Formal Proof Verification

An Introduction to Computer-Assisted Mathematics

Christoph Spiegel

Winter Semester 2024/25

Freie Universität Berlin | Zuse Institute Berlin

Table of contents

Why Do We Need Formal Verification?

The Growing Complexity of Mathematics

Modern proofs can span hundreds or thousands of pages

- **Classification of finite simple groups (1955-2004)**
  tens of thousands of pages by around 100 authors
- **Mochizuki's ABC conjecture proof (2012)**
  500 pages, validity still disputed (Quanta article)
- **Geometric Langlands conjecture proof (2024)**
  800+ pages over 5 papers (Quanta article)

The Growing Complexity of Mathematics

Modern proofs can span hundreds or thousands of pages

- **Classification of finite simple groups (1955-2004)**
  tens of thousands of pages by around 100 authors
- **Mochizuki's ABC conjecture proof (2012)**
  500 pages, validity still disputed (Quanta article)
- **Geometric Langlands conjecture proof (2024)**
  800+ pages over 5 papers (Quanta article)

Traditional peer review is reaching its limits

- Hales' proof of the Kepler conjecture (1998) was published in Annals after 5 years with a note that complete verification was not possible.
- Annals also published contradictory results on smooth toric variety in 2004 and 2006 (the former was finally retracted in 2023).
- The Adam optimizer (ICLR 2015) has received 207,570 citations to date. It was shown in 2017 to not converge for all convex objectives.
- Even peer review of normal papers takes between 6 months and 2 years.

What is Formal Verification of Mathematics?

Traditional Mathematics

- **Written in natural language** with a commonly agreed upon structure through *Lemma, Proposition, Theorem, Conjecture, Remark, Proof, Fact, Remark, Example, ...*
- **Different levels of detail**: minimal for specialists (papers), moderate for researchers (surveys), comprehensive for students (textbooks)
- **Unspecified foundations**: often little agreement or discussion of the axiomatic assumptions.
- **Verified by humans**: *"In mathematics you don't understand things. You just get used to them."*

\+ Easy to write and read

– Can contain errors and requires a common base of knowledge

What is Formal Verification of Mathematics?

Traditional Mathematics

- **Written in natural language** with a commonly agreed upon structure through *Lemma, Proposition, Theorem, Conjecture, Remark, Proof, Fact, Remark, Example, ...*
- **Different levels of detail**: minimal for specialists (papers), moderate for researchers (surveys), comprehensive for students (textbooks)
- **Unspecified foundations**: often little agreement or discussion of the axiomatic assumptions.
- **Verified by humans**: *"In mathematics you don't understand things. You just get used to them."*

\+ Easy to write and read

– Can contain errors and requires a common base of knowledge

Formal Verification

- **Written in formal language** usually following (functional) programming principles.
- **Uniform level of detail**: every step must be justified and can be traced down to its axioms.
- **Specified foundations**: clearly specified axiomatic assumptions (classical vs intuitionistic) and logical framework (type theory vs set theory).
- **Verified by computer** with absolute certainty modulo errors in the core checker.

\+ Guaranteed verification and proofs traceable down to axioms

– (Currently) hard to write and read

Case Studies

Four Color Theorem (1976 / 2005)

- Any map can be colored with at most 4 colors such that no two adjacent regions have the same color. Two early proofs (1879, 1880) were widely accepted but both shown incorrect after 11 years
- Finally proved by Appel and Haken (1976), which was controversial as it was the first major theorem proved using a computer and had a complex human-verifiable portion.
- Werner and Gonthier (2005) formalized the proof in the Coq proof assistant, removing the need to trust separate computer programs.

Case Studies

Four Color Theorem (1976 / 2005)

- Any map can be colored with at most 4 colors such that no two adjacent regions have the same color. Two early proofs (1879, 1880) were widely accepted but both shown incorrect after 11 years
- Finally proved by Appel and Haken (1976), which was controversial as it was the first major theorem proved using a computer and had a complex human-verifiable portion.
- Werner and Gonthier (2005) formalized the proof in the `Coq` proof assistant, removing the need to trust separate computer programs.

Prime Number Theorem (1896 /2005)

- The number of primes less than x is asympt. equal to $x/\log(x)$.
- First proven by Hadamard and de la Vallée Poussin in 1896.
- Selberg and Paul Erdős obtained more elementary proofs in 1948.
- The simplified proof was verified in `Isabelle/HOL` (2005) by Avigad and others, providing a formal machine-checked proof.

Case Studies (contd.)

Jordan Curve Theorem (1887 / 2005)

- Every simple closed curve divides the plane into a bounded (interior) and one unbounded (exterior) region and every continuous path connecting one region to the other intersects the curve.
- For decades it was debated if Jordan's proof from 1887 was correct.
- *"Although the JCT is one of the best known topological theorems, there are many, even among professional mathematicians, who have never read a proof of it."* Tverberg (1980)
- Hales (2005) verified it in `HOL Light` over 60,000 lines. The same year it was also verified in `Mizar` system team in 6,500 lines.

