CS621: Logic and applications

Ramchandra Phawade
Department of Computer Science and Engineering
IIT Dharwad, India

August 1, 2023

Time slots

- 7A : Mon: 14:00 to 14:55
- 7B : Tue: 10:30 to 11:25
- 7C : Thu: 09:00 to 10:25

Evaluation Scheme

1. Assignments+Quiz : 30%
   (2 assignments + 1 Quiz with 10% weight for each)
   Out of two one is a programming assignment.
2. Midsem : 30%
3. Endsem : 40%

Textbooks and References

1. A mathematical introduction to logic
   Herbert B. Enderton
   Elsevier
2. Logic in Computer Science
   Authors: Huth and Ryan
   Cambridge University Press
3. Z3 tool
   SAT/SMT by example by Dennis Yurichev
   https://yurichev.com/SAT_SMT.html

Additional material:

- Logicomix : https://en.wikipedia.org/wiki/Logicomix
- Engines of Logic by Martin Davis

Why should one study this course?

before jumping into the answers/applications,
let us take a look at the history.

Gottfried Leibniz

Born : Leipzig, Germany ; 1646 Died : Hanover, Germany; 1716

Many contributions : philosophy, calculus, logic.

Gottfried Leibniz

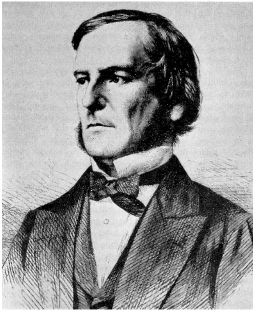
Believed : Human reasoning could be reduced to calculations.
The dream was – Let us compute. (build Machines)

How to represent human reasoning?

Logic! Symbols will have meanings. Create a system (algebra) to manipulate the symbols. Leibinz : calculus ratiocinator.

George Boole

Born : 1815, London; Died : 1864, Ireland.



Contributions:

- Boolean Logic – the basis of calculations in modern computer.
- Turns logic into algebra (Leibniz's dream !)
- Can not caputre all of human thoughts.

Gottlob Frege

Born : 1848, Germany; Died : 1925, Germany.

Contributions:

- Predicate Logic – the modern logic. Language of Mathematics.
- $\forall a, b, c, n \; [\; (a, b, c > 0 \wedge n > 2) \;\rightarrow\; a^n + b^n \neq c^n \;]$
- More powerful than boolean logic. But closer to Leibniz's dream.

Gottlob Frege

- Language of mathematics - predicate logic
- Developed axiomatization of set theory.
- Expressing set theory in terms of logic.

Georg Cantor

Born : 1845, Russia. Died : 1918, Germany.

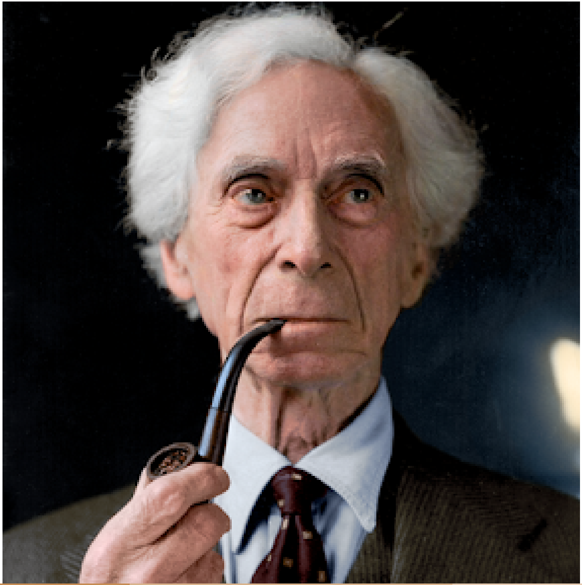
Georg Cantor

Contributions: Infinite sets, cardinality.

- Set of even numbers is of the same size of natural numbers.
- nonintuitive !
- Fierce opposition form Kronecker, Konig, Poincare, Weyl.

Bertrand Russell (1872-1970)

Born : 1872, England; Died : 1970, England



Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \ \mid \ \text{p is shaved by the Barber}\}$.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \ \mid\ \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{ \ p \ \mid \ \text{p \ is \ shaved \ by \ the \ Barber}\}$.
- Assume : Barber shaves himself
  Barber is shaved by the barber.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \ \mid \ \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \ \mid \ \text{p is shaved by the Barber}\}.$
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{ \ p \ \mid \ \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \mid \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

  Barber does not shave himself.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \mid \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

  Barber does not shave himself.

  Then Barber does not belong to the set S.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \mid \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

  Barber does not shave himself.

