Lean:
Past, Present, and Future

Sebastian Ullrich, Lean FRO

FSCD, 2024-07-13

Lean Beginnings

The Strategy Challenge in SMT Solving

Leonardo de Moura[1] and Grant Olney Passmore[2,3]
leonardo@microsoft.com, grant.passmore@cl.cam.ac.uk

[1] Microsoft Research, Redmond
[2] Clare Hall, University of Cambridge
[3] LFCS, University of Edinburgh

Abstract. High-performance SMT solvers contain many tightly integrated, hand-crafted heuristic combinations of algorithmic proof methods. While these heuristic combinations tend to be highly tuned for known classes of problems, they may easily perform badly on classes of problems not anticipated by solver developers. This issue is becoming increasingly pressing as SMT solvers begin to gain the attention of practitioners in diverse areas of science and engineering. We present a challenge to the SMT community: to develop methods through which users can exert strategic control over core heuristic aspects of SMT solvers. We present evidence that the adaptation of ideas of strategy prevalent both within the Argonne and LCF theorem proving paradigms can go a long way towards realizing this goal.

Lean Beginnings

Created as a platform for *white-box automation* by Leonardo de Moura

Intended as a frontend-agnostic answer to the Strategy Challenge

A *lean* kernel: minimized type theory compared to similar systems

Lean 0.1 (2014)

An interactive theorem prover

```
theorem nat_trans3i {a b c d : Nat} (H1 : a = b) (H2 : c = b) (H3 : c = d) : a = d
:= trans (trans H1 (symm H2)) H3
```

Valid Lean 0.1 ~~2~~ ~~3~~ 4 code!

Small set of built-in tactics, more could be added with Lua

```
tactic_macro("simp_no_assump", { macro_arg.Ids },
             function (env, ids)
                if #ids == 0 then
```

Lean 2 (2015)

Added support for inductive types, type theory mostly unchanged since

Optional Homotopy Type Theory mode

Lean’s Type Theory

A dependent type theory based on the *Calculus of Inductive Constructions*

Additions for classical mathematics and program verification:

- A definitionally proof-irrelevant universe of propositions *Prop*
- Quotient types

Adjustments for simplicity and leanness:

- Recursion compiled down to *recursor* higher-order functions
- No universe cumulativity (since Lean 2)

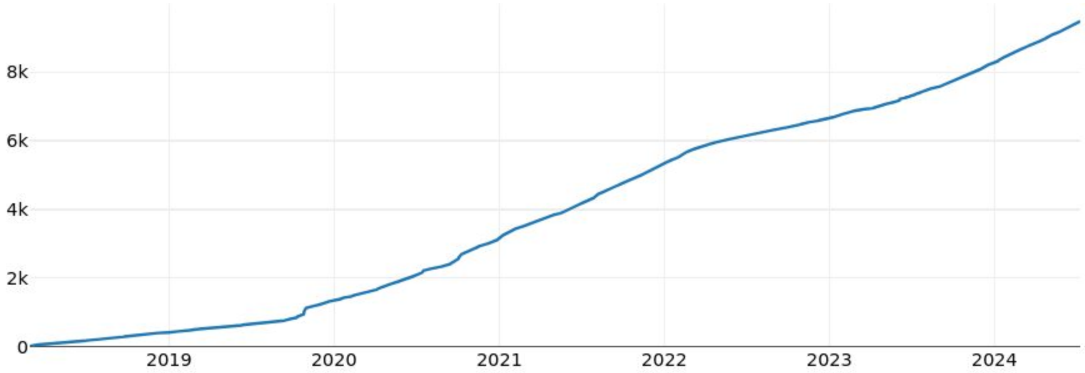
Lean And Mathematicians

Expressive, classical-focused logic

Growing a community and library, first at CMU, then on Zulip

Organization summary

- Number of users: 9,478
- Users active during the last 15 days: 762
- Number of guests: 0
- Total number of messages: 1,452,855
- Number of messages in the last 30 days: 24,687
- File storage in use: 6.3 GB



Start Of My Involvement

leodemoura commented on May 14, 2015

Today, I'm going to cry of happiness :-)
Thanks a lot for going into the source code and making the necessary modifications!
They are very welcome!

