How Computers prove Theorems and why it Matters

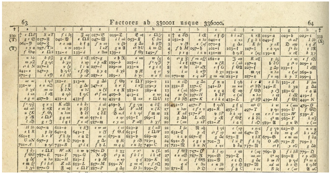
Nima Rasekh

Universität Greifswald

18.12.2024

What is a Computer?

1. In 1797 oder even in 1950 it would be a Person:

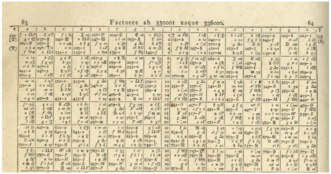


(a) Prime number table by Felkel–Vega (1797)

(b) Human computers (1950)

What is a Computer?

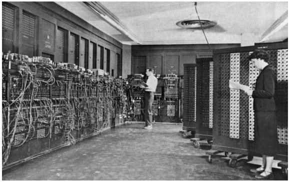
1. In 1797 oder even in 1950 it would be a Person:


63 Factores ab 330001 usque 336000. 64

(a) Prime number table by Felkel–Vega (1797)

(b) Human computers (1950)

2. Starting from the 40s it slowly becomes an electronic device.

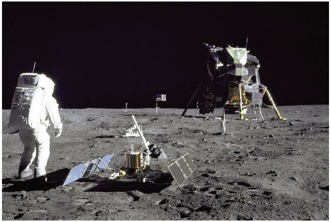


(a) ENIAC (1945)

(b) IBM Blue Gene (2006)

Computers in Science

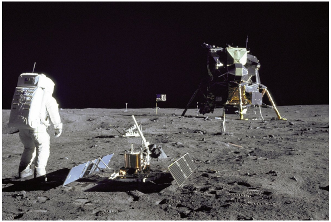
(a) Harvard Computers (1890)



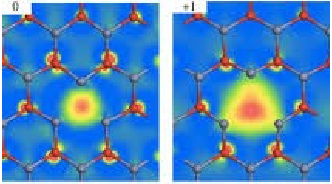
(b) Apollo Guidance Computer (1969)

Computers in Science

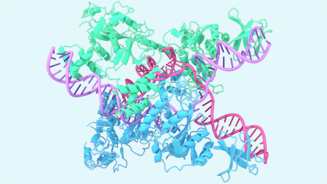
(a) Harvard Computers (1890)



(b) Apollo Guidance Computer (1969)

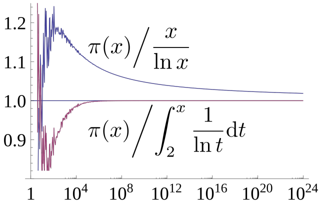


(c) CASTEP (1990)

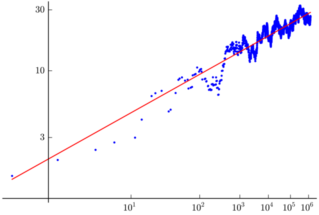


(d) AlphaFold (2018)

Computers in Mathematics

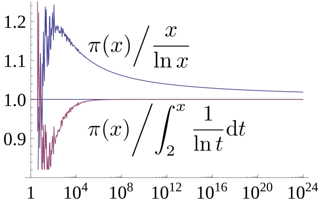


(a) Prime number theorem (1798)

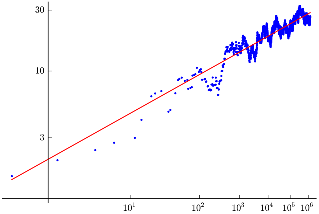


(b) Birch und Swinnerton-Dyer conjecture (1965)

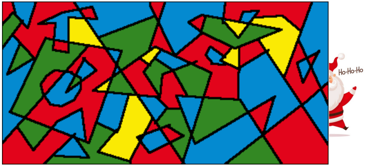
Computers in Mathematics



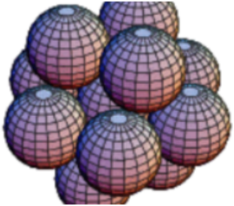
(a) Prime number theorem (1798)



(b) Birch und Swinnerton-Dyer conjecture (1965)



(c) Four color theorem (1967)



(d) Kepler conjecture (1998)

Something is missing: Proofs!

