Welcome

Theory of Automata

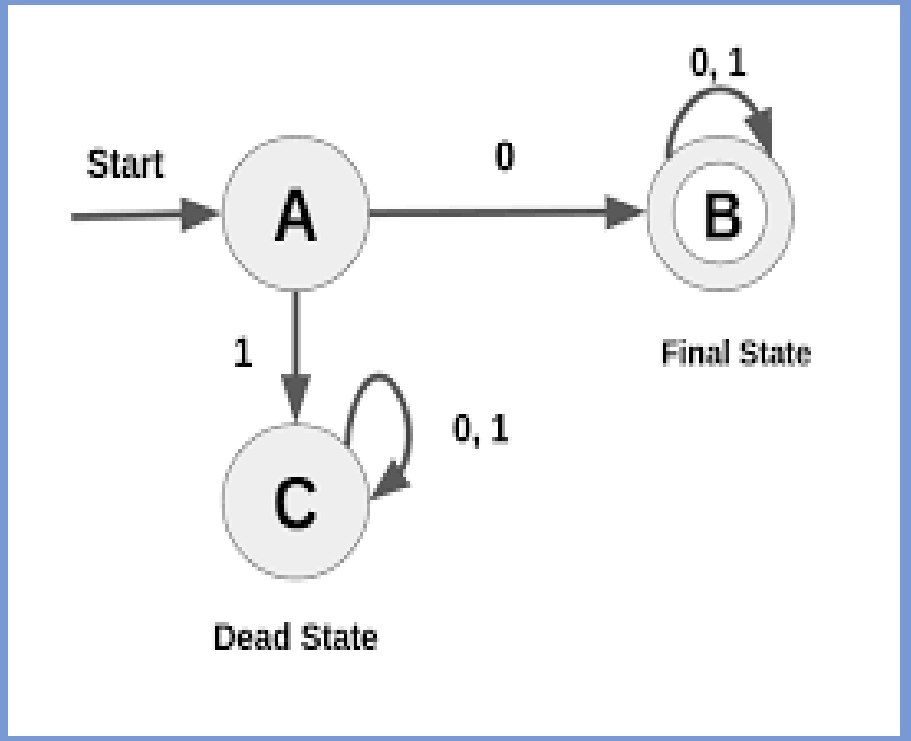
Theory of Automata

- Automata is the plural word of Automaton.
- 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.**

Theory of Automata

- This automaton consists of states and transitions. The **State** is represented by **circles**, and the **Transitions** is represented by **arrows**.

- Automata is the kind of machine which takes some string as input and this input goes through a finite number of states and may enter in the final state.



Applications of Theory of Automata

- **1. Compiler Design**
- **Lexical Analysis:** Regular expressions and finite automata are used to identify tokens in a programming language (like keywords, identifiers, etc.).
- **Syntax Analysis:** Context-free grammars and pushdown automata help in building parsers, which check the syntactic correctness of source code.

Applications of Automata Theory

2. Software for Text Processing

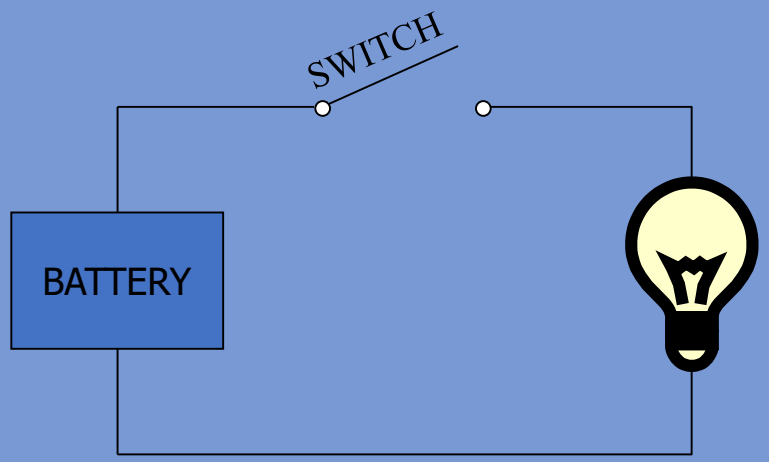
- Text editors, search tools (like grep), and command-line filters often use regular expressions and finite automata to search and manipulate text efficiently.
- Spell checkers and syntax highlighters also rely on finite automata.