Lean 3 (2017)

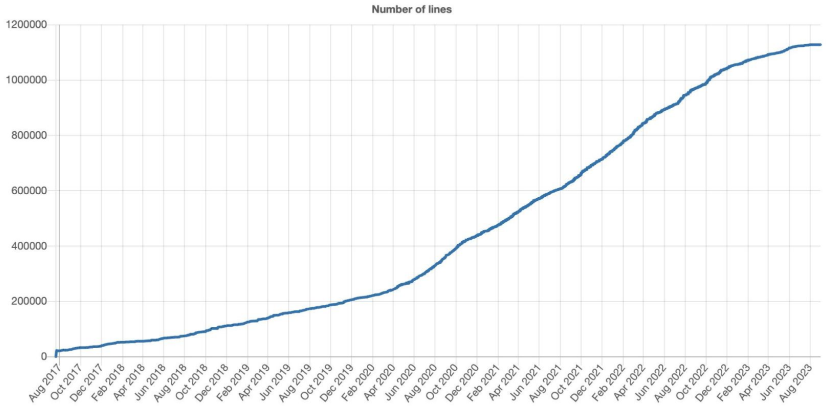
Made Lean a *metaprogramming* language

```
meta def assumption : tactic unit :=
do { ctx ← local_context,
     t   ← target,
     h   ← find t ctx,
     exact h }
<|> fail "assumption tactic failed"
```

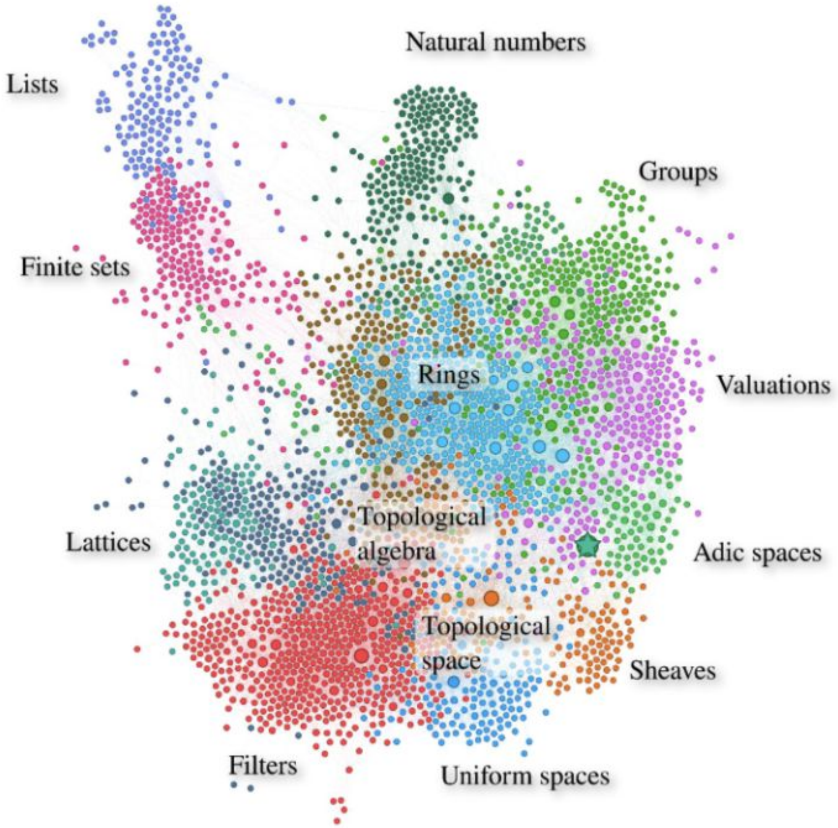
Removed HoTT support

Mathlib spun out as a separate library

Definitions	Theorems	Contributors
66599	122987	310



Mathlib



Natural numbers
Lists
Groups
Finite sets
Rings
Valuations
Topological
algebra
Lattices
Adic spaces
Topological
space
Sheaves
Filters
Uniform spaces

Liquid Tensor Experiment

Nov 2020: Peter Scholze posits formalization challenge

> "I spent much of 2019 obsessed with the proof of this theorem, almost getting crazy over it. In the end, we were able to get an argument pinned down on paper, but I think nobody else has dared to look at the details of this, and so I still have some small lingering doubts."

Liquid Tensor Experiment

Nov 2020: Peter Scholze posits formalization challenge

May 2021: Johan Commelin announces completed Lean formalization of crucial intermediary lemma, with only minor corrections

> “[T]his was precisely the kind of oversight I was worried about when I asked for the formal verification. [...] The proof walks a fine line, so if some argument needs constants that are quite a bit different from what I claimed, it might have collapsed.”

