# Lecture 1

## History of mathematical logic in computer science

*Introduction to Logic for Computer Science*

Prof Hongseok Yang
KAIST

These slides are minor variants of those made by Prof Worrell and Dr Haase for their logic course at Oxford.

# Agenda

1. A historical perspective on logic
2. Recent applications of computational logic in math, CS and philosophy
3. Overview of the course
4. Practicalities

1 A historical perspective on logic

2 Recent applications of computational logic in math, CS and philosophy

3 Overview of the course

4 Practicalities

# What is logic?

# What is logic?

*"[Logic is] ... the name of a discipline which analyzes the meaning of the concepts common to all the sciences, and establishes the general laws governing the concepts."*

*—Alfred Tarski, "Introduction to logic and to the methodology of deductive sciences".*

# What is logic?

*"[Logic is] ... the name of a discipline which analyzes the meaning of the concepts common to all the sciences, and establishes the general laws governing the concepts."*

*—Alfred Tarski, "Introduction to logic and to the methodology of deductive sciences".*

*"To discover truths is the task of all sciences; it falls to logic to discern the laws of truth. ... I assign to logic the task of discovering the laws of truth, not of assertion or thought."*

*—Gottlob Frege, "Der Gedanke. Eine logische Untersuchung".*

# What is logic?

Logic is the study of the principles of correct reasoning.

# Aristotle (384–322 BC)

Syllogism: *"...a discourse in which certain (specific) things having been supposed, something different from the things supposed results of necessity because these things are so."*

All *beings* are *mortal*
All *humans* are *beings*
---
All *humans* are *mortal*

All *B* are *M*
All *H* are *B*
---
All *H* are *M*

# Aristotle (384–322 BC)

Syllogism: *"...a discourse in which certain (specific) things having been supposed, something different from the things supposed results of necessity because these things are so."*

All *beings* are *mortal*
All *humans* are *beings*
---
All *humans* are *mortal*

All *B* are *M*
All *H* are *B*
---
All *H* are *M*

*Aristotle only gave a compendium of valid arguments.*

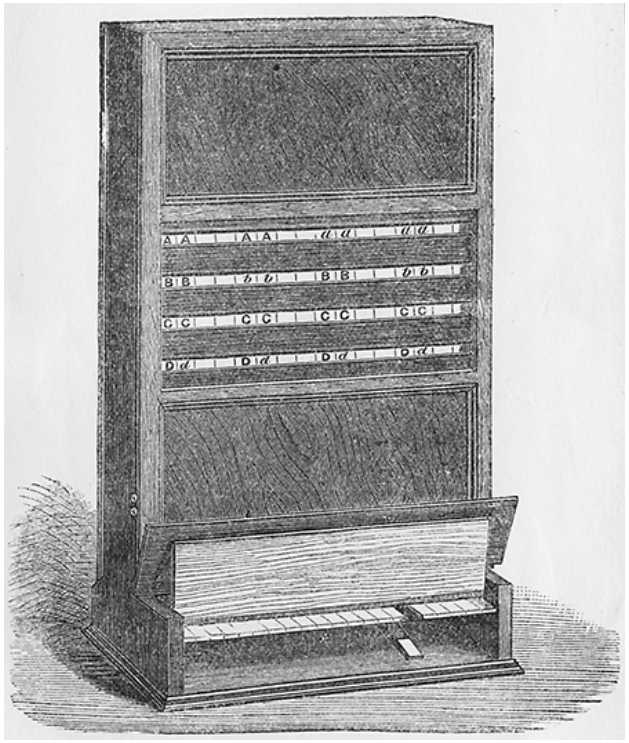
# Gottfried Wilhelm Leibniz (1646–1716)

- Envisioned a calculus that allows for testing *any* argument for validity.

*"It is obvious that if we could find characters or signs suited for expressing all our thoughts as clearly and as exactly as arithmetic expresses numbers or geometry expresses lines [. . . ] all investigations which depend on reasoning would be carried out by transposing these characters and by a species of calculus."*

# George Boole (1815–1864)

- Equational rules for *propositional logic* in 1854.
- Built into the logic piano by William Stanley Jevons.