Applications of Automata Theory

3.Database Query Processing

•Regular expressions and automata are used in query languages (like SQL's LIKE operator) to match patterns.

A simple computer



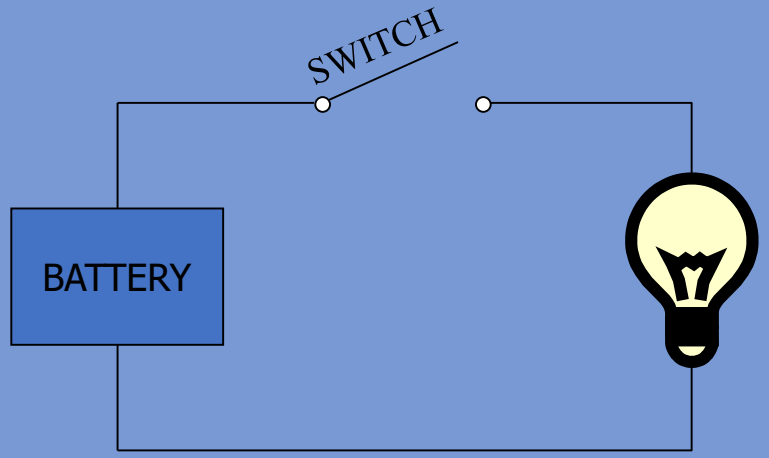
input: switch

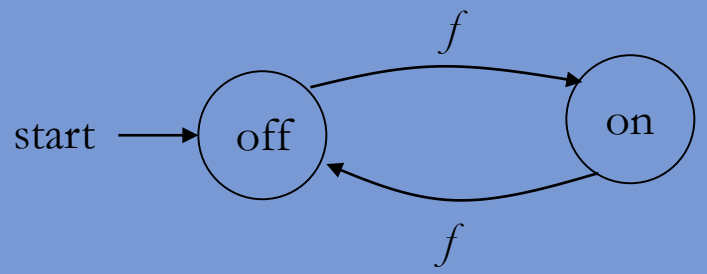
output: light bulb

actions: flip switch

states: on, off

A simple “computer”