Liquid Tensor Experiment

Nov 2020: Peter Scholze posits formalization challenge

May 2021: Johan Commelin announces completed Lean formalization of crucial intermediary lemma, with only minor corrections

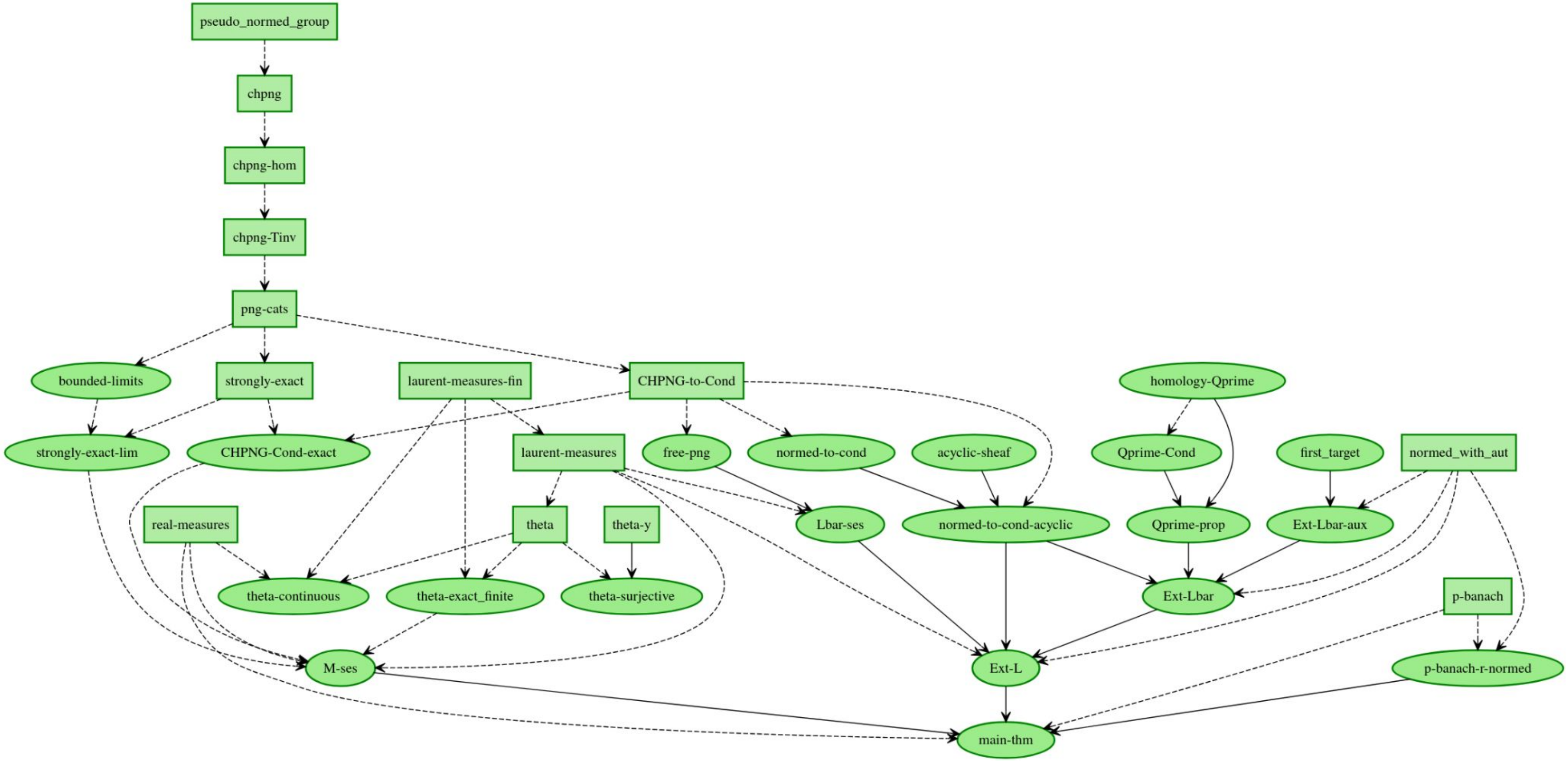
July 2022: Completion of the full challenge in Lean

