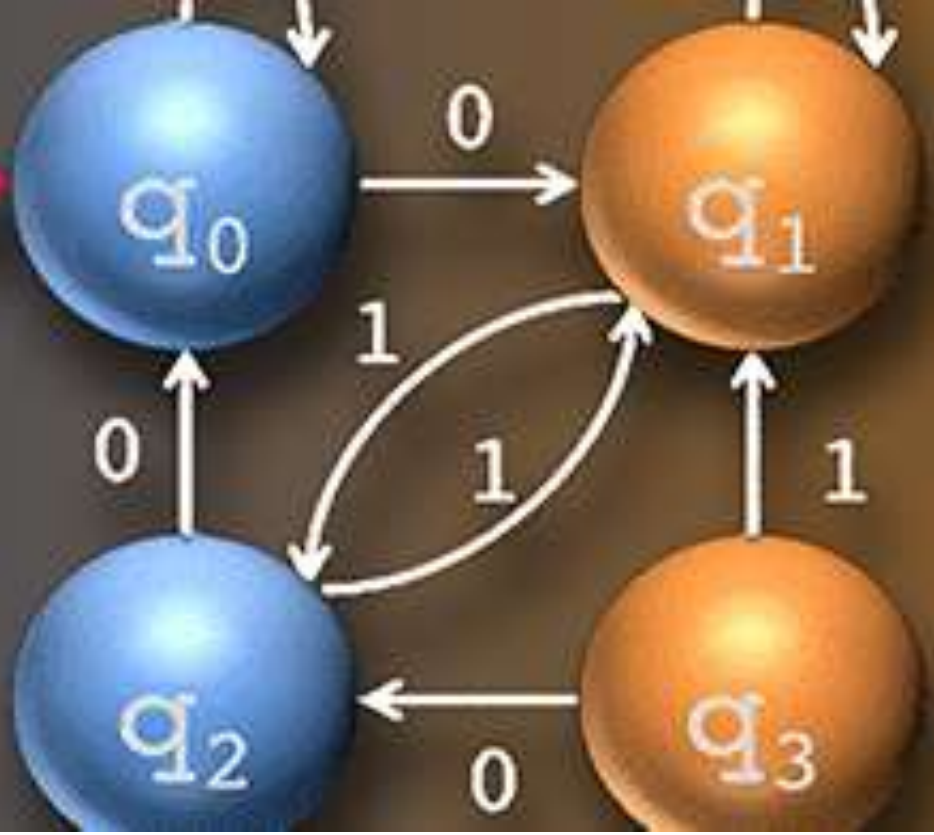
CSE 305

# Theory of COMPUTATION




## Lecture 3

## Introduction (2)

**Md. Mijanur Rahman, Prof. Dr.**

Dept. of Computer Science and Engineering,
Jatiya Kabi Kazi Nazrul Islam University, Bangladesh.
www.mijanrahman.com

# Contents

Introduction


- What is the “Theory of Computation”?
- History: Theory of Computation
- Branches of the Theory of Computation
- **Automata Theory, and Formal Language**
- **Overview of Finite Automata, Context-free Grammars, Pushdown Automata, and Turing Machines**
- **Computability Theory, Complexity Theory, and Models of Computation**
- **Applications of Theory of computation**

# Automata Theory

- Automata theory deals with the definition and properties of various **mathematical models of computers**.
- Mathematicians and Computer Scientists developed this theoretical computer science branch to simplify **the logic of computation by using well defined abstract computational devices (models)**.
- Automata theory is the study of abstract machines (or abstract *mathematical* /computational' machines or systems) and the computational problems that can be solved using these machines. **These abstract machines are called automata**.
- It forms a formal framework for designing and analyzing computing devices, such as biocomputers and quantum computers. These models play a role in several applied areas of computer science.
- One model, called the *finite automaton*, is used in text processing, compilers, and hardware design. Another model, called the *context-free grammar*, is used in programming languages and artificial intelligence.

# Automata Theory

- Automata is originated from the Greek word (Αυτόματα) which means that ***something is doing something by itself***, closely related to "Automation". This automaton consists of *states* and *transitions*.
- **An Automaton is a machine that operates singularly on input and follows a defined pattern or configuration to produce the desired output.** Through automata, we learn how problems and compute functions are solved by the use of automatons.
- A basic computation performed on/by an automaton is defined by the following features:
  - A set of input symbols.
  - The configuration states.
  - Output.