input: switch

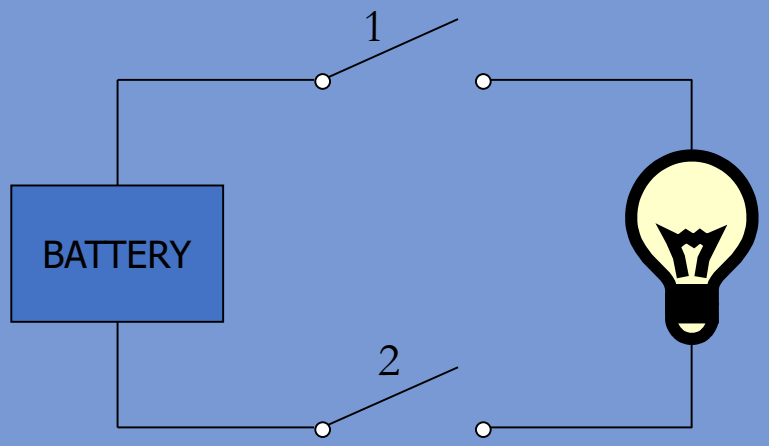
output: light bulb

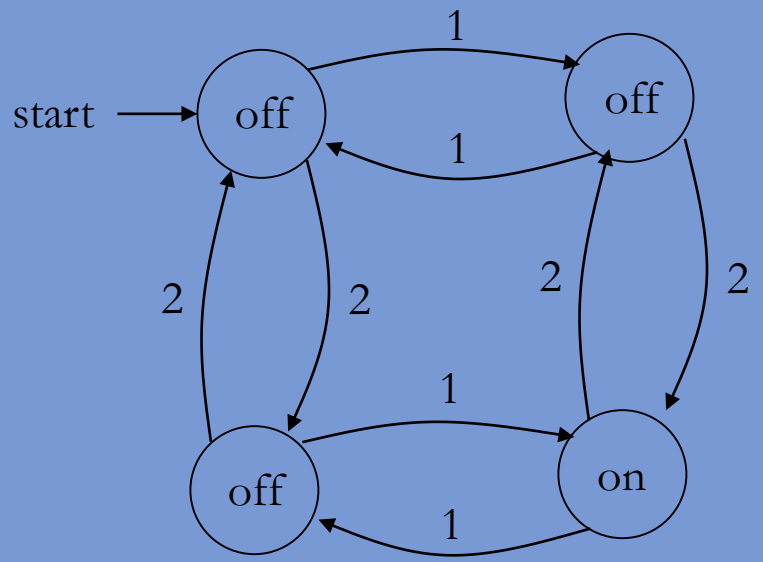
actions: f for “flip switch”

states: on, off

bulb is on if and only if there was an odd number of flips

Another "computer"





inputs: switches 1 and 2

actions: 1 for "flip switch 1"
2 for "flip switch 2"

states: on, off

bulb is on if and only if
both switches were flipped
an odd number of times

Basic Terminologies

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

Basic Terminologies

- **Alphabets**:
  - Alphabets are a finite set of symbols. It is denoted by ∑.
  - Examples:

1. ∑ = {a, b}
2. ∑ = {A, B, C, D}
3. ∑ = {0, 1, 2}
4. ∑ = {0, 1, ..., 5]
5. ∑ = {#, β, Δ}

Basic Terminologies

- **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|.

Basic Terminologies

- Example 2:

1. w = 010
2. Number of Sting |w| = 3

NOTE:

EMPTY STRING or NULL STRING

- Sometimes a string with no symbol at all is used, denoted by (Small Greek letter Lambda) λ or (Capital Greek letter Lambda) Λ, is called an empty string or null string.

- The capital lambda will mostly be used to denote the empty string, in further discussion.

NOTE:

All words are strings, but not all strings are words

Basic Terminologies

- **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***

Words

- Definition:

  Words are strings belonging to some language.

- Example:

  If Σ= {a} then a language L can be defined as L={a^n : n=1,2,3,…..} or L={a,aa,aaa,….}

  Here a,aa,… are the words of L

Valid/In-valid alphabets

- While defining an alphabet, an alphabet may contain letters consisting of group of symbols for example Σ_1= {B, aB, bab, d}.

- Now consider an alphabet

  Σ_2= {B, Ba, bab, d} and a string BababB

Valid/In-valid alphabets

- This BababB (string) can be tokenized in two different ways
  - (Ba), (bab), (B)
  - (B), (abab), (B)

- Which shows that the second group cannot be identified as a string, defined over

  Σ_2= {B, Ba, bab, d}

Valid/In-valid alphabets

- As when this string is scanned by the compiler (Lexical Analyzer), first symbol B is identified as a letter belonging to Σ, while for the second letter the lexical analyzer would not be able to identify, so while defining an alphabet it should be kept in mind that ambiguity should not be created.

Conclusion

- Σ_1= {B, aB, bab, d}
- Σ_2= {B, Ba, bab, d}

- Σ_1 is a valid alphabet while Σ_2 is an in-valid alphabet.

Reverse of a String

- Definition:

  The reverse of a string s denoted by Rev(s) or s^r, is obtained by writing the letters of s in reverse order.

- Example:

  If s = abc is a string defined over

  Σ={a,b,c} then

  Rev(s) or s^r = cba

Reverse of a String

- Example:

  Σ= {B, aB, bab, d}

  s=BaBbabBd

  Tokenizing=(B) (aB) (bab) (B) (d)

  Rev(s)=dBbabaBB

Defining Languages

- The languages can be defined in different ways , such as
  1. Descriptive definition,
  2. Recursive definition,
  3. using Regular Expressions(RE) and
  4. using Finite Automaton(FA) etc.

