Neural theorem proving

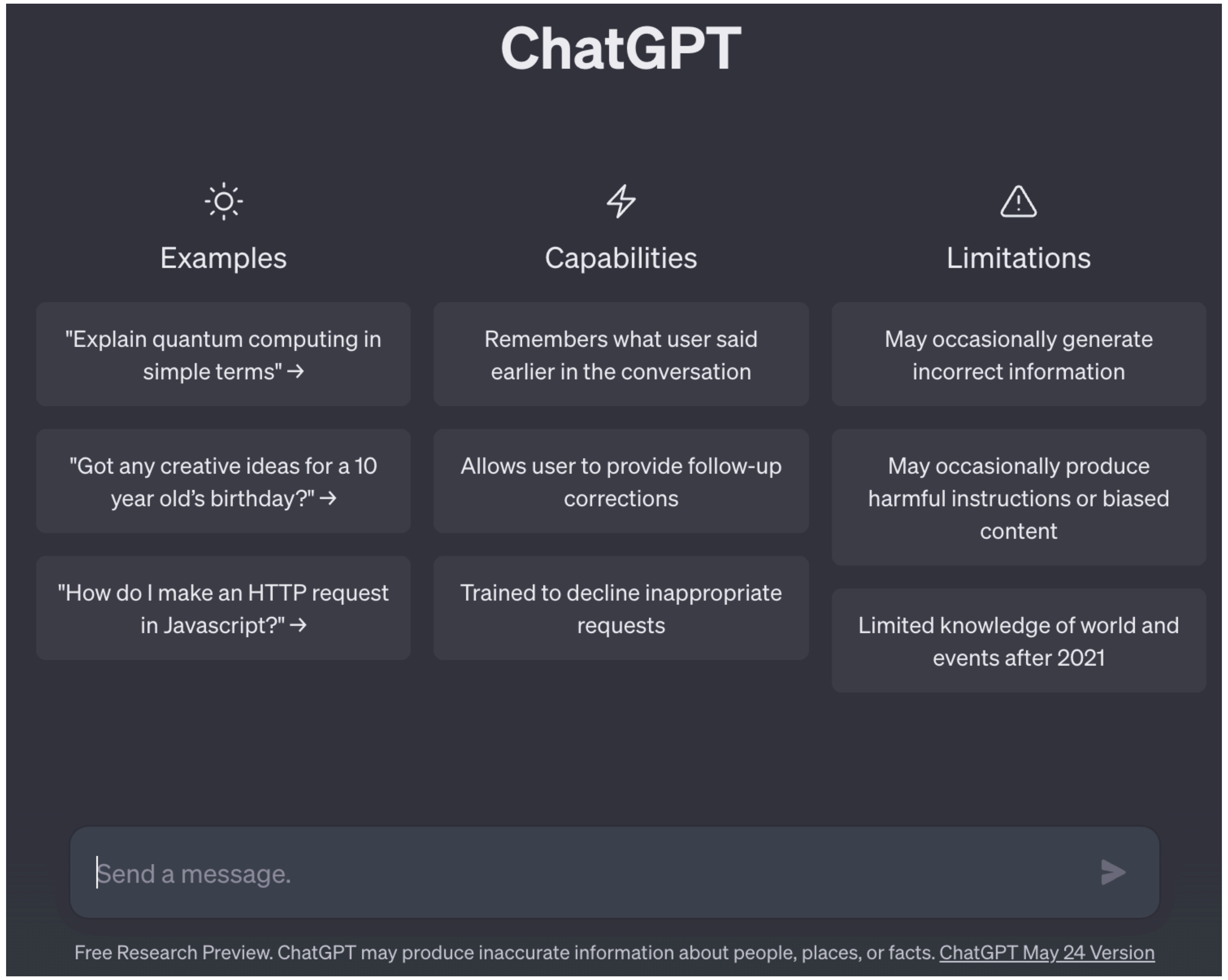
Sean Welleck

Carnegie Mellon University

UNIVERSITY of WASHINGTON

Ai2 Allen Institute for AI

Language models and mathematics



Question: For every $a, b, b \neq a$ prove that

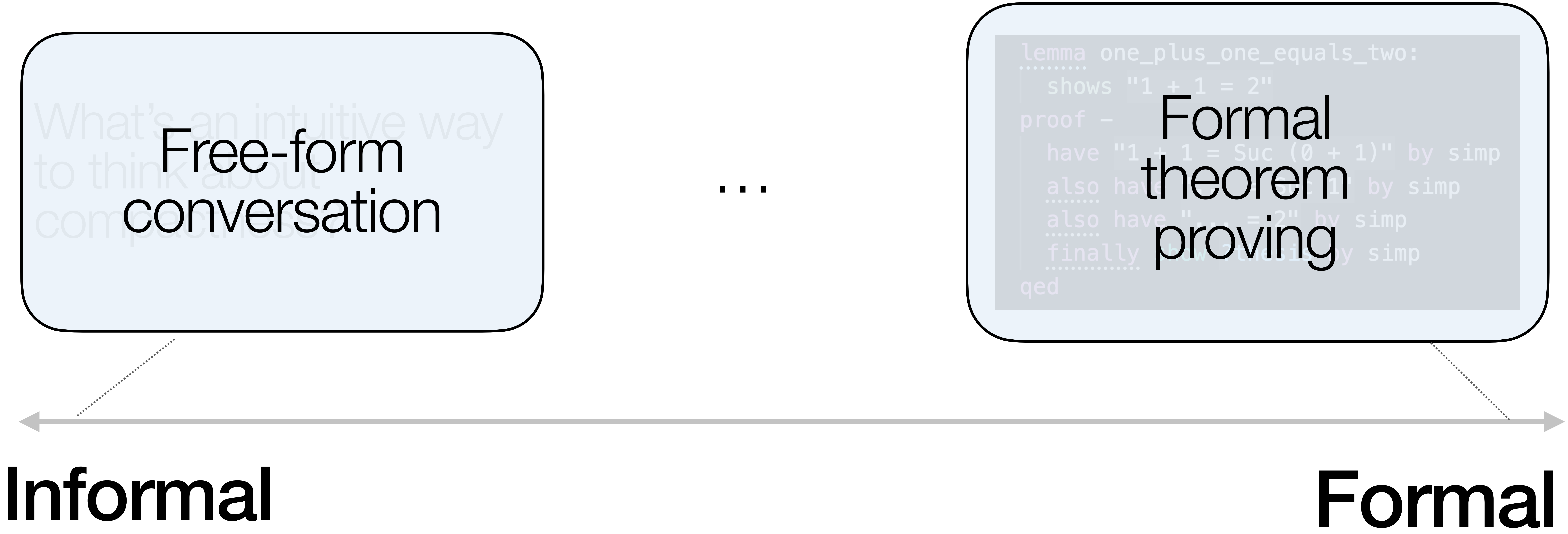
$$\frac{a^2+b^2}{2} > \left(\frac{a+b}{2}\right)^2.$$

Model output:

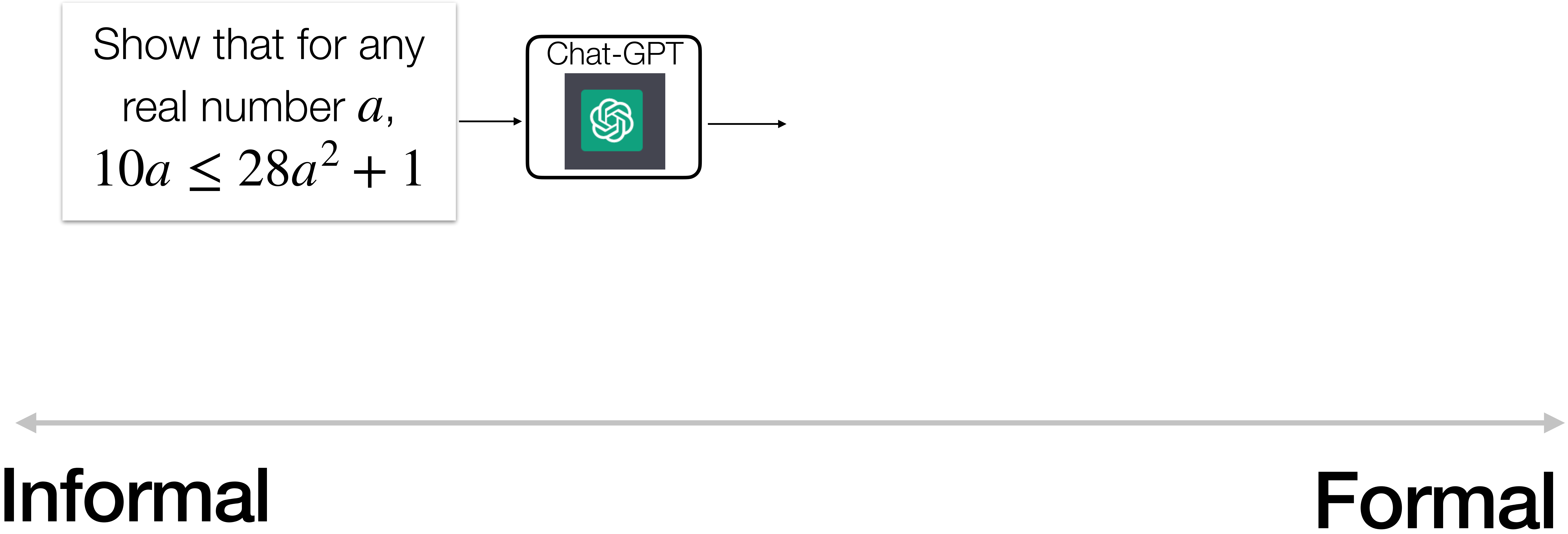
$$\frac{a^2+b^2}{2} > \left(\frac{a+b}{2}\right)^2$$
$$\iff \frac{a^2+b^2}{2} > \frac{a^2+b^2+2ab}{4}$$
$$\iff a^2+b^2 > \frac{a^2+b^2+2ab}{2}$$
$$\iff 2a^2+2b^2 > a^2+b^2+2ab$$
$$\iff a^2+b^2 > 2ab$$
$$\iff a^2+b^2-2ab > 0$$
$$\iff (a-b)^2 > 0$$

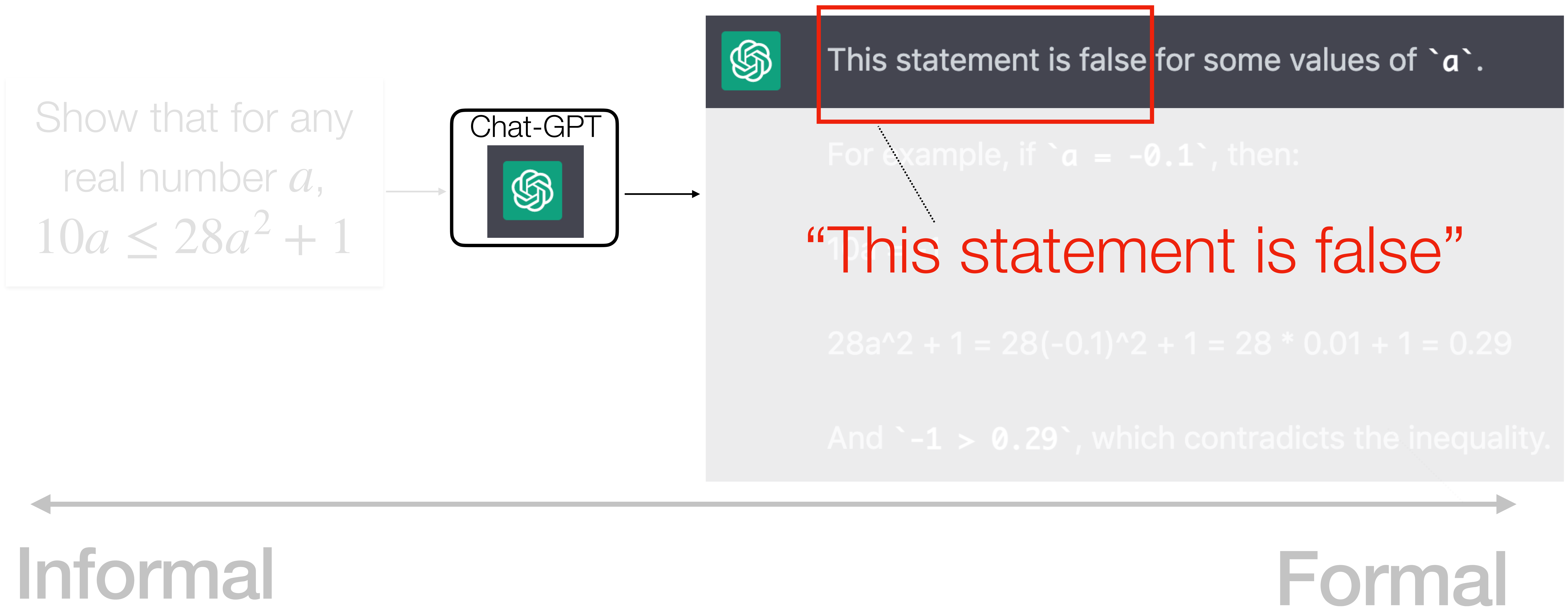
which is true, because the square of a real number is positive.

Minerva Language Model
2022 Poland National Math Exam
[Lewkowycz et al 2022]

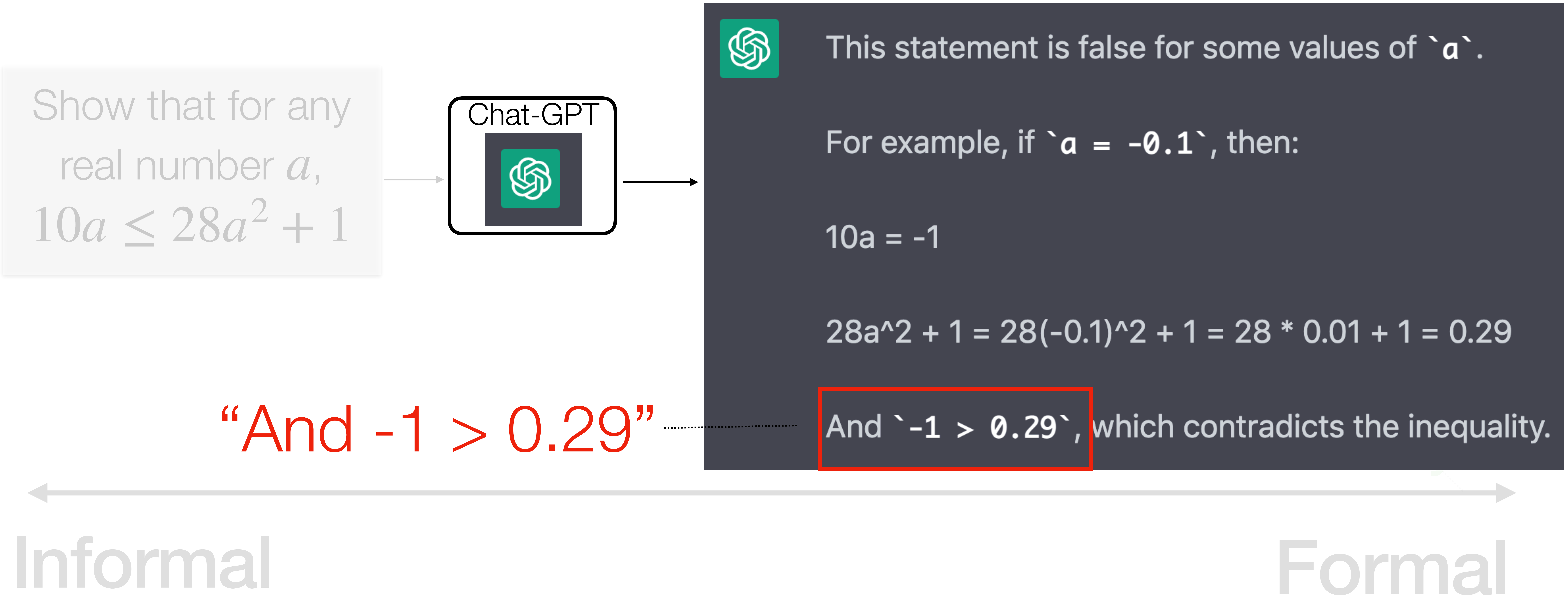


Free-form conversation
...
Formal theorem proving
Informal
Formal


Show that for any real number a, $10a \leq 28a^2 + 1$
Chat-GPT
Informal
Formal



Show that for any real number a, $10a \leq 28a^2 + 1$
Chat-GPT
This statement is false for some values of `a`.
For example, if `a = -0.1`, then:
28a^2 + 1 = 28(-0.1)^2 + 1 = 28 * 0.01 + 1 = 0.29
And `-1 > 0.29`, which contradicts the inequality.
“This statement is false”
Informal
Formal


Show that for any real number a, $10a \leq 28a^2 + 1$
Chat-GPT
This statement is false for some values of `a`.
For example, if `a = -0.1`, then:
10a = -1
28a^2 + 1 = 28(-0.1)^2 + 1 = 28 * 0.01 + 1 = 0.29
And `-1 > 0.29`, which contradicts the inequality.
"And -1 > 0.29"
Informal
Formal

Hard to evaluate

Informal

Formal

Interactive (formal) theorem proving

$1 + 1 = 2$

```
lemma one_plus_one_equals_two:
  shows "1 + 1 = 2"
proof -
  have "1 + 1 = Suc (0 + 1)" by simp
  also have "... = Suc 1" by simp
  also have "... = 2" by simp
  finally show ?thesis by simp
qed
```

Formal

Interactive theorem proving

$1 + 1 = 2$

proof ✓

```
lemma one_plus_one_equals_two:
  shows "1 + 1 = 2"
proof -
  have "1 + 1 = Suc (0 + 1)" by simp
  also have "... = Suc 1" by simp
  also have "... = 2" by simp
  finally show ?thesis by simp
qed
```

Formal

gpt-f

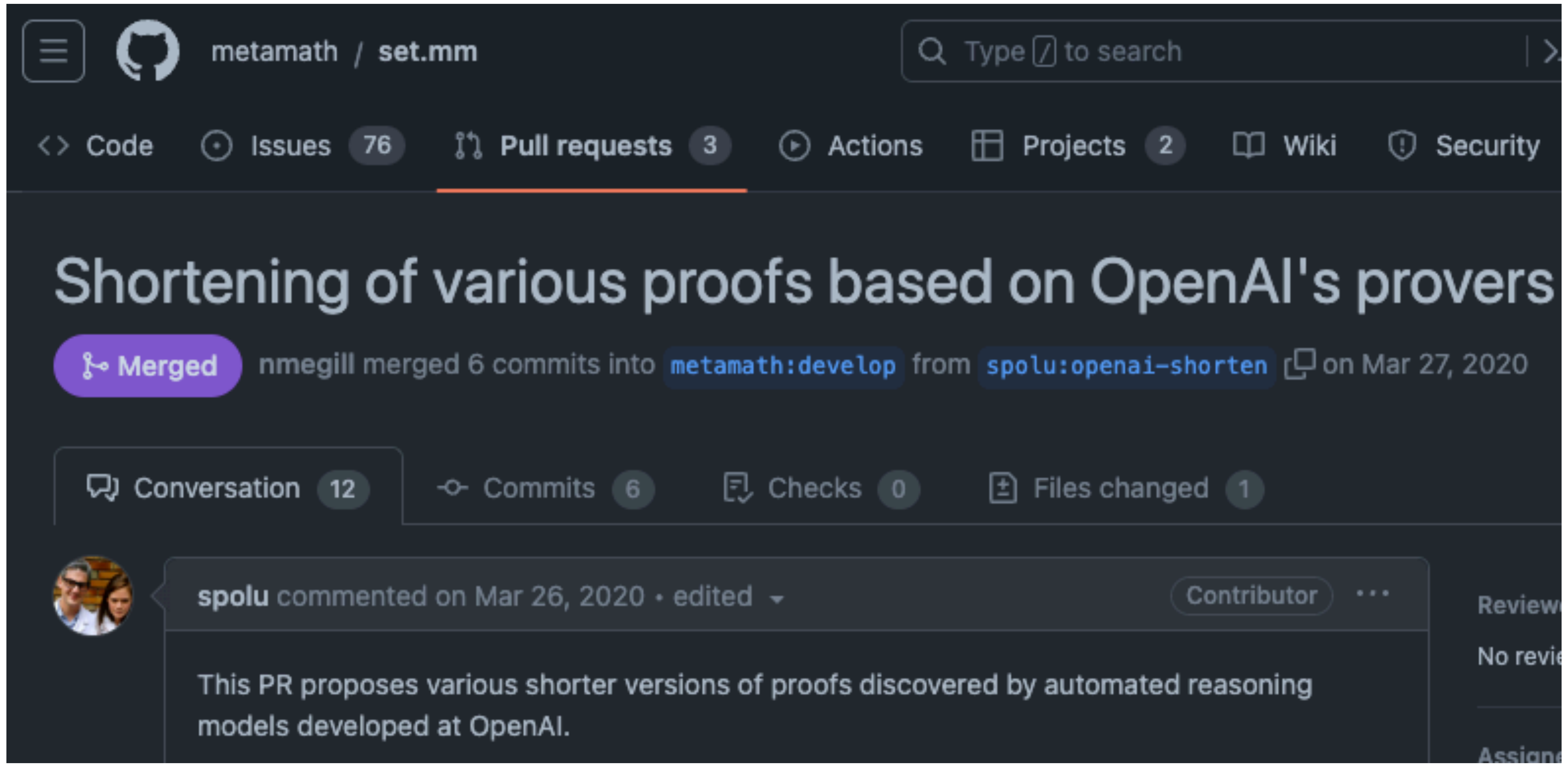
Generative Language Modeling for Automated Theorem Proving

