Automated Reasoning *for the Working Mathematician*

Michael Rawson

University of Southampton

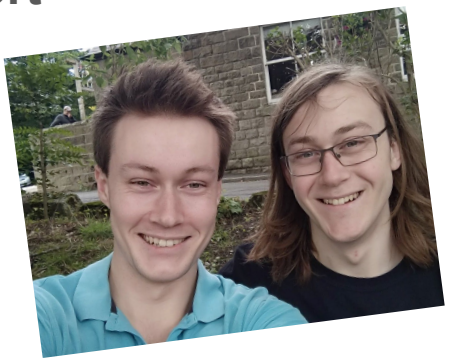
Menü

- What is 'AI'?
  - Why is this about "automated reasoning"?
- Amuse-bouche: **success** in research mathematics
- Tools I think you should know about
  - With **demonstrations!**
- Stuff I think you can forget about 🌶️
- Some suggestions for better computer assistance
- Dessert: the future

Very Short Life Story



- **Failed mathematician**
  - But my brother made it!
- Got interested in logic, became a computer scientist instead
- Wrote BA thesis in the proof assistant *Isabelle*
- Got interested in proof automation
  - PhD in machine learning for automated theorem proving
  - Develop the Vampire theorem prover
- I think mathematicians could **enjoy more computer support**
- I’m here to **listen** to what you want



What is AI?

“AI is a collective name for problems which we do not yet know how to solve properly by computer”

- Bertram Raphael (maybe), 1970ish.



NB: not applied exclusively to LLMs, diffusion models!



Symbolic “versus” Statistical AI, by example

Databases

Logic programming

Symbolic computation

Traditional automated reasoning

Proof checking: Lean, Isabelle, Agda, etc.

Statistical/Bayesian inference

Machine learning

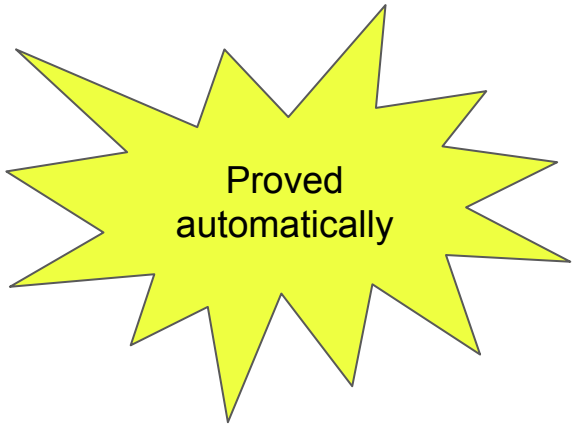
Modern language processing

What is Automated Reasoning?

“The study of automated reasoning helps produce computer programs that allow computers to reason completely, or nearly completely, automatically”

https://en.wikipedia.org/wiki/Automated_reasoning

Some AR Success Stories





For all elements a, b, and c:

1. Associativity: $a \vee (b \vee c) = (a \vee b) \vee c$
2. Commutativity: $a \vee b = b \vee a$
3. *Robbins equation*: $\neg (\neg (a \vee b) \vee \neg (a \vee \neg b)) = a$