- Descriptive definition of language:

  The language is defined, describing the conditions imposed on its words.

- Example:

  The language L of strings of odd length, defined over Σ={a}, can be written as

  L={a, aaa, aaaaa,.....}

- Example:

  The language L of strings that does not start with a, defined over Σ={a,b,c}, can be written as

  L={b, c, ba, bb, bc, ca, cb, cc, ...}

- Example:

  The language L of strings of length 2, defined over Σ={0,1,2}, can be written as

  L={00, 01, 02,10, 11,12,20,21,22}

- Example:

  The language L of strings ending in 0, defined over Σ ={0,1}, can be written as

  L={0,00,10,000,010,100,110,…}

- Example:

  The language **EQUAL**, of strings with number of a's equal to number of b's, defined over Σ={a,b}, can be written as

  {Λ ,ab,aabb,abab,baba,abba,…}

- Example:

  The language **EVEN-EVEN**, of strings with even number of a's and even number of b's, defined over Σ={a,b}, can be written as

  {Λ, aa, bb, aaaa,aabb,abab, abba, baab, baba, bbaa, bbbb,…}

- Example:

  The language **INTEGER**, of strings defined over Σ={-,0,1,2,3,4,5,6,7,8,9}, can be written as

  INTEGER = {…,-2,-1,0,1,2,…}

- Example:

  The language **EVEN**, of stings defined over

  Σ={-,0,1,2,3,4,5,6,7,8,9}, can be written as EVEN = { …,-4,-2,0,2,4,…}

- Example:

  The language $\{a^n b^n\}$, of strings defined over $\Sigma=\{a,b\}$, as $\{a^n b^n : n=1,2,3,\ldots\}$, can be written as

  $\{ab, aabb, aaabbb, aaaabbbb, \ldots\}$

- Example:

  The language $\{a^n b^n a^n\}$, of strings defined over $\Sigma=\{a,b\}$, as $\{a^n b^n a^n : n=1,2,3,\ldots\}$, can be written as

  $\{aba, aabbaa, aaabbbaaa, aaaabbbbaaaa, \ldots$

- Example:

  The language **factorial**, of strings defined over Σ={1,2,3,4,5,6,7,8,9} *i.e.* {1,2,6,24,120,…}

- Example:

  The language **FACTORIAL**, of strings defined over Σ={a}, as $\{a^{n!} : n=1,2,3,\ldots\}$, can be written as {a,aa,aaaaaa,…}.

  It is to be noted that the language FACTORIAL can be defined over any single letter alphabet.

- Example:

  The language **DOUBLEFACTORIAL**, of strings defined over Σ={a, b}, as $\{a^{n!}\, b^{n!} : n=1,2,3,\ldots\}$, can be written as

  {ab, aabb, aaaaaabbbbbb,…}

- Example:

  The language **SQUARE**, of strings defined over Σ={a}, as

  $\{a^{n^2} : n=1,2,3,\ldots\}$, can be written as

  {a, aaaa, aaaaaaaaa,…}

- Example:

  The language **DOUBLESQUARE**, of strings defined over Σ={a,b}, as

  $\{a^{n^2} b^{n^2} : n=1,2,3,\ldots\}$, can be written as

  {ab, aaaabbbb, aaaaaaaaabbbbbbbbb,…}

- Example:

  The language **PRIME**, of strings defined over Σ={a}, as

  $\{a^p : p \text{ is prime}\}$, can be written as
  {aa,aaa,aaaaa,aaaaaaa,aaaaaaaaaaa…}

An Important language

- PALINDROME

  The language consisting of Λ and the strings s defined over Σ such that Rev(s)=s. It is to be denoted that the words of PALINDROME are called palindromes.

- Example:

  For Σ={a,b}, PALINDROME={Λ , a, b, aa, bb, aaa, aba, bab, bbb, ...}

Finite Automata

- Finite automata are used to recognize patterns.
- It takes the string of symbol as input and changes its state accordingly. When the desired symbol is found, then the transition occurs.
- At the time of transition, the automata can either move to the next state or stay in the same state.
- Finite automata have two states, **Accept state** or **Reject state**. When the input string is processed successfully, and the automata reached its final state, then it will accept.