Stanislas Polu
OpenAI
spolu@openai.com

Ilya Sutskever
OpenAI
ilyasu@openai.com

Formal

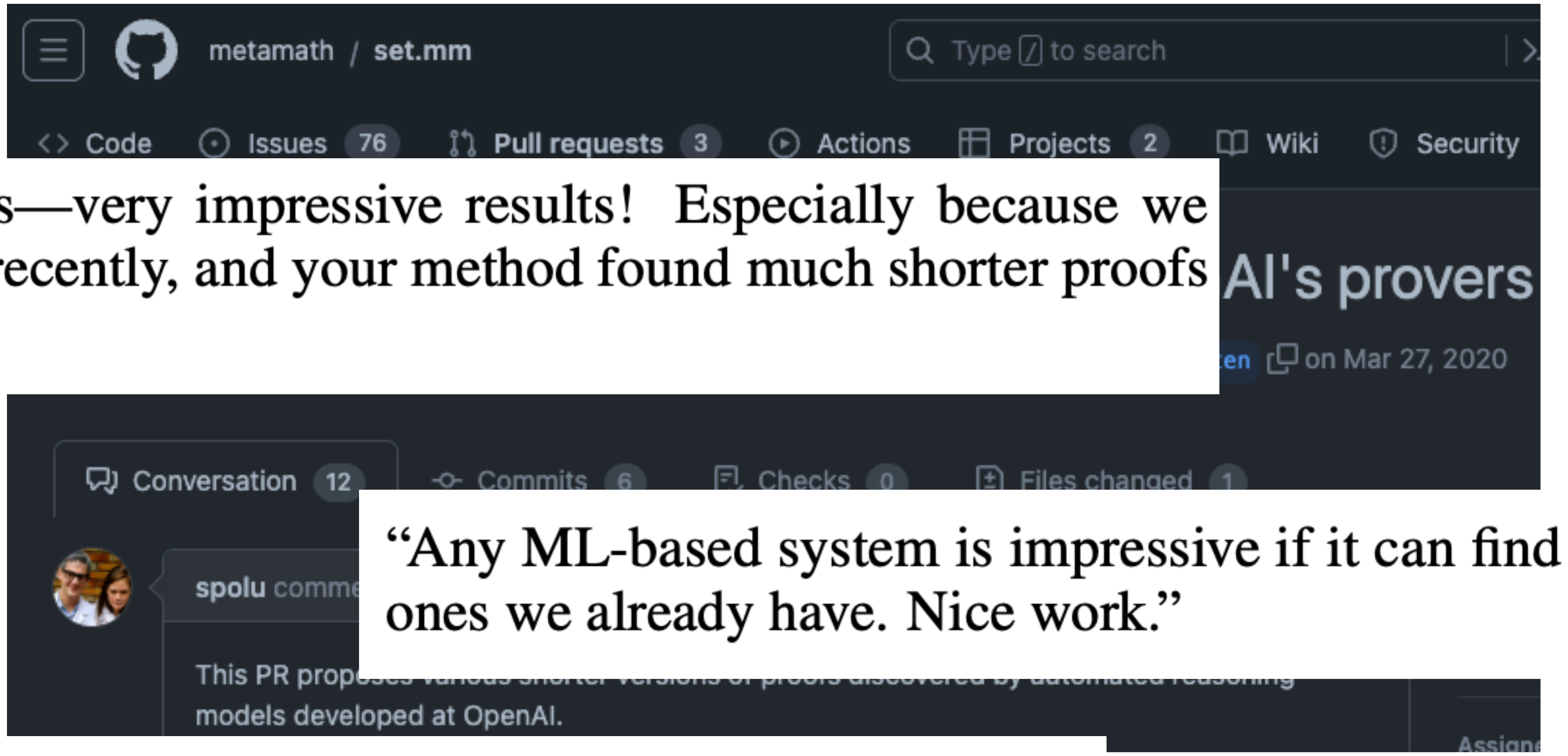
gpt-f



metamath / set.mm
Type / to search
Code
Issues 76
Pull requests 3
Actions
Projects 2
Wiki
Security
Shortening of various proofs based on OpenAI's provers
Merged
nmegill merged 6 commits into metamath:develop from spolu:openai-shorten on Mar 27, 2020
Conversation 12
Commits 6
Checks 0
Files changed 1
spolu commented on Mar 26, 2020 • edited
Contributor
This PR proposes various shorter versions of proofs discovered by automated reasoning models developed at OpenAI.

Formal

gpt-f



"I had a look at the proofs—very impressive results! Especially because we had a global minimization recently, and your method found much shorter proofs nevertheless."

"Any ML-based system is impressive if it can find many shorter proofs than the ones we already have. Nice work."

"The shorter proof is easier to translate. It's more symmetric in that it treats A and B identically. It's philosophically more concise in that it doesn't rely on the existence of a universal class of all sets."

Formal

Demo

If $R \subseteq S$ and $S \subseteq T$ then $R \subseteq T$

LEAN THEOREM PROVER

Microsoft Research

Application 1: mathematics

- Lean Mathlib
  - 1+ million lines of code
  - > 300 contributors
  - Algebra, Linear Algebra, Topology, Analysis, Probability, Geometry, Combinatorics, …

LEAN Community

https://leanprover-community.github.io/

See also Adam Topaz, Formal Mathematics and AI, 2023 NASEM AI+Math Workshop

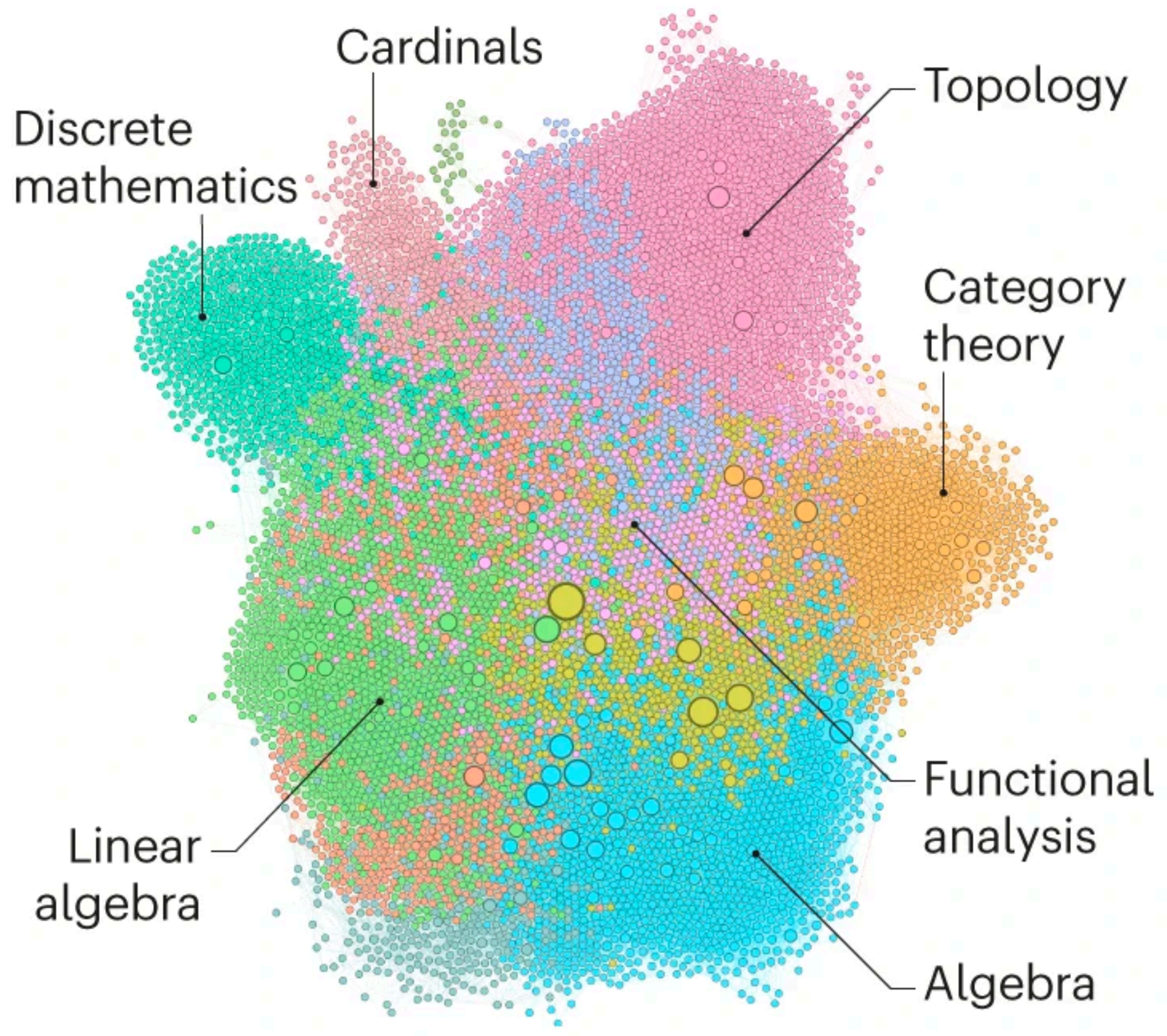
Application 1: mathematics

- Liquid tensor experiment

Mathematicians welcome computer-assisted proof in 'grand unification' theory

Proof-assistant software handles an abstract concept at the cutting edge of research, revealing a bigger role for software in mathematics.

Davide Castelvecchi



Application 1: mathematics

- Education
  - Courses at CMU, Imperial College London, Fordham, JHU, Université Paris-Saclay, …

[1] https://leanprover-community.github.io/courses.html

Machine learning potential

- Mathematician Adam Topaz:
  - “AI Collaborator” ... "It can be useful even if it’s not too smart” [1]
- Automate tedious proofs
- Retrieve definitions/theorems (e.g. h1.trans h2)

[1] Adam Topaz, Formal Mathematics and AI, 2023 NASEM AI+Math Workshop

Application 2: software verification

- Specification
    - E.g. “reverse(reverse(list)) == list”
- Proof: code satisfies specification

See: QED at Large: A Survey of Engineering of Formally Verified Software
Ringer, Palmskog, Sergey, Gligoric, Tatlock. Foundations and Trends in Programming Languages 5.

Application 2: software verification

- Specification
  - E.g. “reverse(reverse(list)) == list”
- Proof: code satisfies specification
- Safety-critical applications
- Certified compilers, low-level systems software

CompCert

seL4
Security. Performance. Proof.

› Defense Advanced Research Projects Agency › Our Research › Proof Engineering, Adaptation, R

Proof Engineering, Adaptation, Repair, and Learning for Software (PEARLS)

See: QED at Large: A Survey of Engineering of Formally Verified Software
Ringer, Palmskog, Sergey, Gligoric, Tatlock. Foundations and Trends in Programming Languages 5.

Machine learning potential

- Make proof assistants easier to use
- Boilerplate, tedious proofs [1]
- Proof re-use, repair, automation [1,2]

> In the end, out of a total of around 550 lemmas, approximately 400 were tedious "infrastructure" lemmas; only the remainder had direct relevance to the meta-theory

Rossberg, Russo, Dreyer, *F-ing Modules*, JFP 2015; from [1]

[1] Brigitte Pientka, Principles of Programming and Proof Languages, 2023 NASEM AI+Math Workshop
[2] Talia Ringer, Concrete Problems in Proof Automation, AITP 2022

ML 1: reasoning

- Tests aspects of reasoning

International Mathematical Olympiad
2020 Problem 2

Prove that $\forall a, b, c, d \in \mathbb{R}$ with $a \geq b \geq c \geq d > 0$ and $a + b + c + d = 1$,

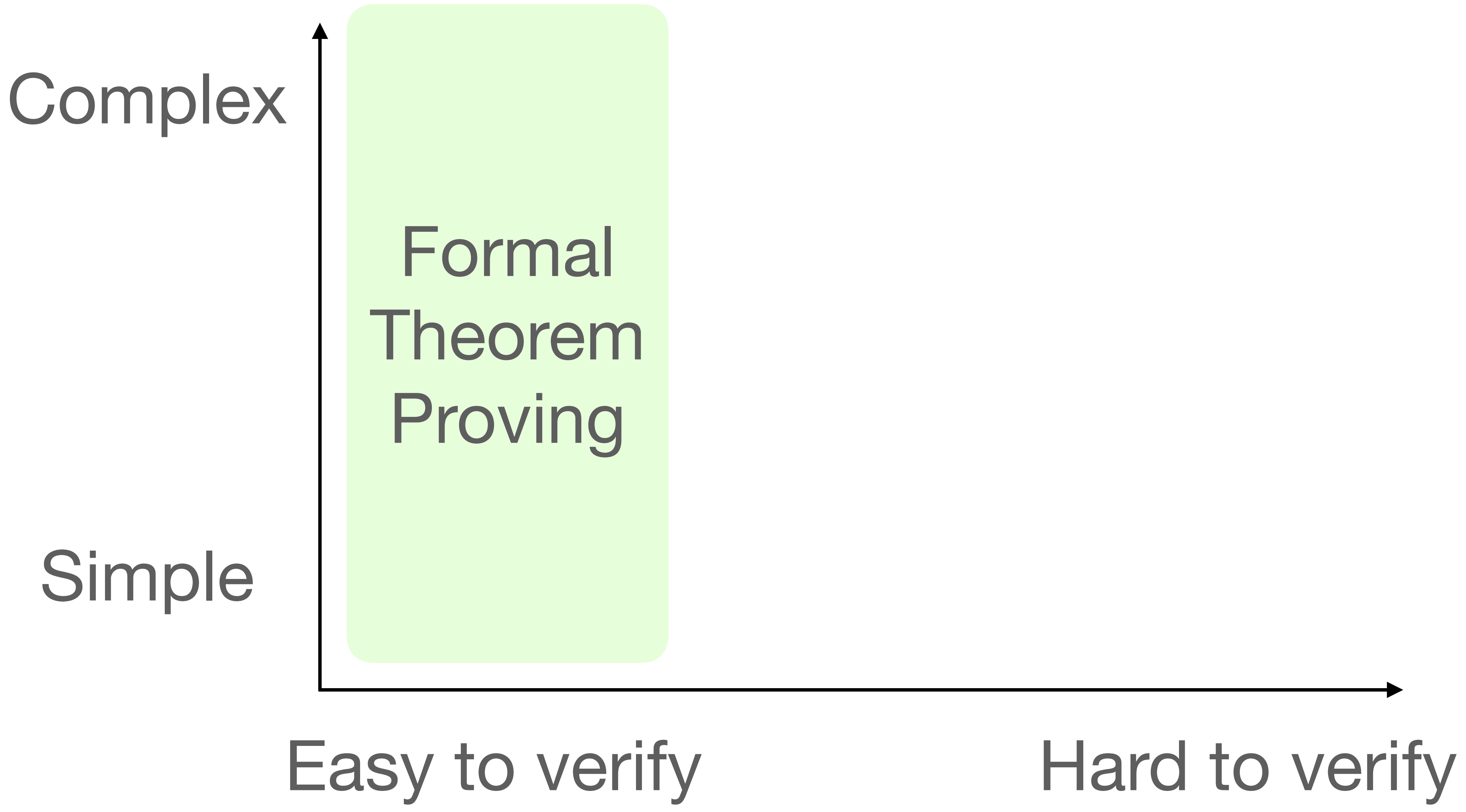
$$(a + 2b + 3c + 4d)a^a b^b c^c d^d < 1.$$

→

Planning
Pattern recognition
Background knowledge
Backtracking
Creativity
Computation
…