Is every Robbins algebra a Boolean algebra?

```
----- Otter 3.0.4c, May 1996 -----
The job was started by mccune on cosmo.mcs.anl.gov, Sat Oct 12 20:54:03 1996
----> UNIT CONFLICT at  18.30 sec ----> 1212 [binary,1211.1,211.1] $F.

Length of proof is 34.  Level of proof is 17.

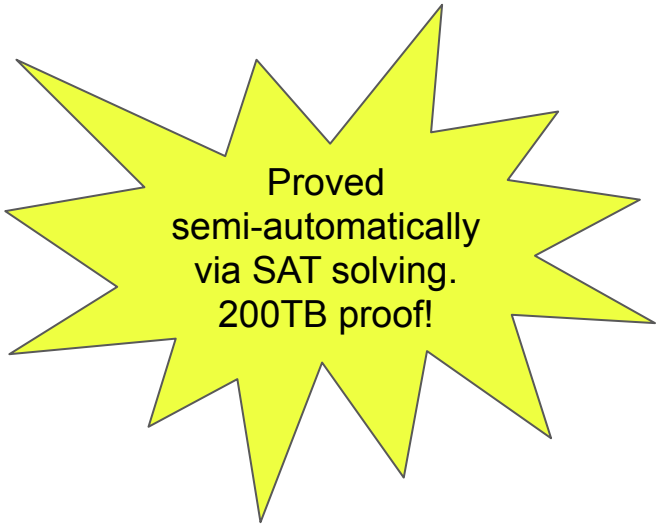
---------------- PROOF ----------------

2 [] x+y=y+x.
3 [] (x+y)+z=x+y+z.
4 [] (x+y)+z=x+y+z.
5 [] n(n(n(x)+y)+n(x+y))=y.
6 [] n(x+y)!=n(x).
64 [para_into,4.1.1.1,2.1.1,demod,3] x+y+z=y+x+z.
71 [para_into,5.1.1.1.2.1,2.1.1] n(n(n(x)+y)+n(y+x))=y.
73 [para_into,5.1.1.1,2.1.1] n(n(x+y)+n(n(x)+y))=y.
75 [para_into,6.1.1.1,2.1.1] n(x+y)!=n(y).
93 [para_into,71.1.1.1,2.1.1] n(n(x+y)+n(n(y)+x))=x.
116 [para_into,75.1.1.1,4.1.1] n(x+y+z)!=n(z).
130 [para_into,93.1.1.1.2,73.1.1] n(n(n(n(x)+y)+x+y)+y)=n(n(x)+y).
132 [para_into,93.1.1.1.2,5.1.1] n(n(n(x+y)+n(x)+y)+y)=n(x+y).
139 [para_into,116.1.1.1.2,64.1.1] n(x+y+z+u)!=n(y+u).
170 [para_from,130.1.1,73.1.1.1.2,demod,3,3] n(n(n(n(x)+y)+x+y+y)+n(n(x)+y))=y.
174 [para_from,130.1.1,5.1.1.1.1,demod,3,3] n(n(n(x)+y)+n(n(n(x)+y)+x+y+y))=y.
190 [para_into,132.1.1.1.1.1.2,64.1.1] n(n(n(x+y+z)+y+n(x)+z)+y+z)=n(x+y+z).
211 [para_into,139.1.1.1,64.1.1] n(x+y+z+u)!=n(x+u).
253 [para_from,170.1.1,93.1.1.1.2] n(n(n(n(x)+y)+n(n(x)+y)+x+y+y)+y)=n(n(x)+y).
260 [para_from,170.1.1,73.1.1.1.2.1.1,demod,3] n(n(n(n(n(x)+y)+x+y+y)+n(n(x)+y)+z)+n(y+z))=z.
346 [para_into,190.1.1.1.2,2.1.1] n(n(n(x+y+z)+y+n(x)+z)+z+y)=n(x+y+z).
423 [para_from,253.1.1,5.1.1.1.1,demod,3,3,3,3] n(n(n(x)+y)+n(n(n(x)+y)+n(n(x)+y)+x+y+y+y))=y.
475 [para_from,260.1.1,73.1.1.1.2,demod,3,3] n(n(n(n(n(x)+y)+x+y+y)+n(n(x)+y)+z+n(y+z))+z)=n(y+z).
486 [para_into,346.1.1.1.1.1.1.1.1,2.1.1,demod,3] n(n(n(x+y+z)+x+n(z)+y)+y+x)=n(z+x+y).
684 [para_into,475.1.1.1.1.1.2.2,2.1.1] n(n(n(n(n(x)+y)+x+y+y)+n(n(x)+y)+n(y+z)+z)+z)=n(y+z).
801 [para_from,684.1.1,73.1.1.1.2.1.1,demod,3] n(n(n(n(n(n(x)+y)+x+y+y)+n(n(x)+y)+n(y+z)+z)+z+u)+n(n(y+z)+u))=u.
848 [para_into,801.1.1.1.1,486.1.1,demod,3] n(n(x+x+n(n(x)+x)+x)+n(n(x+x)+n(n(x)+x)))=n(n(x)+x).
878 [para_into,848.1.1.1.2,93.1.1] n(n(x+x+n(n(x)+x)+x)+x)=n(n(x)+x).
897 [para_into,878.1.1.1.1.1.2,64.1.1] n(n(x+n(n(x)+x)+x+x)+x)=n(n(x)+x).
948 [para_into,897.1.1.1.1.1,64.1.1] n(n(n(n(x)+x)+x+x+x)+x)=n(n(x)+x).
1015 [para_from,948.1.1,73.1.1.1.2.1.1,demod,3] n(n(n(n(n(x)+x)+x+x+x)+x+y)+n(n(n(x)+x)+y))=y.
1036 [para_from,948.1.1,73.1.1.1.2,demod,3,3,3] n(n(n(n(x)+x)+x+x+x+x)+n(n(x)+x))=x.
1053 [para_into,1015.1.1.1.2,423.1.1] n(n(n(n(n(x)+x)+x+x+x)+x+n(n(n(x)+x)+n(n(x)+x)+x+x+x+x))+x)=n(n(n(x)+x)+n(n(x)+x)+x+x+x+x).
1079 [para_from,1036.1.1,93.1.1.1.2] n(n(n(n(x)+x)+n(n(x)+x)+x+x+x+x)+x)=n(n(x)+x).
1112 [para_into,1053.1.1.1.1.1,2.1.1,demod,3] n(n(x+n(n(n(x)+x)+n(n(x)+x)+x+x+x+x)+n(n(n(x)+x)+x+x+x))+x)=n(n(n(x)+x)+n(n(x)+x)+x+x+x+x).
1130 [para_from,1079.1.1,73.1.1.1.2.1.1,demod,3] n(n(n(n(n(x)+x)+n(n(x)+x)+x+x+x+x)+x+y)+n(n(n(x)+x)+y))=y.
1149 [para_into,1112.1.1.1.1.1,64.1.1] n(n(n(n(n(x)+x)+n(n(x)+x)+x+x+x+x)+x+n(n(n(x)+x)+x+x+x))+x)=n(n(n(x)+x)+n(n(x)+x)+x+x+x+x).
1169 [para_into,1130.1.1.1.2,174.1.1] n(n(n(n(n(x)+x)+n(n(x)+x)+x+x+x+x)+x+n(n(n(x)+x)+x+x+x))+x)=n(n(n(x)+x)+x+x+x).
1211 [para_into,1169.1.1,1149.1.1] n(n(n(x)+x)+n(n(x)+x)+x+x+x+x)=n(n(n(x)+x)+x+x+x).
1212 [binary,1211.1,211.1] $F.

------------ end of proof -------------
```

