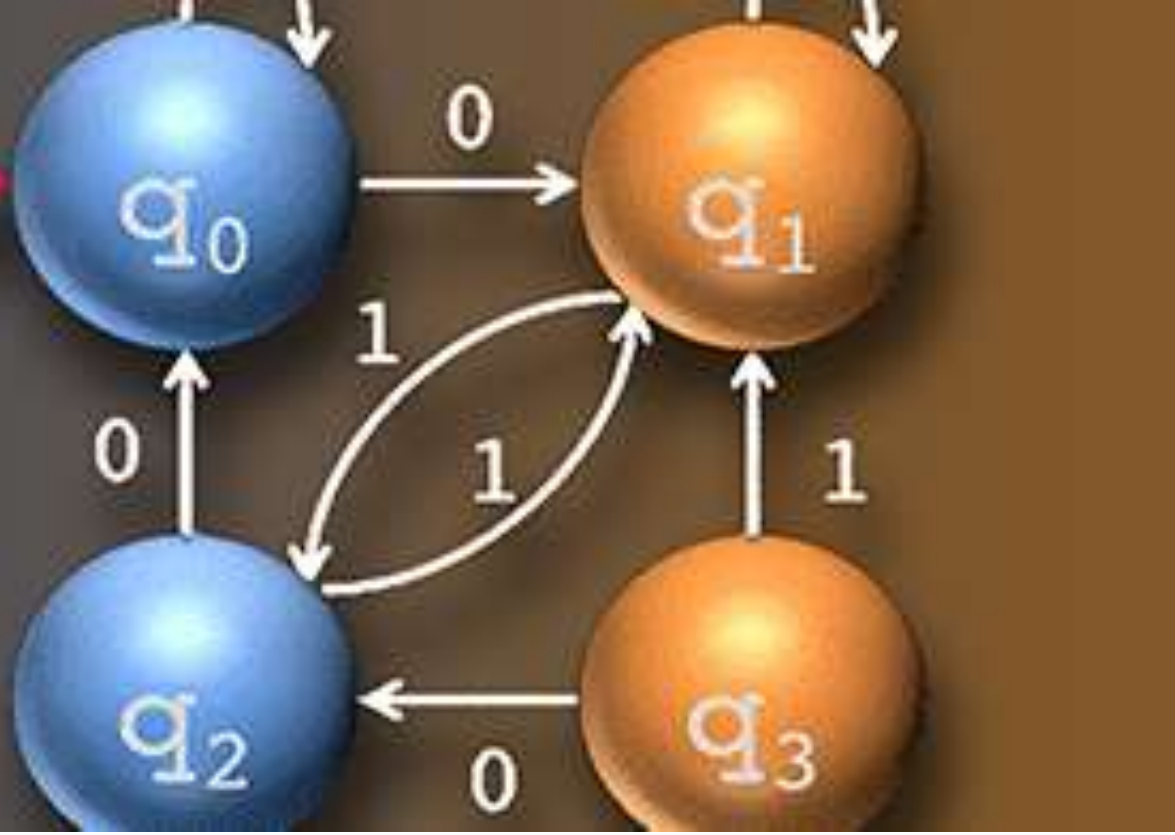
CSE 305

Theory of COMPUTATION



Lecture 2

Introduction (1)

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

Why study Theory of computation (ToC)

- The major reasons about the importance to study of theory of computation are listed below:
  - The importance **to study the theory of computation is to better understand the development of formal mathematical models of computation** that reflect the real-world of computer.
  - **To achieve deep understanding about the mathematical properties of computer hardware and software**.
  - **To understand mathematical definitions of the computation and the algorithms**.
  - **To rectify the limitations of computers and answer what kind of problems can be computed**.

What is the “Theory of Computation”?