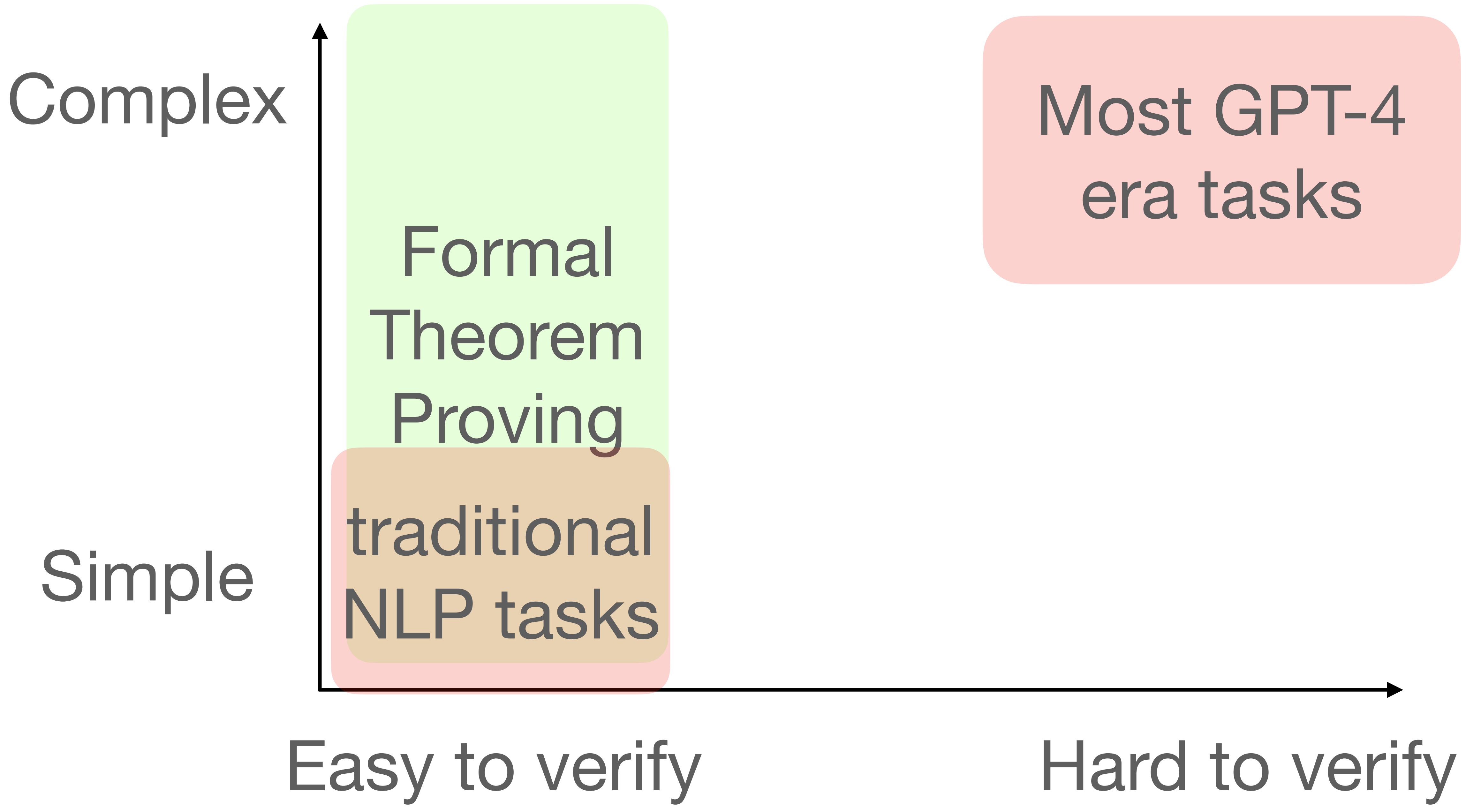
ML 1: reasoning

- Tests aspects of reasoning
- **Verifiable**, yet complex



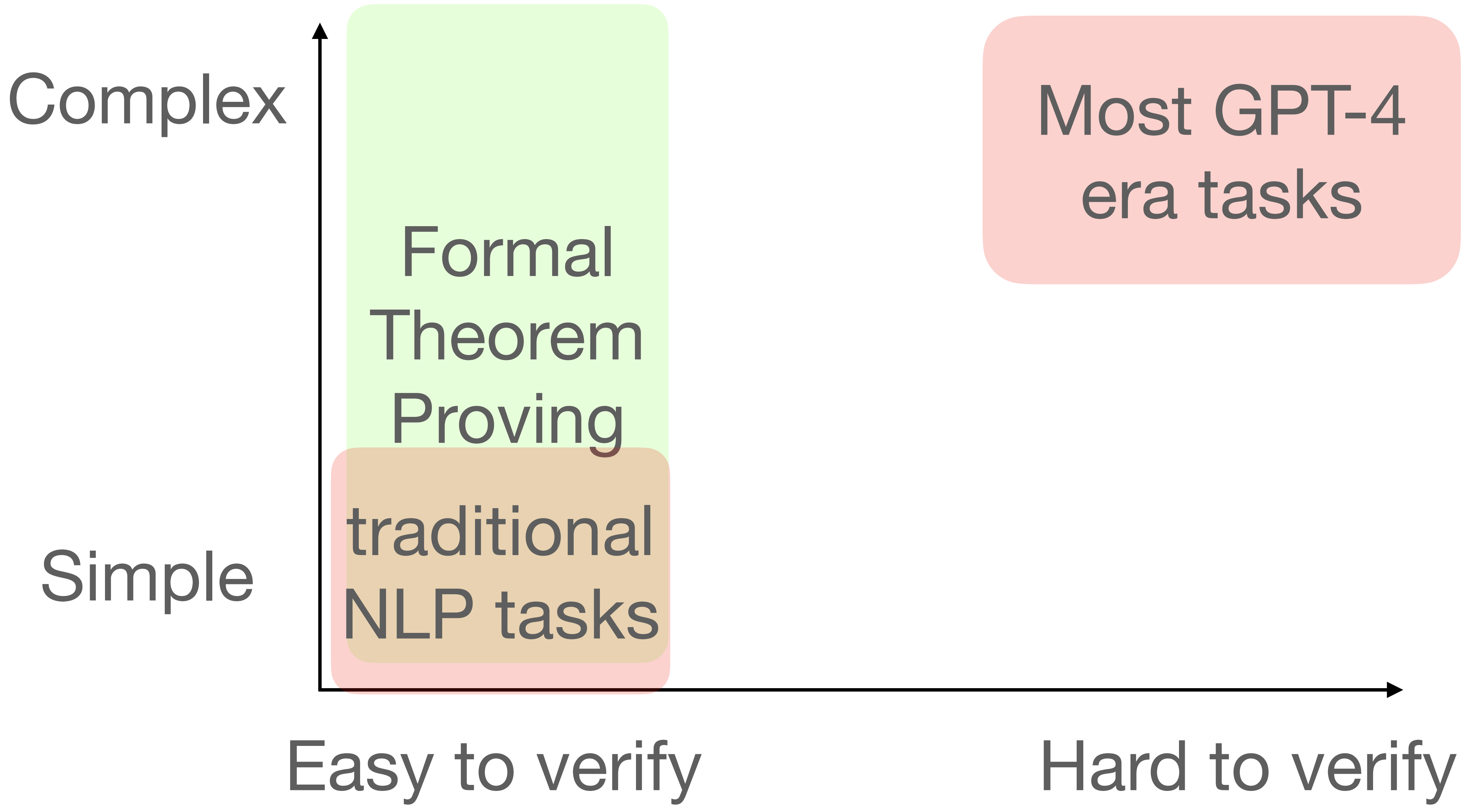
ML 1: reasoning

- Tests aspects of reasoning
- **Verifiable**, yet complex



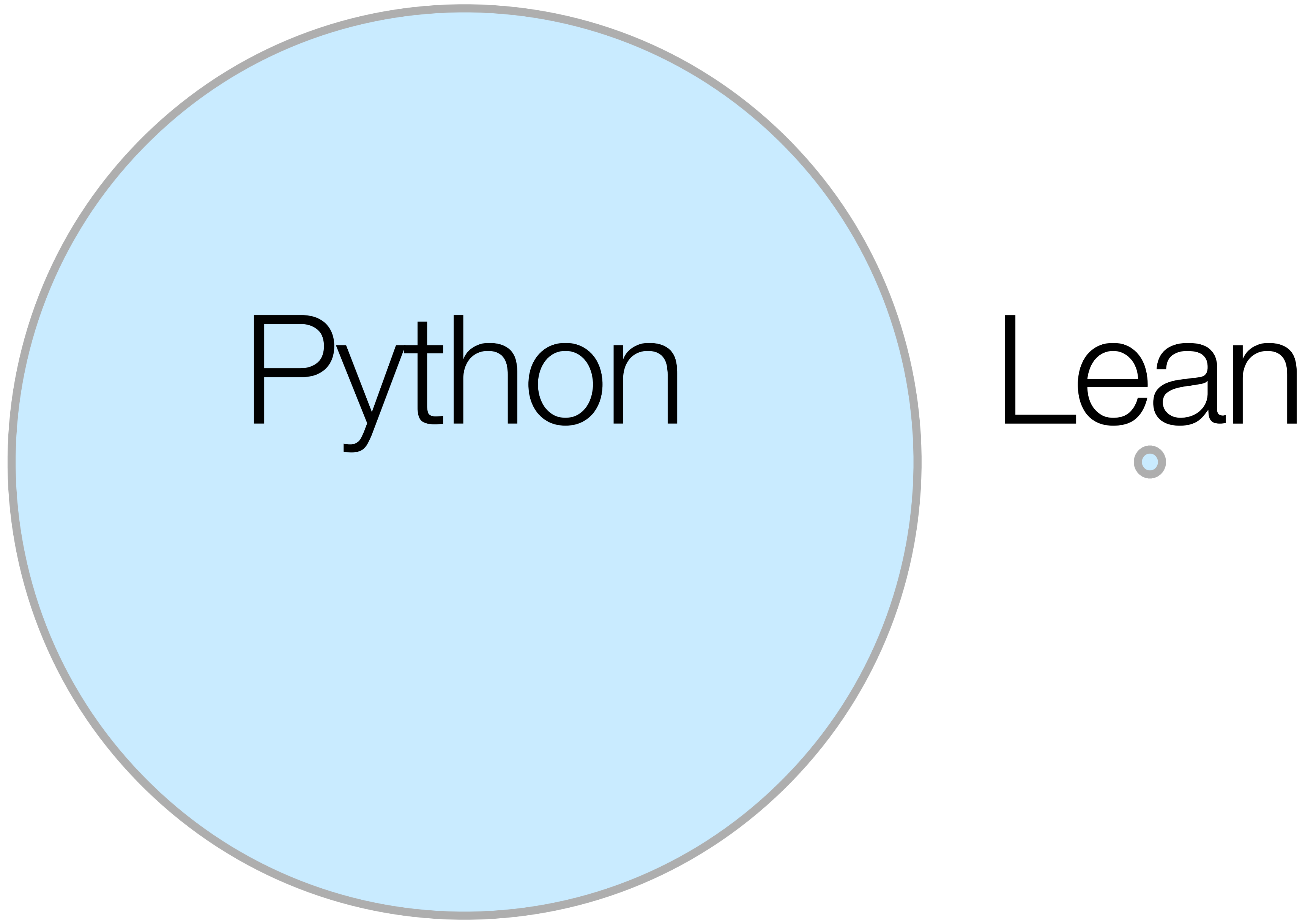
ML 1: reasoning

- Tests aspects of reasoning
- **Verifiable**, yet complex
- **Difficult** in GPT-4 era



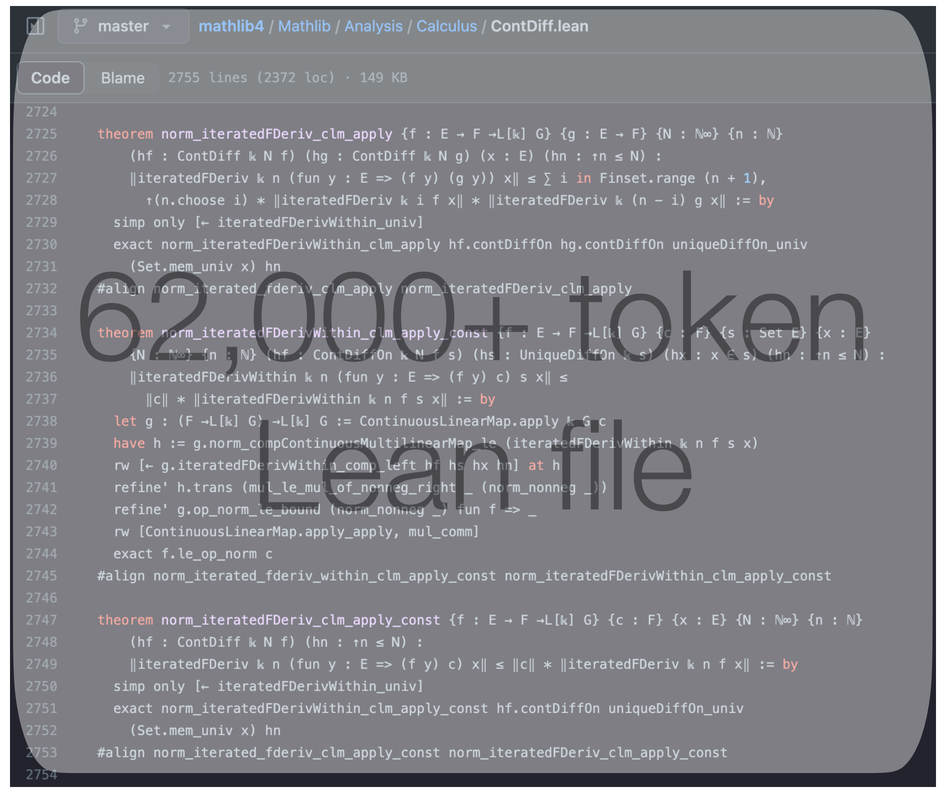
ML 2: code generation

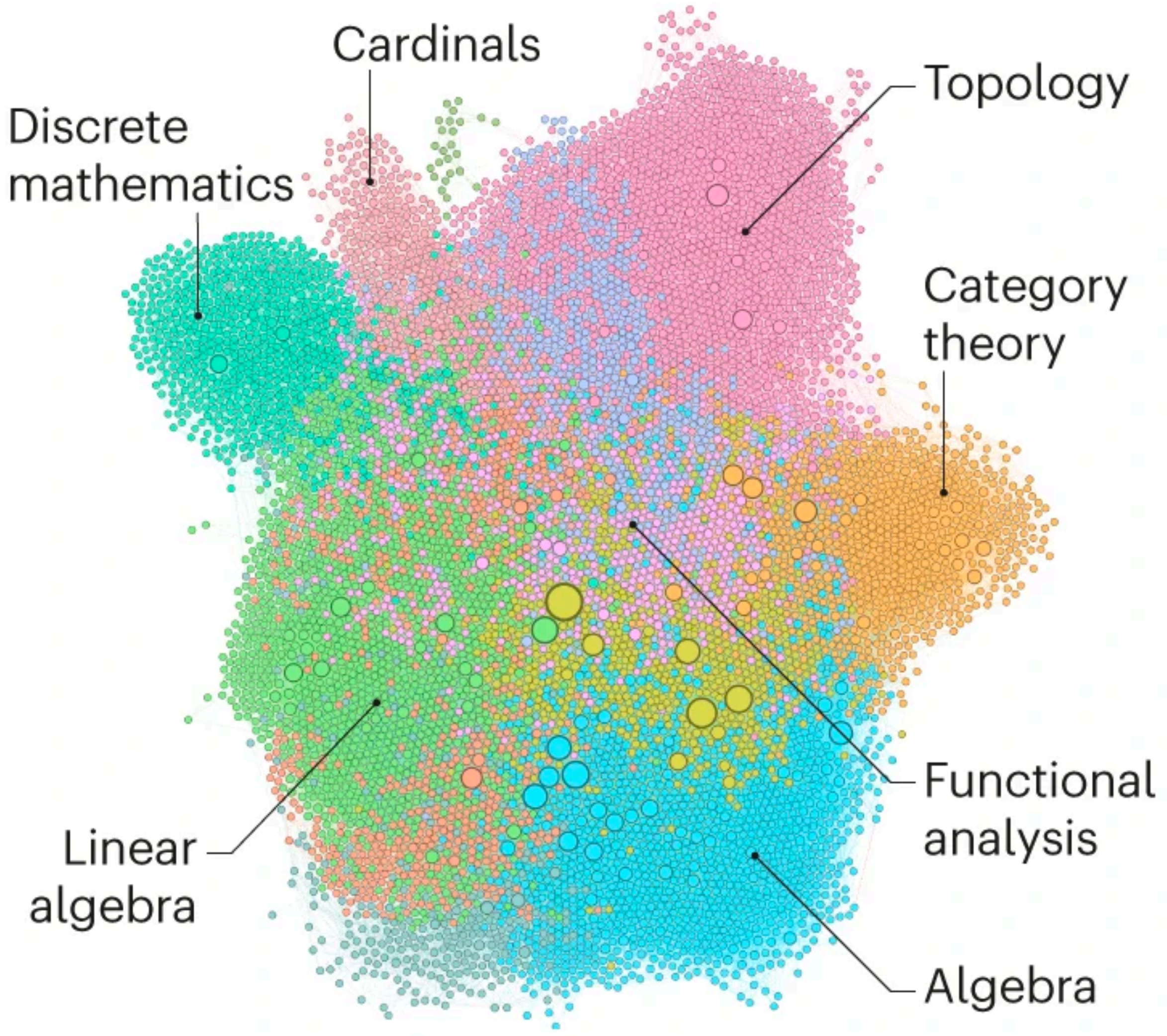
- Low-resource



ML 2: code generation

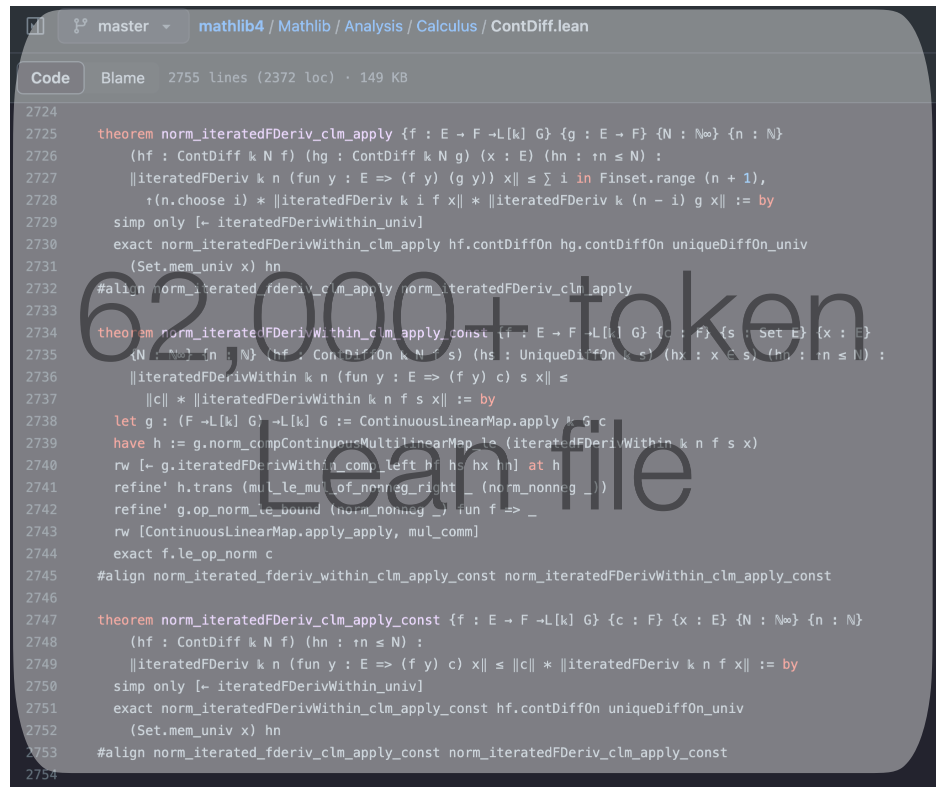
- Low-resource
- Long-range, context-dependent

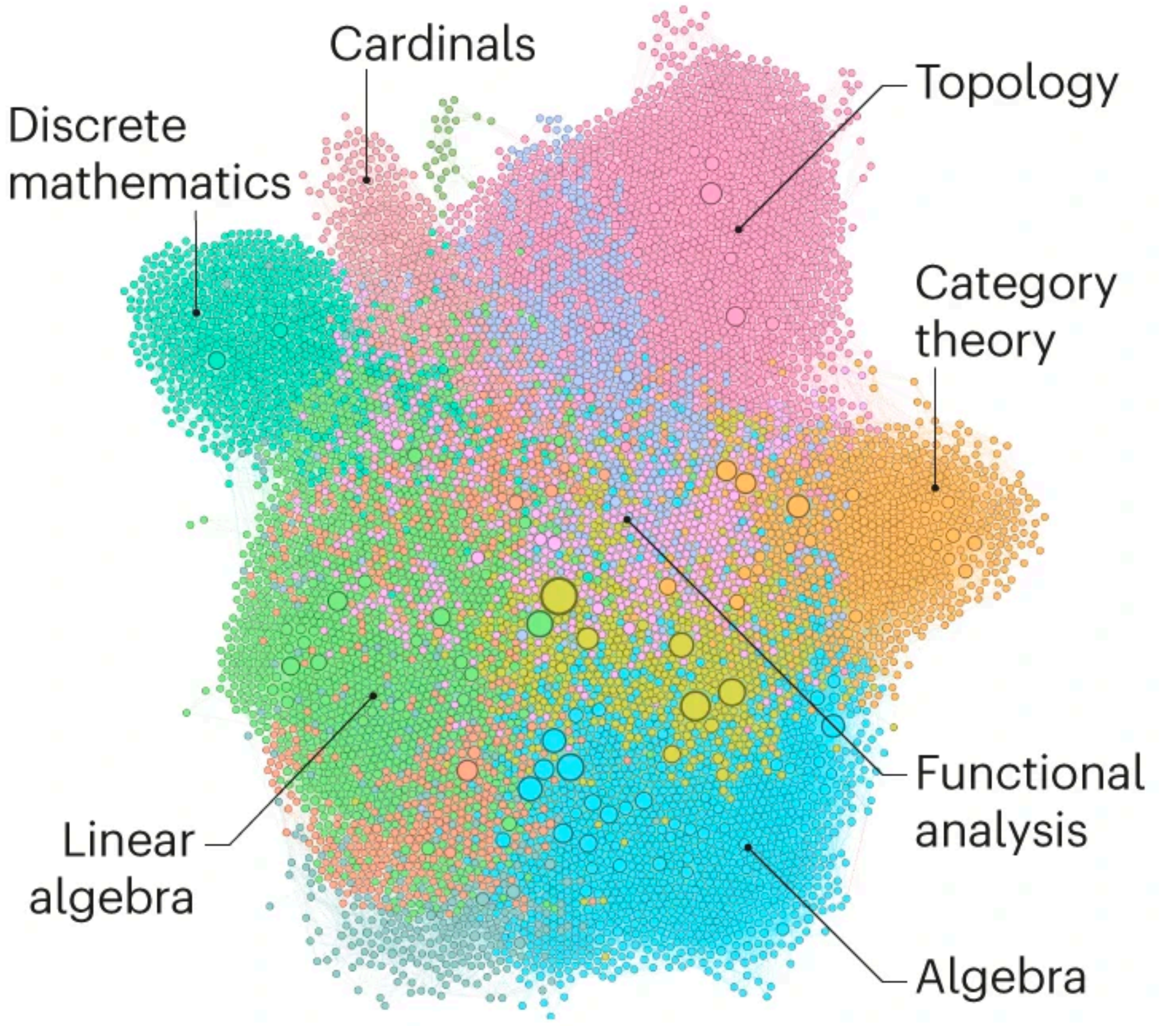




ML 2: code generation

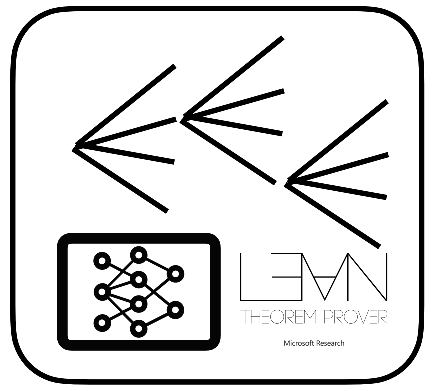
- Low-resource
- Long-range, context-dependent
- **Verifiable**



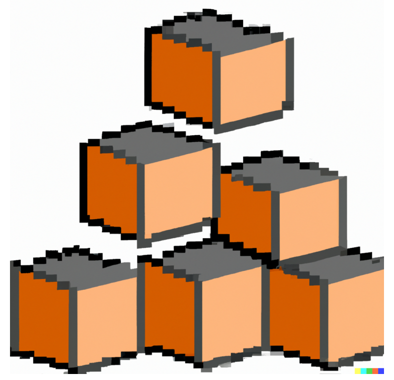


This tutorial

- Part I: next-step prediction
  - Language model suggests next-proof-steps
  - Tree search



- Part II: language cascades
  - Compose language model functions
  - Sketching, correction, tools



Tutorial code: https://github.com/wellecks/ntptutorial

Next-step prediction

- Build a “co-pilot” from scratch

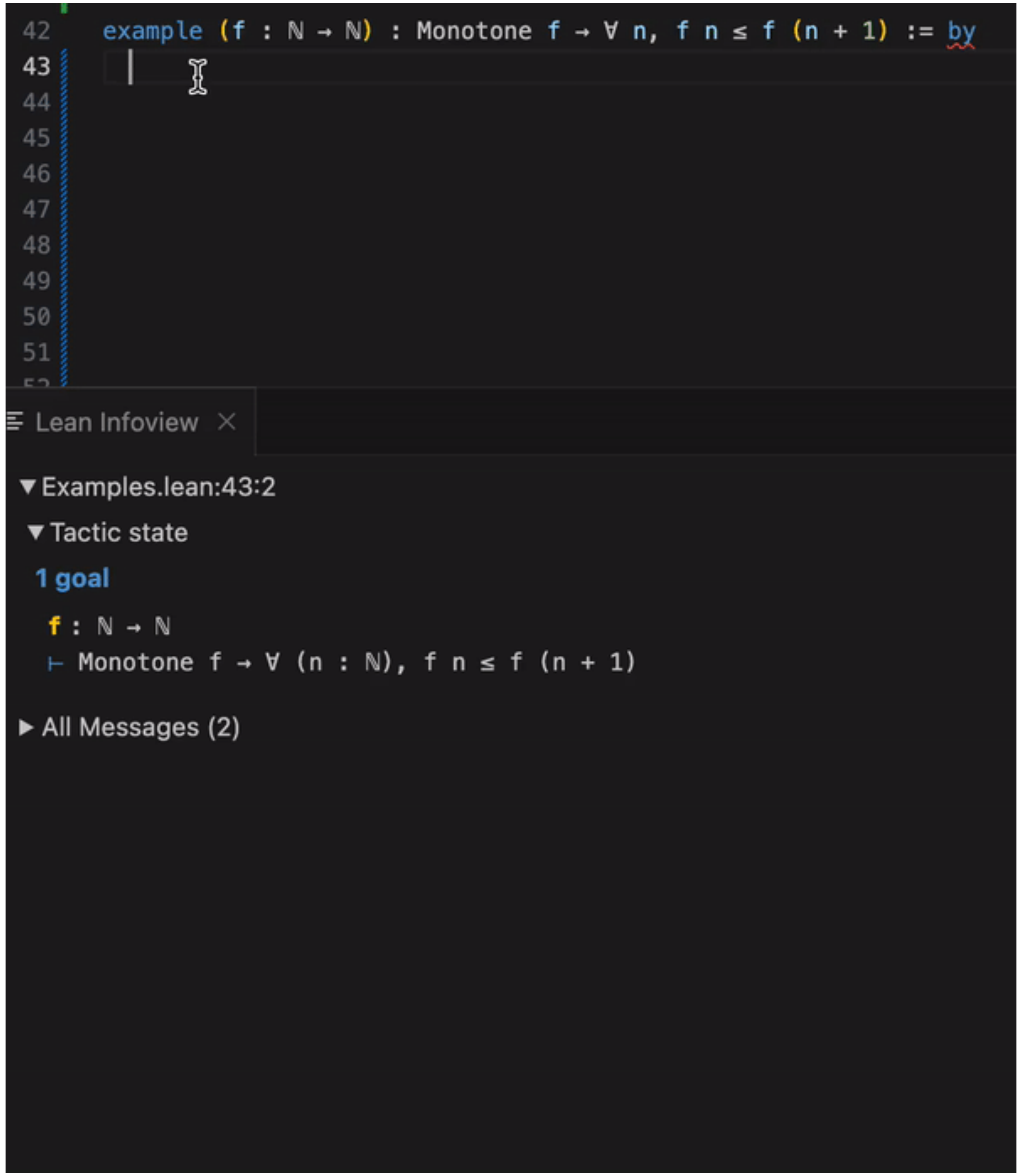
```
example : ∀ (a: ℤ), a + 3 = 0 → a = -3 := by
  intro a ha
  llmstep ""
```

Proof state

```
▼llmstep suggestions
    Try this:
  • 🎉 linarith
  • rw [← sub_eq_zero] at ha
  • apply eq_neg_of_add_eq_zero_left
  • rw [← Int.negSucc_coe] at ha
```

Next-step suggestions

Demo


example (f : ℕ → ℕ) : Monotone f → ∀ n, f n ≤ f (n + 1) := by
Lean Infoview
Examples.lean:43:2
Tactic state
1 goal
f : ℕ → ℕ
⊢ Monotone f → ∀ (n : ℕ), f n ≤ f (n + 1)
All Messages (2)

Topic	Notebook
0. Intro	notebook
1. Data	notebook
2. Learning	notebook
3. Proof Search	notebook
4. Evaluation	notebook
5. `llmsuggest`	notebook

Tutorial code: https://github.com/wellecks/ntptutorial

0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$

0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$
    - x_t : proof state

```
example : R ⊆ S → S ⊆ T → R ⊆ T := by
```