# Automata Theory

- Automata theory is also closely related to **formal language theory**, as the automata are often classified by the class of formal languages they are able to recognize.
- An automaton can be a finite representation of a formal language that may be an infinite set. Automata are used as theoretical models for computing machines, and are used for proofs about computability.

| Grammar | Languages | Automaton | Production rules (constraints) |
|---|---|---|---|
| Type-0 | *Recursively enumerable* | *Turing machine* | $\alpha \to \beta$ (no restrictions) |
| Type-1 | *Context-sensitive* | *Linear-bounded non-deterministic Turing machine* | $\alpha A \beta \to \alpha \gamma \beta$ |
| Type-2 | *Context-free* | Non-deterministic *pushdown automaton* | $A \to \gamma$ |
| Type-3 | *Regular* | *Finite state automaton* | $A \to a$ and $A \to aB$ |

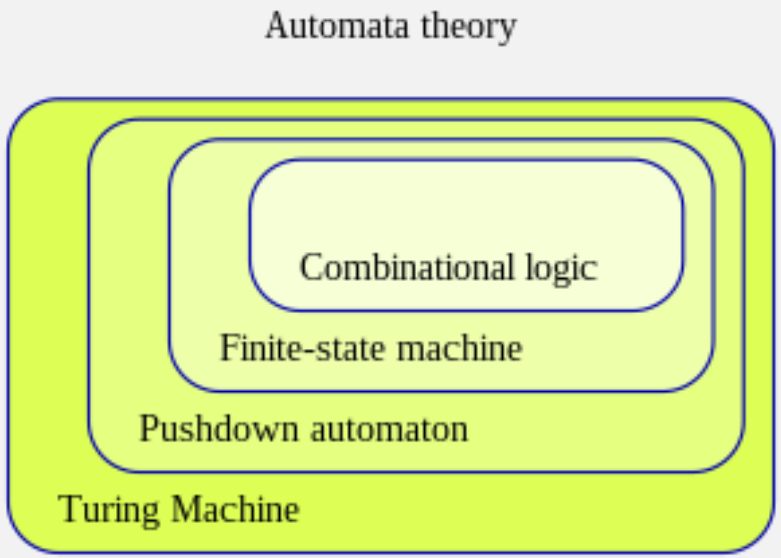
# Automata Theory

- Finally, **Automata Theory** is a theoretical branch of Computer Science and Mathematics, which mainly deals with the logic of computation with respect to simple machines, referred to as automata.
- Automata enables scientists to understand how machines compute the functions and solve problems. The main motivation behind developing **Automata Theory was to develop methods to describe and analyze the dynamic behavior of discrete systems**.
- **Central Question in Automata Theory:** Do these models have the same power, or can one model solve more problems than the other?

# Automata Theory

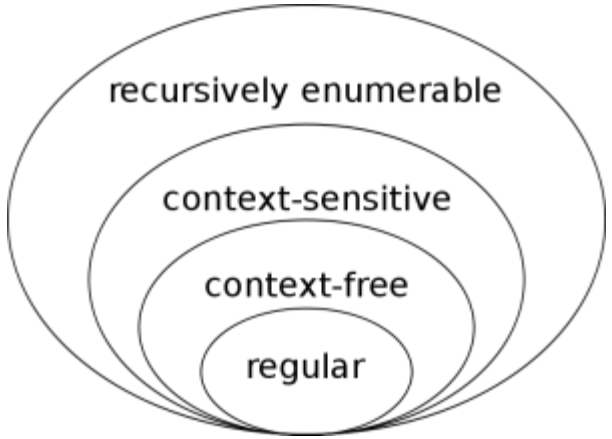
- Main branches of Automata theory
  - **Finite Automata:** These are used in compilers, hardware design and text processing.
  - **Context free grammar:** These are used to define the programming languages and in artificial intelligence.
  - **Pushdown Automata:** A PDA is more powerful than finite automata. It can remember an infinite amount of information
  - **Turing machine:** These are simple abstract models of a real computer, such as your PC at home.