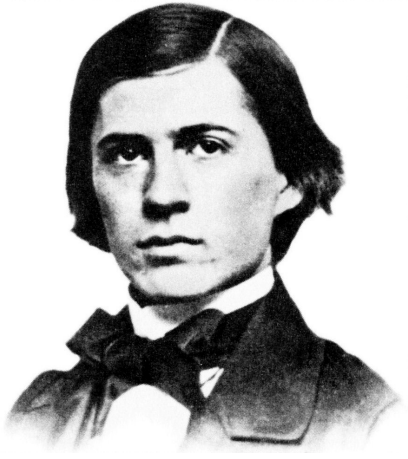


# Gottlob Frege (1848–1925) and Charles Sanders Pierce (1839–1914)

- Generalisation of propositional logic to *predicate logic*.

*"There is just one point where I have encountered a difficulty . . . "*

— Bertrand Russell to Frege, 1902.

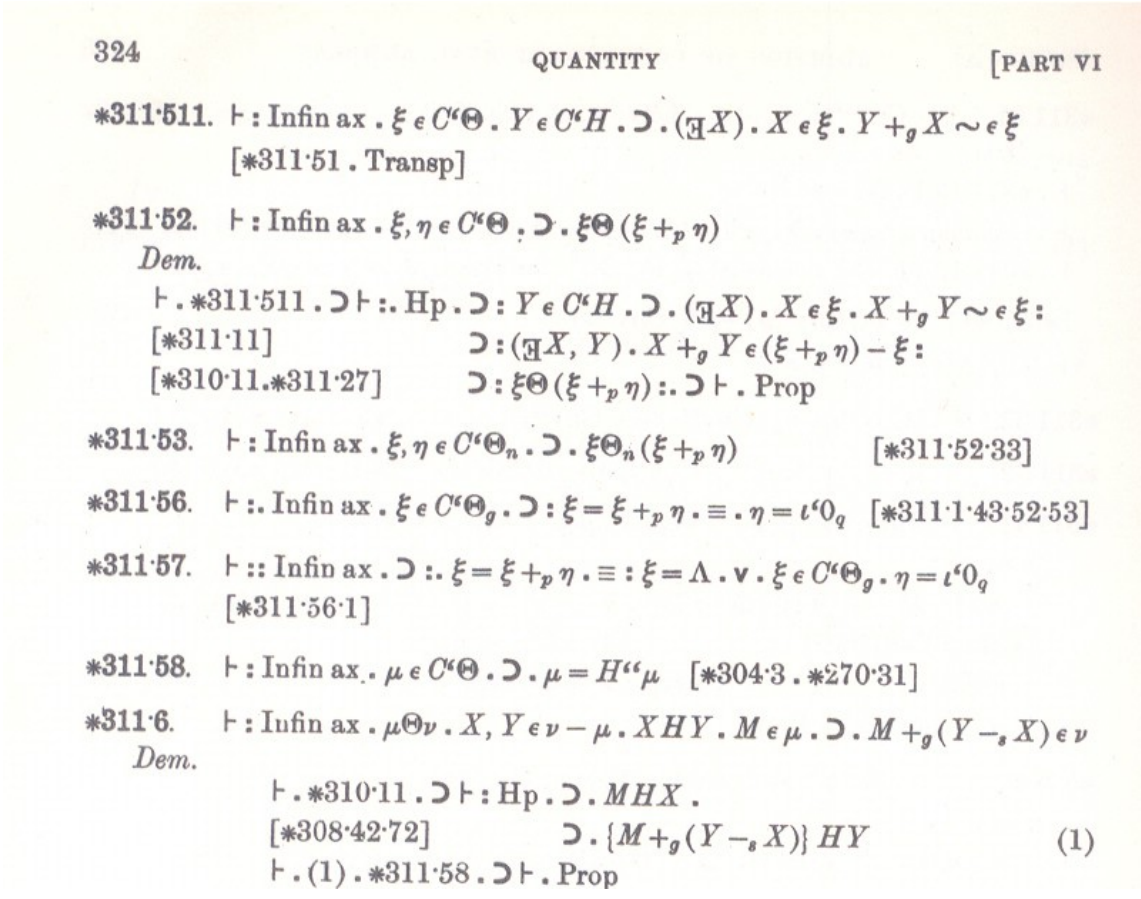


*"Beyond doubt . . . he was one of the most original minds of the later nineteenth century, and certainly the greatest American thinker ever."*

— Bertrand Russell about Pierce, 1959.

# Bertrand Russell (1872–1970) and Alfred North Whitehead (1861–1947)

- Principia Mathematica: attempt to base all of mathematics (set theory, analysis, geometry) rigorously on predicate logic.