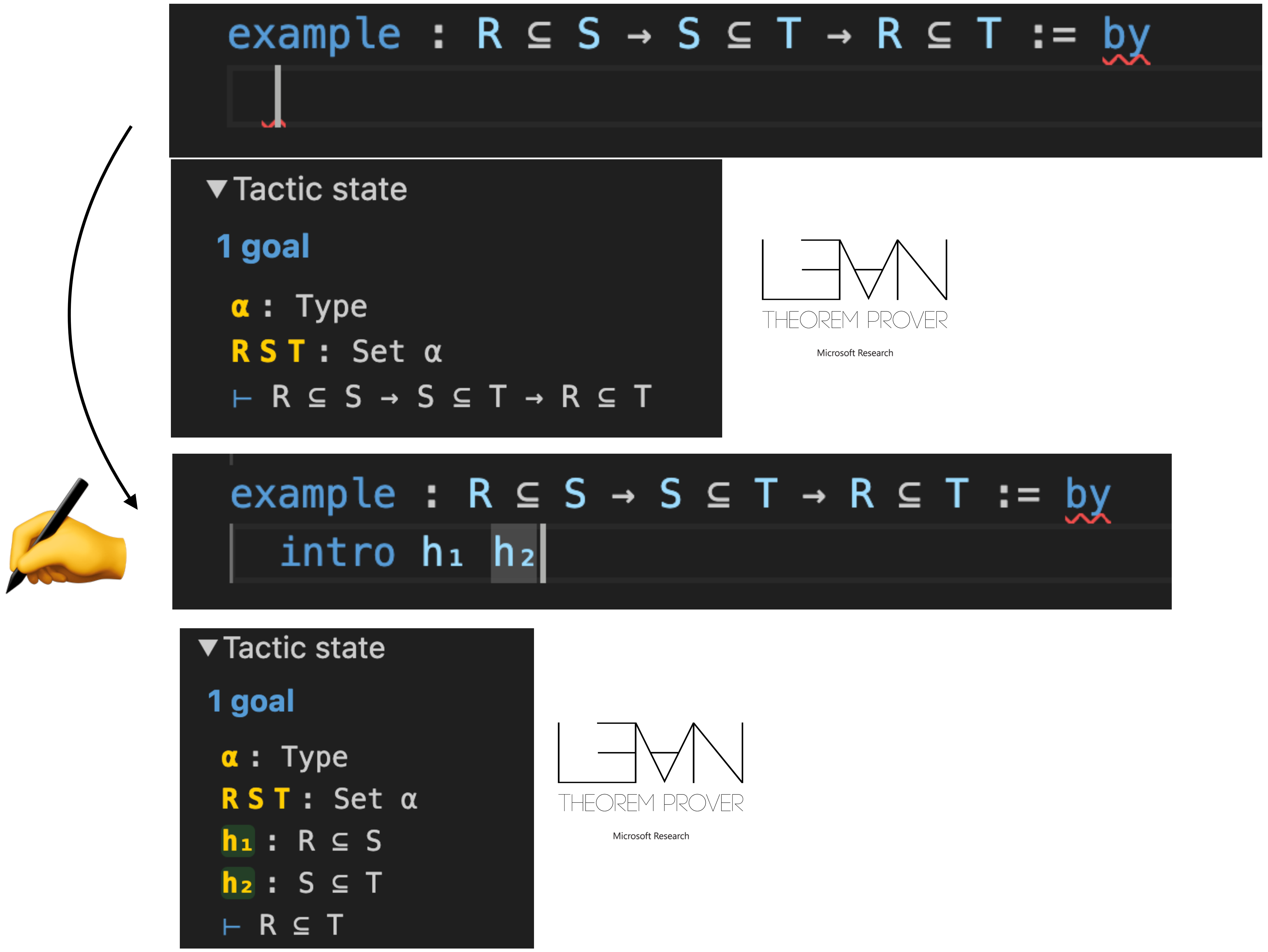
```
▼Tactic state
1 goal
α : Type
R S T : Set α
⊢ R ⊆ S → S ⊆ T → R ⊆ T
```

0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$
    - x_t : proof state
    - y_t : proof step

0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$
    - x_t : proof state
    - y_t : proof step ✍️

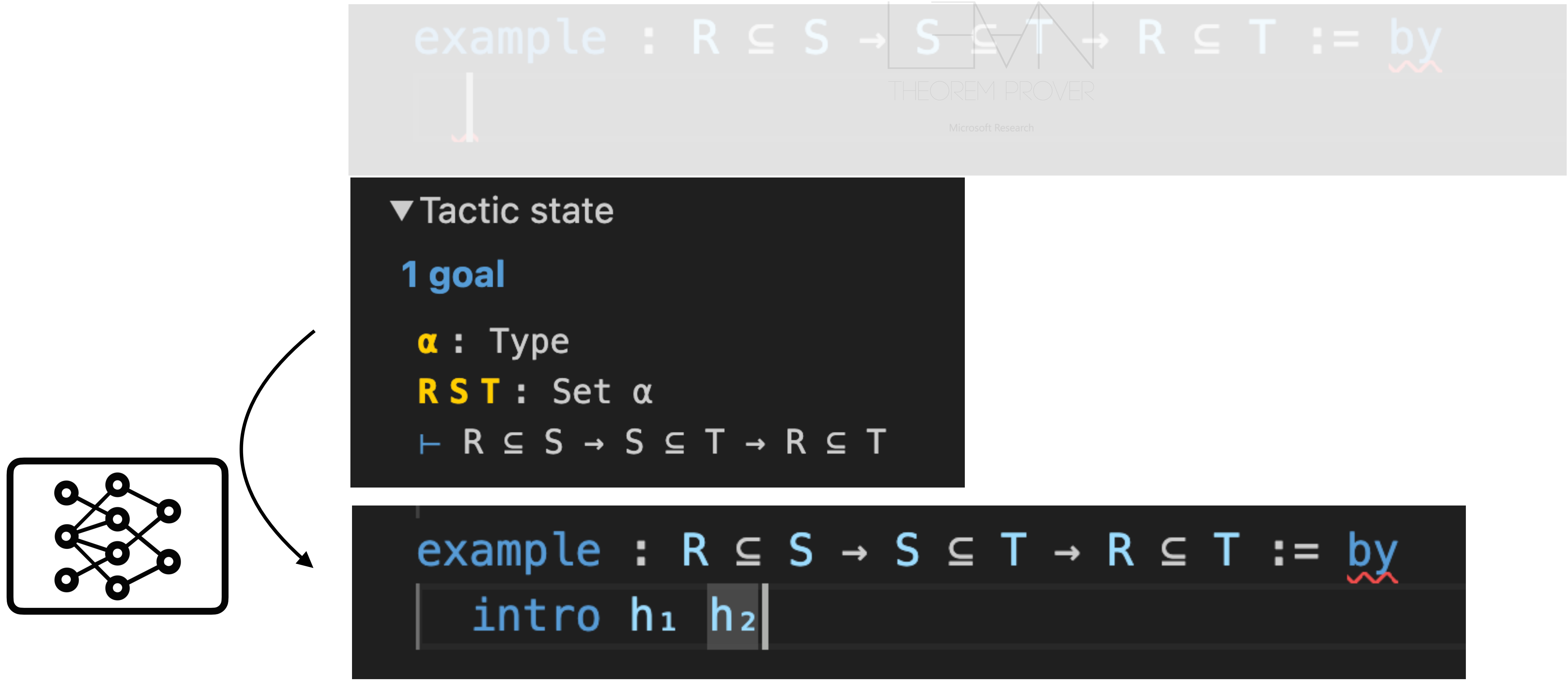


0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$
    - x_t : proof state
    - y_t : proof step
    - x_T : "proof complete"

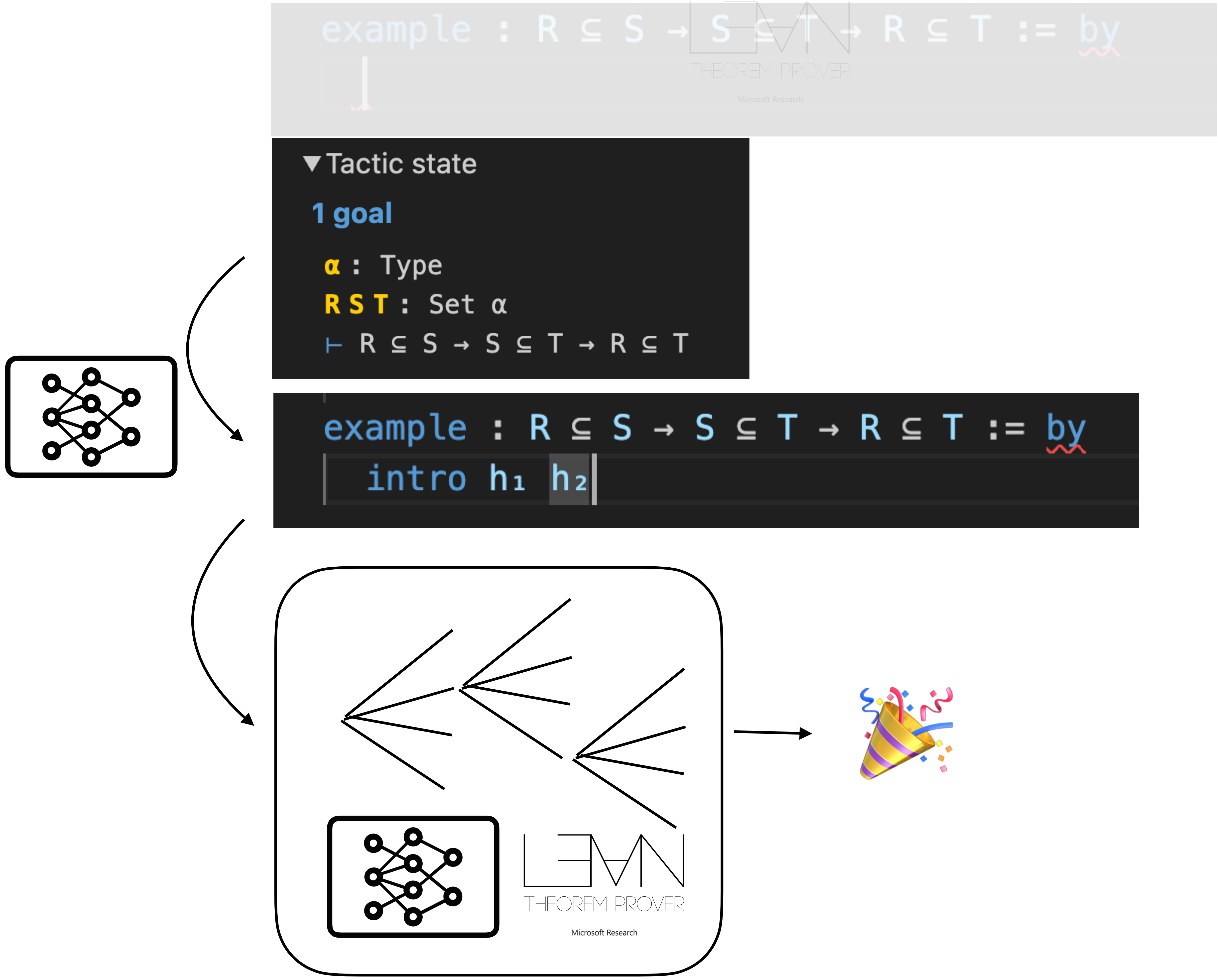
0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$
- Language model:
  - $p_\theta(y_t \mid x_t)$



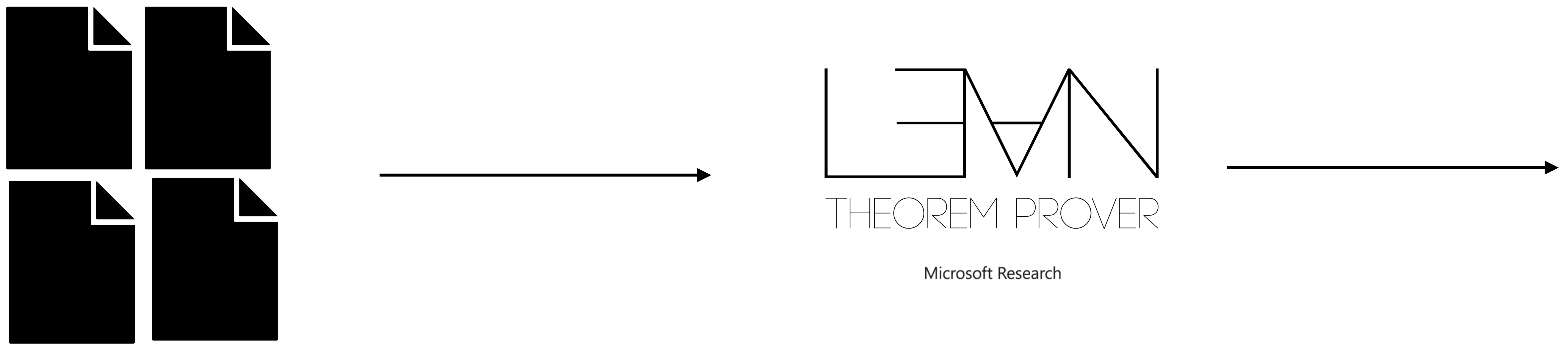
0. Problem setup

- Proof: sequence of (state, step)
  - $(x_0, y_0), \ldots, (x_t, y_t), \ldots (x_T, y_T)$
- Language model:
  - $p_\theta(y_t \mid x_t)$
- Tree search to generate full proof



1. Data

- Extract (proofstate, next-step) pairs from human-written proofs



$$D = \{(x_t, y_t)\}$$

Data

Elaboration → Postprocess

```
example : R ⊆ S → S ⊆ T → R ⊆ T := by
  intro h1 h2
  exact h1.trans h2
```


```
{'args': [{'node': {'args': [...],
                    'info': 'none',
                    'kind': 'Lean.Parser.Command.declModifiers'}},
          {'node': {'args': [...],
                    'info': 'none',
                    'kind': 'Lean.Parser.Command.theorem'}}],
 'info': 'none',
 'kind': 'Lean.Parser.Command.declaration'}
```


$$D = \{(x_t, y_t)\}$$

```
--- x1 ---
m n : ℕ
h : Nat.coprime m n
⊢ Nat.gcd m n = 1
--- y1 ---
rw [Nat.coprime] at h

--- x2 ---
m n : ℕ
h : Nat.gcd m n = 1
⊢ Nat.gcd m n = 1
--- y2 ---
exact h
```

[LeanDojo: Theorem Proving with Retrieval-Augmented Language Models](#)
Yang, Swope, Gu, Chalamala, Song, Yu, Godil, Prenger, Anandkumar, arXiv 2023

Data

- Lean Dojo [Yang et al 2023]
  - repository $\rightarrow \{(x_t, y_t)\}$
  - Dependencies, multiple files, versioning

```
URL = "https://github.com/leanprover-community/mathlib4"
COMMIT = "5a919533f110b7d76410134a237ee374f24eaaad"
repo = LeanGitRepo(URL, COMMIT)
traced_repo = trace(repo)
```

LeanDojo: Theorem Proving with Retrieval-Augmented Language Models
Yang, Swope, Gu, Chalamala, Song, Yu, Godil, Prenger, Anandkumar, arXiv 2023

Data

- Mathlib:
  - 41,944 theorems + proofs
- => training data

```
train   169530
val     4053
test    3606
```

Data

- Mathlib:
  - 41,944 theorems + proofs
- => training data

```
train    169530
val      4053
test     3606
```


```
Number of non-empty training proofs:  41944
{'commit': '5a919533f110b7d76410134a237ee374f24eaaad',
 'end': [308, 76],
 'file_path': 'Mathlib/Analysis/BoxIntegral/Box/Basic.lean',
 'full_name': 'BoxIntegral.Box.withBotCoe_inj',
 'start': [307, 1],
 'traced_tactics': [{'state_after': 'no goals',
                     'state_before': 'ι : Type u_1\n'
                                     'I✝ J✝ : Box ι\n'
                                     'x y : ι → ℝ\n'
                                     'I J : WithBot (Box ι)\n'
                                     '⊢ ↑I = ↑J ↔ I = J',
                     'tactic': 'simp only [Subset.antisymm_iff, ← '
                               'le_antisymm_iff, withBotCoe_subset_iff]'}],
 'url': 'https://github.com/leanprover-community/mathlib4'}
```

2. Learning

- Standard supervised fine-tuning on $D = \{(x_t, y_t)\}$:
  - $$\max_\theta \sum_{(x_t, y_t) \in D} -\log p_\theta(y_t \mid x_t)$$

2. Learning

x_t

```
Input:
[GOAL]ι : Type u_1
I† J† : Box ι
x y : ι → ℝ
I J : WithBot (Box ι)
⊢ ↑I = ↑J ↔ I = J[PROOFSTEP]
```

y_t

```
Output:
simp only [Subset.antisymm_iff, ← le_antisymm_iff, withBotCoe_subset_iff]<|endoftext|>
```

Proof Artifact Co-training for Theorem Proving with Language Models
Han, Rute, Wu, Ayers, Polu. ICLR 2022

2. Learning

https://huggingface.co/wellecks/llmstep-mathlib4-pythia2.8b

```
import transformers

MODEL = 'wellecks/llmstep-mathlib4-pythia2.8b'
model = transformers.GPTNeoXForCausalLM.from_pretrained(MODEL)
tokenizer = transformers.GPTNeoXTokenizerFast.from_pretrained(MODEL)
```

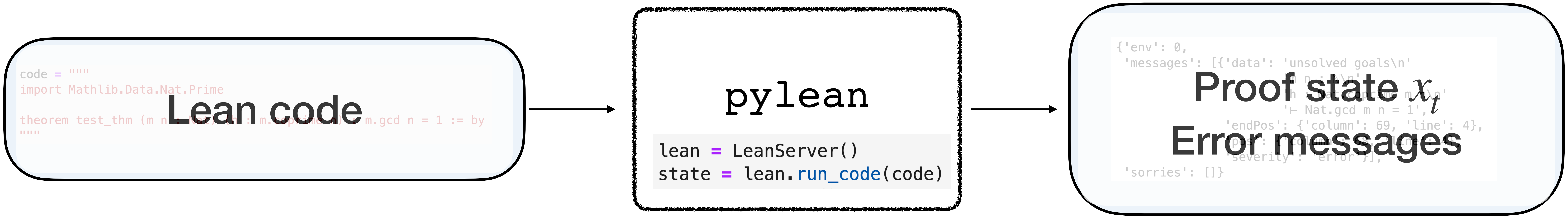
```
prompt = """[GOAL]m n : ℕ
  h : Nat.coprime m n
  ⊢ Nat.gcd m n = 1[PROOFSTEP]"""

input_ids = tokenizer.encode(prompt, return_tensors='pt')
out = model.generate(input_ids)
text = tokenizer.decode(out[0][input_ids.shape[1]:], skip_special_tokens=True)
print(text)
```

```
rw [← h.gcd_eq_one]
```

3. Proof search

- Use next-step predictor $p_\theta(y_t \mid x_t)$ to generate a full proof $y_1, \ldots, y_T$
- We need:
  - **Interaction** with Lean
  - **Algorithm** for search

Interaction | pylean



Zhangir Azerbayev, *pylean.*
https://github.com/zhangir-azerbayev/repl

Interaction | pylean

[4]:
```
# Generate a next step
prompt = f"[GOAL]{get_goal(state)}[PROOFSTEP]"

next_step = generate(prompt)
print(next_step)
```

```
rw [← h.gcd_eq_one]
```

Finally, we can give the generated next step to Lean and receive the next state.