Classes of automata:

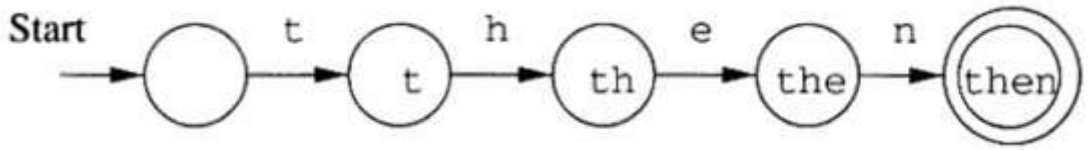


# Formal Language Theory

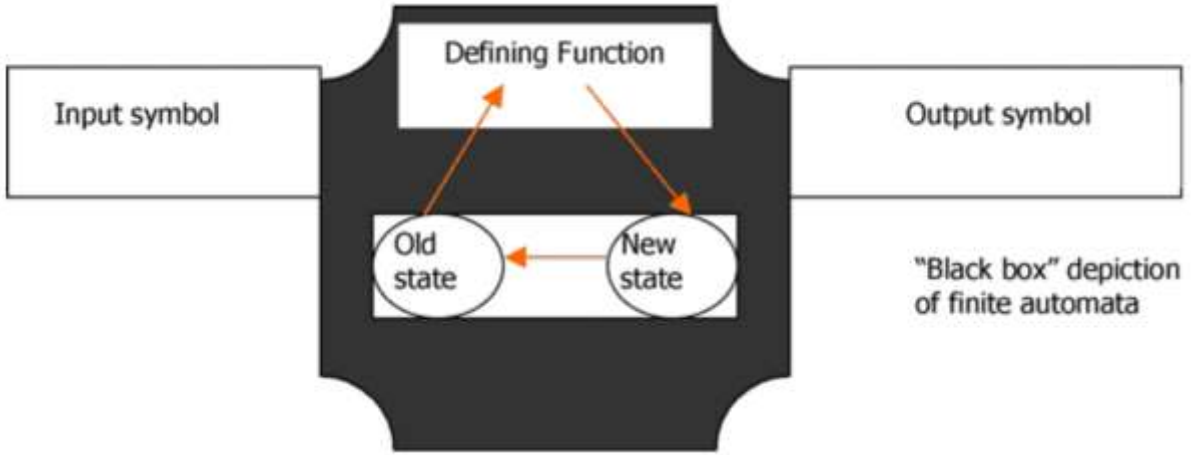
- **Language theory is *a branch of mathematics concerned with describing languages as a set of operations over an alphabet*.** It is closely linked with automata theory, as automata are used to generate and recognize formal languages.
- There are several classes of formal languages, each allowing more complex language specification than the one before it, i.e. Chomsky hierarchy, and each corresponding to a class of automata which recognizes it.
- Because automata are used as models for computation, formal languages are the preferred mode of specification for any problem that must be computed.