324 QUANTITY [PART VI

$*311{\cdot}511.\quad \vdash : \mathrm{Infin\,ax} \,.\, \xi \in C`\Theta \,.\, Y \in C`H \,.\supset.\, (\exists X) \,.\, X \in \xi \,.\, Y +_g X \sim \in \xi$
[$*311{\cdot}51$ . Transp]

$*311{\cdot}52.\quad \vdash : \mathrm{Infin\,ax} \,.\, \xi, \eta \in C`\Theta \,.\supset.\, \xi\Theta(\xi +_p \eta)$
*Dem.*
$\vdash .\, *311{\cdot}511 .\supset \vdash :. \mathrm{Hp} .\supset : Y \in C`H .\supset .\, (\exists X) .\, X \in \xi .\, X +_g Y \sim \in \xi :$
[$*311{\cdot}11$] $\supset : (\exists X, Y) .\, X +_g Y \in (\xi +_p \eta) - \xi :$
[$*310{\cdot}11.*311{\cdot}27$] $\supset : \xi\Theta(\xi +_p \eta) :. \supset \vdash .\, \mathrm{Prop}$

$*311{\cdot}53.\quad \vdash : \mathrm{Infin\,ax} \,.\, \xi, \eta \in C`\Theta_n \,.\supset.\, \xi\Theta_n(\xi +_p \eta)$ [$*311{\cdot}52{\cdot}33$]

$*311{\cdot}56.\quad \vdash :. \mathrm{Infin\,ax} \,.\, \xi \in C`\Theta_g \,.\supset : \xi = \xi +_p \eta \,.\equiv.\, \eta = \iota`0_q$ [$*311{\cdot}1{\cdot}43{\cdot}52{\cdot}53$]

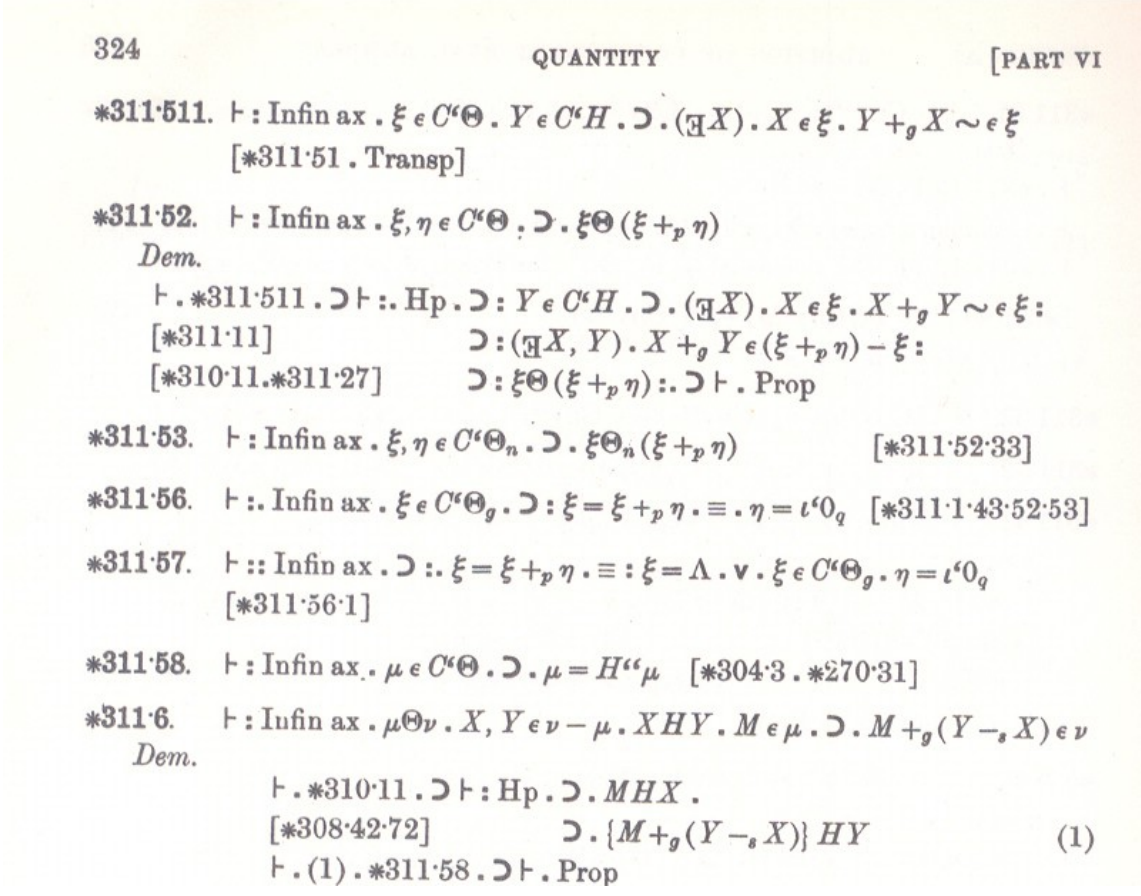
$*311{\cdot}57.\quad \vdash :: \mathrm{Infin\,ax} \,.\supset :. \xi = \xi +_p \eta \,.\equiv : \xi = \Lambda \,.\, \mathrm{v} \,.\, \xi \in C`\Theta_g \,.\, \eta = \iota`0_q$
[$*311{\cdot}56{\cdot}1$]