Is it possible to colour each of the positive integers either red or blue, s.t no Pythagorean triple a/b/c ($a^2 + b^2 = c^2$) are all the same colour?

Theorem 1. *The set* $\{1, \ldots, 7824\}$ *can be partitioned into two parts, such that no part contains a Pythagorean triple, while this is impossible for* $\{1, \ldots, 7825\}$.

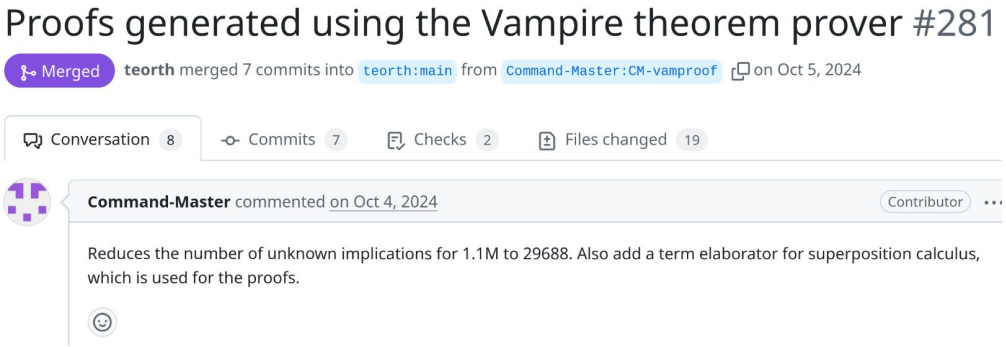


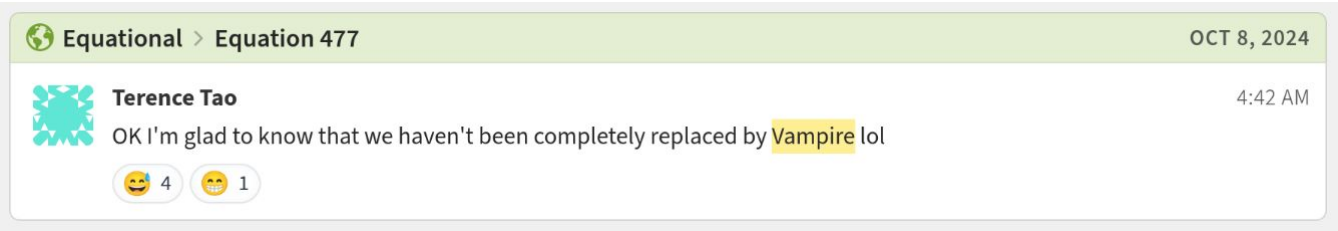


Selected Theoretical Success

- Herbrand’s Theorem
- Complete, *mechanisable* calculi for various logics
  - Including classical FOL
- Complexity and (un)decidability results
  - FOL semi-decidable
  - Presburger arithmetic decidable
  - Boolean satisfiability NP-complete
- Knuth-Bendix completion
  - Buchberger’s algorithm
- Completeness of ordered resolution*
  - And therefore superposition