[5]:
```
code = """
import Mathlib.Data.Nat.Prime

theorem test_thm (m n : Nat) (h : m.coprime n) : m.gcd n = 1 := by

""" + next_step

lean = LeanServer()
state = lean.run_code(code)
lean.proc.close()

pprint(state)
```

```
{'env': 0, 'messages': [], 'sorries': []}
```

Zhangir Azerbayev, *pylean.*
https://github.com/zhangir-azerbayev/repl

Interaction | Lean Dojo

- Lean Dojo
  - Traces entire Lean repository
  - Provides abstractions for interaction

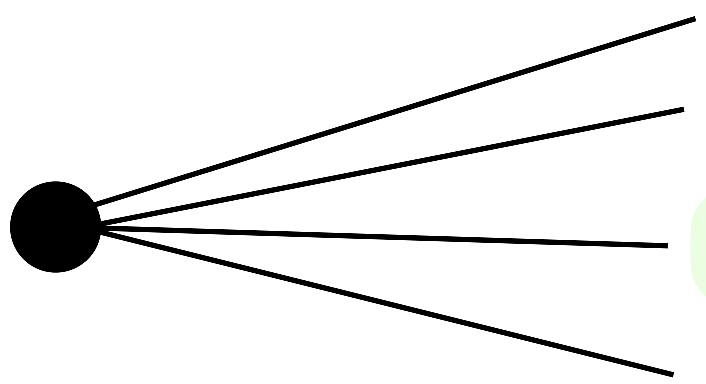
LeanDojo: Theorem Proving with Retrieval-Augmented Language Models
Yang, Swope, Gu, Chalamala, Song, Yu, Godil, Prenger, Anandkumar, arXiv 2023

Best-first search

```
type-checked candidates:
    (-0.066) rintro rfl
    (-0.307) rintro ⟨rfl, rfl⟩
    (-0.035) intro h
    (-0.230) rintro ⟨d, rfl⟩
```

Polu & Sutskever 2020, Han et al. ICLR 2022, Yang et al 2023, …

Best-first search



```
type-checked candidates:
  (-0.066) rintro rfl
  (-0.307) rintro ⟨rfl, rfl⟩
  (-0.035) intro h
  (-0.230) rintro ⟨d, rfl⟩
```

$$\frac{1}{Z}\sum_t \log p_\theta(y_t \mid x_t)$$

"value function"

[Polu & Sutskever 2020, Han et al. ICLR 2022, Yang et al 2023, …]

Best-first search

```
type-checked candidates:
 (-0.066) rintro rfl
 (-0.307) rintro ⟨rfl, rfl⟩
 (-0.035) intro h
 (-0.230) rintro ⟨d, rfl⟩
```

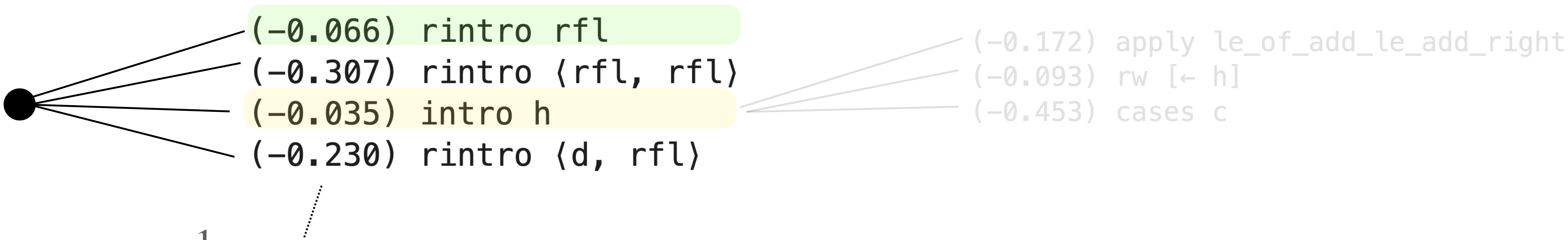
```
(-0.172) apply le_of_add_le_add_right
(-0.093) rw [← h]
(-0.453) cases c
```

$$\frac{1}{Z}\sum_{t}\log p_{\theta}(y_t \mid x_t)$$

"value function"

Polu & Sutskever 2020, Han et al. ICLR 2022, Yang et al 2023, …

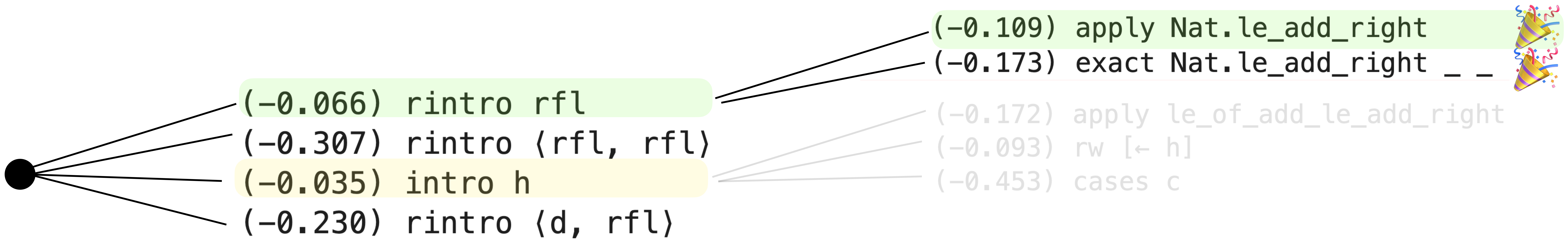
Best-first search



$$\frac{1}{Z}\sum_t \log p_\theta(y_t \mid x_t)$$

Polu & Sutskever 2020, Han et al. ICLR 2022, Yang et al 2023, …

Best-first search

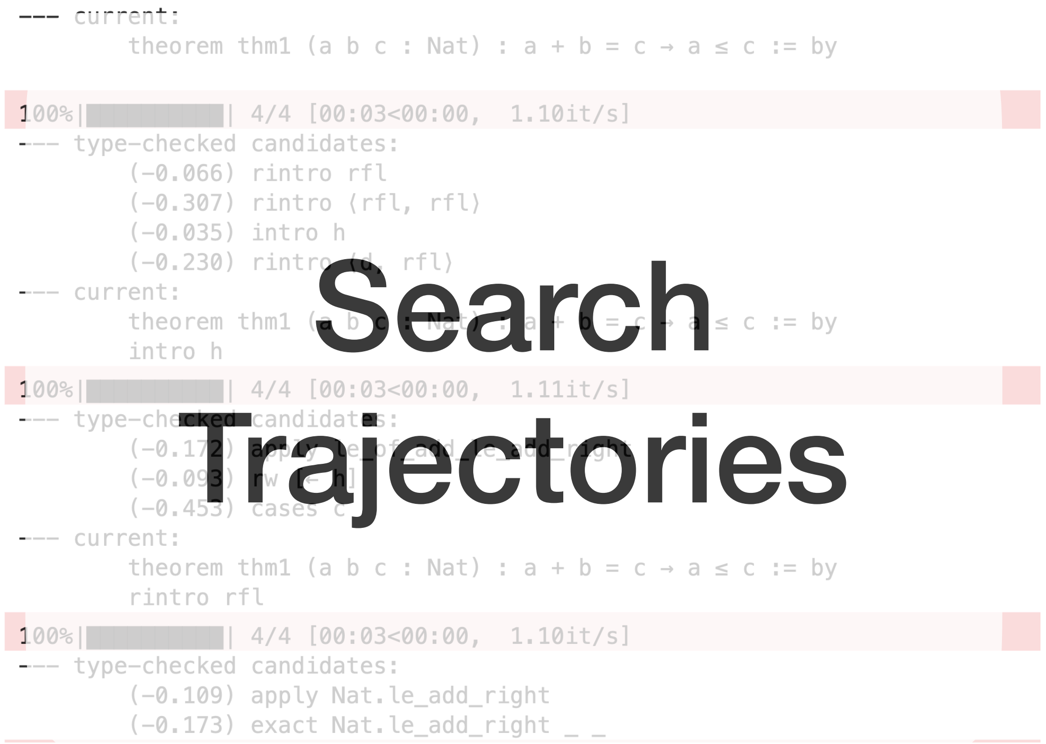


$$\frac{1}{Z}\sum_t \log p_\theta(y_t \mid x_t)$$

Polu & Sutskever 2020, Han et al. ICLR 2022, Yang et al 2023, …

Best-first search

```
proofsearch.best_first_search(
    model, tokenizer, header, theorem_statement,
    max_iters=32,
    num_samples=4,
    temperatures=[0.0],
    verbose=True
)
```



Generated Proof

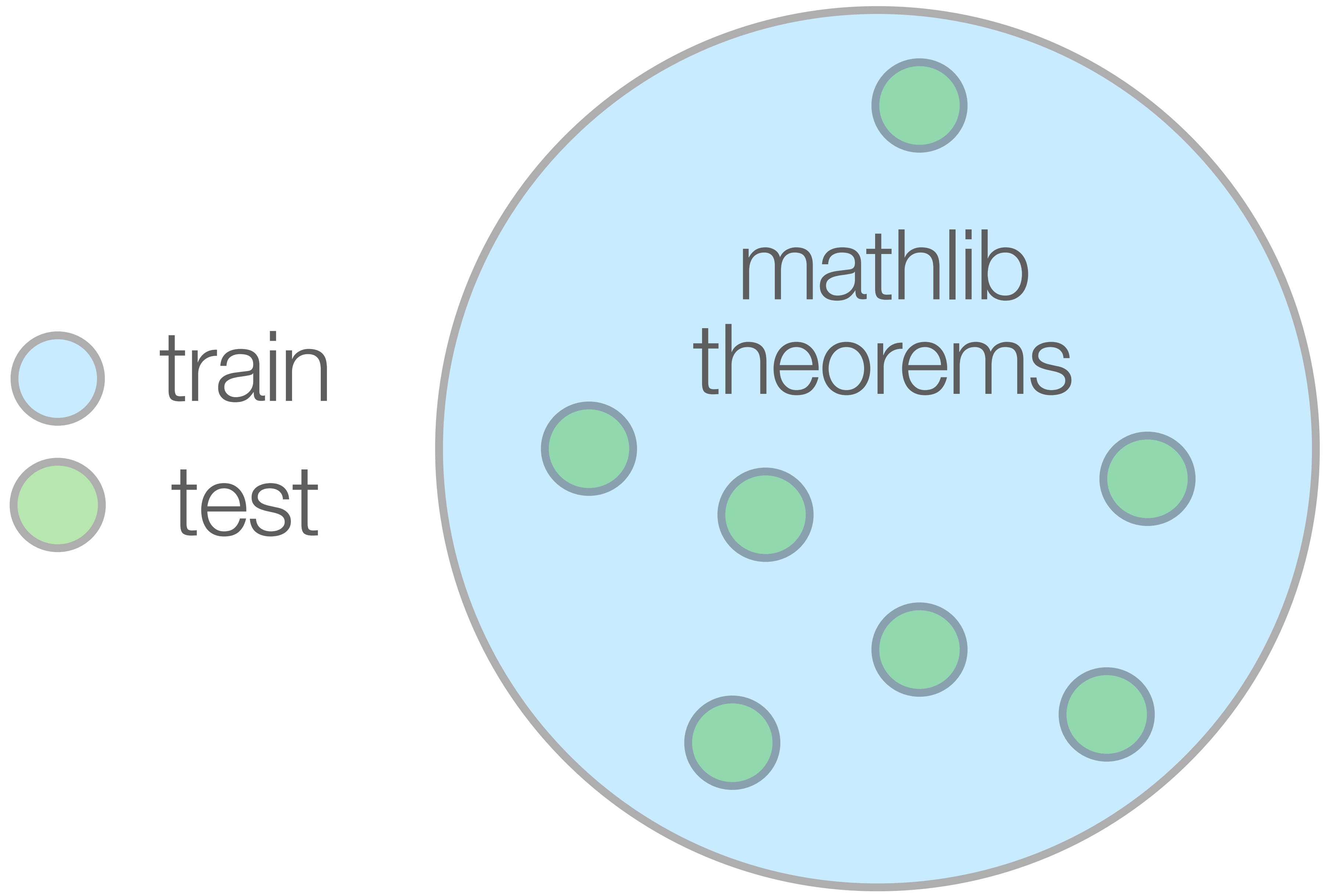
```
SUCCESS!
theorem thm1 (a b c : Nat) : a + b = c → a ≤ c := by
  rintro rfl
  apply Nat.le_add_right
```

3. Proof Search | notebook

https://github.com/wellecks/ntptutorial

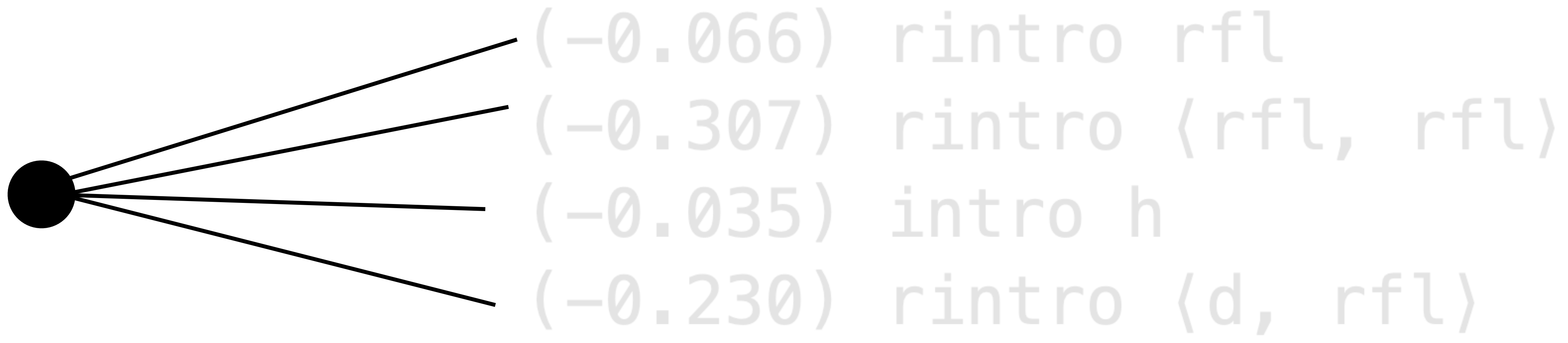
4. Evaluation | in-domain

- Held-out theorems from training distribution



Performance depends on search

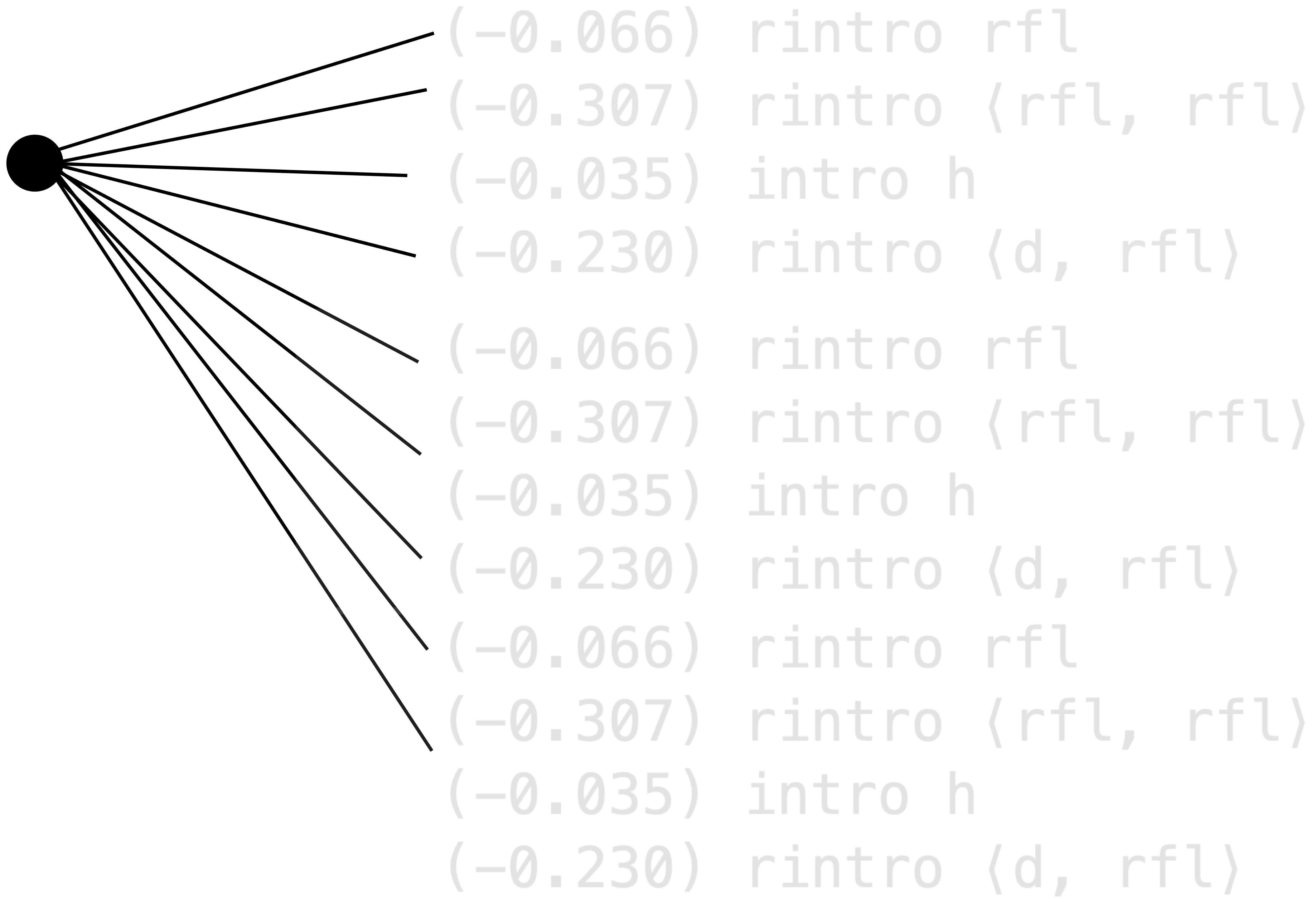
- pass rate $= f(p_\theta, \text{search}, \text{budget})$



- Total compute
- Total time

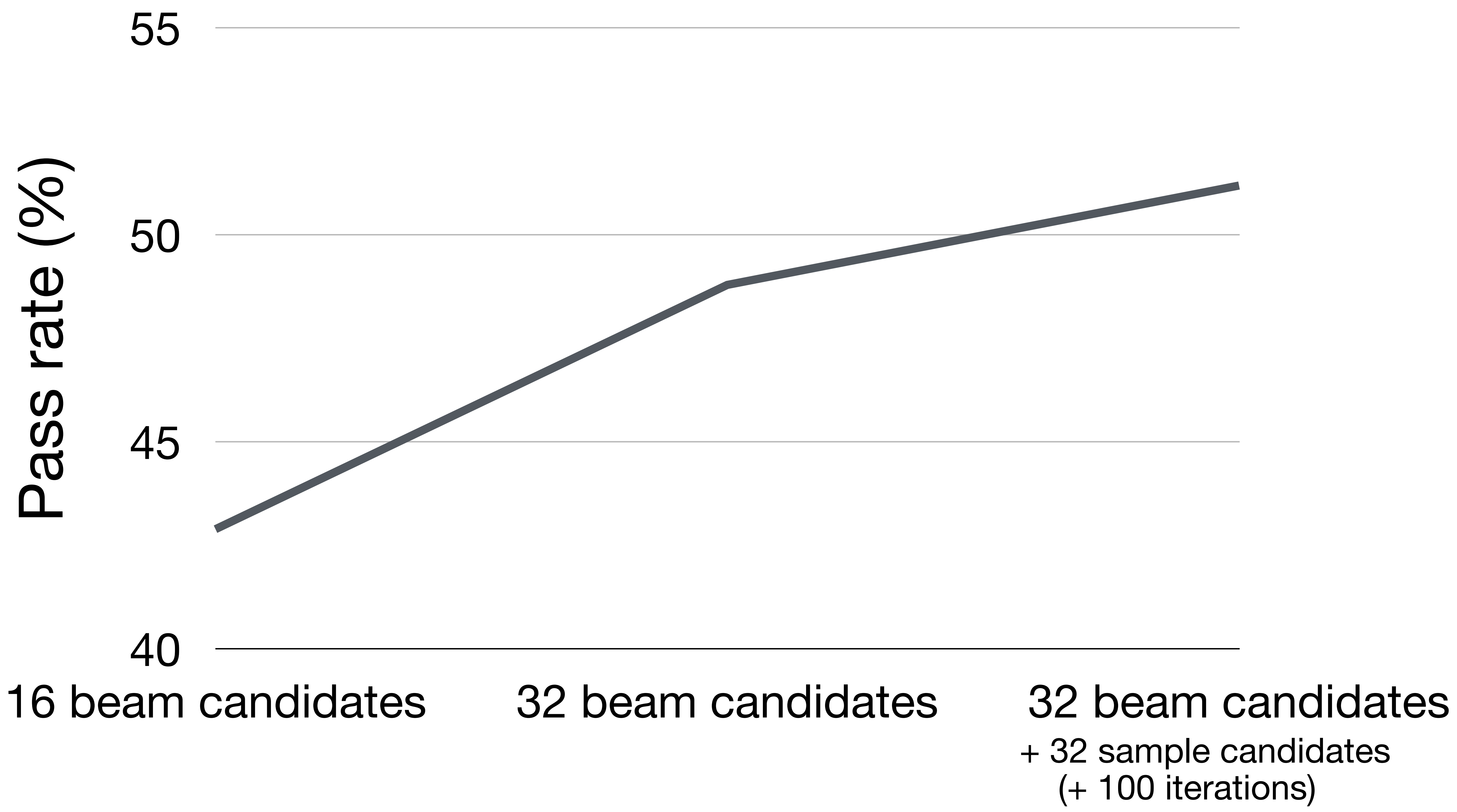
Performance depends on search

- pass rate $= f(p_\theta, \text{search}, \text{budget})$



- Total compute
- Total time

- mathlib4 LeanDojo validation set



10 minute timeout
100 iteration limit per attempt

code: proofsearch_dojo.py

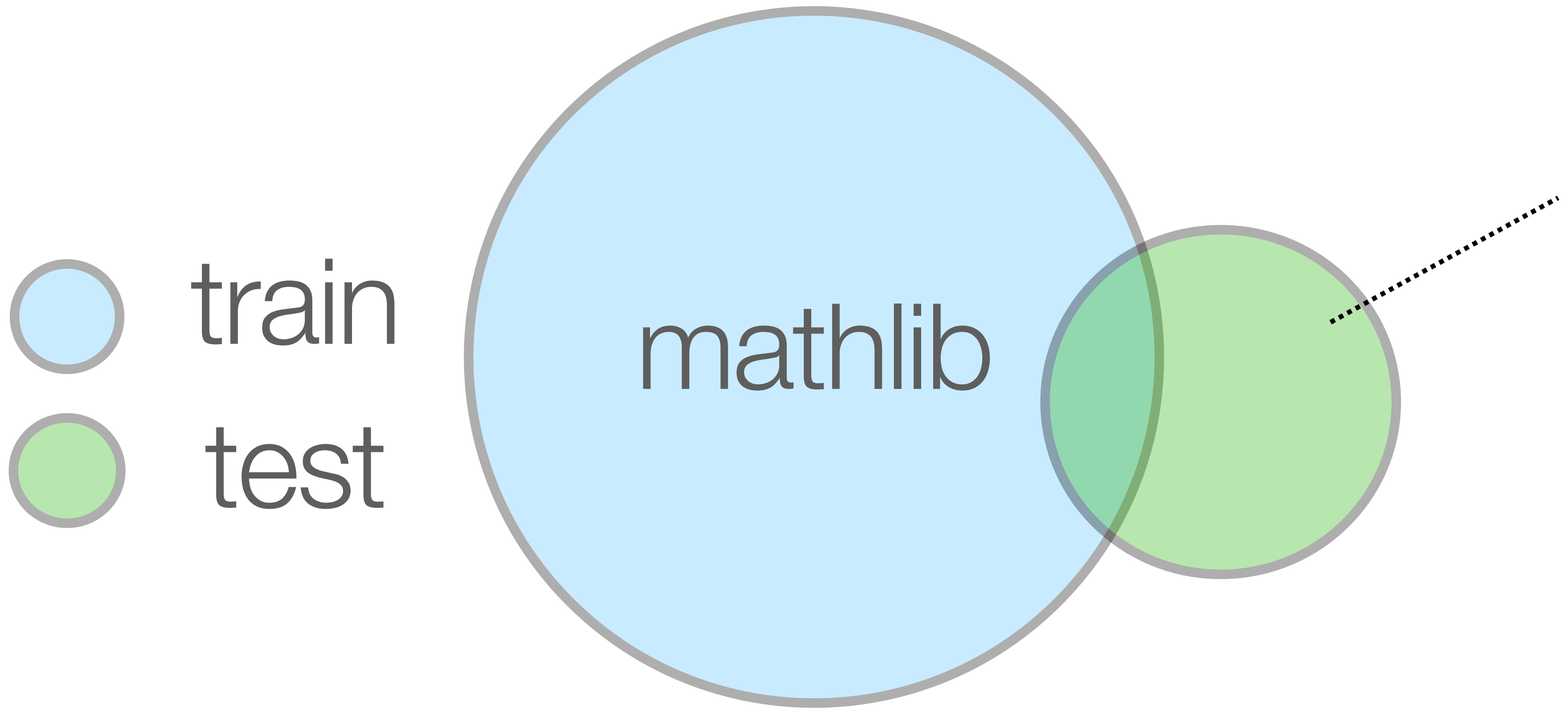
- recent results (Mathlib 3):

