Why Study Formal Languages & Automata Theory?

- Every universal process can be represented by a model.
  - theoretical
  - mathematical.
    - solvable problems.

⇒ Automata theory is the study of abstract mathematical models (hence deals with solvable problems)

- models used are.
  - Finite Automata (FA)
  - Pushdown Automata (PDA)
  - Linear Bounded Automata (LBA)
  - Turing Machines (TM).

⇒ Formal Language (FL)

- abstraction of a general prog. language
- necessary to interpret an abstract mathematical model called an automata.
- FLs are used to define grammars of programming languages

- Automata theory and FLs are the base of current compilers, regular expressions, parsers etc.

Automaton / Automata / Finite state Machine (FSM)

- A finite automata (FA) is a simple idealized machine used to recognise patterns within an input (taken from a character set) called an alphabet
- FA either accepts or rejects an input

accepts → if pattern is defined. e.g., keyword

rejects → if not defined → { wrong keyword ⇐ Syntax error

- A FSM is a model of computation based on a hypothetical machine made of one or more states.
- Only one single state of this machine can be active at the same time ⇒ machine has to transition from one state to another to perform diff. actions.

Finite State Machine (FSM) also called Finite Automaton (FA)

Def
It is a computing device which accepts a string as input and produces the output, accept or reject.

Automaton ← Greek word.
↳ means a machine

Working.

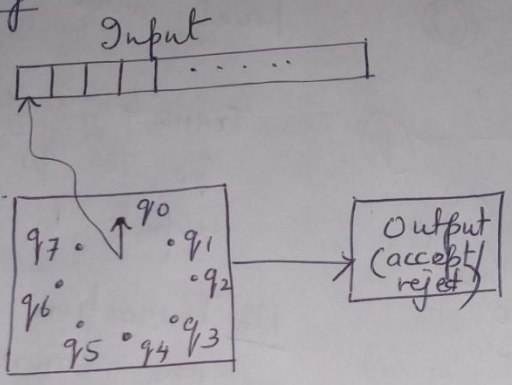


Fig. Block diagram of a FA.

Input: ← string to be processed

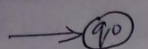
Control Unit: ← number of states identified as $q_0, q_1, \ldots q_7$.
→ Initially machine starts in q_0 and moves to at least one final state.

Output: ← when end of input string is encountered and ~~inp~~ machi

is in final state ⇒ accept

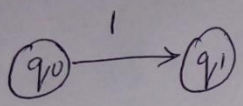
→ When end of string is encountered but final state not reached ⇒ reject

Notation

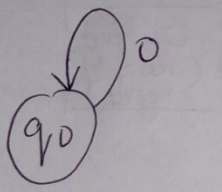


Initial state / start state

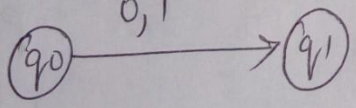
Final state



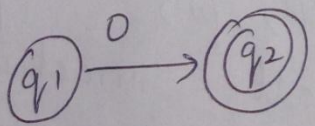
Transition from state q_0 to q_1 on taking input 1.



No Transition of a state on consuming an input 0 (continues in same state).



Transition on taking any of the inputs.



Transition to a final state.

Types of FA

(1) Deterministic Finite Automata (DFA)

(2) Non-deterministic Finite Automata (NFA)

(3) Non-deterministic Finite Automata with ε-moves (ε-NFA)

Applications of FA

(1) used for designing and checking the behavior of digital circuits

(2) used in lexical Analyser of a compiler

(3) for scanning large bodies of text to find occurences of words, phrases, patterns etc.

(4) Analysis of networks or comm. models, gesture recognition

(5) Network design and AI

Symbols.

Alphabet - defined as a finite set of literals
- represented by Σ

e.g;

$\Sigma = \{0, 1\}$ is an alphabet of binary digits

$\Sigma = \{A, B, C, \ldots Z\}$ is an alphabet of uppercase alphabets of English language

String: Finite sequence of symbols formed from some alphabet.

$\Rightarrow \Sigma = \{0, 1\} \Rightarrow$ 011000 is a string

$\Rightarrow$ Length of string $|w|$ is ↓ number of literals

$|w| = 6$

$\Rightarrow$ Empty string $\Leftarrow$ no symbols or literals.
$\Leftarrow$ represented as ε or λ.

Set of strings, including the empty string over a alphabet Σ is denoted by Σ^*.

$\Sigma = \{0, 1\}$ $\Sigma^* = \{\varepsilon, 0, 1, 00, 11, \ldots\}$

$\Sigma_1 = \{0, 1\}$ $\Sigma_2 = \{00, 11, 10, 01\}$

$\Sigma^* = \Sigma+ \cup \varepsilon$