

AI IN FORMAL MATHEMATICS

Josef Urban

Czech Technical University in Prague

EuroProofNet School on Natural Formal Mathematics
June 3, 2025, Bonn



European Research Council
Established by the European Commission

How Do We Automate Math and Science?

- What is mathematical and scientific thinking/intelligence?
- Pattern-matching, analogy, induction/learning from examples
- Learning both **fuzzy hunches** and **crisp algos/heuristics/procedures**
- Deductive reasoning and proving (*btw, computation is reasoning*)
- **intelligently guided** search, exploration and guessing/conjecturing
- Complicated **feedback loops and interplays between all these**
- Using **a lot of previous knowledge** - both for induction and deduction
- Examples from physics (Popper?): deduction pushing us to Relativity, QM
- We need to develop such methods on computers
- Are there any large corpora suitable for nontrivial deduction?
- Yes! **Large libraries of formal proofs and theories**
- **So let's try to develop strong AI on them!**
- In my case done for 25-30 years. Recent overviews:
 - *Learning Guided Automated Reasoning: A Brief Survey. 2024*
 - Zar Goertzel's Phd thesis: *Learning Inference Guidance in ATP. 2023*
 - **AI4REASON**: http://ai4reason.org/PR_CORE_SCIENTIFIC_4.pdf

Quotes: Learning vs. Reasoning vs. Guessing

“C’est par la logique qu’on démontre, c’est par l’intuition qu’on invente.”

(It is by logic that we prove, but by intuition that we discover.)

– Henri Poincaré, *Mathematical Definitions and Education*.

“Hypothesen sind Netze; nur der fängt, wer auswirft.”

(Hypotheses are nets: only he who casts will catch.)

– Novalis, quoted by Popper – *The Logic of Scientific Discovery*

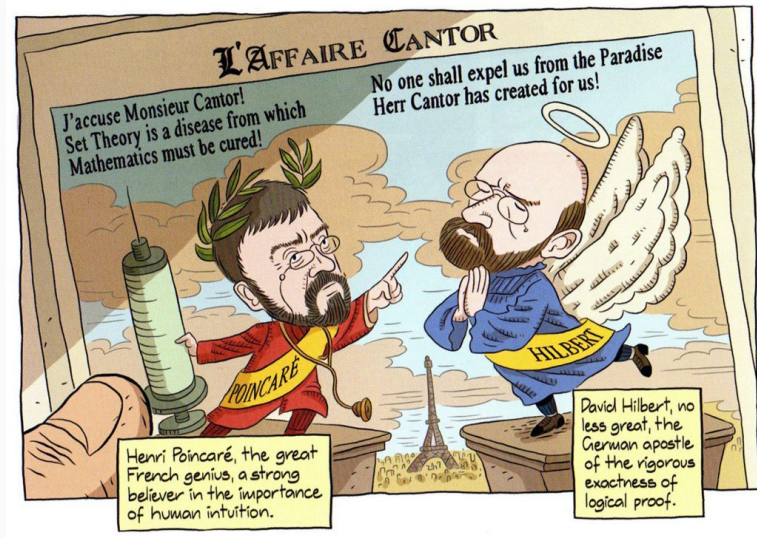
Certainly, let us learn proving, but also let us learn guessing.

– G. Polya - *Mathematics and Plausible Reasoning*

*Galileo once said, "Mathematics is the language of Science." Hence, facing the same laws of the physical world, **alien mathematics** must have a good deal of similarity to ours.*

– R. Hamming - *Mathematics on a Distant Planet*

Intuition vs Formal Reasoning – Poincaré vs Hilbert



[Adapted from: *Logicomix: An Epic Search for Truth* by A. Doxiadis]

Leibniz's/Hilbert's/Russell's Dream: Let Us Calculate!

Solve all (math, physics, law, economics, society, ...) problems by
reduction to logic/computation



[Adapted from: *Logicomix: An Epic Search for Truth* by A. Doxiadis]

A Match Made in Heaven or a Deal with the Devil?

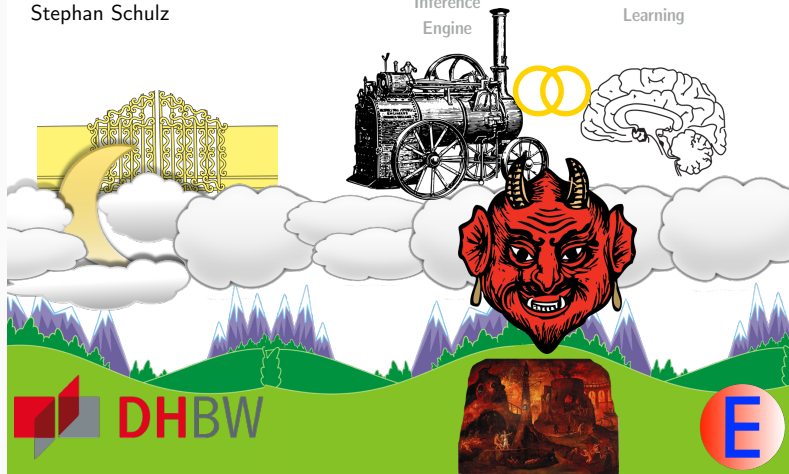
Deduction and Induction

A Match Made in Heaven or a Deal with the Devil?

Stephan Schulz

The
Inference
Engine

Machine
Learning



[Stephan Schulz's talk at AITP'16]

What is Formal Mathematics and Theorem Proving?

- 1900s: Mathematics put on formal logic foundations – **symbolic logic**
- Culmination of a program by Leibniz/Frege/Russell/Hilbert/Church/...
- ... led also to the rise of **computers** (Turing/Church, 1930s)
- ... and rise of AI - Turing's 1950 paper: Learning Machines, Chess, etc.
- 1950s: First AI program: **Logic Theorist** by Newell & Simon
- Formalization of math (60s): combine formal foundations and computers
- **Proof assistants/Interactive theorem provers** and their large libraries:
- Automath (1967), LCF, Mizar, NQTHM, HOL, Coq, Isabelle, ACL2, Lean
- **Automated theorem provers** - search for proofs automatically:
- Otter, Vampire, E, SPASS, Prover9, CVC4, Z3, Satallax, ...
- **more limited logics**: SAT, QBF, SMT, UEQ, ... (DPLL, CDCL, ...)
- **TP-motivated PLs**: ML, Prolog, (*logic programming* - Hayes, Kowalski)

What is Formal Mathematics?

- Formal math (1950/60s): combine formal foundations and the newly available computers
- **Conceptually very simple:**
- Write all your axioms and theorems so that computer understands them
- Write all your inference rules so that computer understands them
- Use the computer to check that your proofs follow the rules
- **But in practice, it turns out not to be so simple**
- Many approaches, still not mainstream, but big breakthroughs recently

Bird's Eye View of ITP Systems by T. Hales



HOL Light

HOL Light has an exquisite minimal design. It has the smallest kernel of any system. John Harrison is the sole



Mizar

Once the clear front-runner, it now shows signs of age. Do not expect to understand the inner workings of this system unless you have been



Coq

Coq is built of modular components on a foundation of dependent type theory. This system has grown one PhD thesis at a time.



Isabelle

Designed for use with multiple foundational architectures, Isabelle's early development featured classical constructions in set theory. However,



Metamath

Does this really work? Defying expectations, Metamath seems to function shockingly well for those who are happy to live without plumbing.



Lean

Lean is ambitious, and it will be massive. Do not be fooled by the name. "Construction area keep out" signs are prominently posted on the perimeter fencing.

F. Wiedijk: Irrationality of $\sqrt{2}$ (informal text)

tiny proof from Hardy & Wright, texts collected by F. Wiedijk:

Theorem 43 (Pythagoras' theorem). $\sqrt{2}$ is irrational.

The traditional proof ascribed to Pythagoras runs as follows. If $\sqrt{2}$ is rational, then the equation

$$a^2 = 2b^2 \tag{4.3.1}$$

is soluble in integers a, b with $(a, b) = 1$. Hence a^2 is even, and therefore a is even. If $a = 2c$, then $4c^2 = 2b^2$, $2c^2 = b^2$, and b is also even, contrary to the hypothesis that $(a, b) = 1$. \square

Irrationality of $\sqrt{2}$ (Formal Proof Sketch)

exactly the same text in Mizar syntax:

```
theorem Th43: :: Pythagoras' theorem
  sqrt 2 is irrational
proof
  assume sqrt 2 is rational;
  consider a,b such that
4_3_1: a^2 = 2*b^2 and
  a,b are relative prime;
  a^2 is even;
  a is even;
  consider c such that a = 2*c;
  4*c^2 = 2*b^2;
  2*c^2 = b^2;
  b is even;
  thus contradiction;
end;
```