Method	`random`	`novel_premises`
`tidy`	23.8	5.4
GPT-4	28.8	7.5
ReProver (ours)	**51.4**	**26.2**
w/o retrieval	47.5	22.9

[LeanDojo: Theorem Proving with Retrieval-Augmented Language Models]
Yang, Swope, Gu, Chalamala, Song, Yu, Godil, Prenger, Anandkumar, arXiv 2023

Evaluation | out-of-domain

- **MiniF2F**: 480 competition problems



Problem **1959 IMO Problems/Problem 1**

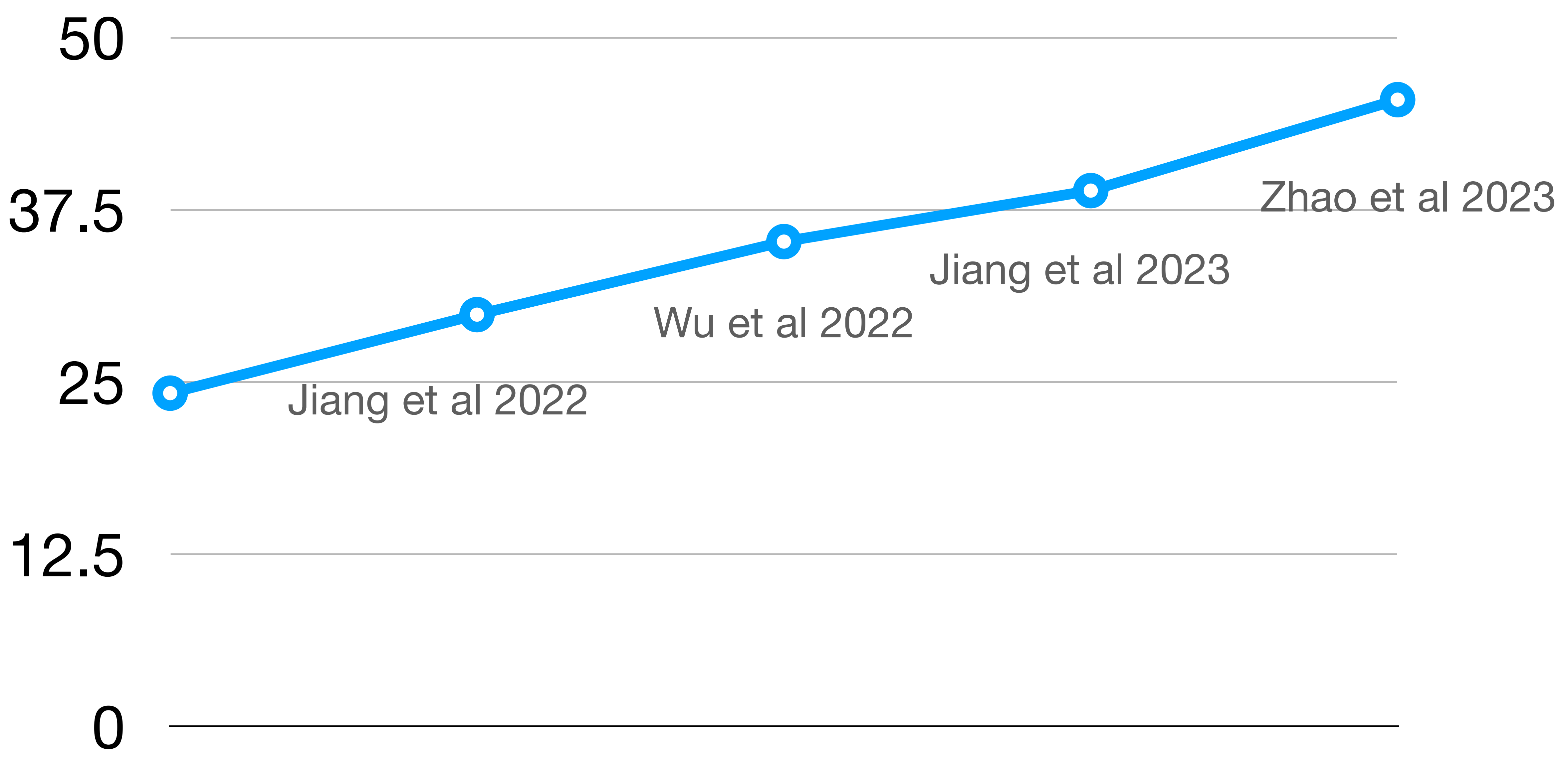
Prove that the fraction $\frac{21n+4}{14n+3}$ is irreducible for every natural number n.

```
theorem imo_1959_p1
  (n : ℕ)
  (h₀ : 0 < n) :
  nat.gcd (21*n + 4) (14*n + 3) = 1 :=
begin
```

MiniF2F: a cross-system benchmark for formal Olympiad-level mathematics
Zheng, Han, Polu ICLR 2022

Evaluation | out-of-domain

- **MiniF2F**: 480 competition problems
  - Benchmark still challenging (e.g. olympiad problems)
  - Lean [Yang et al 2023]: 26.5%

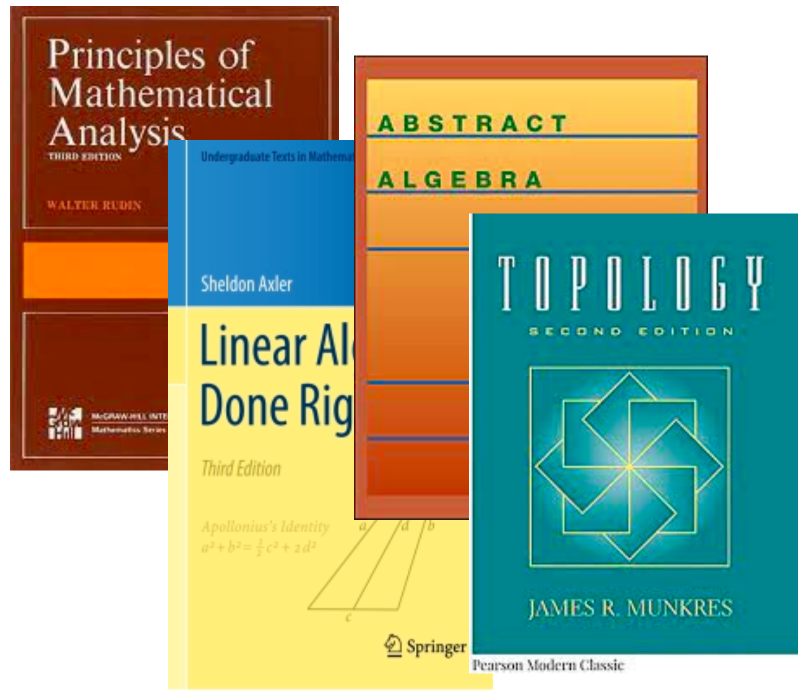


Timeline of MiniF2F performance
(Isabelle proof assistant)

MiniF2F: a cross-system benchmark for formal Olympiad-level mathematics
Zheng, Han, Polu ICLR 2022

Evaluation | out-of-domain

- **ProofNet**: undergraduate textbooks
  - + informal statements/proofs



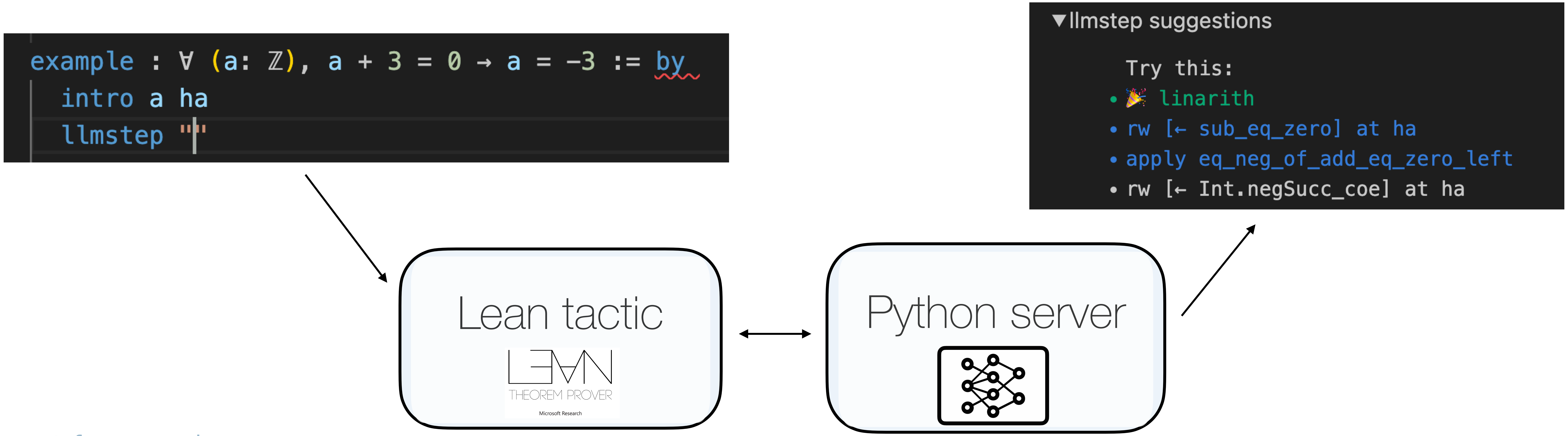

```
theorem exercise_4_5_14 {G : Type*}
  [group G] [fintype G]
  (hG : card G = 312) :
  ∃ (p : ℕ) (P : sylow p G), P.normal
```

ProofNet: Autoformalizing and Formally Proving Undergraduate Mathematics.
Azerbayev, Piotrowski, Schoelkopf, Ayers, Radev, Avigad, arXiv 2023

5. interactive tool | llmstep

https://github.com/wellecks/llmstep

- [L]LM proof step suggestions
- Runs on your own device



Inspired by *gpt-f* suggestions
[Han et al 2022]

5. interactive tool | llmstep

https://github.com/wellecks/llmstep

- [L]LM proof step suggestions
- Runs on your own device
- Simplified version:

5. `llmsuggest`	notebook

https://github.com/wellecks/ntptutorial

Summary

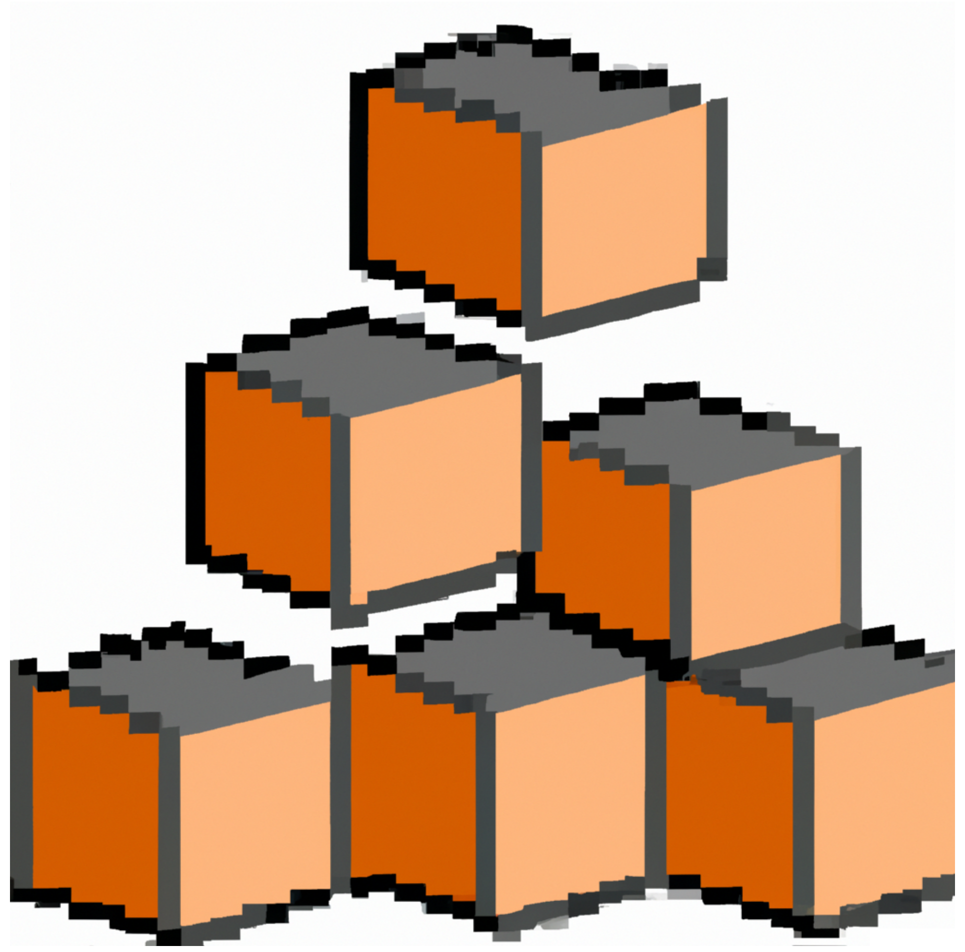
- Next-step suggestion $p_\theta(y_t \mid x_t)$
- Data, learning, search, “human-machine collaboration”

Part II: language cascades

Statement

If gcd$(n, 4) = 1$ and lcm$(n, 4) = 28$, show that n is 7.

→



→

Verified formal proof

```
have c1: "1*28 = n*4"
using assms
    by (smt (z3) prod_gcd_lcm_nat)
then have c2: "n = 1*28/4"
    by auto
then show ?thesis
    by auto
```

Part II: language cascades

Topic	Notebook
1. Language model cascades	notebook
2. Draft, Sketch, Prove	notebook

Tutorial code: https://github.com/wellecks/ntptutorial

Language model cascade

Prompted language model implements a function

$$y \sim p_\theta(y \mid x; P)$$

```
python_code = gpt4(
    prompt,
    """There are 800+72 apples in a barrel.
    How many apples in 2233 barrels?"""
)
```

→

```
(800+72)*2233
```

Language Model Cascades.
Dohan et al 2022

Language model cascade

Chain together functions to form a "cascade"

$$y \sim p_\theta(y \mid x; P)$$

$$z = f(y)$$

```
python_code = gpt4(
    prompt,
    """There are 800+72 apples in a barrel.
    How many apples in 2233 barrels?"""
)
```

→ `(800+72)*2233` → `1947176`

Language Model Cascades.
Dohan et al 2022

Draft, Sketch, Prove

Given informal theorem x_I

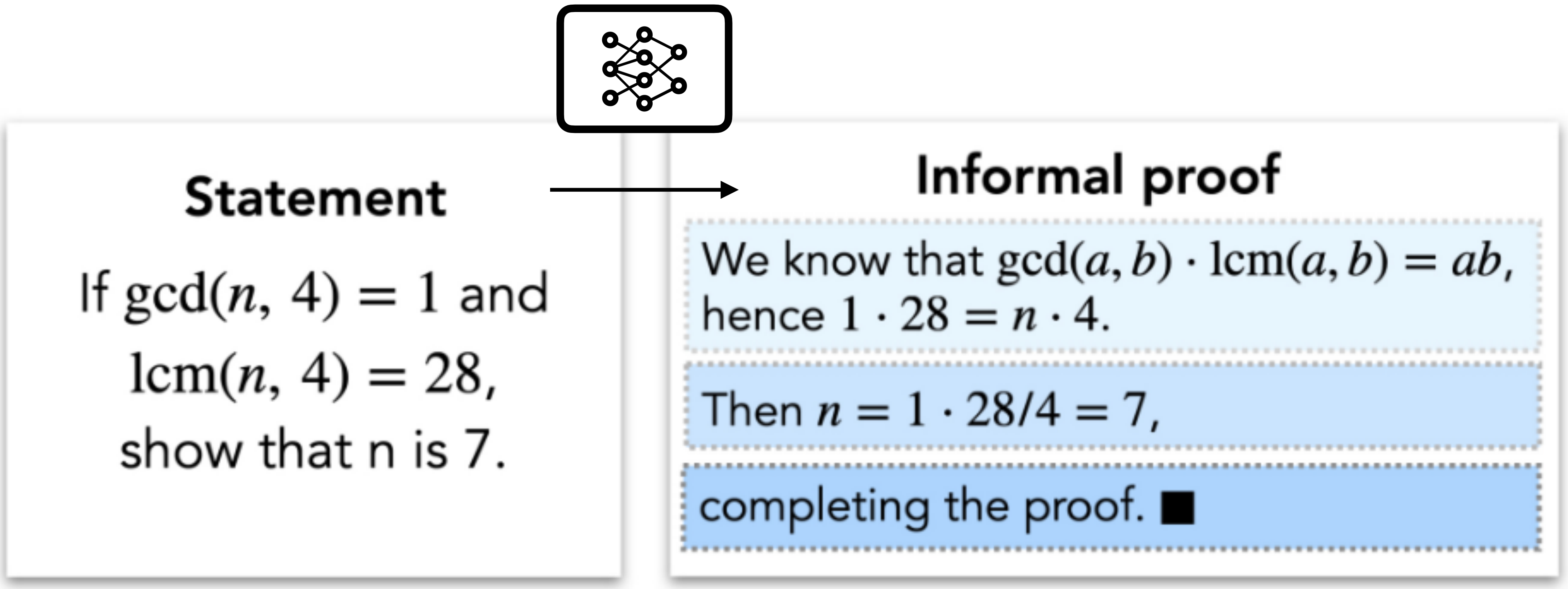
formal theorem x_F

[Draft, Sketch, Prove: Guiding Formal Theorem Provers with Informal Proofs](#)
Jiang, Welleck, Zhou, Lacroix, Liu, Li, Jamnik, Lample, Wu. ICLR 2023

Draft, Sketch, Prove

Given informal theorem x_I
formal theorem x_F

1. Draft $y_I \sim p(\cdot \mid x_I)$

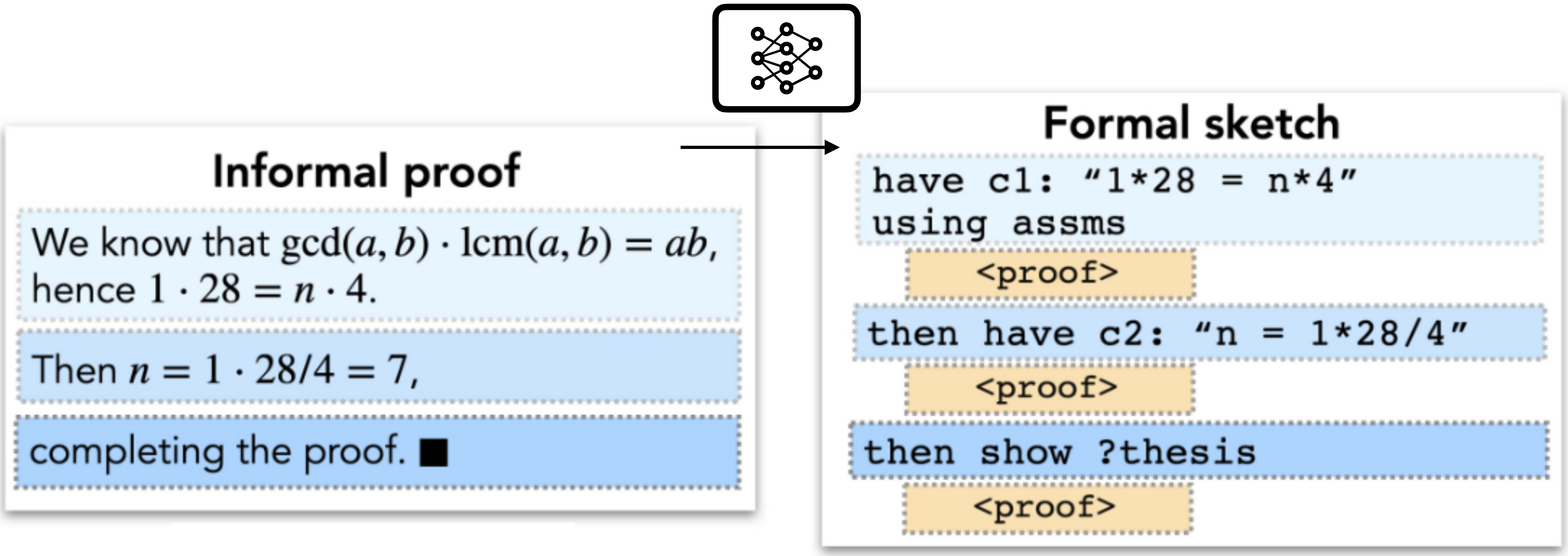


[Draft, Sketch, Prove: Guiding Formal Theorem Provers with Informal Proofs](#)
Jiang, Welleck, Zhou, Lacroix, Liu, Li, Jamnik, Lample, Wu. ICLR 2023

Given informal theorem x_I

formal theorem x_F

1. Draft $y_I \sim p(\cdot \mid x_I)$
2. Sketch $z_F \sim p(\cdot \mid x_F, x_I, y_I)$

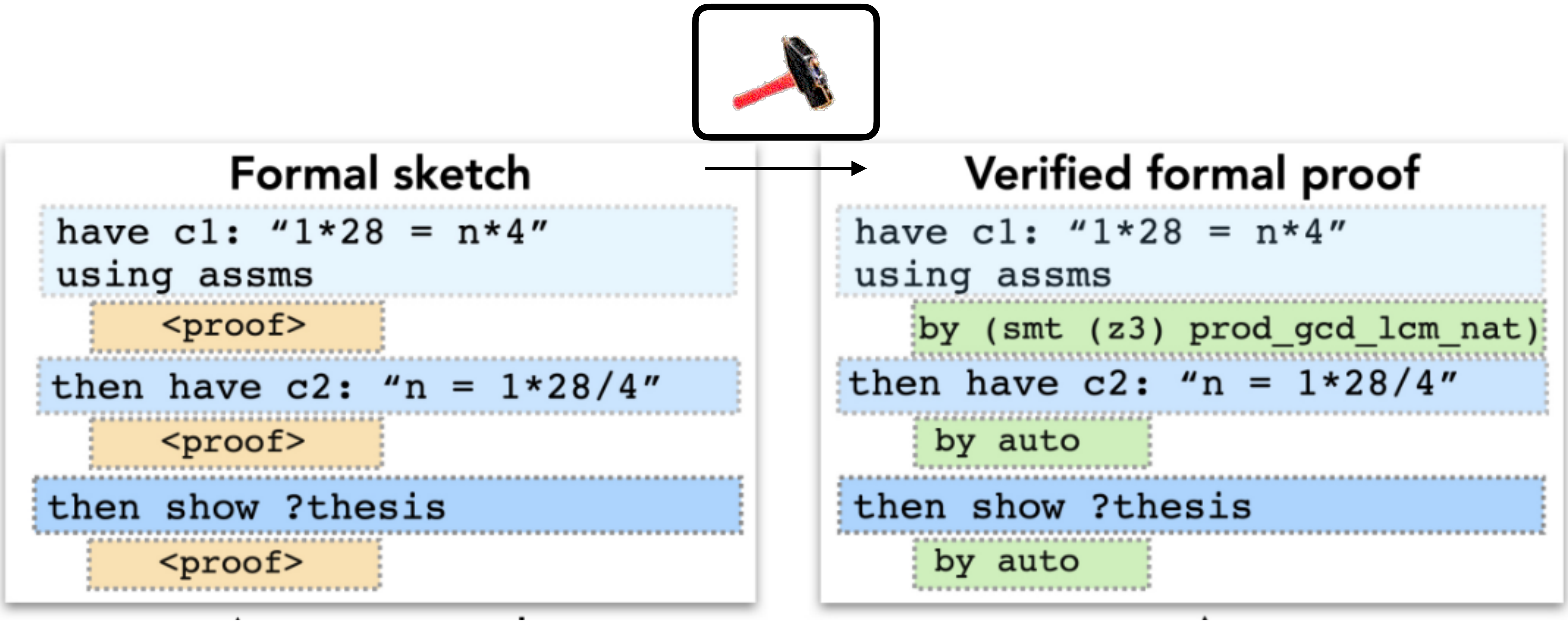


Draft, Sketch, Prove: Guiding Formal Theorem Provers with Informal Proofs
Jiang, Welleck, Zhou, Lacroix, Liu, Li, Jamnik, Lample, Wu. ICLR 2023