Liquid Tensor Experiment

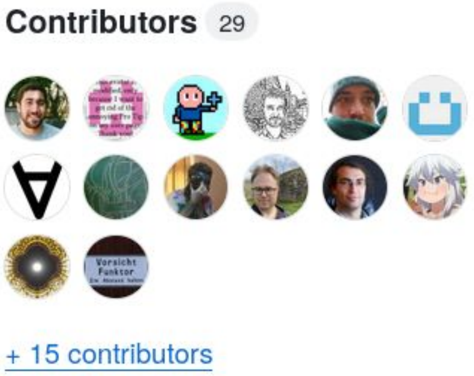
```
variables (p' p : ℝ≥0) [fact (0 < p')] [fact (p' < p)] [fact (p ≤ 1)]

theorem liquid_tensor_experiment (S : Profinite.{0}) (V : pBanach.{0} p) :
  ∀ i > 0, Ext i (ℳ_{p'} S) V ≅ 0 :=
```

Liquid Tensor Experiment



Crowd-Sourced Mathematics



“The beauty of the system: you do not have to understand the whole proof of FLT in order to contribute. The blueprint breaks down the proof into many many small lemmas, and if you can formalise a proof of just one of those lemmas then I am eagerly awaiting your pull request.” – [Kevin Buzzard on the FLT Project]

Liquid Tensor Experiment

Theorem 2.4.15 (Clausen-Scholze)✓

Let $0<p'<p \leq 1$ be real numbers, let S be a profinite set, and let V be a p-Banach space. Let $\mathcal{M}_{p'}(S)$ be the space of real p'-measures on S. Then

$$\mathrm{Ext}^i_{\mathrm{Cond(Ab)}}(\mathcal{M}_{p'}(S), V) = 0$$

for $i \geq 1$.

Proof ▾

Recall from Lemma 2.4.14 the short exact sequence

$$0 \longrightarrow \mathcal{L}_{r'}(S) \longrightarrow \mathcal{L}_{r'}(S) \longrightarrow \mathcal{M}_{p'}(S) \longrightarrow 0.$$

Apply to this $\mathrm{Ext}^*(\_, V)$ to obtain a long exact sequence. Note that T acts on V via multiplication by $\frac{1}{2}$ (by Lemma 2.1.2). Hence we can use Lemma 2.4.13 to obtain isomorphisms between the Ext-groups involving $\mathcal{L}_{r'}(S)$, for $i>0$, and a surjection for $i = 0$. The result follows. □

Only The Beginning

Sphere Eversion, Massot, Nash, and van Doorn, 2020-2022

Fermat’s Last Theorem for regular primes, Brasca et al., 2021-2023

Unit Fractions, Bloom and Mehta, 2022

Consistency of Quine's New Foundations, Wilshaw and Dillies, 2022-2024

Polynomial Freiman-Ruzsa Conjecture, Tao and Dillies, 2023

Prime Number Theorem And Beyond, Kontorovich and Tao, 2024-ongoing

Carleson Project, van Doorn, 2024-ongoing

Fermat’s Last Theorem, Buzzard, 2024-ongoing

Lean 4 (2023)

Current version of Lean

Made Lean a *general-purpose* programming language

Implemented in 120+ kLoC of Lean!