Irrationality of $\sqrt{2}$ in HOL Light

```
let SQRT_2_IRRATIONAL = prove
  (~rational(sqrt(&2))) ,
  SIMP_TAC[rational; real_abs; SQRT_POS_LE; REAL_POS] THEN
  REWRITE_TAC[NOT_EXISTS_THM] THEN REPEAT GEN_TAC THEN
  DISCH_THEN(CONJUNCTS_THEN2 ASSUME_TAC MP_TAC) THEN
  SUBGOAL_THEN (~((&p / &q) pow 2 = sqrt(&2) pow 2))
    (fun th -> MESON_TAC[th]) THEN
  SIMP_TAC[SQRT_POW_2; REAL_POS; REAL_POW_DIV] THEN
  ASM_SIMP_TAC[REAL_EQ_LDIV_EQ; REAL_OF_NUM_LT; REAL_POW_LT;
    ARITH_RULE '0 < q <=> ~(q = 0)'] THEN
  ASM_MESON_TAC[NSQRT_2; REAL_OF_NUM_POW; REAL_OF_NUM_MUL; REAL_OF_NUM_EQ]];
```


Irrationality of $\sqrt{2}$ in Isabelle/HOL

```
theorem sqrt2_not_rational:
  "sqrt (real 2)  $\notin$   $\mathbb{Q}$ "
proof
  assume "sqrt (real 2)  $\in \mathbb{Q}$ "
  then obtain m n :: nat where
    n_nonzero: "n  $\neq$  0" and sqrt_rat: "|sqrt (real 2)| = real m / real n"
    and lowest_terms: "gcd m n = 1" ..
  from n_nonzero and sqrt_rat have "real m = |sqrt (real 2)| * real n" by simp
  then have "real (m2) = (sqrt (real 2))2 * real (n2)"
    by (auto simp add: power2_eq_square)
  also have "(sqrt (real 2))2 = real 2" by simp
  also have "... * real (m2) = real (2 * n2)" by simp
  finally have eq: "m2 = 2 * n2" ..
  hence "2 dvd m2" ..
  with two_is_prime have dvd_m: "2 dvd m" by (rule prime_dvd_power_two)
  then obtain k where "m = 2 * k" ..
  with eq have "2 * n2 = 22 * k2" by (auto simp add: power2_eq_square mult_ac)
  hence "n2 = 2 * k2" by simp
  hence "2 dvd n2" ..
  with two_is_prime have "2 dvd n" by (rule prime_dvd_power_two)
  with dvd_m have "2 dvd gcd m n" by (rule gcd_greatest)
  with lowest_terms have "2 dvd 1" by simp
  thus False by arith
qed
```

Irrationality of $\sqrt{2}$ in Coq

```
Theorem irrational_sqrt_2: irrational (sqrt 2%nat).
intros p q H H0; case H.
apply (main_thm (Zabs_nat p)).
replace (Div2.double (q * q)) with (2 * (q * q));
  [idtac | unfold Div2.double; ring].
case (eq_nat_dec (Zabs_nat p * Zabs_nat p) (2 * (q * q))); auto; intros H1.
case (not_nm_INR _ _ H1); (repeat rewrite mult_INR).
rewrite <- (sqrt_def (INR 2)); auto with real.
rewrite H0; auto with real.
assert (q <> 0%R :=> R); auto with real.
field; auto with real; case p; simpl; intros; ring.
Qed.
```

Irrationality of $\sqrt{2}$ in Metamath

```
{
  $d x y $.
  $( The square root of 2 is irrational. $)
  sqr2irr $p |- ( sqr ` 2 ) e/ QQ $=
    ( vx vy c2 csqr cfv cq wnel wcel wn cv cdiv co wceq cn wrex cz cexp
    cmulc sqr2irrlem3 sqr2irrlem5 bi2rexa mtbir cc0 clt wbr wa wi wb nngt0t
    adantr cr ax0re ltmuldivt mp3an1 nnret zret syl2an mpd ancoms 2re 2pos
    sqrgt0i breq2 mpbii syl5bir cc nncnt mulzer2t syl breqld adant1 sylibd
    exp r19.23adv anc2li elnnz syl6ibr impac r19.22i2 mto elq df-nel mpbir )
    CDEZFGWDFHZIWEWDAJZBJZKLZMZBNOZAPZWKWJANOZWLWFCQLCWGCQLRLMZBNOANOABSWIWM
    ABNNWFWGTUAUBWJWJAPNWFPHZWJWFNHZWNWJWNUCWFUDUEZUFWOWNWJWPWNWIWPBNWNWGNHZW
    IWPUGWNWQUFZWIUCWGRLZWFUDUEZWPWRWTUCWHUDUEZWIWQWNWTXAUHZWQWNUFUCWGUDUEZXB
    WQXCWNWGUIUJWGUHKHZWFUKHZXCXBUGZWQWNUCUXHXDXEXFULUCWGWUFUMUNWGUOWFUPUQURUSW
    IUCWDUDUEXACUTVAVBWDWHUCUDVCVDVEWQWTWPUHWNWQWSUCWFUDWQWGVFHWWSUCMWGVGWGVHV
    IVJVKVLVMNVVOWFVPVQVRVSVTABWDWAUBWDFWBWC $.
  $( [8-Jan-02] $)
}
```

Irrationality of $\sqrt{2}$ in Metamath Proof Explorer

Proof of Theorem sqr2irr			
Step	Hyp	Ref	Expression
1		sqr2irrlem3 10838	$s \vdash \neg \exists x \in \mathbb{N} \exists y \in \mathbb{N} (x \upharpoonright 2) = (2 \cdot (y \upharpoonright 2))$
2		sqr2irrlem5 10840	$s \vdash ((x \in \mathbb{N} \wedge y \in \mathbb{N}) \rightarrow ((\sqrt{2}) = (x/y) \leftrightarrow (x \upharpoonright 2) = (2 \cdot (y \upharpoonright 2))))$
3	2	2rexhiia 2329	$s \vdash (\exists x \in \mathbb{N} \exists y \in \mathbb{N} (\sqrt{2}) = (x/y) \leftrightarrow \exists x \in \mathbb{N} \exists y \in \mathbb{N} (x \upharpoonright 2) = (2 \cdot (y \upharpoonright 2)))$
4	1, 3	mtbir 288	$s \vdash \neg \exists x \in \mathbb{N} \exists y \in \mathbb{N} (\sqrt{2}) = (x/y)$
5		2re 8838	$s \vdash 2 \in \mathbb{R}$
6		2pos 6849	$s \vdash 0 < 2$
7	5, 6	sqrqt0ii 10213	$s \vdash 0 < (\sqrt{2})$
8		breq2 3595	$s \vdash 11 \vdash ((\sqrt{2}) = (x/y) \rightarrow (0 < (\sqrt{2}) \leftrightarrow 0 < (x/y)))$
9	7, 8	mpbii 200	$s \vdash 10 \vdash ((\sqrt{2}) = (x/y) \rightarrow 0 < (x/y))$
10		zre 9029	$s \vdash 12 \vdash (x \in \mathbb{Z} \rightarrow x \in \mathbb{R})$
11	10	adantr 444	$s \vdash 11 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow x \in \mathbb{R})$
12		nnrc 8788	$s \vdash 13 \vdash (y \in \mathbb{N} \rightarrow y \in \mathbb{R})$
13	12	adantl 445	$s \vdash 11 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow y \in \mathbb{R})$
14		nngt0 8807	$s \vdash 12 \vdash (y \in \mathbb{N} \rightarrow 0 < y)$
15	14	adantl 445	$s \vdash 11 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow 0 < y)$
16		gt0div 8083	$s \vdash 11 \vdash ((x \in \mathbb{R} \wedge y \in \mathbb{R} \wedge 0 < y) \rightarrow (0 < x \leftrightarrow 0 < (x/y)))$
17	11, 13, 15, 16	syl3anc 1145	$s \vdash 10 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow (0 < x \leftrightarrow 0 < (x/y)))$
18	9, 17	syl5ibr 210	$s \vdash 9 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow ((\sqrt{2}) = (x/y) \rightarrow 0 < x))$
19		simpl 436	$s \vdash 9 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow x \in \mathbb{Z})$
20	18, 19	jctild 522	$s \vdash 8 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow ((\sqrt{2}) = (x/y) \rightarrow (x \in \mathbb{Z} \wedge 0 < x)))$
21		elnz 9035	$s \vdash 8 \vdash (x \in \mathbb{N} \leftrightarrow (x \in \mathbb{Z} \wedge 0 < x))$
22	20, 21	syl6ibr 216	$s \vdash 7 \vdash ((x \in \mathbb{Z} \wedge y \in \mathbb{N}) \rightarrow ((\sqrt{2}) = (x/y) \rightarrow x \in \mathbb{N}))$
23	22	rexlimdva 2414	$s \vdash 6 \vdash (x \in \mathbb{Z} \rightarrow (\exists y \in \mathbb{N} (\sqrt{2}) = (x/y) \rightarrow x \in \mathbb{N}))$
24	23	impac 598	$s \vdash 5 \vdash ((x \in \mathbb{Z} \wedge \exists y \in \mathbb{N} (\sqrt{2}) = (x/y)) \rightarrow (x \in \mathbb{N} \wedge \exists y \in \mathbb{N} (\sqrt{2}) = (x/y)))$
25	24	reximi2 2396	$s \vdash 4 \vdash (\exists x \in \mathbb{Z} \exists y \in \mathbb{N} (\sqrt{2}) = (x/y) \rightarrow \exists x \in \mathbb{N} \exists y \in \mathbb{N} (\sqrt{2}) = (x/y))$
26	4, 25	mto 165	$s \vdash 3 \vdash \neg \exists x \in \mathbb{Z} \exists y \in \mathbb{N} (\sqrt{2}) = (x/y)$
27		elq 9308	$s \vdash 1 \vdash ((\sqrt{2}) \in \mathbb{Q} \leftrightarrow \exists x \in \mathbb{Z} \exists y \in \mathbb{N} (\sqrt{2}) = (x/y))$
28	26, 27	mtbir 288	$s \vdash \neg (\sqrt{2}) \in \mathbb{Q}$
29		df-nel 3210	$s \vdash 1 \vdash ((\sqrt{2}) \notin \mathbb{Q} \leftrightarrow \neg ((\sqrt{2}) \in \mathbb{Q}))$
30	28, 29	mpbir 198	$s \vdash (\sqrt{2}) \notin \mathbb{Q}$