  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{ p \mid \text{p is shaved by the Barber} \}$.
- Assume : Barber shaves himself
  Barber is shaved by the barber.
  Therefore, Barber belongs to S.
  But,
  Barber shaves all those who do not shave themselves.
  Barber does not shave those who shave themselves.
  Barber does not shave himself.
  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.
  Barber is not shaved by the barber.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \mid \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

  Barber does not shave himself.

  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.

  Barber is not shaved by the barber.

  So, Barber does not belongs to S.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \mid \text{ p is shaved by the Barber}\}$.
- Assume : Barber shaves himself
  Barber is shaved by the barber.
  Therefore, Barber belongs to S.
  But,
  Barber shaves all those who do not shave themselves.
  Barber does not shave those who shave themselves.
  Barber does not shave himself.
  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.
  Barber is not shaved by the barber.
  So, Barber does not belongs to S.
  But Barber shaves all those who do not shave themselves.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p\ \mid\ \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

  Barber does not shave himself.

  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.

  Barber is not shaved by the barber.

  So, Barber does not belongs to S.

  But Barber shaves all those who do not shave themselves.

  Barber shaves himself.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{ \ p \ | \ \text{p is shaved by the Barber} \}$.
- Assume : Barber shaves himself

  Barber is shaved by the barber.

  Therefore, Barber belongs to S.

  But,

  Barber shaves all those who do not shave themselves.

  Barber does not shave those who shave themselves.

  Barber does not shave himself.

  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.

  Barber is not shaved by the barber.

  So, Barber does not belongs to S.

  But Barber shaves all those who do not shave themselves.

  Barber shaves himself.

  Therefore Barber belongs to S.

Russell's paradox

- Barber shaves all those who do not shave themselves.
- $S = \{\ p \mid \text{p is shaved by the Barber}\}$.
- Assume : Barber shaves himself
  Barber is shaved by the barber.
  Therefore, Barber belongs to S.
  But,
  Barber shaves all those who do not shave themselves.
  Barber does not shave those who shave themselves.
  Barber does not shave himself.
  Then Barber does not belong to the set S.
- Assume : Barber does not shave himself.
  Barber is not shaved by the barber.
  So, Barber does not belongs to S.
  But Barber shaves all those who do not shave themselves.
  Barber shaves himself.
  Therefore Barber belongs to S.
  Sets are not defined properly.

David Hilbert

Born : 1862, Könisberg   Died : 1943 : Göttingen, Germany

Program for securing foundations of Mathematics.

Securing foundations of Mathematics: David Hilbert

Securing foundations of Mathematics: David Hilbert

- Formulation of mathematics (Axiomatization and proof calculus).

Securing foundations of Mathematics: David Hilbert

- Formulation of mathematics (Axiomatization and proof calculus).
- Completeness : all true statements about mathematics should be prove in the formalism.

Securing foundations of Mathematics: David Hilbert

- Formulation of mathematics (Axiomatization and proof calculus).
- Completeness : all true statements about mathematics should be prove in the formalism.
- Consistency : No contradiction can be derived in the formulation.

Securing foundations of Mathematics: David Hilbert

- Formulation of mathematics (Axiomatization and proof calculus).
- Completeness : all true statements about mathematics should be prove in the formalism.
- Consistency : No contradiction can be derived in the formulation.
- Decidability : There should be an algorithm to decide the truth of mathematical statements.

Securing foundations of Mathematics: David Hilbert

- Formulation of mathematics (Axiomatization and proof calculus).
- Completeness : all true statements about mathematics should be prove in the formalism.
- Consistency : No contradiction can be derived in the formulation.
- Decidability : There should be an algorithm to decide the truth of mathematical statements.

Principia Mathematica by Russell and Whitehead : One attempt in this direction.

Consistency of arithmetic: David Hilbert

- $\forall a, b, c, n \; [\; (a, b, c > 0 \wedge n > 2) \;] \; \rightarrow \; a^n + b^n \neq c^n$
  - Is there an finite and complete axiomatization of arithmetic which is consistent? (1920)

Kurt Gödel

Born : Brünn (now Czech Republic), 1906; Died : Princeton, 1978.

Major Contributions : Answer is NO!

Godel : Incompleteness Theorems

- First incompleteness theorem–arithmetic.
  Any consistent formalism strong enough in which sufficint arithemetic can be carried out is not complete.

Godel : Incompleteness Theorems

- First incompleteness theorem–arithmetic.
  Any consistent formalism strong enough in which sufficint arithemetic can be carried out is not complete.
- Second incompleteness theorem: Any such formalism can not prove its own consistency.