$*311{\cdot}58.\quad \vdash : \mathrm{Infin\,ax} \,.\, \mu \in C`\Theta \,.\supset.\, \mu = H``\mu$ [$*304{\cdot}3$ . $*270{\cdot}31$]

$*311{\cdot}6.\quad \vdash : \mathrm{Infin\,ax} \,.\, \mu\Theta\nu \,.\, X, Y \in \nu - \mu \,.\, XHY \,.\, M \in \mu \,.\supset.\, M +_g (Y -_s X) \in \nu$
*Dem.*
$\vdash .\, *310{\cdot}11 .\supset \vdash : \mathrm{Hp} .\supset .\, MHX .$
[$*308{\cdot}42{\cdot}72$] $\supset .\, \{M +_g (Y -_s X)\}\, HY$ (1)
$\vdash .\, (1) .\, *311{\cdot}58 .\supset \vdash .\, \mathrm{Prop}$

# Bertrand Russell (1872–1970) and Alfred North Whitehead (1861–1947)

- Principia Mathematica: attempt to base all of mathematics (set theory, analysis, geometry) rigorously on predicate logic.



324 QUANTITY [PART VI

$*311{\cdot}511.\ \vdash : \mathrm{Infin\,ax} . \xi \in C`\Theta . Y \in C`H . \supset . (\exists X) . X \in \xi . Y +_g X \sim \in \xi$
[$*311{\cdot}51$ . Transp]

$*311{\cdot}52.\ \vdash : \mathrm{Infin\,ax} . \xi, \eta \in C`\Theta . \supset . \xi\Theta(\xi +_p \eta)$
*Dem.*
$\vdash . *311{\cdot}511 . \supset \vdash :. \mathrm{Hp} . \supset : Y \in C`H . \supset . (\exists X) . X \in \xi . X +_g Y \sim \in \xi :$
[$*311{\cdot}11$] $\supset : (\exists X, Y) . X +_g Y \in (\xi +_p \eta) - \xi :$
[$*310{\cdot}11 . *311{\cdot}27$] $\supset : \xi\Theta(\xi +_p \eta) :. \supset \vdash . \mathrm{Prop}$

$*311{\cdot}53.\ \vdash : \mathrm{Infin\,ax} . \xi, \eta \in C`\Theta_n . \supset . \xi\Theta_n(\xi +_p \eta)$ [$*311{\cdot}52{\cdot}33$]

$*311{\cdot}56.\ \vdash :. \mathrm{Infin\,ax} . \xi \in C`\Theta_g . \supset : \xi = \xi +_p \eta . \equiv . \eta = \iota`0_q$ [$*311{\cdot}1{\cdot}43{\cdot}52{\cdot}53$]

$*311{\cdot}57.\ \vdash :: \mathrm{Infin\,ax} . \supset :. \xi = \xi +_p \eta . \equiv : \xi = \Lambda . \vee . \xi \in C`\Theta_g . \eta = \iota`0_q$
[$*311{\cdot}56{\cdot}1$]

$*311{\cdot}58.\ \vdash : \mathrm{Infin\,ax} . \mu \in C`\Theta . \supset . \mu = H``\mu$ [$*304{\cdot}3 . *270{\cdot}31$]

$*311{\cdot}6.\ \vdash : \mathrm{Infin\,ax} . \mu\Theta\nu . X, Y \in \nu - \mu . XHY . M \in \mu . \supset . M +_g (Y -_s X) \in \nu$
*Dem.*
$\vdash . *310{\cdot}11 . \supset \vdash : \mathrm{Hp} . \supset . MHX .$
[$*308{\cdot}42{\cdot}72$] $\supset . \{M +_g (Y -_s X)\} HY$ (1)
$\vdash . (1) . *311{\cdot}58 . \supset \vdash . \mathrm{Prop}$

(proves that 1+1=2 on page 379)

# Kurt Gödel (1906–1978)

- No sufficiently strong logical system can be both consistent and complete.

