# Theory Of Computation

Ritwik Jog

Veermata Jijabai Technological Institute, Mumbai

February 13, 2019

# Outline


- Introduction
  - Definition
  - History
- Branches
  - Automata Theory
  - Computability Theory
  - Computational Complexity Theory
- Types of Grammar
- Models of Computation
  - Lambda calculus
  - Combinatory Logic
  - Register Machine

# Introduction: Definition

- Scientific discipline concerned with the study of general properties of computation.
- Methods can be natural ,man-made or imaginary.
- Aims to understand the nature of efficient computation.
- Deals with how efficiently problems can be solved on a model of computation, using an algorithm.

# Introduction: History

The theory of computation can be considered the creation of models of all kinds in the field of computer science.
In the last century it became an independent academic discipline and was separated from mathematics.
Some pioneers of the theory of computation were Alonzo Church, Kurt Gödel, Alan Turing, Stephen Kleene, Rózsa Péter, John von Neumann and Claude Shannon

# Branches: *Automata Theory*

- Automata theory is the study of abstract machines and the computational problems that can be solved using these machines.
- 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.
- Automata are used as theoretical models for computing machines, and are used for proofs about computability.

# Branches: *Computability Theory*

- Computability theory deals primarily with the question of the extent to which a problem is solvable on a computer.
- It is closely related to the branch of mathematical logic called recursion theory.
- This removes the restriction of studying only models of computation which are reducible to the Turing model.

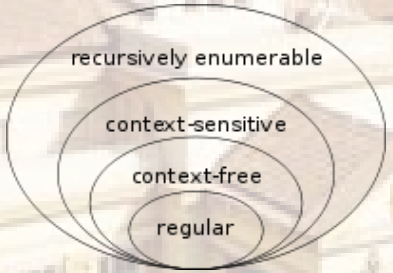


Figure: Set inclusions described by the Chomsky hierarchy

# Branch: *Computational Complexity Theory*

- Complexity theory considers not only whether a problem can be solved at all on a computer, but also how efficiently the problem can be solved.
- Two major aspects are considered: time complexity and space complexity.
- Computer scientists express the time or space required to solve the problem as a function of the size of the input problem.

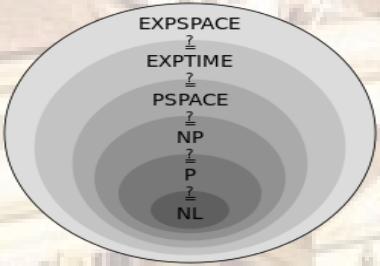


Figure: A representation of the relation among complexity classes

# Types of Grammar

| Grammar | Languages | Automation |
|---|---|---|
| Type-0 | Recursively Enumerable | Turing Machine |
| Type-1 | Context-sensitive | Linear-bounded non-deterministic Turing machine |
| Type-2 | Context-free | Non-deterministic pushdown automaton |
| Type-3 | Regular | Finite State Automation |

Table: Grammar in computational theory

# Models of Computation: *Lambda calculus*

A computation consists of an initial lambda expression (or two if you want to separate the function and its input) plus a finite sequence of lambda terms, each deduced from the preceding term by one application of Beta reduction.

# Models of Computation: *Combinatory Logic*

It is a concept which has many similarities to lambda-calculus, but also important differences exist (e.g. fixed point combinator Y has normal form in combinatory logic but not in lambda-calculus).
Combinatory logic was developed with great ambitions: understanding the nature of paradoxes, making foundations of mathematics more economic, eliminating the notion of variables.

# Models of Computation: *Register Machine*

It is a theoretically interesting idealization of a computer.
Each register can hold a natural number (of unlimited size), and the instructions are simple (and few in number), e.g. only decrementation (combined with conditional jump) and incrementation exist (and halting).
The fact that each register holds a natural number allows the possibility of representing a complicated thing (e.g. a sequence, or a matrix etc.) by an appropriate huge natural number — unambiguity of both representation and interpretation can be established by number theoretical foundations of these techniques.