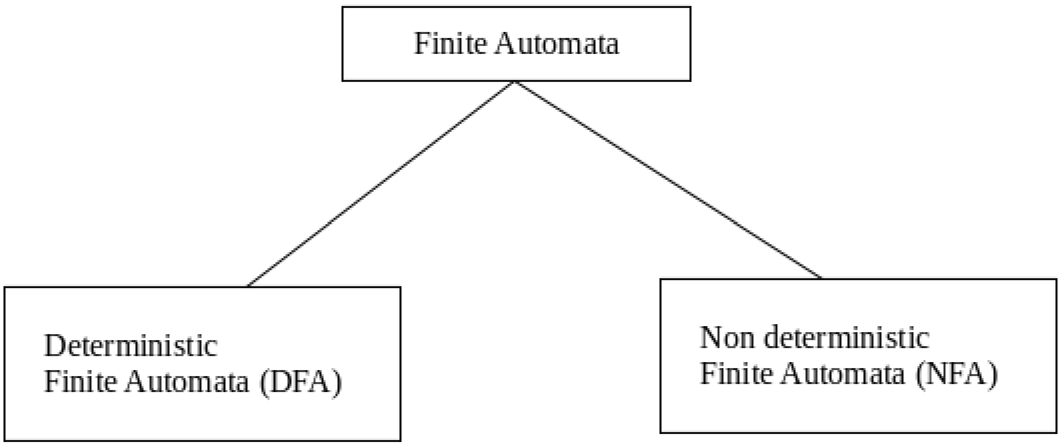
Q = {q0, q1},
Σ = {a, b},
q = q0
F = {q1}
δ: Q X Σ --> Q.

State	**a**	**b**
q0	{q0,q1}	q0
*q1	ϕ	ϕ

Types of Automata:

There are two types of finite automata:

1. DFA(deterministic finite automata)
2. NFA(non-deterministic finite automata)



1. DFA

DFA refers to deterministic finite automata. Deterministic refers to the uniqueness of the computation. In the DFA, the machine goes to one state only for a particular input character. DFA does not accept the null move.

2. NFA

NFA stands for non-deterministic finite automata. It is used to transmit any number of states for a particular input. It can accept the null move.

Some important points about DFA and NFA:

1. Every DFA is NFA, but NFA is not DFA.
2. There can be multiple final states in both NFA and DFA.
3. DFA is used in Lexical Analysis in Compiler.
4. NFA is more of a theoretical concept.