*"Kurt Gödel's achievement in modern logic is singular and monumental - indeed it is more than a monument, it is a landmark which will remain visible far in space and time. [. . .] The subject of logic has certainly completely changed its nature and possibilities with Gödel's achievement."*

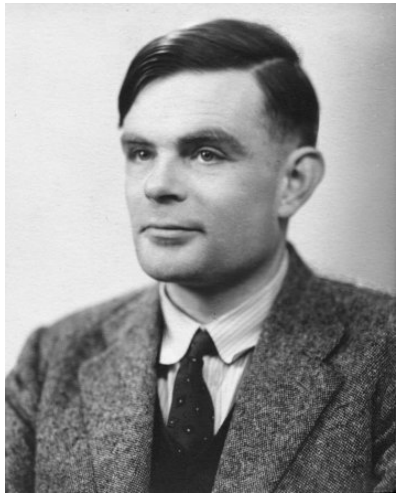
*— John von Neumann*

## Alonso Church (1903–1995) and Alan Turing (1912–1954)

- There is no algorithm that decides whether a given logical argument is valid or not.
- $\lambda$-*calculus* and *Turing machines* lay the foundations for theoretical computer science.

Alonzo Church, A note on the Entscheidungsproblem", *Journal of Symbolic Logic*, 1 (1936), pp 40-41.

Alan Turing, On computable numbers, with an application to the Entscheidungsproblem, *Proceedings of the London Mathematical Society*, Series 2, 42 (1936-7), pp 230-265.



# Claude Shannon (1916–2001) and John Robinson (1930–2016)

Using electrical switches to compute Boolean functions:

Claude Shannon. A Symbolic Analysis of Relay and Switching Circuits, *MSc Thesis*, 1937.

*Resolution* and *unification*: automated reasoning and logic programming:

John Robinson. A Machine-Oriented Logic Based on the Resolution Principle, *Journal of the ACM*, 1965.

**Logic is fundamental to computer science**

*When I was a student, even the topologists regarded mathematical logicians as living in outer space. Today the connections between logic and computers are a matter of engineering practice at every level of computer organization.*

*—Martin Davis, "Influences of Mathematical Logic on Computer Science".*

# Logic is everywhere

- Hardware design.
- Database theory.
- Automated verification.
- Knowledge representation.
- Programming-language theory.
- Complexity theory.
- Constraint satisfaction problems.

## Logic is everywhere

- Hardware design.
- Database theory.
- Automated verification.
- Knowledge representation.
- Programming-language theory.
- Complexity theory.
- Constraint satisfaction problems.
- . . . we won't be focussing on any of the above. Our focus is on foundations of logic.
- But we study the logic from the computational perspective.

1 A historical perspective on logic

2 **Recent applications of computational logic in math, CS and philosophy**

3 Overview of the course

4 Practicalities

# Finding a needle amongst $1,566 \times 10^{349}$ needles

## Erdős discrepancy conjecture

For any $C > 0$ and any infinite sequence $x_1 x_2 x_3 \cdots$ of $+1$'s and $-1$'s, there exist $d, k \in \mathbb{N}$ such that

$$\left| \sum_{1 \leq i \leq k} x_{id} \right| > C.$$

# Finding a needle amongst $1,566 \times 10^{349}$ needles

**Erdős discrepancy conjecture**

For any $C > 0$ and any infinite sequence $x_1 x_2 x_3 \cdots$ of $+1$'s and $-1$'s, there exist $d, k \in \mathbb{N}$ such that

$$\left| \sum_{1 \leq i \leq k} x_{id} \right| > C.$$

- First shown to hold when $\sum_{1 \leq i \leq k} x_{id} \leq 1$ for all $d, k$ by A.R.D. Mathias in 1993. This implies the case $C = 1$.
- Investigated as a Polymath project in 2009-10:
  *"Given how long a finite sequence can be, it seems unlikely that we could answer this question [for $C = 2$] just by a clever search of all possibilities on a computer."*
- B. Konev and A. Lisitsa solve the case $C = 2$ in 2014 using a SAT solver.
- 13Gb of proof that no sequence of length 1161 with discrepancy 2 exists.
- Conjecture proved for all $C > 0$ by T. Tao in 2015.