Given informal theorem x_I

formal theorem x_F

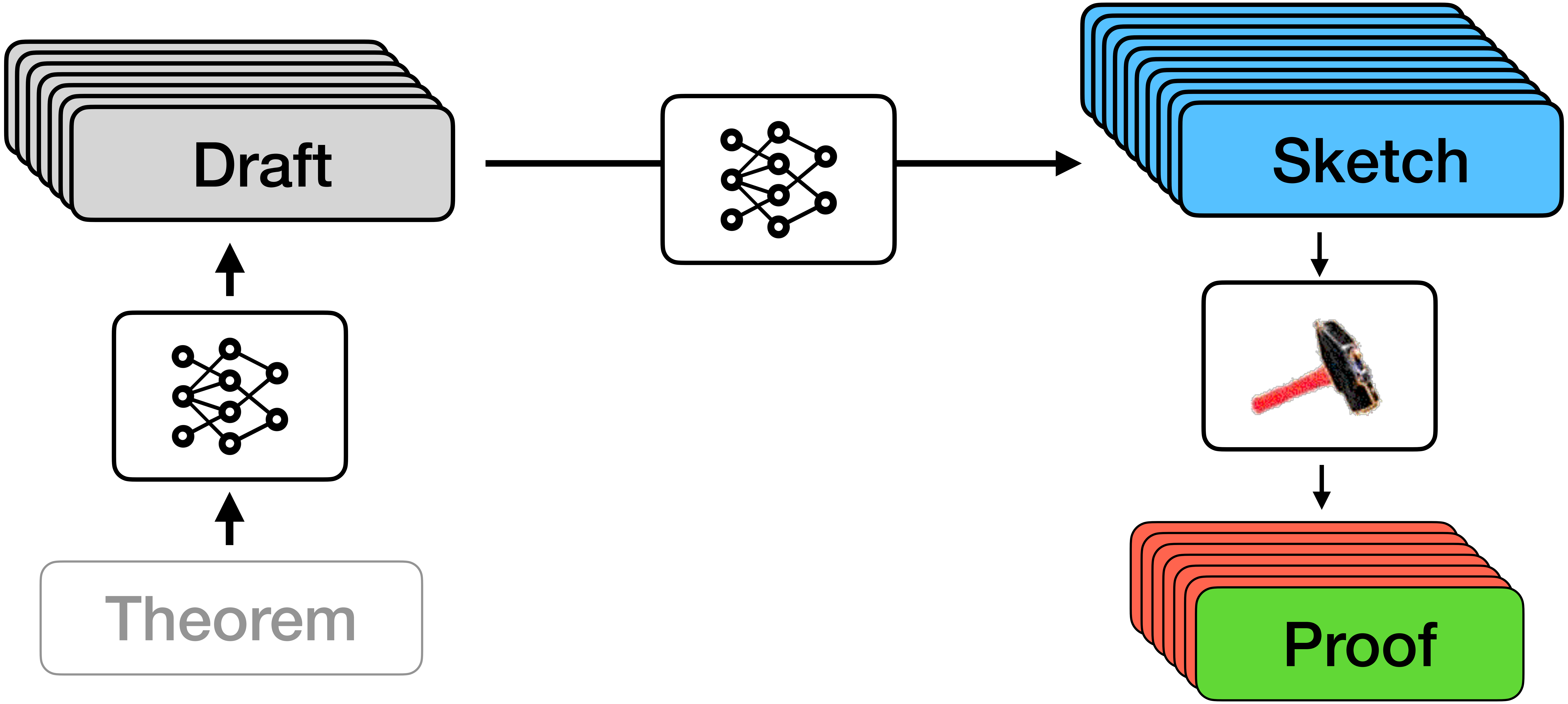
1. Draft $y_I \sim p(\cdot \mid x_I)$
2. Sketch $z_F \sim p(\cdot \mid x_F, x_I, y_I)$
3. Prove $y_F = f(x_F, z_F)$



"Classical" prover
Sledgehammer

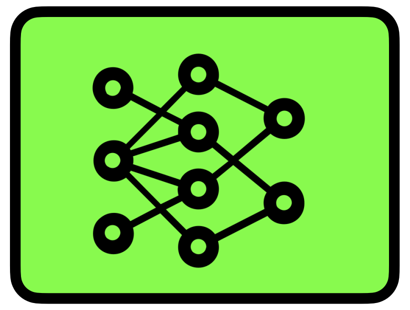
Draft, Sketch, Prove: Guiding Formal Theorem Provers with Informal Proofs
Jiang, Welleck, Zhou, Lacroix, Liu, Li, Jamnik, Lample, Wu. ICLR 2023

Proof search



International Math Olympiad 1959 P.1

Statement: Prove that the fraction $\frac{21n+4}{14n+3}$ is irreducible for every natural number n.



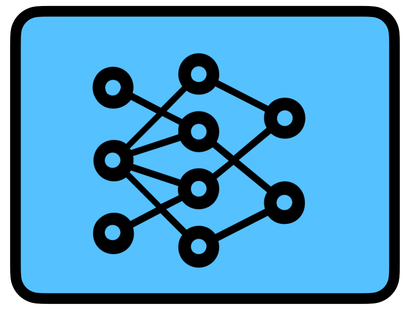
Informal Proof (Minerva 62B):

We must show that $\gcd(21n+4, 14n+3) = 1$.
The Euclidean algorithm gives

$$21n + 4 = 1 \cdot (14n + 3) + 7n + 1$$
$$14n + 3 = 2 \cdot (7n + 1) + 1.$$

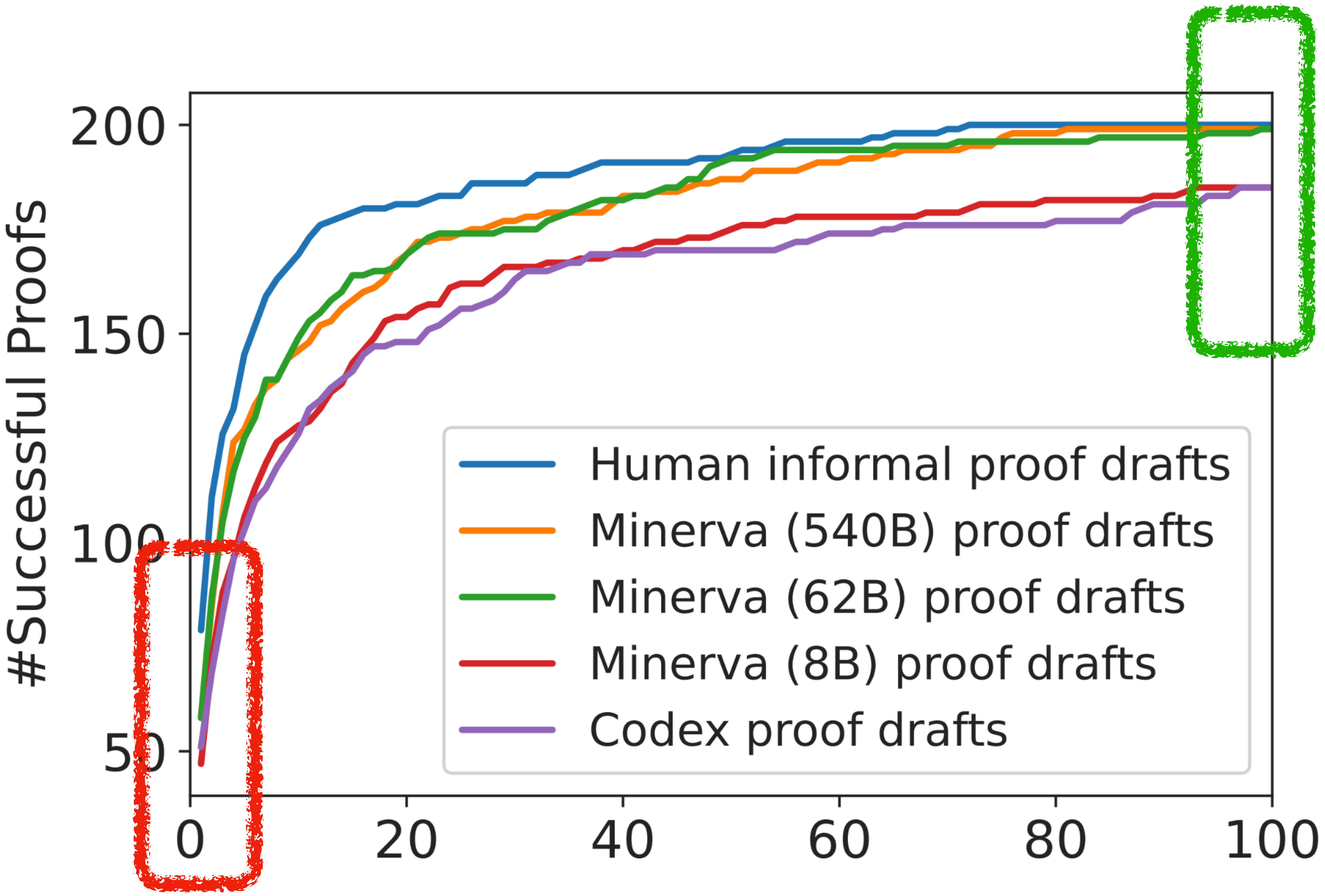
Since $\gcd(7n+1, 1) = 1$, we have $\gcd(21n+4, 14n+3) = 1$.

Formal Proof:



```
theorem imo_1959_p1:
  fixes n :: nat
  shows "gcd (21*n + 4) (14*n + 3) = 1"
proof -
  (* The Euclidean algorithm gives
  21n+4=1\cdot(14n+3)+7n+1
  14n+3=2\cdot(7n+1)+1. *)
  have c0: "21*n + 4 = 1*(14*n + 3) + 7*n + 1"
    by auto  [ATP]
  have c1: "14*n + 3 = 2*(7*n + 1) + 1" using c0
    by auto  [ATP]

  (* Since \gcd(7n+1,1)=1, we have \gcd(21n+4,14n+3)=1. *)
  then have "gcd (7*n + 1) 1 = 1"
    using c1
    by auto  [ATP]
  then have "gcd (21*n + 4) (14*n + 3) = 1"
    using c1
    by (smt (z3) BitM_plus_one ab_semigroup_add_class.add_ac(1)
    add.assoc c0 gcd.commute gcd_add2 gcd_add_mult mult_numeral_1
    numeral_One numeral_eq_Suc numerals(1) semiring_norm(3))  [ATP]
  then show ?thesis
    using c1
    by blast  [ATP]
qed
```

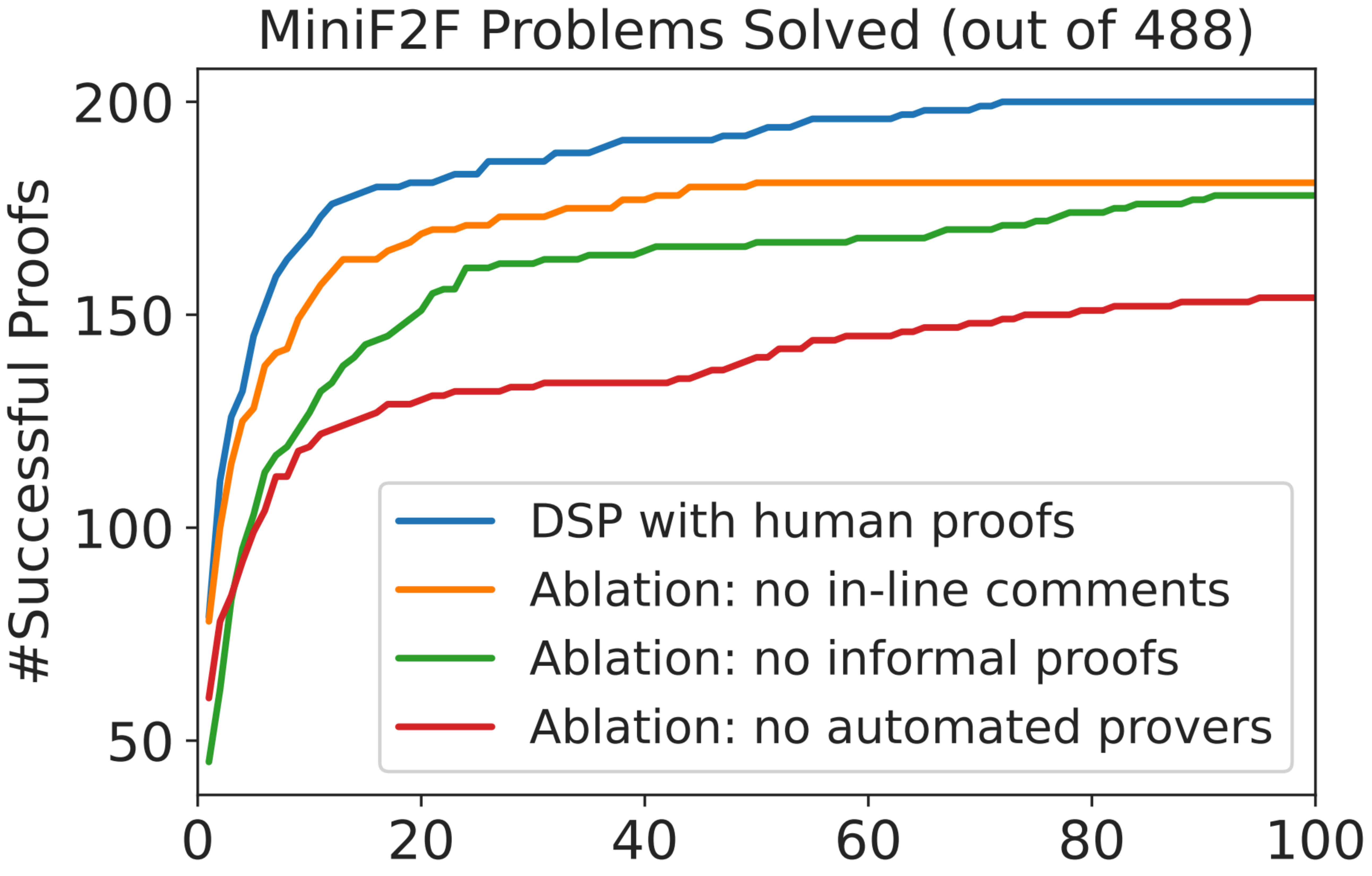
Scaling proof search



trajectories sampled

MiniF2F
480 problems

Ablations



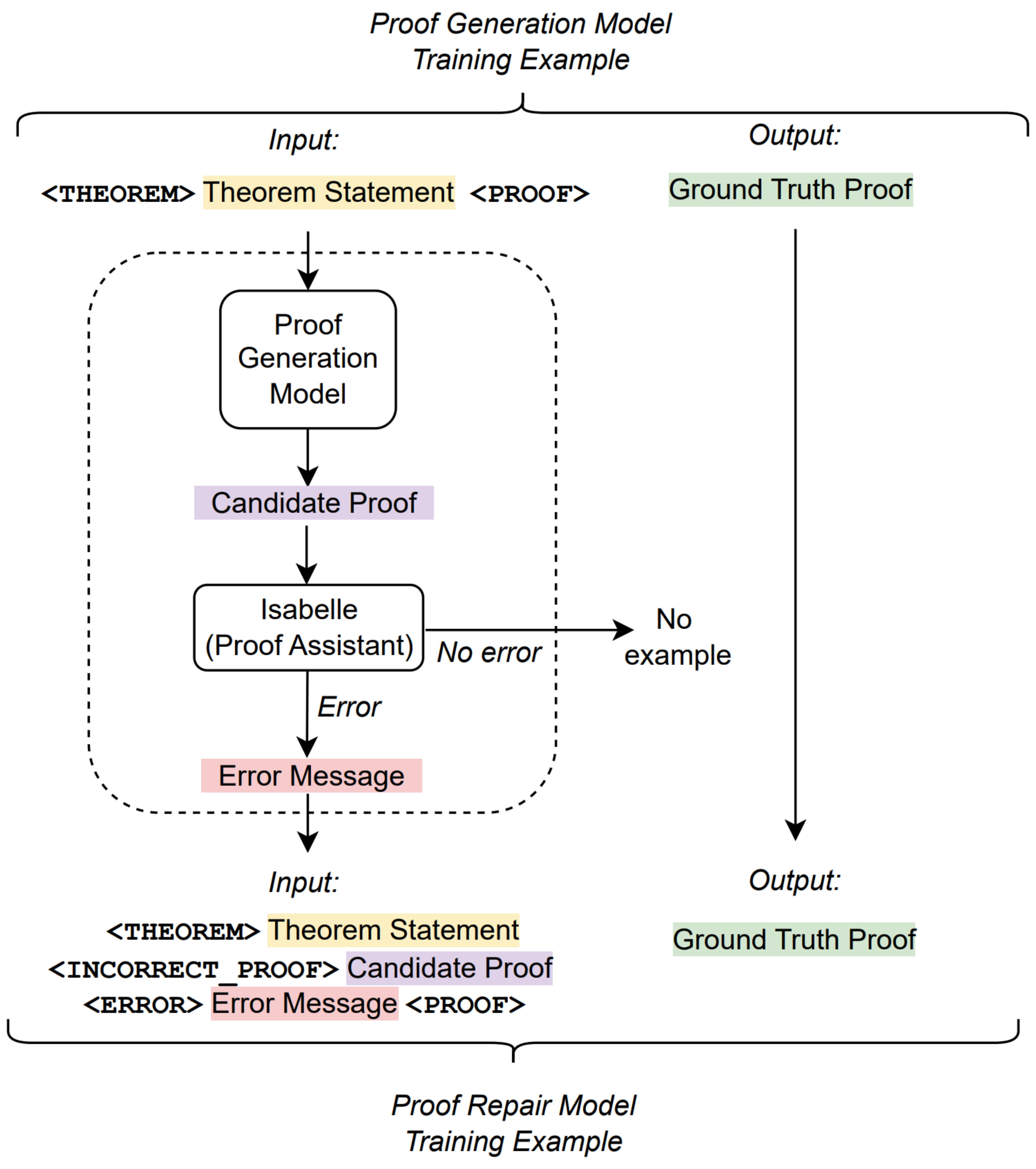
trajectories sampled

MiniF2F
480 problems

Baldur

- Generate $y^{(1)} \sim p_\theta(\,\cdot\,|\,x)$
- Repair $y^{(2)} \sim p_\theta(\,\cdot\,|\,y^{(1)}, x, \text{errors})$

- Train repair module on generator's outputs



Baldur: Whole-proof generation and repair with large language models
First, Rabe, Ringer, Brun. 2023

Baldur

Model	16 samples	64 samples
Baldur 8b generate	34.8%	40.7%
Baldur 8b generate + repair	36.3%*	—
Baldur 8b w/ context	40.9%	47.5%
Baldur 62b w/ context	42.2%	47.9%
Baldur 8b w/ context ∪ Thor	—	65.7%

Table 4: Proof rate of different models.
***The repair approach uses half the number of samples, and then one repair attempt for each sample.**

Baldur: Whole-proof generation and repair with large language models
First, Rabe, Ringer, Brun. 2023

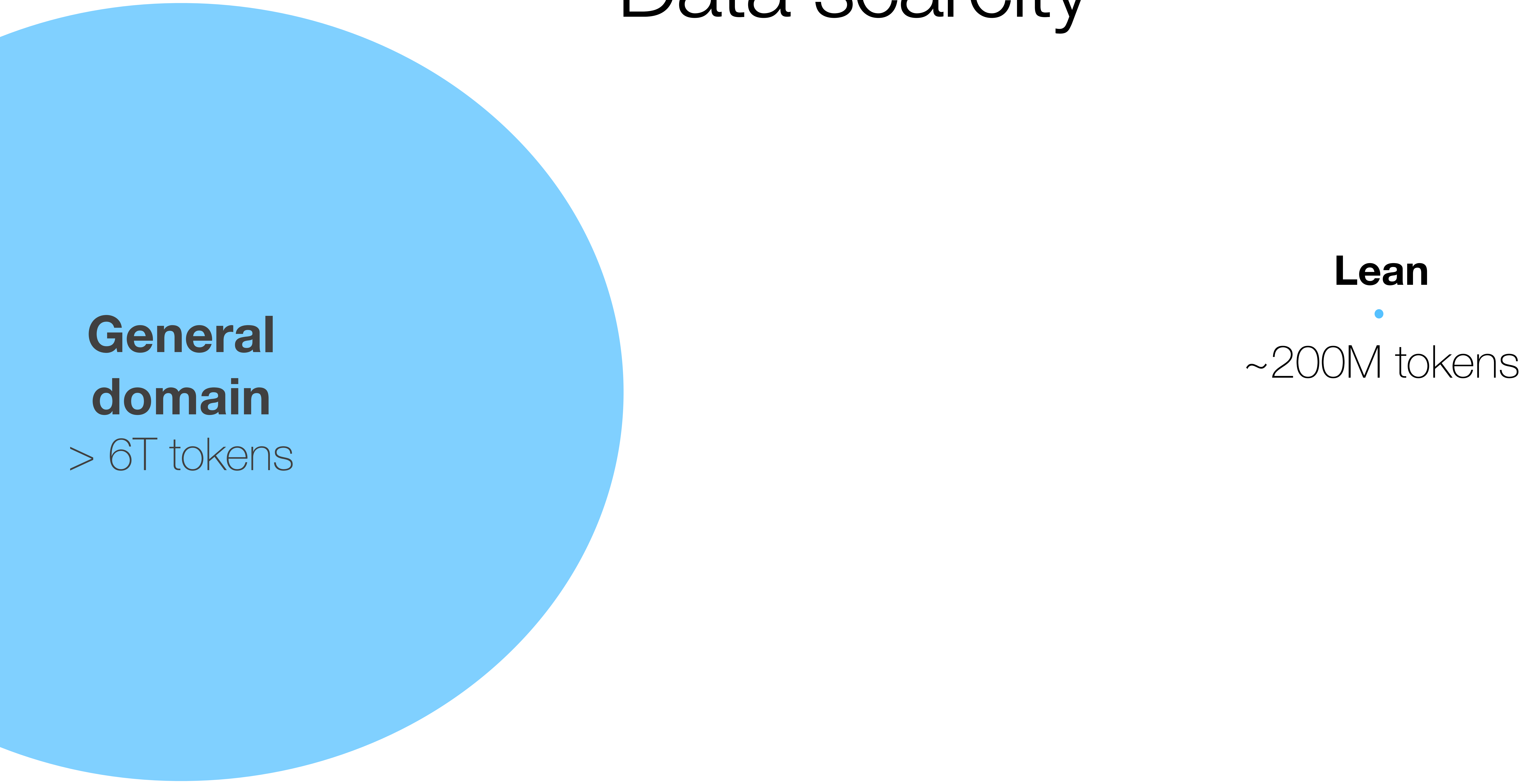
Summary

- Flexible language models
- Chain together in a cascade

(Some) open challenges

- Data
- Context
- Efficiency

Data scarcity



Data scarcity

- Option 1: transfer

General domain
> 6T tokens

Expert domain
~50B tokens

Lean
~200M tokens

- Ongoing project: EleutherAI/math-lm

Data scarcity

- Option 2: synthesize

General domain
> 6T tokens

Expert domain
~50B tokens
Transfer

Synthetic
??