Computers have many applications:

1. Computation
2. Conjecture
3. Proofs through checking finite cases

Something is missing: Proofs!

Computers have many applications:

1. Computation
2. Conjecture
3. Proofs through checking finite cases

What is missing?

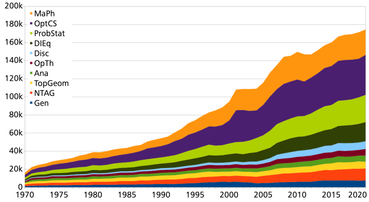
- Proofs!

Questions:

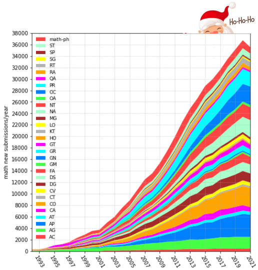
1. Can computers prove things and how?
2. Can computers automate proofs?
3. Can computers prove new theorems?

Why should Computers prove things?

1. There is more mathematics:



Actively publishing persons 1970 - 2020
Klaus Hulek, Olaf Teschke. How do mathematicians publish? EMS Mag. 129



Number of math papers on arXiv
https://info.arxiv.org/help/stats/2021_by_area/index.html

Losing oversight, but more suitable for ML algorithms!

2. Intricate computations cannot be checked.
3. Math is more complicated: we can make more mistakes.

2 Examples

1. **Homotopy Hypothesis:** Voevodsky[1]
   - Published Proof by Voevodsky and Kapranov in 1991
   - Counter example by Simpson in 1998
   - No concrete mistake found until 2013

   "A technical argument by a trusted author, which is hard to check and looks similar to arguments known to be correct, is hardly ever checked in detail."

2. **Condensed Mathematics:** Scholze[2]
   - Complicated proof, hard to check
   - Motivated *Liquid Tensor Experiment*

   "I learnt that it can now be possible to take a research paper and just start to explain lemma after lemma to a proof assistant, until you've formalized it all! I think this is a landmark achievement."

---

1 https://www.ias.edu/ideas/2014/voevodsky-origins

2 https://xenaproject.wordpress.com/2020/12/05/liquid-tensor-experiment/

How can a Computer proof something?

Mathematical statements $\xrightarrow[\text{}]{\text{translated to}}$ Type theory (syntax) $\xrightarrow[\text{computers}]{\text{entered into}}$ Programming code e.g. Lean, Coq, ... $\xrightarrow[\text{assistants}]{\text{via proof}}$ Formally verified mathematics

How can a Computer proof something?

Mathematical statements $\rightsquigarrow$ translated to $\rightsquigarrow$ Type theory (syntax) $\rightsquigarrow$ entered into computers $\rightsquigarrow$ Programming code e.g. Lean, Coq, ... $\rightsquigarrow$ via proof assistants $\rightsquigarrow$ Formally verified mathematics

Example

For all sentences p, q, r we have: $p \wedge (q \vee r) \leftrightarrow (p \wedge q) \vee (p \wedge r)$

```
example (p q r : Prop) : p ∧ (q ∨ r) ↔ (p ∧ q) ∨ (p ∧ r) := by
  apply Iff.intro
  . intro h
    apply Or.elim (And.right h)
    . intro hq
      apply Or.inl
      apply And.intro
      . exact And.left h
      . exact hq
    . intro hr
      apply Or.inr
      apply And.intro
      . exact And.left h
      . exact hr
  . intro h
    apply Or.elim h
    . intro hpq
      apply And.intro
      . exact And.left hpq
      . apply Or.inl
        exact And.right hpq
    . intro hpr
      apply And.intro
      . exact And.left hpr
      . apply Or.inr
        exact And.right hpr
```

```
▼ Tactic state
1 goal
▼ case mpr.left.right.h
p q r : Prop
h : p ∧ q ∨ p ∧ r
hpq : p ∧ q
⊢ q
▼ Expected type
p q r : Prop
h : p ∧ q ∨ p ∧ r
hpq : p ∧ q
⊢ ∀ {a b : Prop}, a → a ∨ b
▶ All Messages (26)
```

But is there Formalization of real Mathematics?