Today: Computers Checking Large Math Proofs



Researchers Find 40,000-Year-Old Figurative Paintings in Bornean Cave

HOME ASTRONOMY SPACE EXPLORATION ARCHAEOLOGY PALEONTOLOGY BIOLOGY PHYSICS MEDICINE GENETICS GEOLOGY MORE

Scientists Deliver Formal Proof of Famous Kepler Conjecture

Jun 16, 2017 by News Staff / Source

« Previous | Next »

Published in
Mathematics

Tagged as
Johannes Kepler
Kepler conjecture

Follow
You Might Like



Researchers Develop First-Ever 3D Numerical Model of Melting Snowflake



Researchers Develop Mathematical Model for How Innovations

An international team of mathematicians led by University of Pittsburgh **Professor Thomas Hales** has delivered a formal proof of the **Kepler conjecture**, a famous problem in discrete geometry. The team's **paper** is published in the journal *Forum of Mathematics, Pi*.



LATEST NEWS



SPHERE Captures Young Exoplanet Beta Pictoris b Orbiting around Its Star

Nov 13, 2018 | Astronomy



Mirace eatoni: Newly-Discovered Cretaceous Bird Lived Among Dinosaurs, Was Strong Flier

Nov 13, 2018 | Paleontology



Juno Takes Closer Look at Jupiter's Magnificent, Swirling Clouds

Nov 13, 2018 | Space Exploration



Physicists Solve Structure of Unusually Complex Form of Nitrogen

Nov 13, 2018 | Physical Chemistry



Natural Compound Protects Hypertensive Rats against Heart Disease

Nov 13, 2018 | Medicine



Inventive Orangutans Make Hook Tools to Retrieve Food

Nov 12, 2018 | Biology



Researchers Find 40,000-Year-Old Figurative Paintings in Bornean Cave

Nov 12, 2018 | Archaeology



Hubble Sees Lensing Galaxy Cluster,

cdn.sci-news.com/images/enlarge3/image_4960e-Kepler-Conjecture.jpg

Today's Applications

PHYS.ORG Nanotechnology ▾ Physics ▾ Earth ▾ Astronomy & Space ▾ Technology ▾ Chemistry ▾ Biology ▾ Other Sciences ▾


[f](#) [t](#) [r](#) [e](#) [m](#)

search

Home » Other Sciences » Mathematics » October 12, 2012

Six-year journey leads to proof of Feit-Thompson Theorem

October 12, 2012 by Rob Knies, Microsoft



Georges Gonthier.

At 5:46 p.m. on Sept. 20, Georges Gonthier, principal researcher at Microsoft Research Cambridge, sent a brief email to his colleagues at the Microsoft Research-Inria Joint Centre in Paris. It read, in full: "This is really the End."


Those five innocuous words heralded the culmination of a project that had consumed more than six years and resulted in the formal proof of the **Feit-Thompson Theorem**, the first major step of the classification of finite simple groups.

The theorem, first proved by Walter Feit and John Griggs Thompson in 1963 and also known as the Odd-Order Theorem, states that in mathematical group theory, every finite group of odd order is solvable.


Featured

Last comments


Popular




Gaia spots a 'ghost' galaxy next door 19 hours ago 81




How plants evolved to make ants their servants Nov 12, 2018 21



Physicists build fractal shape out of electrons Nov 12, 2018 0



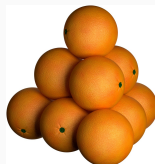
Dark matter 'hurricane' offers chance to detect axions 18 hours ago 36



How to drive a robot on Mars Nov 12, 2018 2

Big Example: The Flyspeck project

- Kepler conjecture (1611): The most compact way of stacking balls of the same size in space is a pyramid.



$$V = \frac{\pi}{\sqrt{18}} \approx 74\%$$

- Formal proof finished in 2014
- 20000 lemmas in geometry, analysis, graph theory
- All of it at <https://code.google.com/p/flyspeck/>
- All of it **computer-understandable and verified** in HOL Light:
- `polyhedron s /\ c face_of s ==> polyhedron c`
- However, this took **20 – 30 person-years!**

Learning vs Reasoning – Alan Turing 1950 – AI



- 1950: *Computing machinery and intelligence* – AI, Turing test
- “We may hope that machines will eventually compete with men in *all purely intellectual fields*.” (regardless of his 1936 undecidability result!)
- last section on *Learning Machines*:
- “But which are the best ones [fields] to start [learning on] with?”
- “... Even this is a difficult decision. Many people think that a very abstract activity, like the *playing of chess*, would be best.”
- Why not try with *math*? It is much more (universally?) expressive ...
- (formal) math as a *universal/science-complete game*, *semantic sweetspot*

History and Motivation for AI/ML/TP

- Intuition vs Formal Reasoning – Poincaré vs Hilbert, Science & Method
- Turing's 1950 paper: Learning Machines, learn Chess?, undecidability??
- Lenat, Langley, etc: manually-written heuristics, learn Kepler laws,...
- Denzinger, Schulz, Goller, Fuchs – late 90's, ATP-focused:
- **Learning from Previous Proof Experience**
- My MSc (1998): Try ILP to learn **explainable** rules/heuristics from Mizar
- Since: Use large formal math (Big Proof) corpora: Mizar, Isabelle, HOL
- ... to combine/develop symbolic/statistical deductive/inductive ML/TP/AI
- ... hammer-style methods, feedback loops, internal guidance, ...
- More details – AGI'18 keynote: <https://bit.ly/3qifhg4>
- **AI vs DL**: Ben Goertzel's Prague talk: <https://youtu.be/Zt2HSTuGBn8>
- **Big AI visions**: automate/verify math, science, law, (Leibniz, McCarthy, ..)
- Practical impact: boost today's large ITP verification projects

Quick intro: *Prove/Learn feedback loop* on formal math

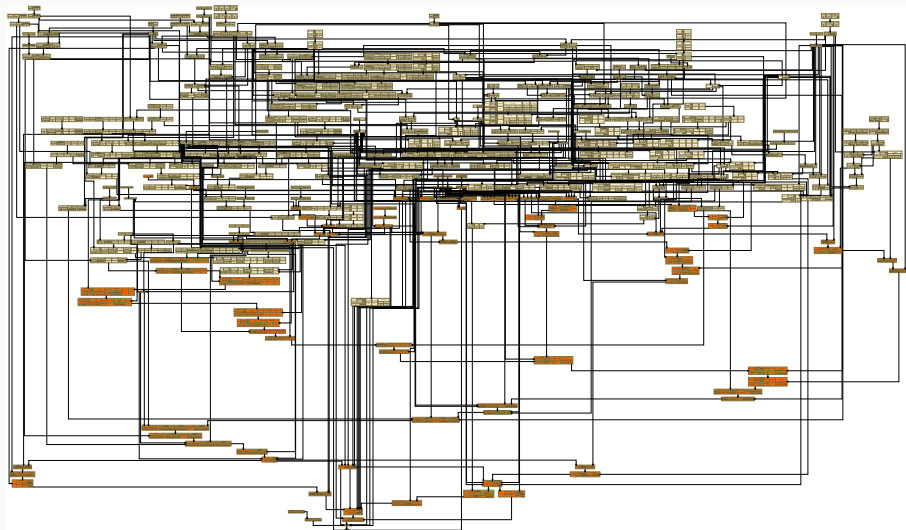
- Done on 57880 Mizar Mathematical Library formal math problems in 2019
- Efficient **ML-guidance inside the best ATPs** like E prover (ENIGMA)
- *Training* of the ML-guidance is *interleaved* with *proving* harder problems
- Ultimately a **70% improvement** over the original E strategy:
- ... from 14933 proofs to 25397 proofs (all in 10s CPU - no cheating)

	S	$S \odot M_9^0$	$S \oplus M_9^0$	$S \odot M_9^1$	$S \oplus M_9^1$	$S \odot M_9^2$	$S \oplus M_9^2$	$S \odot M_9^3$	$S \oplus M_9^3$
solved	14933	16574	20366	21564	22839	22413	23467	22910	23753
$S\%$	+0%	+10.5%	+35.8%	+43.8%	+52.3%	+49.4%	+56.5%	+52.8%	+58.4
$S+$	+0	+4364	+6215	+7774	+8414	+8407	+8964	+8822	+9274
$S-$	-0	-2723	-782	-1143	-508	-927	-430	-845	-454

	$S \odot M_{12}^3$	$S \oplus M_{12}^3$	$S \odot M_{16}^3$	$S \oplus M_{16}^3$
solved	24159	24701	25100	25397
$S\%$	+61.1%	+64.8%	+68.0%	+70.0%
$S+$	+9761	+10063	+10476	+10647
$S-$	-535	-295	-309	-183