# Problem from Ramsey theory

**Boolean Pythagorean triples problem**

For which $n \in \mathbb{N}$, can we colour integers $1, 2, \ldots, n$ with red or blue such that no three integers $a, b, c$ with the same colour satisfy the Pythagorean equation $a^2 + b^2 = c^2$?

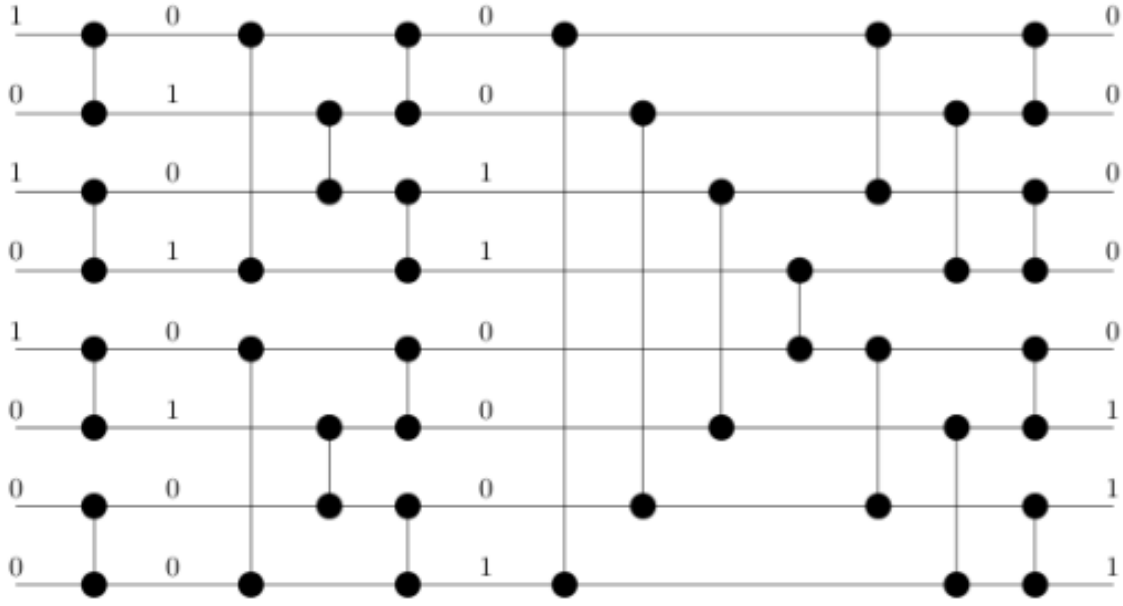
# Problem from Ramsey theory

**Boolean Pythagorean triples problem**

For which $n \in \mathbb{N}$, can we colour integers $1, 2, \ldots, n$ with red or blue such that no three integers $a, b, c$ with the same colour satisfy the Pythagorean equation $a^2 + b^2 = c^2$?

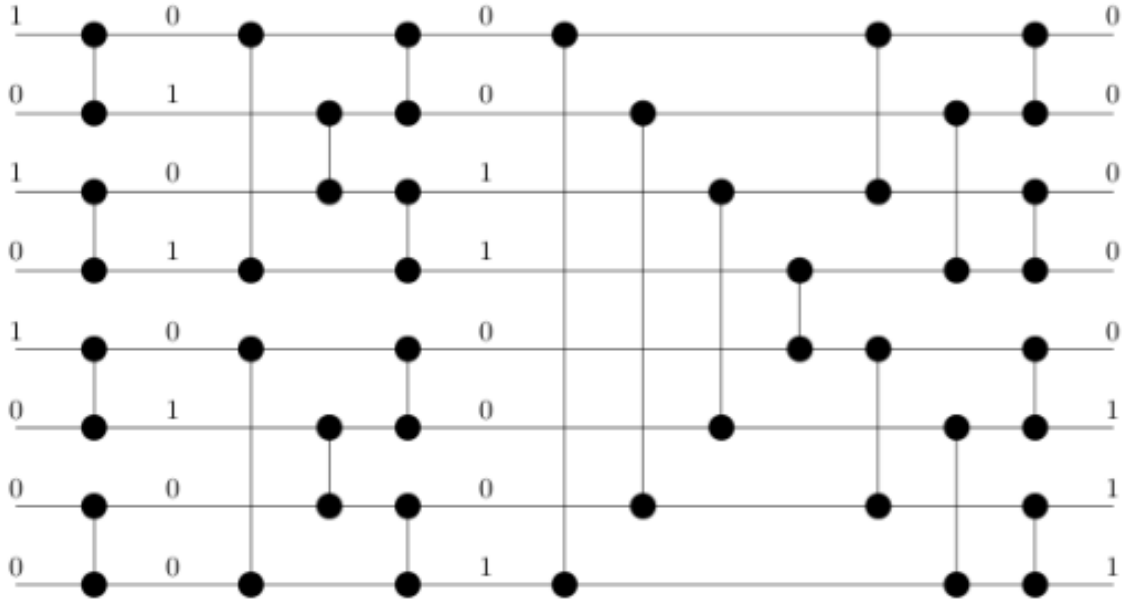
- Heule, Kullman, and Marek resolved this problem using a SAT solver in 2016.
- Their answer: precisely those $n$'s with $n \leq 7824$.
- The solver spent 4 CPU years of computation, and generated a 200 terabyte propositional proof, which was later compressed to 68GB.