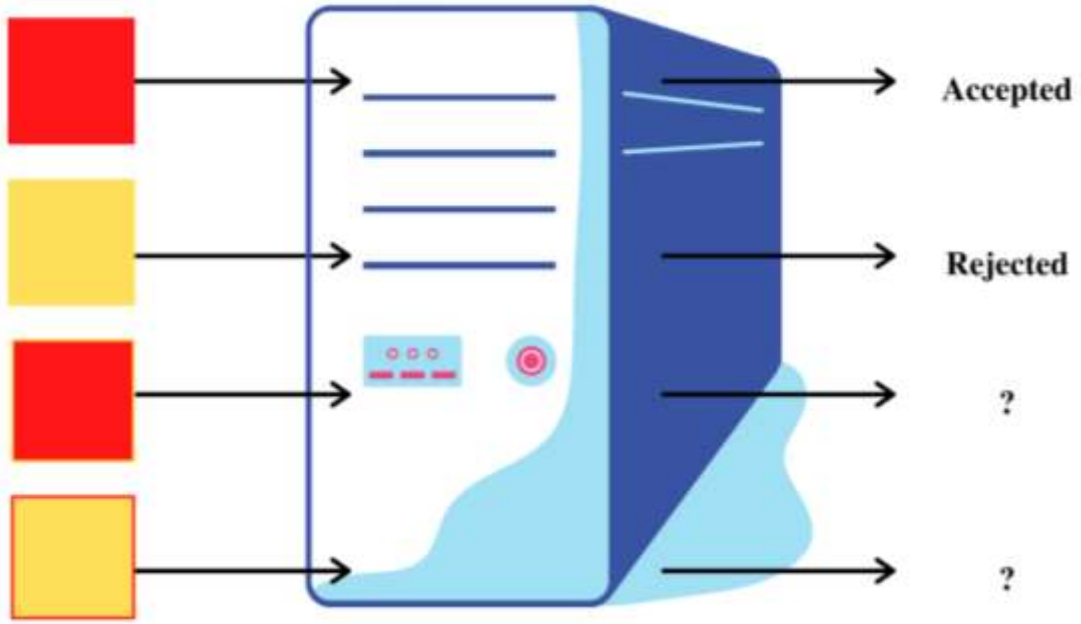
- Real-world computers perform computations that by nature run like mathematical models to solve problems in systematic ways.
- The essence of the theory of computation is **to help develop mathematical and logical models** that run efficiently and to the point of halting. Since all machines that implement logic apply TOC, studying **TOC gives learners an insight into computer hardware and software limitations**.
- Computation is the movement and alteration which occurs during the transition of data or the processing of data based on a set of operations. **The theory of computation includes the fundamental mathematical properties of computer hardware, software and their applications**.
- It is a computer science branch which **deals with how a problem can be solved efficiently by using an algorithm on a model of computation**.

What is the “Theory of Computation”?

- The Theory of Computation rests on the fact that computers can’t solve all problems. A given machine would have limitations, and the Theory of Computation aims to discover these. The computational model is given inputs and decides whether or not it can process the information using the developed algorithm.
- For example, you can create a machine and design it so that it only accepts red objects. The algorithm is pretty straightforward, as represented by the image below. The red square is accepted, and the model rejects the yellow square.

What is the “Theory of Computation”?

- The Theory of Computation can also help determine if a model needs improvement. In the earlier example, the developer may want to introduce other inputs to see how the model treats them.
- What happens when a red square with a yellow border is introduced to the machine? How about when the border is red, but the inside of the object is yellow?



What is the “Theory of Computation”?

- Key considerations of computational problems
  - What can and cannot be computed.
  - Speed of such computations.
  - The amount of memory in use during such computations.
- The theory of computation forms the basis for:
  - Writing efficient algorithms that run in computing devices.
  - Programming language research and their development.
  - Efficient compiler design and construction.
- Three main areas: automata, computability, and complexity

History: Theory of Computation

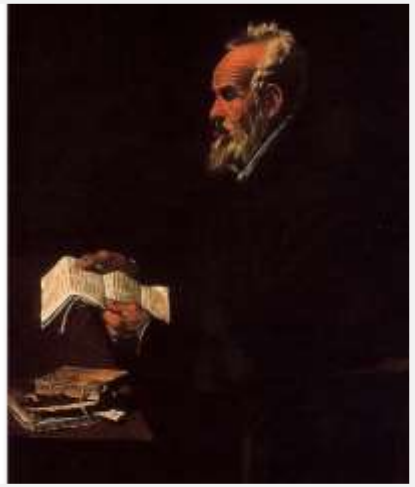
- The theory of computation can be considered the creation of models of all kinds in the field of computer science. Therefore, mathematics and logic are used. In the last century it became an independent academic discipline and was separated from mathematics.

- Some pioneers of the theory of computation were **Ramon Llull** (Computation theory)**, Alonzo Church** (mathematical logic and theoretical computer science)**, Kurt Gödel** (mathematical logic and proof theory)**, Alan Turing** (Turing machines)**, Stephen Kleene** (regular expressions)**, Rózsa Péter, John von Neumann, Claude Shannon, Warren McCulloch and Walter Pitts** (finite automata), **Noam Chomsky** (Chomsky hierarchy and formal language), **Marcel-Paul Schützenberger** (Formal language), **Michael O. Rabin and Dana Scott** (nondeterministic finite automata), **Juris Hartmantis and Richard Stearns** (time and space complexity), **Stephen Cook and Richard Karp** (NP-complete problems).