- **75% of the Mizar corpus** (43414) reached in July 2021 - higher times and many prove/learn cycles: https://github.com/ai4reason/ATP_Proofs
- Details in our Mizar60 paper: <https://arxiv.org/abs/2303.06686>

Can you do this in 4 minutes? (359-step ATP proof)



Can you do this in 4 minutes? (human-written code)

```
theorem 7h31: :: BORSUK 5:31
  for A being Subset of R^1
  for a, b being real number st a < b & A = RAT (a,b) holds
  Cl A = [.a,b.]
proof
  let A be Subset of R^1; :: thesis:
  let a, b be real number ; :: thesis:
  assume that
  A1: a < b and
  A2: A = RAT (a,b) ; :: thesis:
  reconsider ab = [.a,b.], RT = RAT as Subset of R^1 by NUMBERS:12, TOPMETR:17;
  reconsider RR = RAT /\ [.a,b.] as Subset of R^1 by TOPMETR:17;
  A3: the carrier of R^1 /\ (Cl ab) = Cl ab by XBOOLE:3:26;
  A4: Cl RR c= (Cl RT) /\ (Cl ab) by HME:TOPC:11;
  thus Cl A c= [.a,b.] :: according to XBOOLE:0:def 10 :: thesis:
proof
  let x be set ; :: according to TARSKI:def 3 :: thesis:
  assume x in Cl A ; :: thesis:
  then x in (Cl RT) /\ (Cl ab) by A2, A4;
  then x in the carrier of R^1 /\ (Cl ab) by A3;
  hence x in [.a,b.] by A1, A3, Th14; :: thesis:
end;
thus [.a,b.] c= Cl A :: thesis:
proof
  let x be set ; :: according to TARSKI:def 3 :: thesis:
  assume A5: x in [.a,b.] ; :: thesis:
  then reconsider p = x as Element of RealSpace by METRIC_1:def 13;
  A6: a <= p by A5, XXREAL_1:1;
  A7: p <= b by A5, XXREAL_1:1;
  per cases by A7, XXREAL_0:1;
  suppose A8: p < b ; :: thesis:
  now :: thesis:
  let r be real number ; :: thesis:
  reconsider pp = p + r as Element of RealSpace by METRIC_1:def 13, XXREAL_0:def 1;
  set pr = min (pp,((p + b) / 2));
  A9: min (pp,((p + b) / 2)) <= (p + b) / 2 by XXREAL_0:17;
  assume A10: r > 0 ; :: thesis:
  p < min (pp,((p + b) / 2))
  proof
    per cases by XXREAL_0:15;
    suppose min (pp,((p + b) / 2)) = pp ; :: thesis:
      hence p < min (pp,((p + b) / 2)) by A10, XXREAL_1:29; :: thesis:
    end;
    suppose min (pp,((p + b) / 2)) = (p + b) / 2 ; :: thesis:
      hence p < min (pp,((p + b) / 2)) by A8, XXREAL_1:29; :: thesis:
    end;
  end;
end;
then consider Q being rational number such that
A11: p < Q and
A12: Q < min (pp,((p + b) / 2)) by RAT_1:7;
(p + b) / 2 < b by A8, XXREAL_1:29;
then min (pp,((p + b) / 2)) < b by A9, XXREAL_0:2;
then A13: Q < b by A12, XXREAL_0:2;
min (pp,((p + b) / 2)) <= pp by XXREAL_0:17;
then A14: (min (pp,((p + b) / 2))) - p <= pp - p by XXREAL_1:9;
reconsider P = Q as Element of RealSpace by METRIC_1:def 13, XXREAL_0:def 1;
P - p < (min (pp,((p + b) / 2))) - p by A12, XXREAL_1:9;
then P - p < r by A14, XXREAL_0:2;
then dist (p,P) < r by A11, Th14;
then A15: P in Ball (p,r) by METRIC_1:11;
a < Q by A6, A11, XXREAL_0:2;
then A16: Q in [.a,b.] by A13, XXREAL_1:4;
Q in RAT by RAT_1:def 2;
then Q in A by A2, A16, XBOOLE:0:def 4;
hence Ball (p,r) meets A by A15, XBOOLE:0:3; :: thesis:
end;
hence x in Cl A by GOBOARD6:92, TOPMETR:def 6; :: thesis:
```

Intro2: *Search/Check/Learn* feedback loop on OEIS

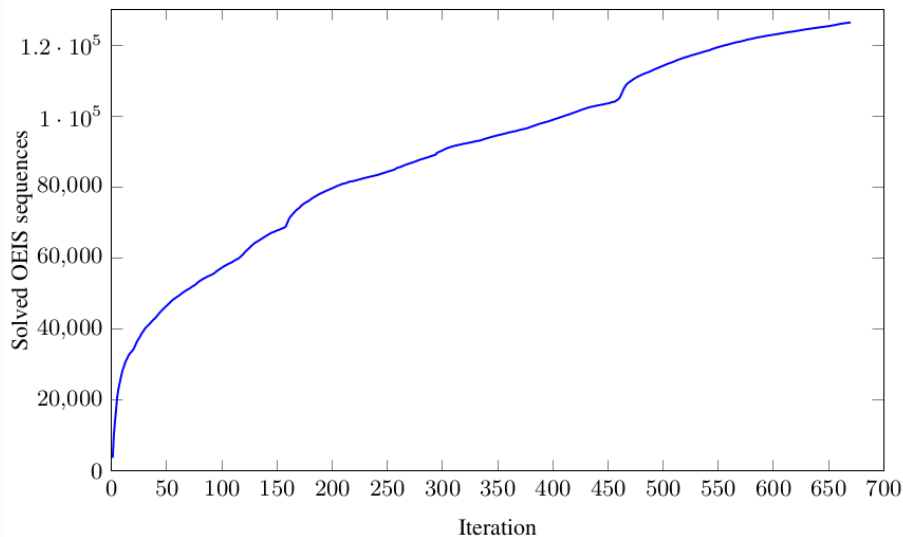
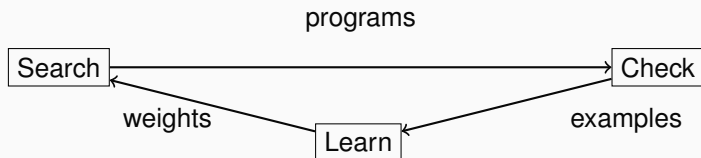


Figure 12: Number y of solved OEIS sequences after x iterations

Search-Verify-Train Positive Feedback Loop (OEIS)



- Small Turing-complete DSL for our programs, e.g.:

$$2^x = \prod_{y=1}^x 2 = \text{loop}(2 \times x, \mathbf{x}, 1)$$

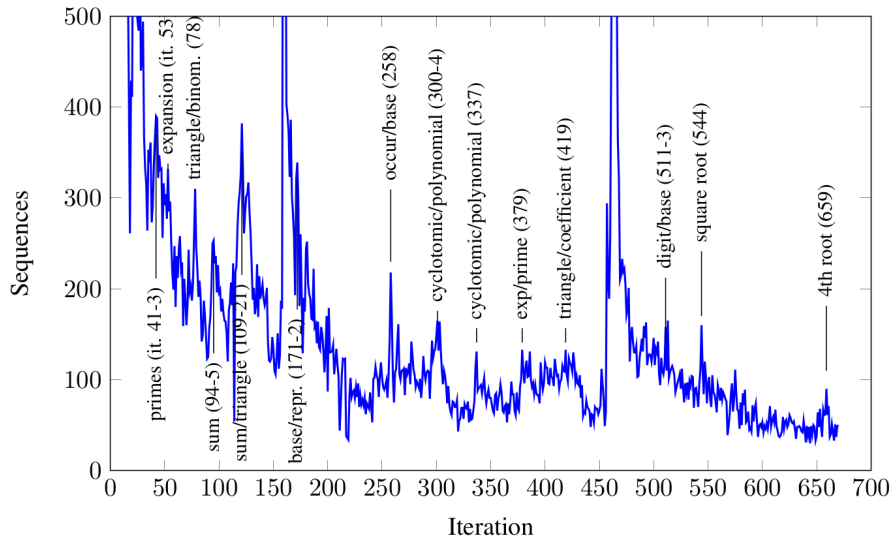
$$\mathbf{x}! = \prod_{y=1}^x y = \text{loop}(y \times x, \mathbf{x}, 1)$$

- Analogous to our Prove/Learn feedback loops in learning-guided proving (since 2006 – Machine Learner for Automated Reasoning – MaLAREa))
- However, OEIS allows much faster feedback on *symbolic conjecturing*
- 670 iterations and still refuses to plateau - counters RL wisdom?
- Since it interleaves symbolic breakthroughs and statistical learning?
- Cheap: The electricity bill is only \$1k-\$3k, you can do this at home
- ~4.5M explanations invented: 50+ different characterizations of primes

Some Invented Explanations for OEIS (“Alien Coder”)