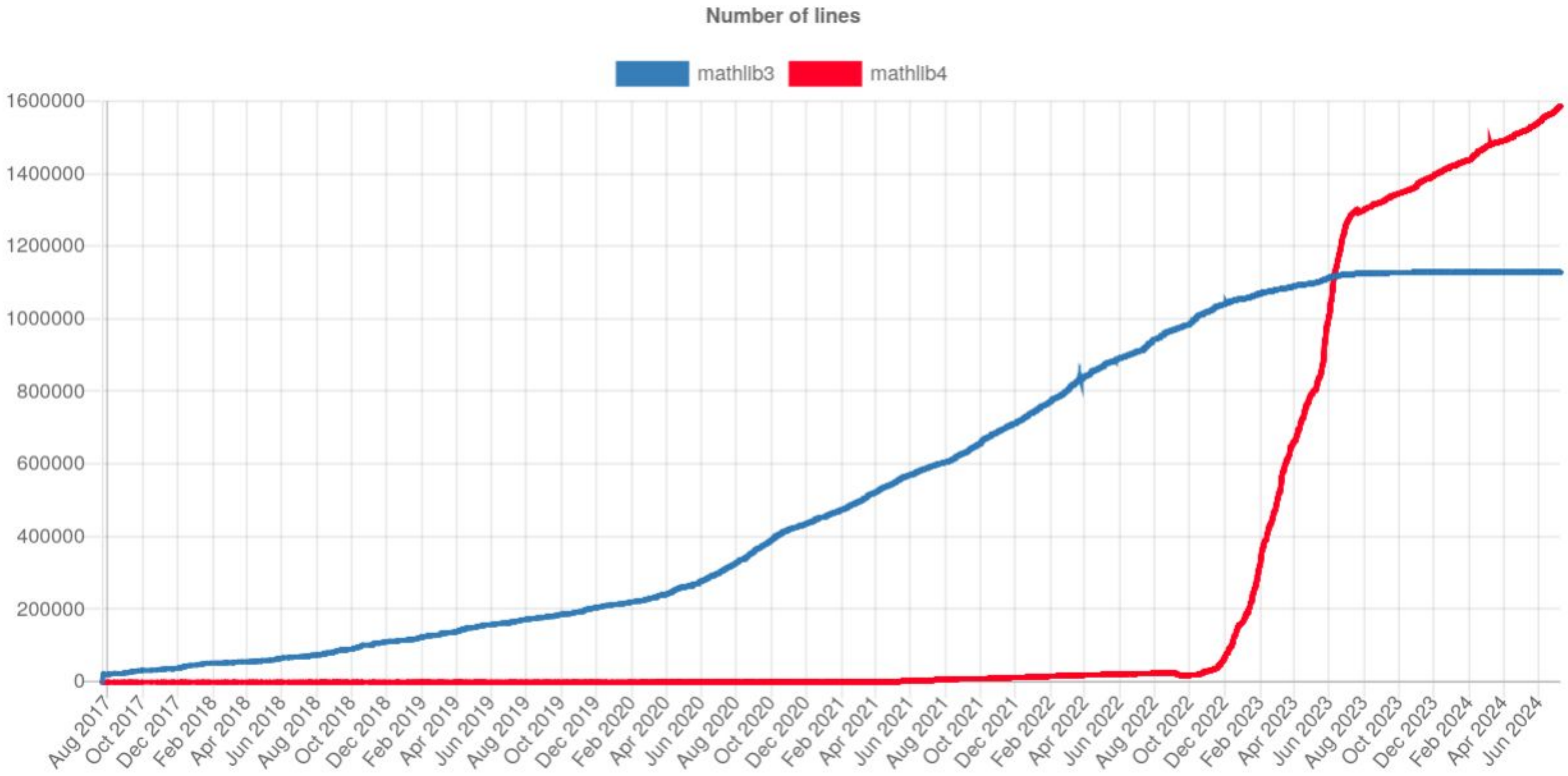
Opened up parser and elaborator for complex notations, embedded languages, …

```
#doc (Post) "Functional induction" =>

%%%
authors := ["Joachim Breitner"]
date := ⟨2024, 5, 17⟩
categories := [technical]
%%%

Proving properties of recursive function
```

Mathlib 4



Number of lines
mathlib3
mathlib4
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
Aug 2017
Oct 2017
Dec 2017
Feb 2018
Apr 2018
Jun 2018
Aug 2018
Oct 2018
Dec 2018
Feb 2019
Apr 2019
Jun 2019
Aug 2019
Oct 2019
Dec 2019
Feb 2020
Apr 2020
Jun 2020
Aug 2020
Oct 2020
Dec 2020
Feb 2021
Apr 2021
Jun 2021
Aug 2021
Oct 2021
Dec 2021
Feb 2022
Apr 2022
Jun 2022
Aug 2022
Oct 2022
Dec 2022
Feb 2023
Apr 2023
Jun 2023
Aug 2023
Oct 2023
Dec 2023
Feb 2024
Apr 2024
Jun 2024

Software Verification in Lean 4

Formalizing Cedar in Lean: QED

Models	Lean LOC	Dafny LOC	L/D
Data	246		
Spec	951	1707	.56
Validator	532	1189	.45
Total	**1729**	**2896**	**.60**

Proofs	Lean LOC	Dafny LOC	L/D
Data	681		
Authz	350	394	.89
Validator	4686	3110	1.5
Total	**5717**	**3504**	**1.6**

Verification	Lean (s)	Dafny (s)	L/D
All proofs	185	519	.36

Testing (per request)	Rust (µs)	Lean (µs)	Dafny Java (µs)
abac	7	4	3325
abac-typed	7	5	3410

Productivity: proved validator soundness in 18 person-days

Maintainability: replaced Mathlib with Std in a couple of hours

SampCert

A verified implementation using Lean and Mathlib of the discrete Gaussian sampler for differential privacy, the composition and postprocessing of zero concentrated differential privacy, and some simple queries.