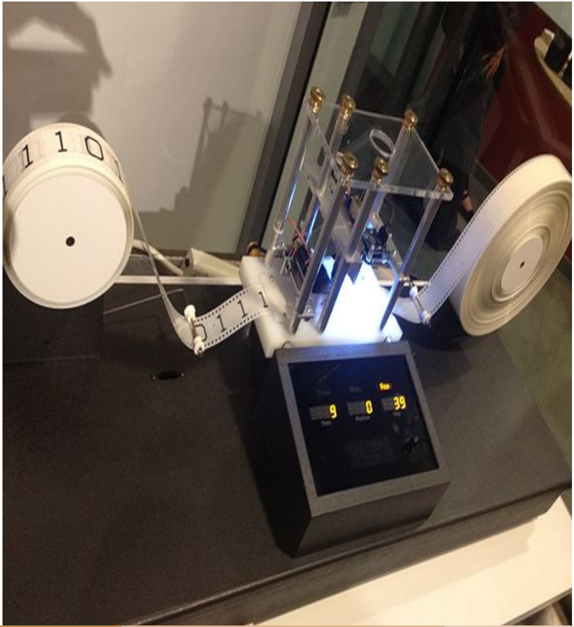
Entscheidungsproblem : David Hilbert

- $\forall a, b, c, n \; [\; (a, b, c > 0 \wedge n > 2) \;] \;\rightarrow\; a^n + b^n \neq c^n$
- Is there an "algorithm" that can take such a mathematical statement as input and say if it is true or false. (1900)

Mathematical notion of computation : Turing Machines

Figure: Alan Turing (1912-1954)

Turing machines



Logic Applications

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.
- As a database Query language -Codd's relational databases, SQL.

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.
- As a database Query language -Codd's relational databases, SQL.
- Programming languages : design, analysis, implementation-type theory, separation logics, concurrent separation logics.
  RUST (2020 ACM doctoral thesis awaerd), verified compilers,..

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.
- As a database Query language -Codd's relational databases, SQL.
- Programming languages : design, analysis, implementation-type theory, separation logics, concurrent separation logics.
  RUST (2020 ACM doctoral thesis awaerd), verified compilers,..
- Reasoning about knowledge – epistemic logics, beliefs. 1950s by Hintikka.

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.
- As a database Query language -Codd's relational databases, SQL.
- Programming languages : design, analysis, implementation-type theory, separation logics, concurrent separation logics.
  RUST (2020 ACM doctoral thesis awaerd), verified compilers,..
- Reasoning about knowledge – epistemic logics, beliefs. 1950s by Hintikka.
  I know that you know it, but you do not know that I know that you know it.

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.
- As a database Query language -Codd's relational databases, SQL.
- Programming languages : design, analysis, implementation-type theory, separation logics, concurrent separation logics.
  RUST (2020 ACM doctoral thesis awaerd), verified compilers,..
- Reasoning about knowledge – epistemic logics, beliefs. 1950s by Hintikka.
  I know that you know it, but you do not know that I know that you know it.
  protocols – design, verification; AI.

Logic Applications

- Descriptive complexity–characterizations of complexity classes
  R. Fagin (1974), Immerman, Moshe Vardi.
- As a database Query language -Codd's relational databases, SQL.
- Programming languages : design, analysis, implementation-type theory, separation logics, concurrent separation logics.
  RUST (2020 ACM doctoral thesis awaerd), verified compilers,..
- Reasoning about knowledge – epistemic logics, beliefs. 1950s by Hintikka.
  I know that you know it, but you do not know that I know that you know it.
  protocols – design, verification; AI.
- Automated verification of chips: LTL, CTL, automata theoretic approaches.

Formal Verification

- Systems – Automata, Different kinds of machines, programs,
- Property– specified by some suitable logic
- Does the system satisfy the given property?

Formal Verification

- Systems – Automata, Different kinds of machines, programs,
- Property– specified by some suitable logic
- Does the system satisfy the given property?
- Automated systems

Formal Verification

- Systems – Automata, Different kinds of machines, programs,
- Property– specified by some suitable logic
- Does the system satisfy the given property?
- Automated systems
  Is goal achievable? planning, vefification.

Formal Verification

- Systems – Automata, Different kinds of machines, programs,
- Property– specified by some suitable logic
- Does the system satisfy the given property?
- Automated systems
  Is goal achievable? planning, vefification.
  verification of systems with machine learning components.
  2020 ACM doctoral thesis award.

What is in the course?

- formalization of proofs, theory, consistency, completeness, soundness, decidability
- propositional logic
- FOL – proof mechanism, undecidability, expressibility
- Decidable fragments–Presburger arithmetic
- Decision procedures for First Order Theories. SAT/SMT solvers.

Thank you