- <https://oeis.org/A4578>: Expansion of $\sqrt{8}$ in base 3:
loop2(((y * y) div (x + y)) + y, y, x + x, 2, loop((1 + 2) * x, x, 2)) mod (1 + 2)
- <https://oeis.org/A4001>: Hofstadter-Conway \$10k seq: $a(n) = a(a(n-1)) + a(n-a(n-1))$ with $a(1) = a(2) = 1$:
loop(push(loop(pop(x), y-x, pop(x)), x) + loop(pop(x), x-1, x), x - 1, 1)
- <https://oeis.org/A40>: prime numbers:
 $2 + \text{compr}((\text{loop}(x * y, x, 2) + x) \bmod (2 + x), x)$
- <https://oeis.org/A30184>: Expand $\eta(q) * \eta(q^3) * \eta(q^5) * \eta(q^{15})$ in powers of q (elliptic curves):
loop(push(loop((pop(x) * loop(if (pop(x) mod y) <= 0 then (x - loop(if (x mod (1 + (y + y))) <= 0 then (x + x) else x, 2, y)) else x, y, push(0, y))) + x, y, push(0, x)), x) div y, x, 1)
- <https://oeis.org/A51023>: Wolfram's \$30k Rule 30 automaton:
loop2(y, y div 2, x, 1, loop2(loop2(((y div (0 - (2 + 2))) mod 2) + x) + x, y div 2, y, 1, loop2(((y mod 2) + x) + x, y div 2, y, 1, x)), 2 + y, x, 0, 1)) mod 2
- <https://oeis.org/A2580>: $\sqrt[3]{2}$ Hales's blog: <https://t.ly/tHs1d>

Some Automatic Technology Jumps



Serious Math Conjecturing – Elliptic Curves

- **Sander Dahmen:** *Here are some OEIS labels related to elliptic curves (and hence modular forms), ordered by difficulty. It would be interesting to know if some of these appear in your results.*
- A006571 A030187 A030184 A128263 A187096 A251913
- **JU:** *We have the first three:*
- A6571 : `loop((push(loop((pop(x) * loop(if (pop(x) mod y) <= 0 then ((if (y mod loop(1 + (x + x), 2, 2)) <= 0 then (x - y) else x) - y) else x, y, push(0, y))) + x, y, push(0, x)), x) * 2) div y, x, 1)`
- A30187 : `loop(push(loop((pop(x) * loop(if (pop(x) mod y) <= 0 then (x - loop(if (x mod (((2 + y) * y) - 1)) <= 0 then (x + x) else x, 2, y)) else x, y, push(0, y))) + x, y, push(0, x)), x) div y, x, 1)`
- A30184 : `loop(push(loop((pop(x) * loop(if (pop(x) mod y) <= 0 then (x - loop(if (x mod (1 + (y + y))) <= 0 then (x + x) else x, 2, y)) else x, y, push(0, y))) + x, y, push(0, x)), x) div y, x, 1)`

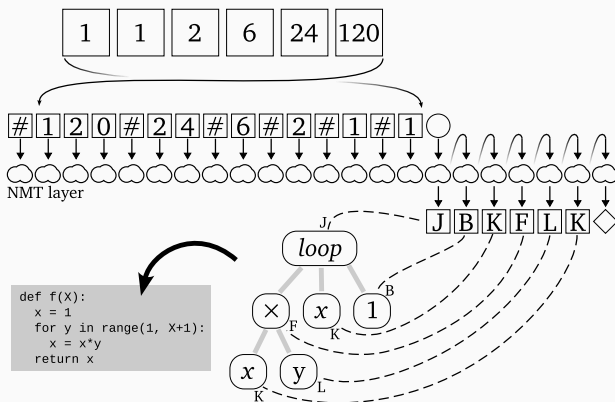
A6571: Expansion of $q * \text{Product}_{k \geq 1} (1 - q^k)^2 * (1 - q^{11*k})^2$

A30187: Expansion of $\eta(q) * \eta(q^2) * \eta(q^7) * \eta(q^{14})$ in powers of q .

A30184: Expansion of $\eta(q) * \eta(q^3) * \eta(q^5) * \eta(q^{15})$ in powers of q .

Encoding OEIS for Language Models

- Input sequence is a **series of digits**
- Separated by an additional token # at the integer boundaries
- Output program is a **sequence of tokens** in Polish notation
- Parsed by us to a syntax tree and **translatable to Python**
- Example: $a(n) = n!$



Search-Verify-Train Feedback Loop for OEIS

- **search phase:** LM synthesizes many programs for input sequences
- typically 240 candidate programs for each input using **beam search**
- **84M programs** for OEIS in several hours on the GPU (depends on model)
- **checking phase:** the millions of programs **efficiently evaluated**
- resource limits used, **fast indexing** structures for OEIS sequences
- check if the program generates *any* OEIS sequence (**hindsight replay**)
- we keep the **shortest** (Occams's razor) and **fastest** program (efficiency)
- from iter. 501, we also keep the program with the **best speed/length ratio**
- **learning phase:** LM **trains to translate** the “solved” OEIS sequences into the best program(s) generating them
- from iter. 336: **train LMs to transform** (generalization, optimization)
- our learned version of human-coded methods like **ILP and compilation**

Search-Verify-Train Feedback Loop

- The weights of the LM either trained from **scratch** or **continuously updated**
- This yields *new minds vs seasoned experts* (who have seen it all)
- We also train experts on varied selections of data, in varied ways
- **Orthogonality**: common in theorem proving - different experts help
- Each iteration of the self-learning loop discovers **more solutions**
- ... also **improves/optimizes existing solutions**
- The **alien mathematician** thus self-evolves
- Occam's razor and efficiency are used for its **weak supervision**
- Quite different from today's LLM approaches:
- LLMs do **one-time** training on everything human-invented
- Our alien instead **starts from zero knowledge**
- Evolves increasingly nontrivial skills, may **diverge from humans**
- **Turing complete** (unlike Go/Chess) – arbitrary complex algorithms

Sample of Learning Approaches

- **neural networks** (**statistical ML**, old!) – backprop, SGD, deep learning, convolutional, recurrent, attention/transformers, tree NNs, graph NNs, etc.
- **decision trees, random forests, gradient boosted trees** – find good classifying attributes (and/or their values); more **explainable**, often SoTA
- **support vector machines** – find a good classifying hyperplane, possibly after non-linear transformation of the data (*kernel methods*)
- **k-nearest neighbor** – find the k nearest neighbors to the query, combine their solutions, good for *online learning* (important in ITP)
- **naive Bayes** – compute probabilities of outcomes assuming complete (naive) independence of characterizing features, i.e., just multiplying probabilities: $P(y|\mathbf{x}) = P(x_1|y) * P(x_2|y) * \dots * P(x_n|y) * P(y)/P(\mathbf{x})$
- **inductive logic programming** (**symbolic ML**) – generate logical explanation (program) from a set of ground clauses by generalization
- **genetic algorithms** – evolve large population by crossover and mutation
- various **combinations** of statistical and symbolic approaches
- **supervised, unsupervised, online/incremental, reinforcement** learning (actions, explore/exploit, cumulative reward)

Learning – Features and Data Preprocessing

- **Extremely important** - if irrelevant, there is no way to learn the function from input to output (“garbage in garbage out”)
- **Feature discovery/engineering** – a big field, a bit overshadowed by DL
- **Deep Learning (DL)** – deep neural nets that **automatically find important high-level features** for a task, can be structured (tree/graph NNs)
- **Data Augmentation and Selection** – how do we generate/select more/better data to learn on?
- **Latent Semantics, PCA, dimensionality reduction**: use linear algebra (eigenvector decomposition) to discover the most similar features, make approximate equivalence classes from them; or just use *hashing*
- **word2vec and related/neural methods**: represent words/sentences by *embeddings* (in a high-dimensional real vector space) learned by predicting the next word on a large corpus like Wikipedia
- **math and theorem proving**: syntactic/semantic/computational patterns/abstractions/programs
- How do we **represent** math data (formulas, proofs, models) in our mind?

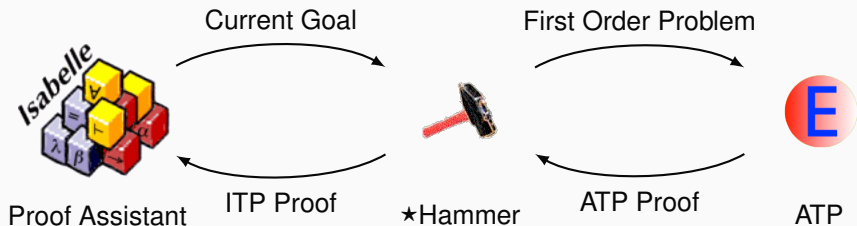
Using Learning to Guide Theorem Proving

- **high-level**: pre-select lemmas from a large library, give them to ATPs
- **high-level**: pre-select a good ATP strategy/portfolio for a problem
- **low-level**: guide every inference step of ATPs (tableau, superposition)
- **low-level**: guide every kernel step of LCF-style ITPs
- **mid-level**: guide application of tactics in ITPs, learn new tactics
- **mid-level**: invent suitable strategies/procedures for classes of problems
- **mid-level**: invent suitable conjectures for a problem
- **mid-level**: invent suitable concepts/models for problems/theories
- **proof sketches**: explore stronger/related theories to get proof ideas
- **theory exploration**: develop interesting theories by conjecturing/proving
- **feedback loops**: (dis)prove, learn from it, (dis)prove more, learn more, ...
- **autoformalization**: (semi-)automate translation from \LaTeX to formal
- ...