Formal Definition of FA

- A finite automaton is a collection of 5-tuple (Q, ∑, δ, q0, F), where:
  1. Q: finite set of states
  2. ∑: finite set of the input symbol
  3. q0: initial state
  4. F: **final** state
  5. δ: Transition function (δ -> Greek words tilde).

Finite Automata Model:

- Finite automata can be represented by input tape and finite control.
- **Input tape:**
  - It is a linear tape having some number of cells. Each input symbol is placed in each cell.
- **Finite control:**
  - The finite control decides the next state on receiving particular input from input tape.
  - The tape reader reads the cells one by one from left to right, and at a time only one input symbol is read.

Finite Automata Model:

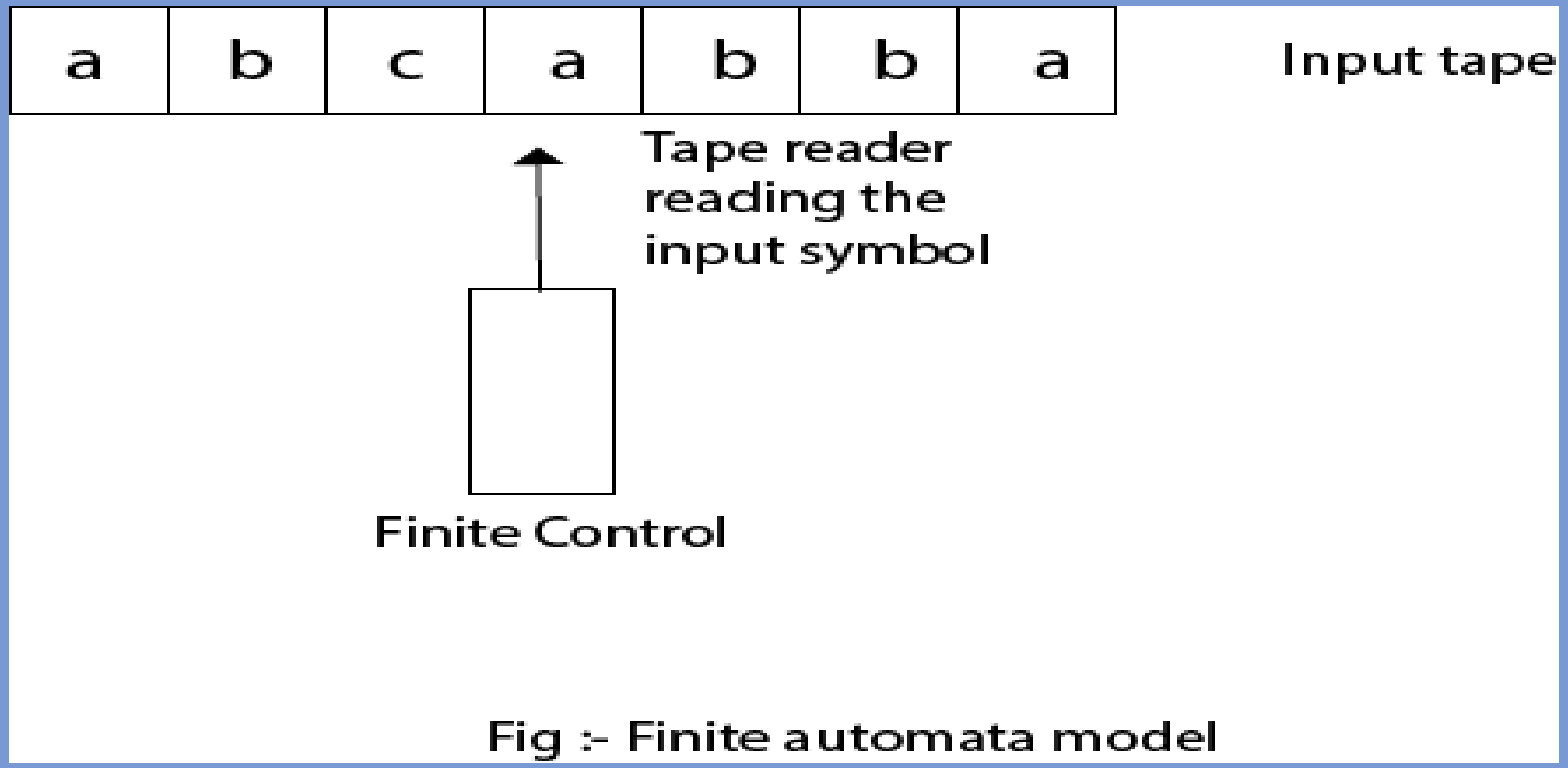
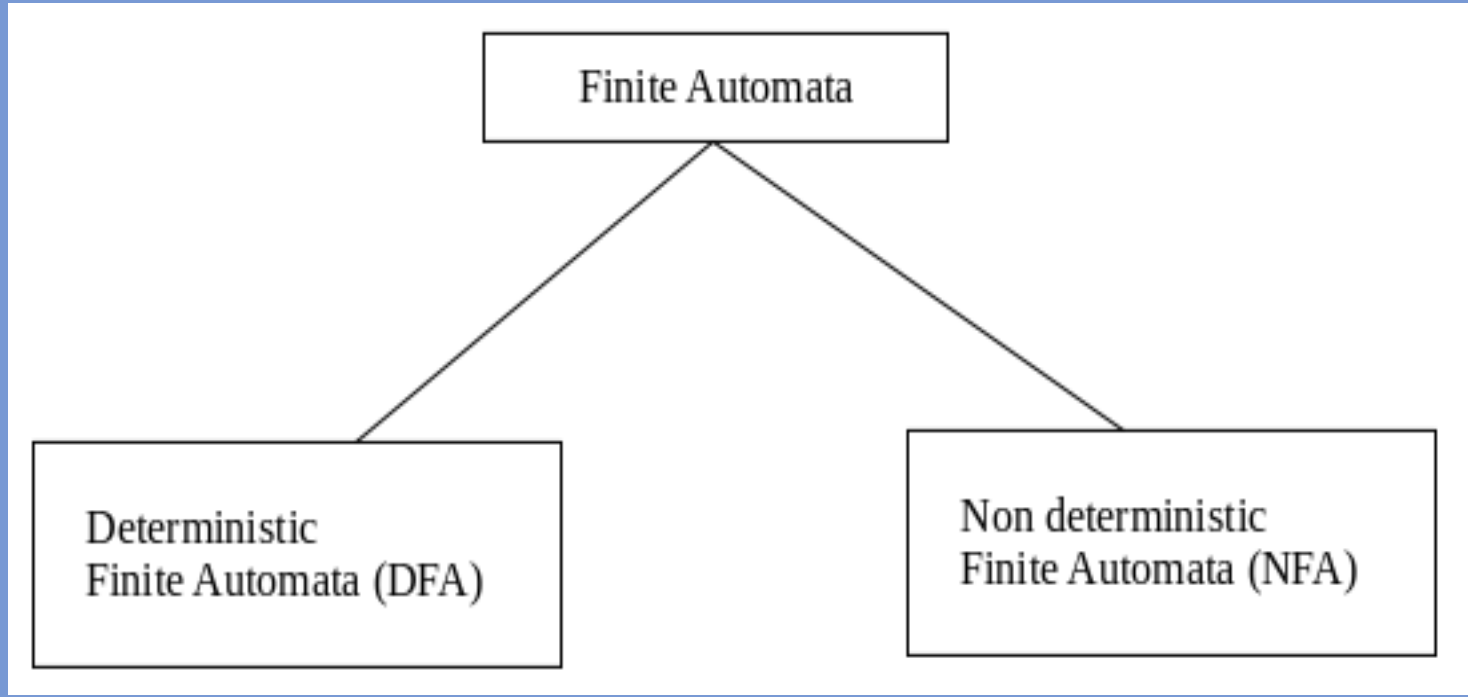


Fig :- Finite automata model

Types of Automata



Thanks