# Finite Automata (FA)




**A finite automaton modeling recognition of “then”.**



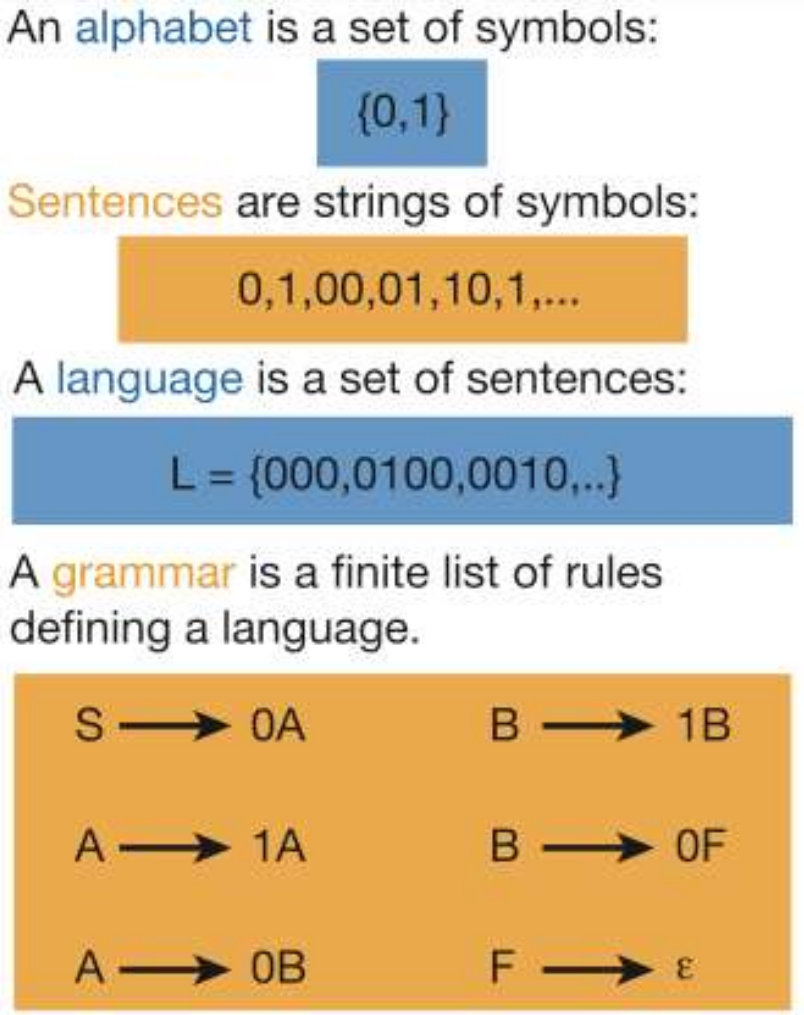

- **FA** is a computer model that is inferior in its computation ability. This model is fit for devices with limited memory. **This is a finite collection of states with rules (transition functions) for traversing through the states depending on the input symbol.** FA accepts or rejects input strings while reading the strings from left to right.
- It is a simple abstract machine with five elements that define its functioning and processing of problems.
- **The tuples are - Q:** *Finite set of states*, **∑:** *Set of input symbols*, **q:** *Initial state*, **F:** *Set of final states*, *and* **δ:** *Transition function*.
- Finite Automata is useful in building text editors/text preprocessors. FA are poor models of computers. They can only perform simple computational tasks.

# Context-Free Grammars (CFGs)

- **Context free grammar is a formal grammar which is used to generate all possible strings in a given formal language.**
- ***Definition*** − A context-free grammar (CFG) consisting of a finite set of grammar rules is a quadruple **(N, T, P, S)** where
  - **N** is a set of non-terminal symbols.
  - **T** is a set of terminals where **N ∩ T = NULL.**
  - **P** is a set of rules, **P: N → (N ∪ T)***, i.e., the left-hand side of the production rule **P** does have any right context or left context.
  - **S** is the start symbol.
- In CFG, the start symbol is used to derive the string. You can derive the string by repeatedly replacing a non-terminal by the right hand side of the production, until all non-terminal have been replaced by terminal symbols.

# Languages & Grammars

- **Alphabet:** An alphabet is a set of symbols.
- **Languages:** A language is a collection of words/sentences of finite length all constructed from a finite alphabet of symbols.
- **Grammars:** A grammar is a finite list of rules defining a language. It can be regarded as a device that enumerates the sentences of a language - nothing more, nothing less.



# Pushdown Automata (PDA)

- Pushdown automata is a way to implement a CFG in the same way we design DFA for a regular grammar. **A DFA can remember a finite amount of information, but a PDA can remember an infinite amount of information**.

- Pushdown automata is simply an NFA augmented with an "**external stack memory**". The addition of stack is used to provide a last-in-first-out memory management capability to Pushdown automata.

- A PDA is more powerful than FA. Any language which can be acceptable by FA can also be acceptable by PDA. PDA also accepts a class of language which even cannot be accepted by FA. Thus PDA is much more superior to FA.