AI/TP Examples and Demos

- **ENIGMA/hammer proofs of Pythagoras** : <https://bit.ly/2MVPA7> (more at <http://grid01.ciirc.cvut.cz/~mptp/enigma-ex.pdf>) and simplified Carmichael <https://bit.ly/3oGBdRz>,
- **3-phase ENIGMA**: <https://bit.ly/3C0Lwa8>, <https://bit.ly/3BWqR6K>
- **Long trig proof from 1k axioms**: <https://bit.ly/2YZ0OgX>
- **Extreme Deepire/AVATAR proof of $\epsilon_0 = \omega^{\omega^{\omega^{\dots}}}$** <https://bit.ly/3Ne4WNX>
- **Hammering demo**: <http://grid01.ciirc.cvut.cz/~mptp/out4.ogv>
- **TacticToe on HOL4**: http://grid01.ciirc.cvut.cz/~mptp/tactictoe_demo.ogv
- **TacticToe longer**: <https://www.youtube.com/watch?v=BO4Y8ynwT6Y>
- **Tactician for Coq**: <https://blaauwbroek.eu/papers/cicm2020/demo.mp4>, <https://coq-tactician.github.io/demo.html>
- **Inf2formal over HOL Light**: <http://grid01.ciirc.cvut.cz/~mptp/demo.ogv>
- **QSynt: AI rediscovers the Fermat primality test**: <https://www.youtube.com/watch?v=24oejR9wsXs>

Today's AI-ATP systems (★-Hammers)

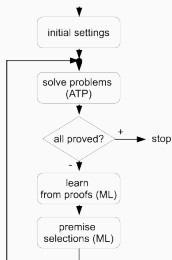


How much can it do?

- Mizar / MML – MizAR
- Isabelle (Auth, Jinja) – Sledgehammer
- Flyspeck (including core HOL Light and Multivariate) – HOL(y)Hammer
- HOL4 (Gauthier and Kaliszyk)
- CoqHammer (Czajka and Kaliszyk) - about 40% on Coq standard library
≈ 40-45% success by 2016, 60% on Mizar as of 2021

High-level feedback loops – MALARea, ATPBoost

- Machine Learner for Autom. Reasoning (2006) – infinite hammering
- feedback loop interleaving **ATP** with **learning premise selection**
- both syntactic and **semantic** features for characterizing formulas:
- evolving set of finite (counter)models in which formulas evaluated
- winning AI/ATP benchmarks (MPTPChallenge, CASC 08/12/13/18/20)
- ATPBoost (Piotrowski) - recent incarnation focusing on multiple proofs



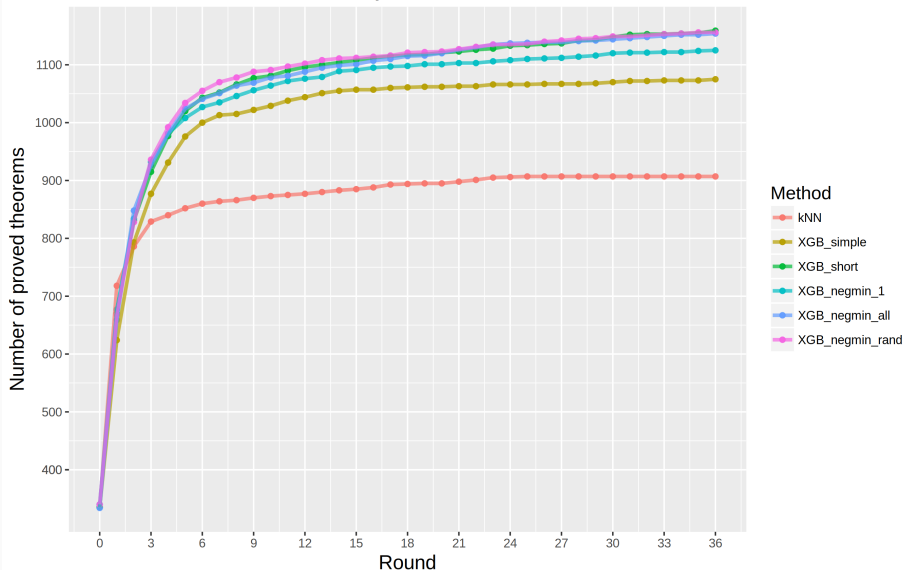
Results - Chromium

Startpage x scheduler x w time (Unix) x Startpage x Samuel Al x Schedule x Keynotes x Results x +

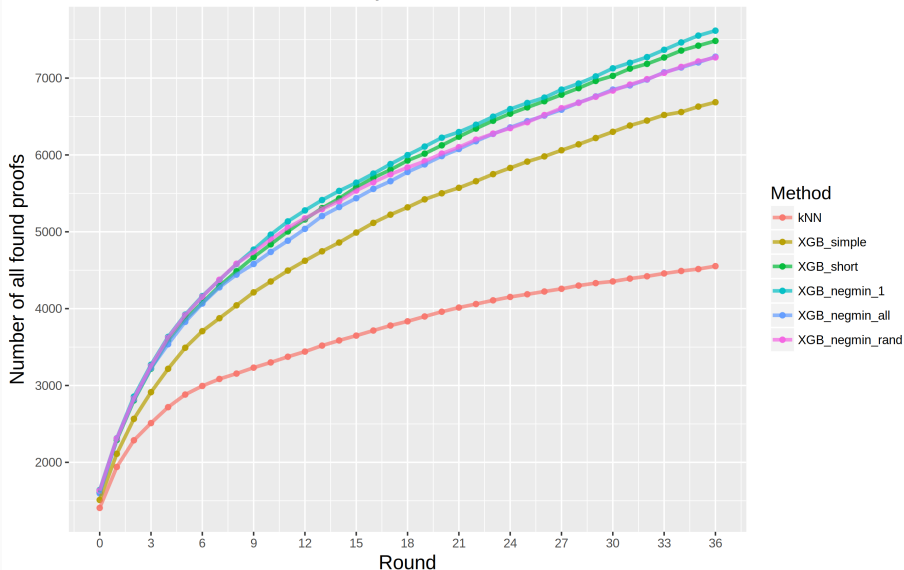
Not secure | ttp://CASC/J10/WWW/leq/DivisionSummary1.html

Large Theory Batch Problems	MaLARea 8.9	E LTB-2.5	Prover LTB-3.1	Zipperpin LTB-2.0	Leo-III LTB-1.5	ATPBoost 1.8	GKC LTB-0.5.1	Leo-III LTB-1.4
Solved/10000	7054/10000	3393/10000	3164/10000	1699/10000	1413/10000	1237/10000	493/10000	134/10000
Solutions	7054 70%	3393 33%	3163 31%	1699 16%	1413 14%	1237 12%	493 4%	134 1%

Prove-and-learn loop on MPTP2078 data set



Prove-and-learn loop on MPTP2078 data set



Various Improvements and Additions

- Model-based features for **semantic similarity** [IJCAR'08]
- Features encoding **term matching/unification** [IJCAI'15]
- Various learners: weighted k-NN, boosted trees (LightGBM, XGBoost)
- **Matching and transferring concepts** and theorems between libraries (Gauthier & Kaliszyk) – allows “superhammers”, conjecturing, and more
- **Lemmatization** – extracting and considering millions of low-level lemmas
- LSI, word2vec, neural models, definitional embeddings (with Google)
- Learning in **binary setting** from many **alternative proofs**
- Negative/positive mining (ATPBoost - Piotrowski & JU, 2018)
- **Stateful** neural methods: RNNs and Transformers (Piotrowski & JU, 2020) (smooth transition from fact selection to **conjecturing** – Jakubuv & JU 2020)
- **Currently strongest**: Name-independent graph neural nets (Olsak, 2020) (generalize very well to new terminology/lemmas)

Low-level: Statistical Guidance of Connection Tableau

- learn guidance of every clausal inference in connection tableau (leanCoP)
- set of first-order clauses, *extension* and *reduction* steps
- proof finished when all branches are *closed*
- a lot of *nondeterminism*, requires backtracking
- *Iterative deepening* used in leanCoP to ensure completeness
- good for learning – the tableau *compactly represents the proof state*

Clauses:

$$c_1 : P(x)$$

$$c_2 : R(x, y) \vee \neg P(x) \vee Q(y)$$

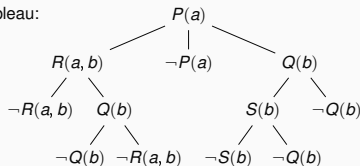
$$c_3 : S(x) \vee \neg Q(b)$$

$$c_4 : \neg S(x) \vee \neg Q(x)$$

$$c_5 : \neg Q(x) \vee \neg R(a, x)$$

$$c_6 : \neg R(a, x) \vee Q(x)$$

Closed Connection Tableau:



Statistical Guidance of Connection Tableau – rICoP

- 2018: strong learners via C interface to OCAML (**boosted trees**)
- **remove iterative deepening**, the prover can go arbitrarily deep
- added **Monte-Carlo Tree Search** (MCTS) (inspired by AlphaGo/Zero)
- MCTS search nodes are sequences of clause application
- a good heuristic to **explore new vs exploit** good nodes:

$$UCT(i) = \frac{w_i}{n_i} + c \cdot p_i \cdot \sqrt{\frac{\ln N}{n_i}} \quad (\text{UCT - Kocsis, Szepesvari 2006})$$

- learning both **policy (p)** (clause selection) and **value (w)** (state evaluation)
- clauses represented not by names but also by features (generalize!)
- **binary** learning setting used: | proof state | clause features |
- mostly term walks of length 3 (trigrams), **hashed** into small integers
- **many iterations of proving and learning**
- More recently also with GNNs (Olsak, Rawson, Zombori, ...)