Case Studies (contd.)

Feit-Thompson theorem (1962 / 2012)

- Feit (1962) and Thompson (1963) proved that every finite group of odd order is solvable in 255 pages long proof.
- A revised proof was published accross two books, (Bender & Glauberman 1994) and (Peterfalvi 2000, part I), which is even longer, but written in a more leisurely style.
- A fully formal proof in `Coq` was announced 2012 by Georges Gonthier and fellow researchers at Microsoft Research and Inria.

The Kepler Conjecture (1998 / 2015)

- Proof by Thomas Hales (1998) which in part checks 23 000 nonlinear inequalities in computer
- Accepted by Annals of Mathematics after 5 years of review but with a note stating that full verification was not possible
- Led to formal verification project with 20+ people (Flyspeck project) using a combination of `Isabelle` and `HOL Light` (completed 2015)

Case Studies (contd.)

Liquid Tensor Experiment

- Peter Scholze suggested verifying a result of Clausen and his on liquid vector space in Lean in 2020. The project was completed in 2022.
- The project gained widespread attention in the mathematical community and demonstrated feasibility of formalizing *complex proofs about complex mathematical objects* (more or less) in real time.
- Peter Scholze noted that the formalization experience taught him *"what actually makes the proof work"*

Fermat's Last Theorem

- Fermat's Last Theorem (1637) states that no three positive integers a, b, and c satisfy the equation $a^n + b^n = c^n$ for any $n \geq 2$.
- The first successful proof is due to Andrew Wiles and takes over 100 pages that were published in Annals of Mathematics in 1995.
- It is the last non-formalized result on the list of "top 100" mathematical theorems. Kevin Buzzard is leading a project to formalize it in Lean.

The Landscape of Formal Proof Verification

Formal Systems and Verification

A **formal system** is an abstract structure consisting of:

- **Formal language**: A set of well-formed formulas (strings of symbols from an alphabet formed by a formal grammar)
- **Deductive system**: Rules of inference and axioms that allow for the derivation of theorems

Formal Systems and Verification

A **formal system** is an abstract structure consisting of:

- **Formal language**: A set of well-formed formulas (strings of symbols from an alphabet formed by a formal grammar)
- **Deductive system**: Rules of inference and axioms that allow for the derivation of theorems

Formal proofs are sequences of well-formed formulas (WFF) using axioms or previous WFFs. **Formal verification** is the application of formal proofs to show that a system satisfies its specification.

Formal systems have applications beyond mathematics:

- CompCert – Formally verified C compiler
- ProVerif – automatic cryptographic protocol verifier
- DAIDALUS – NASA formally verifies algorithms for UAV
- The FM 8501 was the first formally verified microprocessor
- Mondex Smart Card – Payment system certified to ITSEC Level E6

Logic Foundations for Formal Verification

- Logical Systems (Hierarchy by Quantification)
  - **Propositional Logic**: Statements without quantification. $(P \vee Q)$
  - **First-Order Logic (FOL)**: Quantification over objects: $\forall x \exists y\, R(x, y)$
  - **Second-Order Logic**: Quantification over properties: $\exists P \forall x\, P(x)$
  - **Higher-Order Logic (HOL)**: Quantifies over properties of properties.

Logic Foundations for Formal Verification

- Logical Systems (Hierarchy by Quantification)
  - **Propositional Logic**: Statements without quantification. $(P \vee Q)$
  - **First-Order Logic (FOL)**: Quantification over objects: $\forall x \exists y\ R(x, y)$
  - **Second-Order Logic**: Quantification over properties: $\exists P \forall x\ P(x)$
  - **Higher-Order Logic (HOL)**: Quantifies over properties of properties.

- Set Theory (all objects are constructed from sets)
  - Historically dominant foundation for mathematics. Provides axioms for constructing all mathematical objects from sets.
  - Zermelo-Fraenkel set theory (ZFC) addresses foundational issues.
  - Expressible in both first-order and second-order logic.

Logic Foundations for Formal Verification

- Logical Systems (Hierarchy by Quantification)
  - **Propositional Logic**: Statements without quantification. $(P \vee Q)$
  - **First-Order Logic (FOL)**: Quantification over objects: $\forall x \exists y\, R(x, y)$
  - **Second-Order Logic**: Quantification over properties: $\exists P \forall x\, P(x)$
  - **Higher-Order Logic (HOL)**: Quantifies over properties of properties.