Here are some recent developments:

1. **Number theory:** Fermats Last Theorem, since 2024[3]
2. **Analysis:** Carleson's Theorem, since 2024[4]
3. **Algebra:** Liquid Tensor Experiment, 2020 - 2022[5]
4. **Geometry:** Sphere Eversion, 2020 - 2022[6]
5. **Topology:** Homotopy group $\pi_4(S^3)$, 2016 - 2022[7]
6. **Geometry:** Kepler conjecture, 2003 - 2015[8]

---

3 https://github.com/ImperialCollegeLondon/FLT

4 https://github.com/fpvandoorn/carleson

5 https://github.com/leanprover-community/lean-liquid

6 https://github.com/leanprover-community/sphere-eversion

7 https://github.com/agda/cubical/tree/master/Cubical/Homotopy/Group/Pi4S3

8 https://github.com/flyspeck/flyspeck

Where can this lead us?

1. Journal Submission with Formalization[9]
2. Automatic Proofs via AI[10]
3. Applications in Teaching[11,12,13,14,15]

```
import Verbose.English.ExampleLib
import Verbose.English.Statements

set_option verbose.suggestion_widget true

Exercise "Continuity implies sequential continuity"    declaration uses 'sorry'
  Given: (f : ℝ → ℝ) (u : ℕ → ℝ) (x₀ : ℝ)
  Assume: (hu : u converges to x₀) (hf : f is continuous at x₀)
  Conclusion: (f ∘ u) converges to f x₀
Proof:
  Let's prove that ∀ ε > 0, ∃ N, ∀ n ≥ N, |(f ∘ u) n - f x₀| ≤ ε
  Fix ε > 0
  By hf applied to ε using that ε > 0 we get δ such that (δ_pos : δ > 0) (hδ : ∀ (x : ℝ), |x - x₀| ≤ δ → |f x - f x₀| ≤ ε)
  By hu applied to δ using that δ > 0 we get N such that hN : ∀ n ≥ N, |u n - x₀| ≤ δ
  Let's prove that N works: ∀ n ≥ N, |(f ∘ u) n - f x₀| ≤ ε

  sorry
QED
```

```
▼ Tactic state
1 goal
f : ℝ → ℝ
u : ℕ → ℝ
x₀ : ℝ
hu : u converges to x₀
hf : f is continuous at x₀
ε : ℝ
ε_pos : ε > 0
δ : ℝ
δ_pos : δ > 0
hδ : ∀ (x : ℝ), |x - x₀| ≤ δ → |f x - f x₀| ≤ ε
N : ℕ
hN : ∀ n ≥ N, |u n - x₀| ≤ δ
⊢ ∀ n ≥ N, |(f ∘ u) n - f x₀| ≤ ε
▼ Suggestions
Use shift-click to select sub-expressions.
```

9 https://xenaproject.wordpress.com/2023/11/04/formalising-modern-research-mathematics-in-real-time/

10 https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/

11 https://impermeable.github.io/

12 https://gihanmarasingha.github.io/modern-maths-pages/

13 https://cs22.io/

14 https://hhu-adam.github.io/

15 https://github.com/PatrickMassot/lean-verbose

What does that mean for me?

Here are some possible first steps:

1. Stay updated.
   - Keep track of milestones: LTE[16], AlphaProof[17], ...
   - Formalization at (German) universities: Bonn[18], Düsseldorf[19]
2. Try Lean.[20]
3. Formalize couple first definitions, lemmas in your area of research.
4. Integrate formalization into teaching (e.g. exercises).
5. Formalize a major theorem.

If there are questions, please ask!

16 https://leanprover-community.github.io/blog/posts/lte-final/

17 https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/

18 https://florisvandoorn.com/

19 https://hhu-adam.github.io/

20 https://adam.math.hhu.de/#/g/leanprover-community/nng4

Formalization in Lean

How many  ?

Demonstration