Theory of computation

B. Srivathsan

Chennai Mathematical Institute

http://www.cmi.ac.in/~sri/Courses/TOC2013

Why do this course?

Credits

Contents of this talk are picked from / inspired by:

- Wikipedia
- Sipser: *Introduction to the theory of computation*
- Kleene: *Introduction to metamathematics*
- Emerson: *The beginning of model-checking: A personal perspective*
- Scott Aaronson's course at MIT: *Great ideas in theoretical CS*

1900 - 1940

(Illustrations from Logicomix. Published by Bloomsbury)

“ Those who don’t shave themselves are shaved by the barber ”

“ Those who don’t shave themselves are shaved by the barber ”

Who will shave the barber ?

“ Those who don’t shave themselves are shaved by the barber ”

Who will shave the barber ?

Bertrand Russell (1872 - 1970)

Russell’s paradox (1901)

questioned Cantor’s set theory

Foundational crisis of mathematics

New schools of thought emerged in early 20^{th} century

Intuitionist: *Brouwer*

Formalist: *Russell, Whitehead, Hilbert*

Hilbert's programme

David Hilbert (1862 - 1943)

Goal: convert Mathematics to mechanical manipulation of symbols

$\forall$: `for all` $\quad$ $\exists$: `there exists` $\quad$ $\wedge$: `and`

- There are infinitely many primes

$$\forall q\, \exists p\, \forall x, y\ [\ p > q \wedge (x, y > 1 \rightarrow xy \neq p)\ ]$$

- Fermat's last theorem

$$\forall a, b, c, n\ [\ (a, b, c > 0 \wedge n > 2) \rightarrow a^n + b^n \neq c^n\ ]$$

- Twin prime conjecture

$$\forall q\, \exists p\, \forall x, y\ [\ p > q \wedge (x, y > 1 \rightarrow\ (xy \neq p \wedge xy \neq p + 2)\ )\ ]$$

$\forall$: `for all` $\exists$: `there exists` $\wedge$: `and`

- There are infinitely many primes

$$\forall q\, \exists p\, \forall x, y\ [\ p > q \wedge (x, y > 1 \rightarrow xy \neq p)\ ]$$

- Fermat's last theorem

$$\forall a, b, c, n\ [\ (a, b, c > 0 \wedge n > 2) \rightarrow a^n + b^n \neq c^n\ ]$$

- Twin prime conjecture

$$\forall q\, \exists p\, \forall x, y\ [\ p > q \wedge (x, y > 1 \rightarrow\ (xy \neq p \wedge xy \neq p + 2)\ )\ ]$$

Hilbert's Entscheidungsproblem (1928)

Is there an "algorithm" that can take such a mathematical statement as input and say if it is true or false?

λ-calculus

Alonzo Church (1903 - 1995)

Turing machines

Alan Turing (1912 - 1954)

Answer to **Entscheidungsproblem** is No (1935 - 1936)

Intuitively, an algorithm meant

“a process that determines the solution in a finite number of operations”

Intuitively, an algorithm meant

"a process that determines the solution in a finite number of operations"

Intuitive notion **not adequate** to show that an algorithm **does not exist** for a problem!

Church-Turing thesis

Intuitive notion of algorithms

$$\equiv$$

Turing machine algorithms

Church-Turing thesis

Intuitive notion of algorithms

$$\equiv$$

Turing machine algorithms

A prototype for a computing machine!

Church-Turing thesis

Intuitive notion of algorithms

$\equiv$

Turing machine algorithms

A prototype for a computing machine!

Advent of digital computers in the 40's

1900 - 1940

Precise notion of algorithm

1940 - 1975

How “**efficient**” is an algorithm?

What is the “**optimal**” way of solving a problem?

Can we do better than just “**brute-force**”?

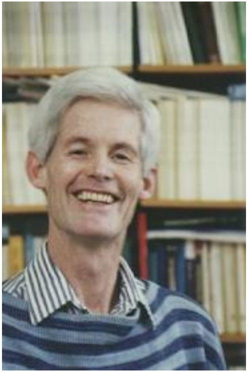
Computational complexity

Juris Hartmanis (Born: 1928)

Richard Stearns (Born: 1936)

Classification of algorithms based on **time** and **space** (1965)

NP-completeness (1971 - 1973)



Stephen Cook (Born: 1939)

Leonid Levin (Born: 1948)

Richard Karp (Born: 1935)

Identified an important class of **intrinsically difficult** problems

An easy solution to one would give an easy solution to the other!

Some examples...

- Easy problems: sorting, finding shortest path in a graph
- Hard problems: scheduling classes for university

Computationally hard problems very important for **cryptographers**!

1900 - 1940

Precise notion of algorithm

1940 - 1975

Hardness of problems

1975 - present

Computer programs (esp. large ones)

are prone to **ERRORS**

Computer programs (esp. large ones)

are prone to **ERRORS**

Is there a way to

specify formally what a program is intended to do, and

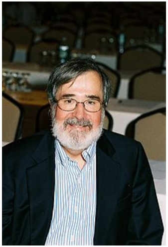
verify automatically if the program satisfies the specification

Temporal logic

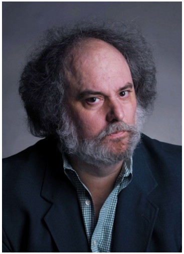
Amir Pnueli (1941 - 2009)

Introduced a formalism to **specify** intended behaviours of programs (1977)

Model-checking



Edmund Clarke (Born: 1945)



Allen Emerson (Born: 1954)

Joseph Sifakis (Born: 1946)

Automatically verify a program against its specification (1981)

1900 - 1940

Precise notion of algorithm

1940 - 1975

Hardness of problems

1975 - present

Correctness of programs

1900 - 1940

Precise notion of algorithm (Theory of computation)

1940 - 1975

Hardness of problems (Computational complexity theory)

1975 - present

Correctness of programs (Formal verification)

1900 - 1940

Precise notion of algorithm (Theory of computation)

1940 - 1975

Hardness of problems (Computational complexity theory)

1975 - present

Correctness of programs (Formal verification)

This course: Theory of computation + bit of Computational complexity

Problems $\to$ languages

Decision problems

Questions for which the answer is either **Yes** or **No**

- Is the sum of 5 and 8 equal to 12?
- Is 19 a prime number?
- Is the graph G_1 connected?