- Set Theory (all objects are constructed from sets)
  - Historically dominant foundation for mathematics. Provides axioms for constructing all mathematical objects from sets.
  - Zermelo-Fraenkel set theory (ZFC) addresses foundational issues.
  - Expressible in both first-order and second-order logic.
- Type Theory (all objects have a type)
  - Resolve paradoxes by assigning types to mathematical entities.
  - Rules instead of axioms; Closely linked to intuitionistic logic.
  - **Simple Type Theory** closely corresponds to Higher-Order Logic while **Dependent Type Theory** allows types to depend on terms.

Classical vs Intuitionistic Logic

- **Classical Logic (Non-Constructive)**:
  - Law of Excluded Middle: always either P or $\neg P$.
  - Axiom of Choice (AoC) accepted; allows non-constructive proofs and existence arguments without explicit constructions.

Classical vs Intuitionistic Logic

- **Classical Logic (Non-Constructive)**:
  - Law of Excluded Middle: always either P or $\neg P$.
  - Axiom of Choice (AoC) accepted; allows non-constructive proofs and existence arguments without explicit constructions.

- **Intuitionistic Logic (Constructive)**:
  - A statement is true only if there is an explicit construction (proof).
  - $\neg P$ means "P is refutable", not that we know P
  - AoC controversial; acceptable if explicit choices constructed.

Classical vs Intuitionistic Logic

- **Classical Logic (Non-Constructive)**:
  - Law of Excluded Middle: always either P or $\neg P$.
  - Axiom of Choice (AoC) accepted; allows non-constructive proofs and existence arguments without explicit constructions.

- **Intuitionistic Logic (Constructive)**:
  - A statement is true only if there is an explicit construction (proof).
  - $\neg P$ means "P is refutable", not that we know P
  - AoC controversial; acceptable if explicit choices constructed.

  Dependent Type Theory is inherently constructive, though classical logic can still be used explicitly if added as axioms.

  Foundations matter and impact proof formalization and verification and surprisingly subtle points can matter (e.g. the notion of equality).

Major Formal Proof Verification Systems

System	Year	Foundation/Logic	Math Library
Lean	2013	DTT / Int. + Classical	mathlib
Coq	1989	CIC (DTT) / Intuitionistic	MathComp, UniMath
Isabelle	1986	HOL / Type Theory	AFP
HOL Light	1996	HOL / Type Theory	Flyspeck
Mizar	1973	TG / Classical	Mizar Mathematical Lib.
Agda	2007	UDTT / Intuitionistic	Agda Std. Lib.
nqthm/ACL2	1990	FORL / Classical	ACL2 Books
PVS	1992	HOL+DTT / Classical	NASA Libraries
Metamath	2005	ZFC / Classical	set.mm

The Future of Computer-Assisted Mathematics

Computers vs Humans

Paraphrasing Kevin Buzzard:

- Computers have been better at *computing* for 60 years.
- Computers have been better at chess for 30 years.
- Computers have been better at Go for 10 years.
- Computers are now better than most humans at mathematics, but don't (yet) know their own limits.

Does Formal Proof Verification open the door to fully automated reasoning, perhaps surpassing even the best mathematicians?

Computational tools actually have a long history ...

1965 Birch and Swinnerton-Dyer conjecture based on **numerical evidence**

1976 **Proof by exhaustion** of the four color theorem

1996 McCune prove that Robbins algebras are boolean using **ATP**

1998 Hales proves the Kepler conjecture by systematically solving **LPs**

2016 Resolution of Boolean Pythagorean triples problem using **SAT solvers**

2017 Schur number five is determined in a two petabytes proof

2021 An important connection in Knot theory is discovered using **ML**

2024 The fifth Ramsey number is at most 46

In 2024 Deepmind (Google) announced that they can solve (certain) IMO problems at silver-medal level.

The state right now ...

- Right now **full automation still seems far away** and formal proof verification can be quite tedious.
- There are **increasingly more powerful tools** that make the task somewhat easier, but **understanding the fundamentals is still very much necessary**.
- LLMs have a decent understanding of FPV languages (as for most programming languages) but **programming a proof is even less forgiving than normal programming**.
- Mathematics has become a **benchmark to demonstrate reasoning** capabilities and established mathematicians are starting to work towards a **future in which the work of a mathematician might look very different** (making FPV knowledge very valuable).

Course Outline

`P01` Introduction

`P02` Logic (Reasoning, Connectives, Negation, Quantifiers)

`P03` Set Theory (Subsets, Complements, Intersections, Unions, Set Families)

`P04` Type Theory (Dependent and Inductive types, Structures, Type Classes, Prop.)

`P05` Natural Numbers (Definition, Addition, Multiplication, Powers, Inequalities)

`P06` **Project:** Infinitude of primes (Euclid, Folklore, Fürstenberg, Goldbach)

`P07` **Project:** Mantel's Theorem (Handshake Lemma, CS Proof, AM-GM Proof)

Let's look at some concrete examples ...