AI in Lean

OpenAI: Solving (some) formal math olympiad problems

Meta AI: Teaching AI advanced mathematical reasoning

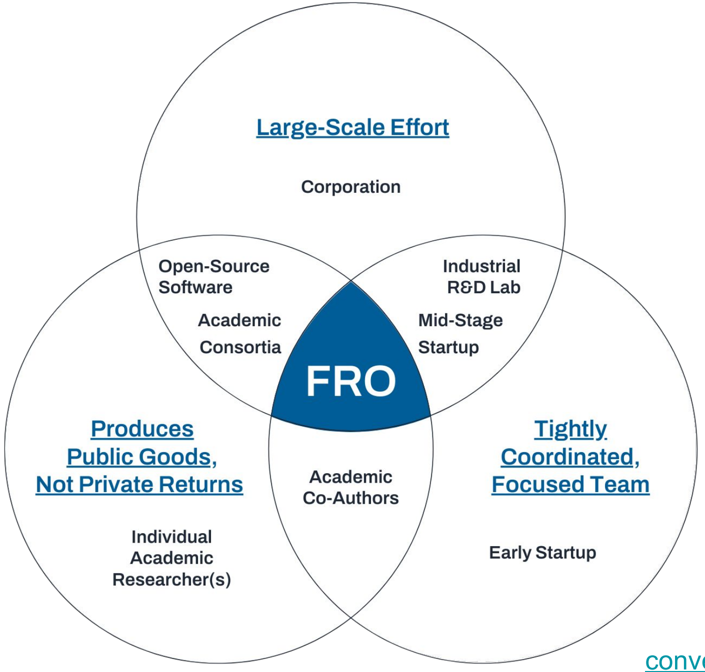
DeepMind: Scalable AI Safety via Doubly-Efficient Debate

LeanDojo: open source models, datasets, and code for Lean

Morph labs is developing ML models for Lean and moogle.ai

Focused Research Organization (FRO)

A new type of nonprofit startup for science developed by Convergent Research



convergentresearch.org

The Lean FRO

A non-profit organization dedicated to the development of Lean

Missions:

- Address **scalability**, **usability**, and **proof automation** in Lean.
- Support formal mathematics.
- Achieve self-sustainability in 5 years.

Supported by Simons Foundation International, Alfred P. Sloan Foundation, and Richard Merkin

lean-fro.org

The Lean FRO

Leo de Moura (AWS)
Chief Architect, Co-Founder

Sebastian Ullrich
Head of Engineering, Co-Founder

Corinna Calhoun
Chief Operating Officer

Henrik Böving
Research Software Engineer

Joachim Breitner
Senior Research Software Engineer

David Thrane Christiansen
Senior Research Software Engineer

Johan Commelin
Mathematical Research Engineer

Markus Himmel
Research Software Engineer

Marc Huisinga
Research Software Engineer

Mac Malone
Research Software Engineer

Kyle Miller
Research Software Engineer

Kim Morrison
Senior Research Software Engineer

Sofia Rodrigues
Research Software Engineer

The Lean FRO: Year 1

Documentation: Verso authoring tool, lean-lang.org/blog/

System: incremental proof processing

Packaging: Reservoir package registry

Automation: functional induction, `omega`

Library: verified hash map, bitvector

Infrastructure: Mathlib cache hosting, continuous benchmarking

… and many other improvements across the system

The Lean FRO: Roadmap

Documentation: reference manual, metaprogramming guide

System: parallel processing, module system

Packaging/Infrastructure: cloud build & cache

Automation: SMT primitives, and more

Library: more data structures, programming fundamentals

Conclusion

10 years of Lean, a system that grew with the ambitions of its users

Introductory reading:

- Functional Programming in Lean
- Theorem Proving in Lean 4
- Mathematics in Lean

lean-lang.org