Praise from Tao





Things You Should Know About

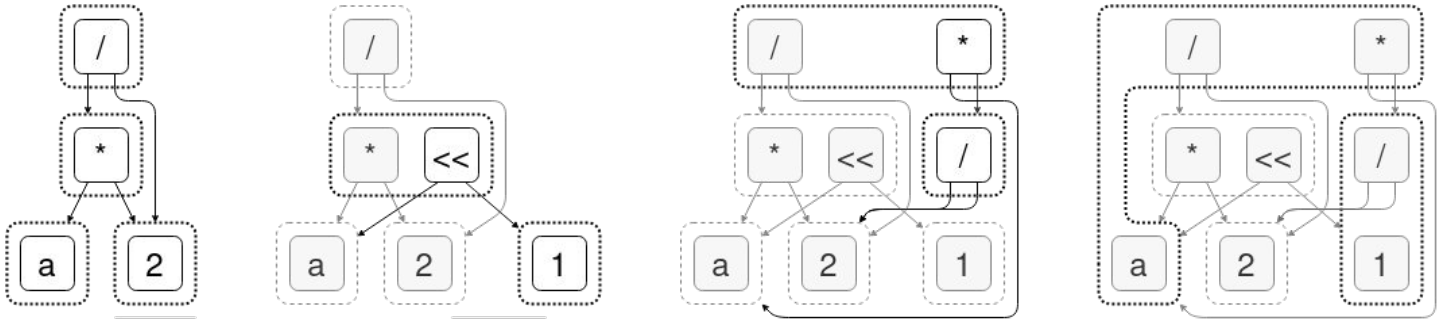
Things I think you already know

- Programming languages
- Computer algebra: SymPy, MAGMA, Maple…

E-graphs

https://egraphs-good.github.io/

- Maintain a congruence relation
- Query it
- Add new terms and equate them
- Extract least member
- Requires some programming



Satisfiability Solving

- Given a set of constraints, try and find a *solution*
  - Or show one doesn't exist
- Constraints expressed over variables with *finite* domain
  - Therefore decidable!
- Very often *Boolean* satisfiability: **SAT solving**
- SAT solving should take exponential time
- In practice, it doesn't
- Modern SAT solvers *really very good*
- To prove G from F: show F, ¬G unsatisfiable



SAT demonstration

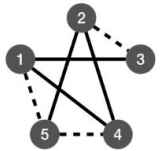
variables: ~60k
clauses: ~300k

Satisfiability Modulo Symmetries

- Cool recent research!
- Find only *canonical* solutions
- https://sat-modulo-symmetries.readthedocs.io/

Dynamic SAT approach: SMS





partially defined graph P

- SAT modulo Symmetries [Kirchweger, Sz. 2021]
- IPASIR-UP interface [Fazekas et al. 2023]

https://simons.berkeley.edu/sites/default/files/2024-07/TFCS24-2%20Slides%20-%20Stefan%20Szeider.pdf

Satisfiability Modulo Theory

- What if the Boolean variables *mean* something?
  - Like x **>** 0
  - Or f(x) **=** g(y, 10)
- “Theory” usually (but not always) decidable
- Various theories, including
  - Equality and uninterpreted functions
  - (non) linear and real arithmetic
  - Integer arithmetic
- Theories combine naturally (!)
- Slower than SAT, but more expressive



SMT demonstration

Finite Model Builders

- Given some (quantified) first-order formulas, is there a finite model of size k?
- This could be a *counterexample* or a *witness*.