# Optimal sorting networks




- Consist of wires and comparators, input flows from left to right.
- Depth is the maximum number of comparators an input can encounter.
- D. Knuth and R. Floyd found optimal depths for $n = 1, \ldots, 8$ in 1960s.
- D. Bundala and J. Závodný found optimal depths for $n = 11, \ldots, 16$ in 2014 using a SAT solver.

# Optimal sorting networks




- Consist of wires and comparators, input flows from left to right.
- Depth is the maximum number of comparators an input can encounter.
- D. Knuth and R. Floyd found optimal depths for $n = 1, \ldots, 8$ in 1960s.
- D. Bundala and J. Závodný found optimal depths for $n = 11, \ldots, 16$ in 2014 using a SAT solver.

| $n$ | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $d$ | 5 | 5 | 6 | 6 | 7 | 7 | 8 | 8 | 9 | 9 | 9 | 9 |

# Leibniz's ontological proof

## Leibniz's proof for the existence of god

1. Definition: God is a being having all perfections.
2. Definition: A perfection is a simple and absolute property.
3. Existence is a perfection.
4. If existence is part of the essence of a thing, then it is a necessary being.
5. If it is possible for a necessary being to exist, then a necessary being does exist.
6. It is possible for a being to have all perfections.
7. Therefore, a necessary being (God) does exist.

# Leibniz's ontological proof

## Leibniz's proof for the existence of god

1. Definition: God is a being having all perfections.
2. Definition: A perfection is a simple and absolute property.
3. Existence is a perfection.
4. If existence is part of the essence of a thing, then it is a necessary being.
5. If it is possible for a necessary being to exist, then a necessary being does exist.
6. It is possible for a being to have all perfections.
7. Therefore, a necessary being (God) does exist.

- Leibniz's "algebra of concepts" formalised in the theorem prover Isabelle/HOL by Bentert, Benzmüller, Streit and Paleo in 2016.
- Showed validity and invalidity of Leibniz's proof in the algebra of concepts depending on the interpretation of some informal statements.

1 A historical perspective on logic

2 Recent applications of computational logic in math, CS and philosophy

3 **Overview of the course**

4 Practicalities

# Focus

- Foundations.
- Computational aspects in particular.
- Computability and algorithms for satisfiability/validity problems.
- Based on the course by Prof Worrell and Prof Haase at Oxford.

# Topics

- Propositional logic.
  - The SAT problem.
  - Polynomial-time algorithms for Horn, 2-SAT, and XOR formulas.
  - Resolution.
  - DPLL with clause learning.
  - Lower bounds for resolution proofs.
  - Compactness theorem.
- First-order logic.
  - Prenex normal form and Skolemisation.
  - Herbrand models and ground resolution.
  - Unification and resolution for predicate logic.
  - Undecidability of satisfiability.
  - Compactness theorem.
  - Logical theories and quantifier elimination.

# Topics - Basics

- Propositional logic.
  - The SAT problem.
  - Polynomial-time algorithms for Horn, 2-SAT, and XOR formulas.
  - Resolution.
  - DPLL with clause learning.
  - Lower bounds for resolution proofs.
  - Compactness theorem.
- First-order logic.
  - Prenex normal form and Skolemisation.
  - Herbrand models and ground resolution.
  - Unification and resolution for predicate logic.
  - Undecidability of satisfiability.
  - Compactness theorem.
  - Logical theories and quantifier elimination.