Statistical Guidance of Connection Tableau – rlCoP

- On 32k Mizar40 problems using 200k inference limit
- nonlearning CoPs:

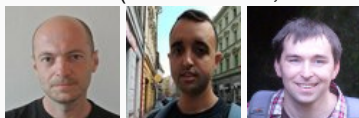
System	leanCoP	bare prover	rlCoP no policy/value (UCT only)
Training problems proved	10438	4184	7348
Testing problems proved	1143	431	804
Total problems proved	11581	4615	8152

- rlCoP with policy/value after 5 proving/learning iters on the training data
- $1624/1143 = 42.1\%$ improvement over leanCoP on the testing problems

Iteration	1	2	3	4	5	6	7	8
Training proved	12325	13749	14155	14363	14403	14431	14342	14498
Testing proved	1354	1519	1566	1595	1624	1586	1582	1591

ENIGMA (2017): Guiding the Best ATPs like E Prover

- ENIGMA (Jan Jakubuv, Zar Goertzel, Karel Chvalovsky, others)



- The proof state are two large heaps of clauses *processed/unprocessed*
- learn on E's proof search traces, put classifier in E
- positive examples: clauses (lemmas) used in the proof
- negative examples: clauses (lemmas) not used in the proof
- 2021 **multi-phase architecture** (combination of different methods):
 - fast gradient-boosted decision trees (GBDTs) used in 2 ways
 - fast logic-aware graph neural network (GNN - Olsak) run on a GPU server
 - logic-based subsumption using fast indexing (discrimination trees - Schulz)
- The GNN scores many clauses (context/query) together in a large graph
- Sparse - vastly more efficient than transformers ($\sim 100k$ symbols)
- 2021: leapfrogging and Split&Merge:
- aiming at learning **reasoning/algo components**

Feedback prove/learn loop for ENIGMA on Mizar data

- Done on 57880 Mizar problems recently
- Serious ML-guidance breakthrough applied to the best ATPs
- Ultimately a **70% improvement** over the original strategy in 2019
- From 14933 proofs to 25397 proofs (all 10s CPU - no cheating)
- Went up to 40k in more iterations and 60s time in 2020
- 75% of the Mizar corpus reached in July 2021 - higher times and many

runs: https://github.com/ai4reason/ATP_Proofs

	S	$S \odot M_9^0$	$S \oplus M_9^0$	$S \odot M_9^1$	$S \oplus M_9^1$	$S \odot M_9^2$	$S \oplus M_9^2$	$S \odot M_9^3$	$S \oplus M_9^3$
solved	14933	16574	20366	21564	22839	22413	23467	22910	23753
$S\%$	+0%	+10.5%	+35.8%	+43.8%	+52.3%	+49.4%	+56.5%	+52.8%	+58.4
$S+$	+0	+4364	+6215	+7774	+8414	+8407	+8964	+8822	+9274
$S-$	-0	-2723	-782	-1143	-508	-927	-430	-845	-454

	$S \odot M_{12}^3$	$S \oplus M_{12}^3$	$S \odot M_{16}^3$	$S \oplus M_{16}^3$
solved	24159	24701	25100	25397
$S\%$	+61.1%	+64.8%	+68.0%	+70.0%
$S+$	+9761	+10063	+10476	+10647
$S-$	-535	-295	-309	-183

ENIGMA Anonymous: Learning from patterns only

- The GNN and GBDTs only learn from formula **structure, not symbols**
- Not from symbols like $+$ and $*$ as Transformer & Co.
- E.g., learning on additive groups thus transfers to multiplicative groups
- **Evaluation** of old-Mizar ENIGMA on 242 new Mizar articles:
 - 13370 **new theorems**, $> 50\%$ of them with **new terminology**:
 - The 3-phase ENIGMA is **58%** better on them than unguided E
 - While **53.5%** on the old Mizar (where this ENIGMA was trained)
 - Generalizing, analogizing and transfer abilities **unusual in the large transformer models**

More Low-Level Guidance of Various Creatures

- Neural (TNN) clause selection in **Vampire** (Deepire - M. Suda):
Learn from clause *derivation trees only*
Not looking at what it says, just who its ancestors were.
- Fast and surprisingly good: Extreme Deepire/AVATAR proof of
 $\epsilon_0 = \omega^{\omega^{\omega^{\dots}}}$ <https://bit.ly/3Ne4WNX>
- 1193-long proof takes *about the same resources as one GPT-3/4 reply*
- GNN-based guidance in **iProver** (Chvalovsky, Korovin, Piepenbrock)
- New (*dynamic data*) way of training
- Led to **doubled** real-time performance of iProver's instantiation mode
- **CVC5**: neural & GBDT instantiation guidance (Piepenbrock, Jakubuv)
- very recently 20% improvement on Mizar
- **Hints** method for Otter/Prover9 (Veroff):
- boost inferences on clauses that match a lemma used in a related proof
- **symbolic ML** - can be combined with statistical - **proof completion vectors**

TacticToe: mid-level ITP Guidance (Gauthier'17,18)



- TTT learns from human and its own tactical HOL4 proofs
- No translation or reconstruction needed - native tactical proofs
- Fully integrated with HOL4 and easy to use
- Similar to rICoP: policy/value learning for applying tactics in a state
- Demo: http://grid01.ciirc.cvut.cz/~mptp/tactictoe_demo.ogv
- However much more technically challenging - a real breakthrough:
 - tactic and goal state recording
 - tactic argument abstraction
 - absolutization of tactic names
 - nontrivial evaluation issues
 - these issues have often more impact than adding better learners
- policy: which tactic/parameters to choose for a current goal?
- value: how likely is this proof state succeed?
- 66% of HOL4 toplevel proofs in 60s (better than a hammer!)
- similar followup work for HOL Light (Google), Coq, Lean, ...

Tactician: Tactical Guidance for Coq (Blaauwbroek'20)



- Tactical guidance of Coq proofs
- Technically very challenging to do right - the Coq internals again nontrivial
- 39.3% on the Coq standard library, 56.7% in a union with CoqHammer (orthogonal)
- Fast approximate hashing for k-NN makes a lot of difference
- Fast re-learning more important than “cooler”/slower learners
- Fully integrated with Coq, should work for any development
- **User friendly, installation friendly, integration/maintenance friendly**
- **Demo:** <https://blaauwbroek.eu/papers/cicm2020/demo.mp4>,
<https://coq-tactician.github.io/demo.html>
- Took several years, but could become a common tool for Coq formalizers
- Recently GNNs added, a major comparison of k-NN, GNN and LMs (Graph2Tac - <https://arxiv.org/abs/2401.02949>)

Neural Autoformalization (Wang et al., 2018)

- generate about 1M Latex - Mizar pairs synthetically (quite advanced)
- train neural seq-to-seq translation models (Luong – NMT)
- evaluate on about 100k examples
- many architectures tested, some work much better than others
- very important latest invention: attention in the seq-to-seq models
- more data crucial for neural training
- Recent addition: unsupervised MT methods (Lample et al 2018) – no need for aligned data, improving a lot!
- Type-checking not yet internal (boosting well-typed data externally)

Neural Autoformalization data

Rendered \LaTeX

Mizar

If $X \subseteq Y \subseteq Z$, then $X \subseteq Z$.

Tokenized Mizar

$X \subseteq Y \ \& \ Y \subseteq Z$ implies $X \subseteq Z$;

\LaTeX

$X \subseteq Y \ \& \ Y \subseteq Z$ implies $X \subseteq Z$;

Tokenized \LaTeX

If $\$X \subseteq Y \subseteq Z\$,$ then $\$X \subseteq Z\$.$

If $\$ X \subseteq Y \subseteq Z \$,$ then $\$ X \subseteq Z \$.$

Neural Fun – Performance after Some Training

Rendered
 \LaTeX

Input \LaTeX

Correct

Snapshot-
1000

Snapshot-
2000

Snapshot-
3000

Snapshot-
4000

Snapshot-
5000

Snapshot-
6000

Snapshot-
7000

Suppose s_8 is convergent and s_7 is convergent . Then $\lim(s_8+s_7) = \lim s_8 + \lim s_7$

Suppose $\{ s_{8} \}$ is convergent and $\{ s_{7} \}$ is convergent . Then $\lim (\{ s_{8} \} + \{ s_{7} \}) \mathrel{=} \lim \{ s_{8} \} + \lim \{ s_{7} \}$.

seq1 is convergent & seq2 is convergent implies $\lim (\text{seq1} + \text{seq2}) = (\lim \text{seq1}) + (\lim \text{seq2})$;