Order n	# Groups[8]	Abelian	Non-Abelian
0	0	0	0
1	1	1	0
2	1	1	0
3	1	1	0
4	2	2	0
5	1	1	0
6	2	1	1
7	1	1	0
8	5	3	2
9	2	2	0
10	2	1	1

FMB demonstration

Automated Theorem Provers

- …but not the LLM-flavoured ones!
  - This has become confused in the last few years.
- Given: premises in classical FOL
  - Could also be extensions of FOL or even HOL or other logics
- ATP derives more and more consequences
- Either get to falsum (QED) or run out of patience/RAM
- Essentially *brute force*, but
  - Fancy preprocessing
  - Fancy calculus deriving very few consequences
  - Redundancy eliminations

ATP demonstration

Things you might be OK to forget about

AI hype

I don’t know who needs to hear this, but:

- Silicon Valley is not above dishonest press releases
- LLMs are not immune to decidability/complexity results or hard problems
- AI labs are not capable of “disrupting” mathematics
- **Point and laugh at the hubris.**
- This is not to say there’s nothing useful in AI research!
    - But take it all with a bucket of salt

An easy argument against “LLMs can prove things”

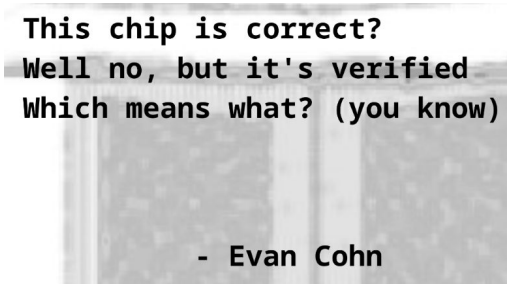
- Proof requires n tokens.
- Each token is sampled with probability p_n
  - Let’s be generous and say p close to 1 and uniform
- Therefore: chance of correct proof approximately p^n.
- “Blah blah chain of thought blah blah self-correcting blah beam search blah scale”
- **Whatever**.
- “AI labs” need to get serious about GOFAI, logical calculi
- When (if) they do, then I might pay some attention

Proof “assistants”🌶️

It's blatantly clear
You stupid machine, that what
I tell you is true

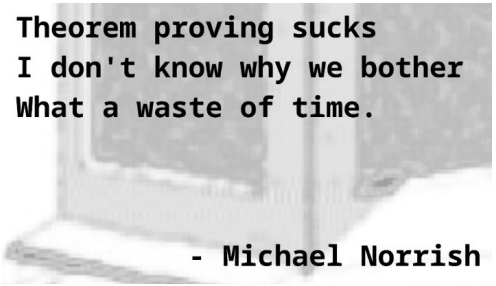
- Michael Norrish

- Things like Lean, Isabelle, Coq, Agda
- I very much doubt that mathematics is **easier** in a proof **assistant**.
  - And I have some experience with very good ones!
- This alone might put you off
- Voevodsky has a point: the computer catches mistakes
- Jury’s still out on the ease/mistakes tradeoff
- Still keen? OK then…



https://www.cl.cam.ac.uk/research/hvg/haiku.html

Your first year in computer-checked mathematics



- Spend some serious time setting one up
  - Remember that software is the devil
  - Rage-quit at least once
- Contract a nasty case of constructivism
- Perhaps read the (amusingly bonkers) QED manifesto
  - Antidote: *QED manifesto revisited*, Freek Wiedijk
- Realise it's still very difficult to phrase non-trivial statements
  - And even harder to work with them
- Have a software update break your formalisation
- Do no actual mathematics
- Start looking over your shoulder for Kevin Buzzard

def

```
cast.{u} {α β : Sort u} (h : α = β) (a : α) : β
```

Cast across a type equality. If `h : α = β` is an equality of types, and `a : α`, then `a : β` will usually not typecheck directly, but this function will allow you to work around this and embed `a` in type `β` as `cast h a : β`.

It is best to avoid this function if you can, because it is more complicated to reason about terms containing casts, but if the types don't match up definitionally sometimes there isn't anything better you can do.

For more information: Equality

The Future

Proposal: more civilised assistance

- Interface (and installation) designed for humans, not programmers
- Much reduced emphasis on checking
  - Designed to catch 99.999% of human error, *not* 100%
- No QED mind-rot: *articles* stand alone
- “Frontend” for various well-behaved symbolic routines
  - Say: I want a Ramsey (3, 4) graph, not <some constraint soup>
- Perhaps including “best effort” proof search
- Should match *existing* mathematical practice
- **I’m not aware of any such**
- I’m not sure how to do it

Automated Reasoning: not dead!



- Narrative: traditional AR has plateaued
  - LLMs are the replacement
  - Maybe guardrailed by some proof checker
- Reality: both are current and have strengths and weaknesses
  - We don’t know how to combine them yet!
- AR **still improving rapidly**
  - Can read empirical results about this
  - New techniques evolving all the time
  - In the last few years, the SAT people started using an *entirely new type of deduction*!
  - First-order ATPs prove never-ATP’d-before theorems all the time
  - Need help from mathematicians! I think the theory of proof search is interesting.

Questions