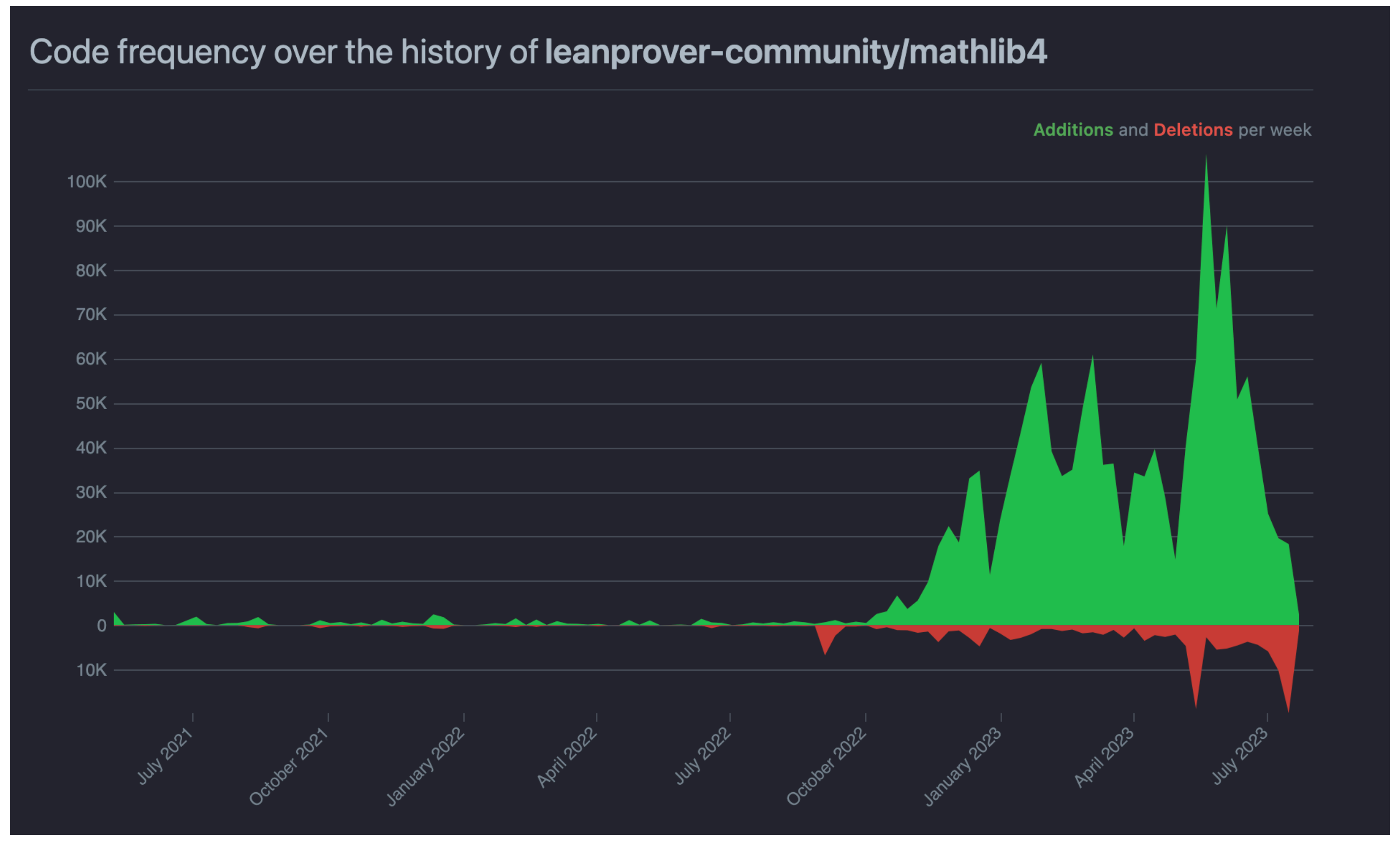
Lean
~200M tokens

- E.g. see Expert Iteration [Polu et al ICLR 2023],
  Autoformalization [Wu et al Neurips 2022]

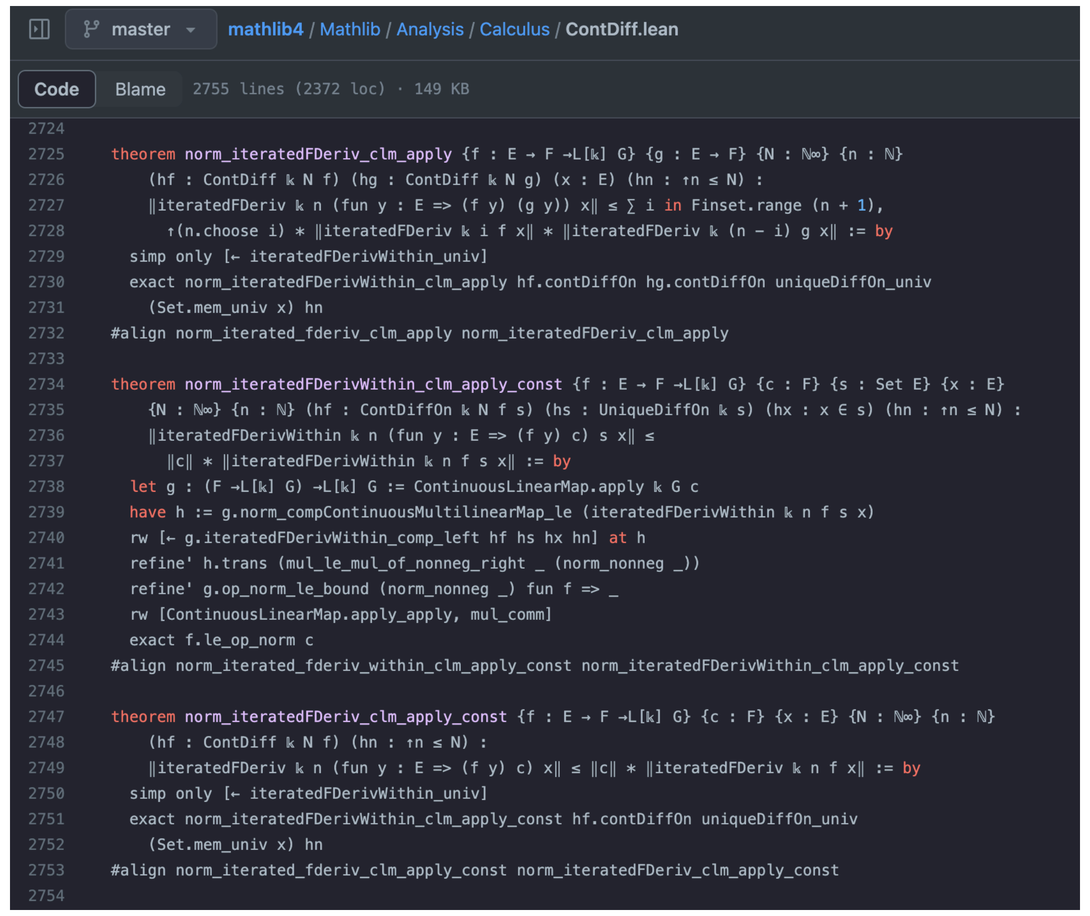
Context

$$p_\theta(y_t \mid x_t)$$

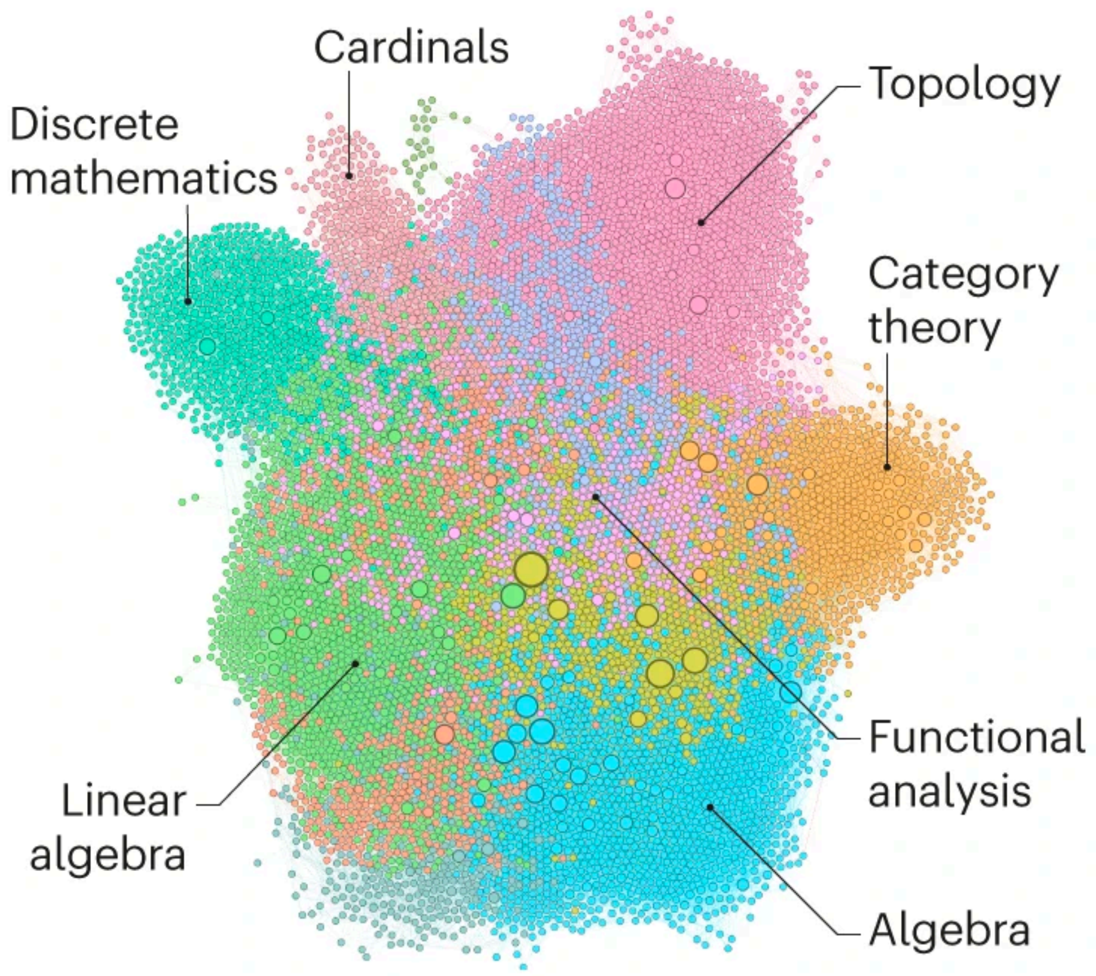
Changing code



Intermediate definitions/lemmas

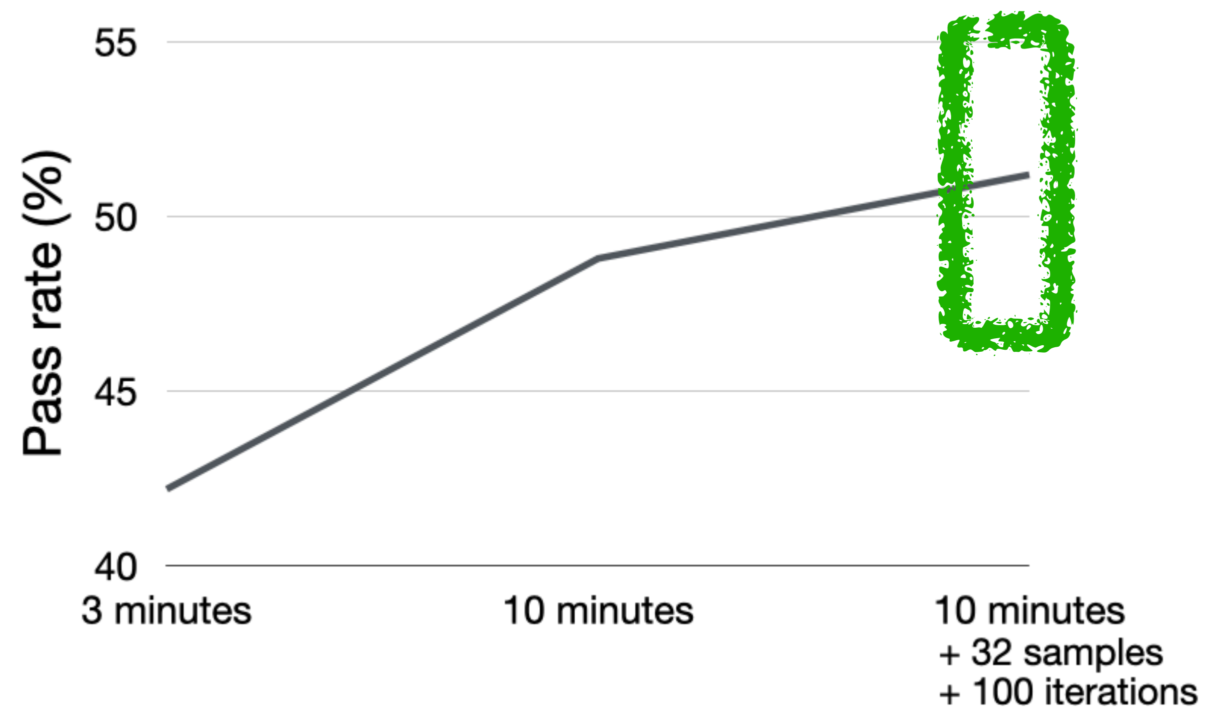


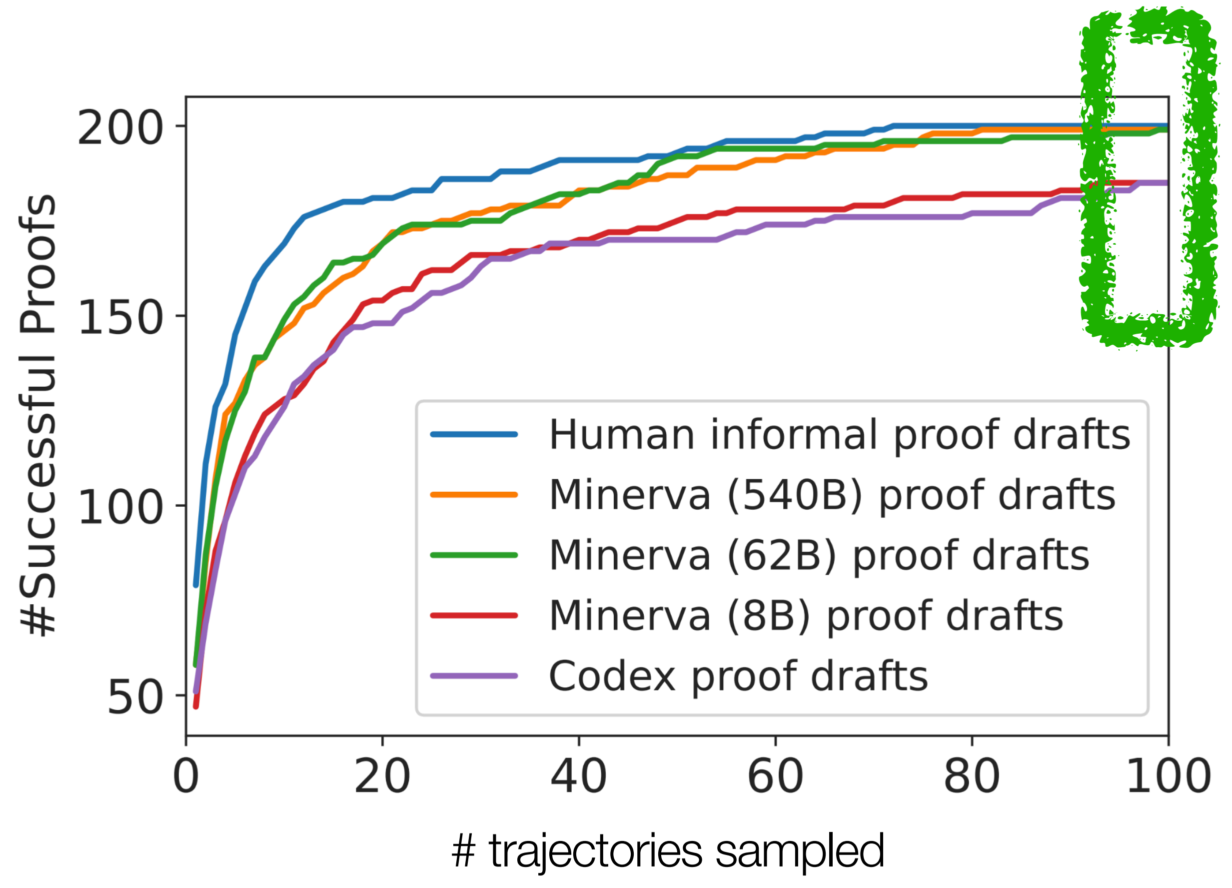

```
theorem norm_iteratedFDeriv_clm_apply {f : E → F →L[𝕜] G} {g : E → F} {N : ℕ∞} {n : ℕ}
    (hf : ContDiff 𝕜 N f) (hg : ContDiff 𝕜 N g) (x : E) (hn : ↑n ≤ N) :
    ‖iteratedFDeriv 𝕜 n (fun y : E => (f y) (g y)) x‖ ≤ ∑ i in Finset.range (n + 1),
      ↑(n.choose i) * ‖iteratedFDeriv 𝕜 i f x‖ * ‖iteratedFDeriv 𝕜 (n - i) g x‖ := by
  simp only [← iteratedFDerivWithin_univ]
  exact norm_iteratedFDerivWithin_clm_apply hf.contDiffOn hg.contDiffOn uniqueDiffOn_univ
    (Set.mem_univ x) hn
#align norm_iterated_fderiv_clm_apply norm_iteratedFDeriv_clm_apply

theorem norm_iteratedFDerivWithin_clm_apply_const {f : E → F →L[𝕜] G} {c : F} {s : Set E} {x : E}
    {N : ℕ∞} {n : ℕ} (hf : ContDiffOn 𝕜 N f s) (hs : UniqueDiffOn 𝕜 s) (hx : x ∈ s) (hn : ↑n ≤ N) :
    ‖iteratedFDerivWithin 𝕜 n (fun y : E => (f y) c) s x‖ ≤
      ‖c‖ * ‖iteratedFDerivWithin 𝕜 n f s x‖ := by
  let g : (F →L[𝕜] G) →L[𝕜] G := ContinuousLinearMap.apply 𝕜 G c
  have h := g.norm_compContinuousMultilinearMap_le (iteratedFDerivWithin 𝕜 n f s x)
  rw [← g.iteratedFDerivWithin_comp_left hf hs hx hn] at h
  refine' h.trans (mul_le_mul_of_nonneg_right _ (norm_nonneg _))
  refine' g.op_norm_le_bound (norm_nonneg _) fun f => _
  rw [ContinuousLinearMap.apply_apply, mul_comm]
  exact f.le_op_norm c
#align norm_iterated_fderiv_within_clm_apply_const norm_iteratedFDerivWithin_clm_apply_const

theorem norm_iteratedFDeriv_clm_apply_const {f : E → F →L[𝕜] G} {c : F} {x : E} {N : ℕ∞} {n : ℕ}
    (hf : ContDiff 𝕜 N f) (hn : ↑n ≤ N) :
    ‖iteratedFDeriv 𝕜 n (fun y : E => (f y) c) x‖ ≤ ‖c‖ * ‖iteratedFDeriv 𝕜 n f x‖ := by
  simp only [← iteratedFDerivWithin_univ]
  exact norm_iteratedFDerivWithin_clm_apply_const hf.contDiffOn uniqueDiffOn_univ
    (Set.mem_univ x) hn
#align norm_iterated_fderiv_clm_apply_const norm_iteratedFDeriv_clm_apply_const
```



Efficiency

- Real users:
  - run on own device (e.g. laptop)
- Large inference costs





Thank you

Tutorial code: https://github.com/wellecks/ntptutorial

Topic	Notebook
0. Intro	notebook
1. Data	notebook
2. Learning	notebook
3. Proof Search	notebook
4. Evaluation	notebook
5. `llmsuggest`	notebook

Topic	Notebook
1. Language model cascades	notebook
2. Draft, Sketch, Prove	notebook

Carnegie Mellon University

Incoming Assistant Professor, Jan. 2024