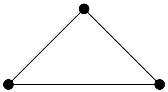


Figure: G_1

Is the sum of 5 and 8 equal to 12 ?

Is the sum of 5 and 8 equal to 12 ?

$$L_{add} = \{\, (a, b, c) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N} \mid a + b = c \,\}$$

$$\{\, (1, 3, 4),\ (5, 9, 14),\ (0, 2, 2),\ \ldots \}$$

Is the sum of 5 and 8 equal to 12 ?

$$L_{add} \;=\; \{\, (a,b,c) \in \mathbb{N}\times\mathbb{N}\times\mathbb{N} \;\mid\; a+b=c \,\}$$

$$\{\, (1,3,4),\; (5,9,14),\; (0,2,2),\; \ldots \}$$

Is $(5,8,12) \in L_{add}$?

Is the sum of 5 and 8 equal to 12 ?

$$L_{add} \;=\; \{\, (a,b,c) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N} \;\mid\; a+b=c \,\}$$

$$\{\, (1,3,4),\; (5,9,14),\; (0,2,2),\; \dots \}$$

Is $(5,8,12) \in L_{add}$?

Is 19 a prime number ?

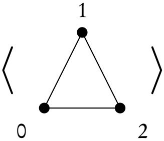
$$L_{prime} \;=\; \{\, x \in \mathbb{N} \;\mid\; x \text{ is prime} \,\}$$

$$\{\, 1,\; 2,\; 3,\; 5,\; 7,\; 11,\; \dots \}$$

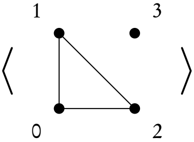
Is $19 \in L_{prime}$?

Encoding graphs

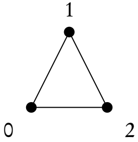
$\langle$ graph $\rangle$:= no. of vertices \$ edge relation



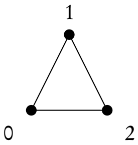
$3 \,\$\, (0,1)(1,2)(0,2)$



$4 \,\$\, (0,1)(1,2)(0,2)$



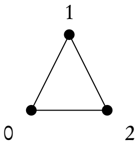
Is graph G_1 connected?



Is graph G_1 connected?

$$L_{conn} = \{ \langle G \rangle \mid G \text{ is connected} \}$$

Is $\langle G_1 \rangle \in L_{conn}$?



Is graph G_1 connected?

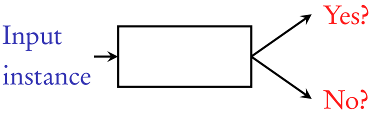
$$L_{conn} = \{ \langle G \rangle \mid G \text{ is connected} \}$$

Is $\langle G_1 \rangle \in L_{conn}$?

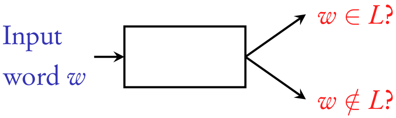
$$\equiv$$

Is $3 \, \$ \, (0,1)(1,2)(0,2) \in L_{conn}$?

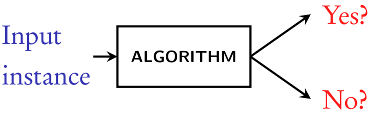
Decision problem P



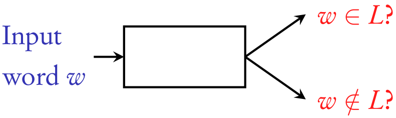
Language L



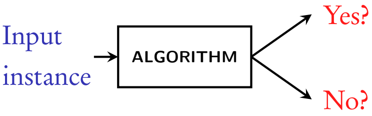
Decision problem P



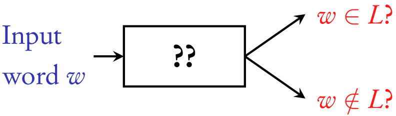
Language L



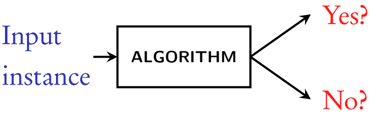
Decision problem P



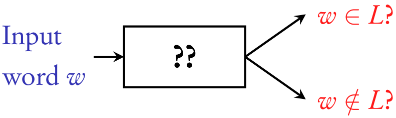
Language L



Decision problem P

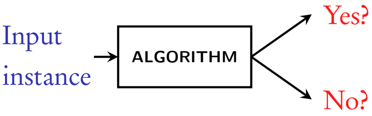


Language L

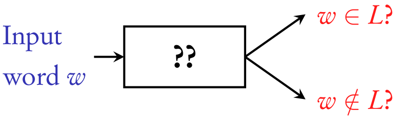


We answer “ ?? ” in this course

Decision problem P



Language L



We answer " **??** " in this course

Main challenge: How to get a **finite representation** for languages?