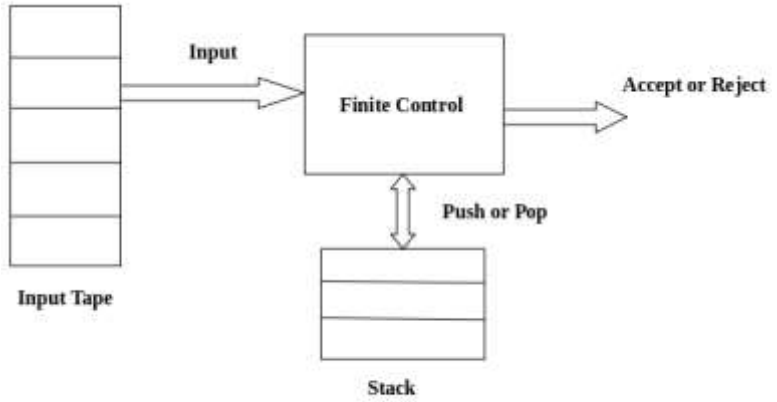


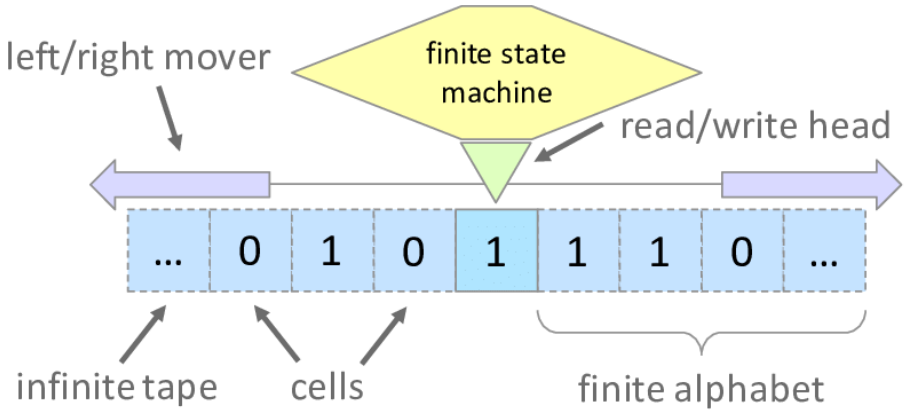

Fig: Pushdown Automata

# Turing Machine

- **A Turing machine is a mathematical model of computation that defines an abstract machine that manipulates symbols on a strip of tape according to a table of rules.**
- Despite the model's simplicity, given any computer algorithm, a Turing machine capable of implementing that algorithm's logic can be constructed.
- Computer scientists study the Turing machine because it is simple to formulate, can be analyzed and used to prove results, and because it represents what many consider the most powerful possible reasonable model of computation. So in principle, **any problem that can be solved by a Turing machine can be solved by a computer that has a finite amount of memory.**
- **Note:**
  - An **abstract machine**, also called an abstract computer, is a theoretical computer used for defining a model of computation
  - A typical abstract machine consists of a definition in terms of input, output, and the set of allowable operations used to turn the former into the latter. The best-known example is the Turing machine.

# Turing Machine

- **A Turing machine consists of a tape of infinite length on which read and writes operation can be performed.**
- The tape consists of infinite cells on which each cell either contains input symbol or a special symbol called blank.
- It also consists of a head pointer which points to cell currently being read and it can move in both directions.



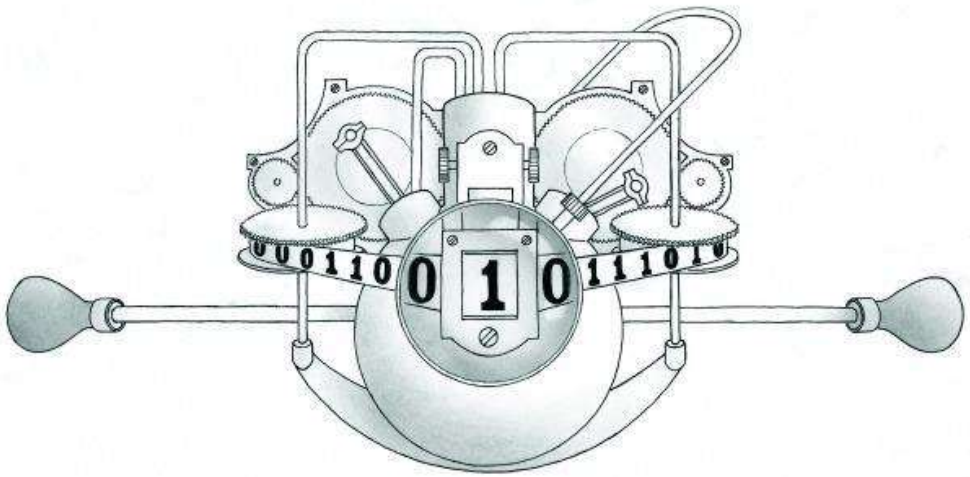



Figure: **Turing machine**

# Write Short Notes:

- **Biocomputers (or Biological computers):**
  - A computer that uses components of biological origin, such as molecules of DNA, instead of electrical components to perform digital or real computations.
- **Quantum computers:**
  - A computer that uses the properties of quantum physics (such as the probability of an object's state before it is measured – instead of just 1s and 0s)) to store data and perform computations.
- **Abstract computer:**
  - An abstract machine, also called an abstract computer, is a theoretical computer used for defining a model of computation.

# ?

## THE END