History: Theory of Computation

- **Ramon Llull** (Computation theory)
- **Alonzo Church** (mathematical logic and theoretical computer science)

Ramon Llull was a philosopher, theologian, poet, missionary, and Christian apologist from the Kingdom of Majorca.



Alonzo Church was an American mathematician and logician who made major contributions to mathematical logic and the foundations of theoretical computer science. He is best known for the lambda calculus, Church–Turing thesis, proving the unsolvability of the

History: Theory of Computation

- **Kurt Gödel** (mathematical logic and proof theory)
- **Alan Turing** (Turing machines)

Kurt Friedrich Gödel was a logician, mathematician, and philosopher. Considered along with Aristotle and Gottlob Frege to be one of the most significant logicians in history, Gödel had an immense effect upon scientific and philosophical thinking in the 20th century, a time when others s

Alan Mathison Turing was an English mathematician, computer scientist, logician, cryptanalyst, philosopher, and theoretical biologist. Turing was highly influential in the development of theoretical computer science, providing a formalisation of the concepts of algorithm and

History: Theory of Computation

- **Stephen Kleene** (regular expressions)
- **Rózsa Péter** (finite automata)

Stephen Cole Kleene was an American mathematician. One of the students of Alonzo Church, Kleene, along with Rózsa Péter, Alan Turing, Emil Post, and others, is best known as a founder of the branch of mathematical logic known as recursion theory, which subsequently helped to

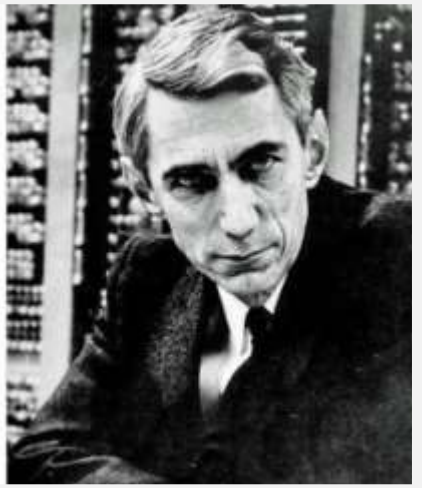
Rózsa Péter, born **Rózsa Politzer**, was a Hungarian mathematician and logician. She is best known as the "founding mother of recursion theory".

History: Theory of Computation

- **John von Neumann**
- **Claude Shannon** (finite automata)

John von Neumann was a Hungarian-American mathematician, physicist, computer scientist, engineer and polymath. Von Neumann was generally regarded as the foremost mathematician of his time and said to be "the last representative of the great

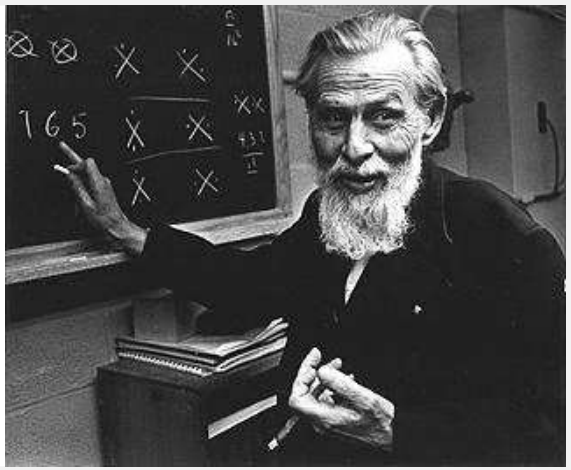
Claude Elwood Shannon was an American mathematician, electrical engineer, and cryptographer known as "the father of information theory". Shannon founded information theory with a landmark paper, "A Mathematical Theory of Communication", which he



History: Theory of Computation

- **Warren McCulloch and Walter Pitts** (finite automata)

Warren Sturgis McCulloch was an American neurophysiologist and cybernetician, known for his work on the foundation for certain brain theories and his contribution to the cybernetics movement. Along with Walter Pitts, McCulloch created computational models based on mathematical algorithr



Walter Harry Pitts, Jr. was a logician who worked in the field of computational neuroscience. He proposed landmark theoretical formulations of neural activity and generative processes that influenced diverse fields such as cognitive sciences and psychology, philosophy,

History: Theory of Computation

- **Noam Chomsky** (Chomsky hierarchy and formal language)
- **Marcel-Paul Schützenberger** (Formal language)

Avram Noam Chomsky is an American linguist, philosopher, cognitive scientist, historian, social critic, and political activist. Sometimes called "the father of modern linguistics", Chomsky is also a major figure in analytic philosophy and one of the founders of the field of cogniti

Marcel-Paul "Marco" Schützenberger was a French mathematician and Doctor of Medicine. He worked in the fields of formal language, combinatorics, and information theory. In addition to his formal results in mathematics, he was "deeply involved in [a] struggle agains

History: Theory of Computation

- **Michael O. Rabin and Dana Scott** (nondeterministic finite automata)

Michael Oser Rabin is an Israeli mathematician and computer scientist and a recipient of the Turing Award.