# Topics – Algorithms or semi-algorithms

- Propositional logic.
  - The SAT problem.
  - Polynomial-time algorithms for Horn, 2-SAT, and XOR formulas.
  - Resolution.
  - DPLL with clause learning.
  - Lower bounds for resolution proofs.
  - Compactness theorem.
- First-order logic.
  - Prenex normal form and Skolemisation.
  - Herbrand models and ground resolution.
  - Unification and resolution for predicate logic.
  - Undecidability of satisfiability.
  - Compactness theorem.
  - Logical theories and quantifier elimination.

# Topics – Computability and complexity

- Propositional logic.
  - The SAT problem.
  - Polynomial-time algorithms for Horn, 2-SAT, and XOR formulas.
  - Resolution.
  - DPLL with clause learning.
  - Lower bounds for resolution proofs.
  - Compactness theorem.
- First-order logic.
  - Prenex normal form and Skolemisation.
  - Herbrand models and ground resolution.
  - Unification and resolution for predicate logic.
  - Undecidability of satisfiability.
  - Compactness theorem.
  - Logical theories and quantifier elimination.

1 A historical perspective on logic

2 Recent applications of computational logic in math, CS and philosophy

3 Overview of the course

4 **Practicalities**

## Course Materials and Information

- The following webpage and KLMS are the main sources:

  `https://github.com/hongseok-yang/logic24`

  Check them regularly.

- Lecture notes, slides, and homework sheets all at the webpage.

- Announcements and Q&A at KLMS.

## Teaching staffs

- Lecturer: Prof Hongseok Yang (Email: hongseok00@gmail.com).
- TA1: Mr Taeyoung Kim (Email: taeyoungkim21@kaist.ac.kr).
- TA2: Mr Jaehui Hwang (Email: hwcmi62@kaist.ac.kr).
- Hongseok's office hours: 6pm - 7pm on Monday in his office (3403, E3-1).
- TAs' office hours: To be announced.

# Evaluation

- Homework (20%) – 4 problem sheets.
- Programming project and report (30%) – optimised SAT solver based on DPLL with clause learning.
- Final exam (50%).

# Programming project

- Implementation of a SAT solver that uses the DPLL algorithm and other optimisations (20%).
- Report on the description and analysis of the implementation and its design decision (10%). At most 4 pages excluding bibliography and figures.
- Submission deadline: 23:59 of 20 May 2024 (Monday).
- Submit it in KLMS.
- All quoted sentences from existing literature should be marked so explicitly with pointers to their sources. If not, zero mark.

# Honour code

- We adopt a very strict policy for handling the violation of the standard honour code.
- If a student is found to cheat in a test or copy answers or code from friends' or other souces, the student will get F.

# Recommended textbooks

- Lecture notes in the course webpage.
- *'The Calculus of Computation: Decision Procedures with Applications to Verification'*, A. Bradley and Z. Manna.
- *'Logic in Computer Science: Modelling and Reasoning about Systems'*, M. Huth and M. Ryan.
- *'The Art of Computer Programming, 4B'*, D. Knuth.

# Further Literature

- *'Logicomix: An Epic Search for Truth'*, A. Doxiadis and C. Papadimitriou.
- *'Handbook of Practical Logic and Automated Reasoning'*, J. Harrison.
- *'Mathematical Logic for Computer Science'*, M. Ben-Ari.
- *'Logic for Computer Scientists'*, U. Schöning.
- *'Gödel, Escher, Bach: an Eternal Golden Braid'*, D. Hofstadter.

## Questions to check your background knowledge

- Let $X$ be a non-negative real-valued random variable, and $c > 0$ be a positive real number. Prove that

$$P(X > c) \leq \frac{\mathbb{E}(X)}{c}.$$

- What is the time complexity of solving a system of linear equations $Ax = b$ using Gaussian elimination?
- Explain the following notions: NP, NP-complete, regular, recursive, and recursively enumerable.
- Consider a subset $L \subseteq \Sigma^*$ over a finite $\Sigma$. Show that if both $L$ and $L^c$ are recursively enumerable, then $L$ is recursive.
- Assume that $L, L' \subseteq \Sigma^*$ over a finite $\Sigma$ are regular. Let $f : \Sigma \to \Sigma'$ be a map to a finte $\Sigma'$. Show that $L \cap L'$ is regular. Also, prove that $f(L) = \{f(x_1) \dots f(x_n) \mid x_1 \dots x_n \in L\}$ is regular.
- Prove: the program (`y=1; while (x>0) y=y*x; x=x-1;`) computes the factorial of `x`.