$x \text{ in dom } f \text{ implies } (x * y) * (f | (x | (y | (y | y)))) = (x | (y | (y | (y | y))))$;

seq is summable implies seq is summable ;

seq is convergent & $\lim \text{seq} = 0$ implies $\text{seq} = \text{seq}$;

seq is convergent & $\lim \text{seq} = \lim \text{seq}$ implies $\text{seq1} + \text{seq2}$ is convergent ;

seq1 is convergent & $\lim \text{seq2} = \lim \text{seq2}$ implies $\lim_{\text{inf}} \text{seq1} = \lim_{\text{inf}} \text{seq2}$;

seq is convergent & $\lim \text{seq} = \lim \text{seq}$ implies $\text{seq1} + \text{seq2}$ is convergent ;

seq is convergent & seq9 is convergent implies $\lim (\text{seq} + \text{seq9}) = (\lim \text{seq}) + (\lim \text{seq9})$;

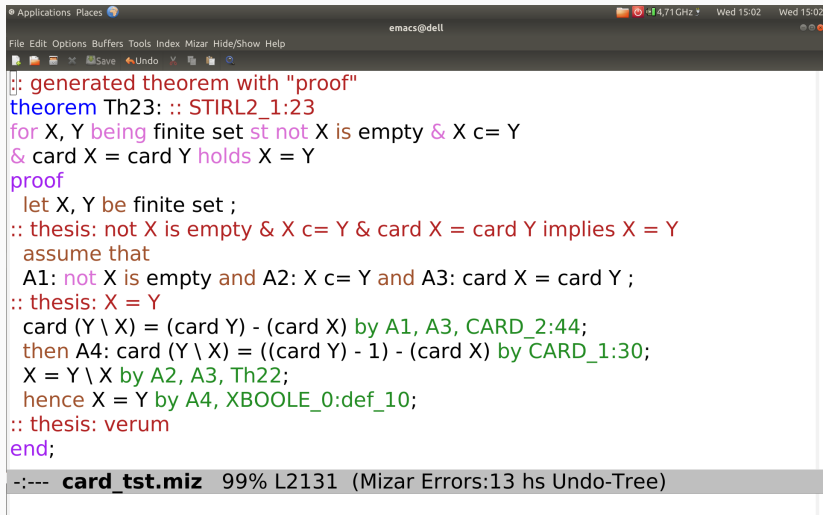
More on Conjecturing in Mathematics

- **Targeted**: generate intermediate lemmas (cuts) for a harder conjecture
- **Unrestricted** (theory exploration):
 - Creation of interesting conjectures based on the previous theory
 - One of the most interesting activities mathematicians do (how?)
 - Higher-level AI/reasoning task - can we learn it?
 - If so, we have solved math:
 - ... just (recursively) **divide** Fermat into many subtasks ...
 - ... and **conquer** (I mean: **hammer**) them away

Conjecturing and Proof Synthesis by Neural Methods

- Karpathy'15 - RNN experiments with generating fake Math over Stacks
- I have tried to use that for formal math in 2016 but it looked weak
- GPT (-2,3) looks stronger
- Renewed experiments in 2020 (JU & J. Jakubuv: First Neural Conjecturing Datasets and Experiments. C1CM'20) on:
 - All Mizar articles, stripped of comments and concatenated together (78M)
 - Articles with added context/disambiguation (156M) (types, names, thesis)
 - TPTP proofs of 28271 Mizar/MPTP theorems by E/ENIGMA (658M)
 - Just the conjecture and premises needed for the 28271 proofs printed in prefix notation
- Quite interesting results, server for Mizar authors
- Quickly taken up by others on HOL, Isabelle, MetaMath ...
- **Caveat:** Watch for "model pretraining" on undisclosed corpora - often GitHub/math repos that may contain (translations of) the testing data

Can you find the flaw(s) in this fake GPT-2 proof?



```
Applications Places emacs@dell
File Edit Options Buffers Tools Index Mizar Hide/Show Help
Save Undo
:: generated theorem with "proof"
theorem Th23: :: STIRL2_1:23
for X, Y being finite set st not X is empty & X c= Y
& card X = card Y holds X = Y
proof
  let X, Y be finite set ;
  :: thesis: not X is empty & X c= Y & card X = card Y implies X = Y
  assume that
  A1: not X is empty and A2: X c= Y and A3: card X = card Y ;
  :: thesis: X = Y
  card (Y \ X) = (card Y) - (card X) by A1, A3, CARD_2:44;
  then A4: card (Y \ X) = ((card Y) - 1) - (card X) by CARD_1:30;
  X = Y \ X by A2, A3, Th22;
  hence X = Y by A4, XBOOLE_0:def_10;
  :: thesis: verum
end;
-:--- card_tst.miz 99% L2131 (Mizar Errors:13 hs Undo-Tree)
```

Figure: Fake full declarative GPT-2 “Mizar proof” - typechecks!

Gibberish Generator Provoking Algebraists

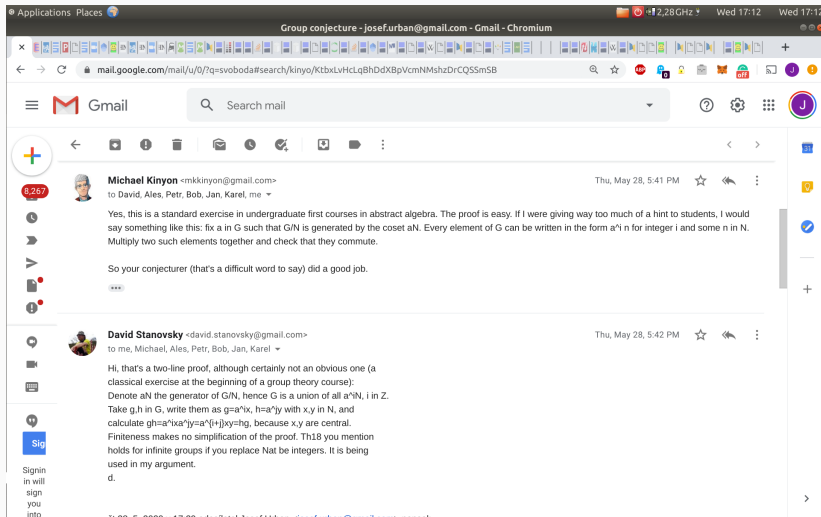


Figure: First successes in making mathematicians comment on AI.

A correct conjecture that was too hard to prove

Kinyon and Stanovsky (algebraists) confirmed that this cut is valid:

```
theorem Th10: :: GROUPE_1:10
for G being finite Group
for N being normal Subgroup of G st
N is Subgroup of center G & G ./ N is cyclic
holds G is commutative
```

The generalization that avoids finiteness:

```
for G being Group
for N being normal Subgroup of G st
N is Subgroup of center G & G ./ N is cyclic
holds G is commutative
```

More cuts

- In total 33100 in this experiment
- Ca 9k proved by trained ENIGMA
- Some are clearly false, yet quite natural to ask:

`theorem :: SIN COS 10:17`

`sec is increasing on [0, pi/2)`

leads to conjecturing the following:

`Every differentiable function is increasing.`

Caveats, Comments and Dark Sides of ML

- **Training on testing data** – used to be a big taboo
- Useful as a sanity check: if the trained predictor doesn't predict well on the training data, there is a problem (**garbage in garbage out**)
- For simple ML systems training on testing data is less troublesome (e.g. naive Bayes with few features)
- Even for the GNN its performance on train/test split is similar
- For large systems like LLMs, this may however be a big issue - memorization vs generalization
- **"pretraining"**: e.g. word embeddings useful for many NLP applications
- capture a lot of "latent semantics" of words, similar for pretrained LLMs
- However, they can also store/memorize any dataset they have seen on internet/GitHub
- E.g. formal proofs written in Lean instead of Coq, Mizar instead of Isabelle, HOL4 instead of HOL light
- Proofwiki/Arxiv versions of the proofs and the important lemmas in them (later used e.g. for conjecturing in the formal setting)

Caveats, Comments and Dark Sides of ML

- Use of synthetic data
- Can be very useful
- In general, there is a large field of data augmentation related to unsupervised and semi supervised learning
- e.g., try to generate many related easier problems, solve those, and learn on them to attack a harder problem
- a major issue is training and testing on an oversaturated dataset:
- even though the test data are formally disjoint from the training set, they can be very close to many training examples
- the synthetic data (and their parameterized generators) may in various hidden/clandestine ways target the test set
- a serious issue: unreleased synthetic data/generators are today used to claim major "breakthroughs"

Caveats, Comments and Dark Sides of ML

- **Leaks from the future:**
- chronological eval vs random split where the split contains the names of lemmas not yet seen in the chronological evaluation
- chronological evaluation is obviously the best but many ML systems are not easily updatable
- weaker learners like naive Bayes behaved similarly under the random split and in the chronological evaluation
- **anonymous settings** (typically not LLMs) largely mitigate this - eg the symbol independent GNN, or the binary setting when using GBDTs with anonymous features
- in nonanonymous settings, in-context learning can perhaps be used, this however so far requires a lot of resources
- also, there could be just a single evaluation - e.g. on new articles in a new version of the library (our Mizar 60 eval quite surprising)

Thanks and Advertisement

- Thanks for your attention!
- To push AI methods in math and theorem proving, we organize:
- **AITP – Artificial Intelligence and Theorem Proving**
- September 2025, Aussois, France, aitp-conference.org
- ATP/ITP/Math vs AI/ML/AGI people, Computational linguists
- Discussion-oriented and experimental