Dana Stewart Scott is an American logician who is the emeritus Hillman University Professor of Computer Science, Philosophy, and Mathematical Logic at Carnegie Mellon University; he is now retired and lives in Berkeley, California. His work on automata theory ear

History: Theory of Computation

- **Juris Hartmantis and Richard Stearns** (time and space complexity)

Juris Hartmanis is a prominent computer scientist and computational theorist who, with Richard E. Stearns, received the 1993 ACM Turing Award "in recognition of their seminal paper which established the foundations for the field of computational complexity theory".

Richard Edwin Stearns is a prominent computer scientist who, with Juris Hartmanis, received the 1993 ACM Turing Award "in recognition of their seminal paper which established the foundations for the field of computational complexity theory". In 1994 he was inducted as a Fellow of the Associa

History: Theory of Computation

- **Stephen Cook and Richard Karp** (NP-complete problems).

Stephen Arthur Cook, is an American-Canadian computer scientist and mathematician who has made major contributions to the fields of complexity theory and proof complexity. He is a university professor at the University of Toronto, Department of Computer Science and

Richard Manning Karp is an American computer scientist and computational theorist at the University of California, Berkeley. He is most notable for his research in the theory of algorithms, for which he received a Turing Award in 1985, The Benjamin Franklin Medal in

History: Theory of Computation

- **The theory of abstract automata was developed in the mid-20th century in connection with finite automata**. Automata theory was initially considered a branch of mathematical systems theory, studying the behavior of discrete-parameter systems.
- **Before 1930's:**
  - Alan Turing Studied an abstract machine that had all the capabilities of today's computers to solve problems. A. Turing's goal was to describe precisely that boundary between what a computing machines could do and what it could not do.
  - **In the year 1936** Alan Turing invented *Turing machine*, and proved that there exists an *unsolvable problem*.
- **1931's to 1950's:**
  - Simpler kinds of machines were used which we called 'Finite Automata'. These automata originally proposed to model brain function, turned out to be extremely useful for a variety of other purposes like designing software's to checking the behavior of digital circuit used in computers etc.
  - **In the 1940's** the stored-program computers were built.

History: Theory of Computation

- **Late 1950's to 1960's:**
  - N. Chomsky began the study of formal 'grammars' that are not strictly belongs to the machines, but these grammars have closer relationships to abstracts automata.
  - **In the year 1956** Kleene invented *regular expressions* and proved equivalence of regular expression or limited automata. Also, in this year, Chomsky described *Chomsky hierarchy*, which organized the languages recognized by the different automata into the hierarchical classes.
  - **In the 1959** Rabin and Scott introduced *nondeterministic finite automata* and proved its equivalence to finite automata.
  - **1950's-1960's** more works on grammars, languages, and compilers. In present world these grammars serves as the basis of some important software components, including parts of compilers.
- **After 1960's:**
  - **In the year 1965** Hartmantis and Stearns defined *time complexity*, and Lewis, Hartmantis and Stearns defined *space complexity*. **In the year 1971** Cook showed the 1st *NP-complete problem*, the *satisfiability* problem. **In the year 1972** Karp Showed many other NP-complete problems.
- The theory of computational complexity also took shape in the 1960s. By the end of the decade, **automata theory came to be seen as "the pure mathematics of computer science"**.

Branches of the Theory of Computation

- Three main branches of theories make up what the Theory of Computation is. These are:
  - **Automata Theory and Language:** Refers to the analysis of how machines work to solve a problem.
  - **Computability Theory:** Pertains to determining which problems a machine can solve and which ones it can't.
  - **Computational Complexity Theory:** Addresses the issue of the efficiency of the machine when solving a problem.
- One of the most famous inventions that embody these concepts is the **Turing machine created by Alan Turing in the 1930s**. The idea is that a Turing machine can run any problem that algorithms can solve. In reverse, anything that an algorithm can't do can't be done by a Turing machine.

?

THE END