https://github.com/auto-res/HTPS-RAG

Automated Theorem Proving by HyperTree Proof Search with Retrieval-Augmented Tactic Generator

Sho Sonoda (RIKEN), Naoto Onda (OMRON SINIC X), Kei Tsukamoto (The University of Tokyo),
Fumiya Uchiyama (The University of Tokyo), Akiyoshi Sannai (Kyoto University),
Wataru Kumagai (OMRON SINIC X)

https://github.com/auto-res/HTPS-RAG

Motivations

- **Large Language Models (LLMs) do best at generating languages ever**
- **So, we want to develop an AI system for automated ML research**

Input:
“Estimate the generalization error of the following network $f(x;\theta) = W\sigma(Vx)$”

LLM

Output:
“$Risk[f] \leq EmpRisk[f] + \sqrt{\frac{C}{n}}$ with high probability where c is …”

- **But, LLMs are known to hallucinate, and have limited math abilities**

- **Possible solutions are**
  - **Prompt engineering --- in natural language**
  - **Using external computers --- in formal language (this study)**

- **In this study, we develop an LLM that perform Automated Theorem Proving (ATP) by interacting with Proof Assistant (Lean)**

https://github.com/auto-res/HTPS-RAG

Sequence Diagram of interactive proof by LLM and Proof Assistant

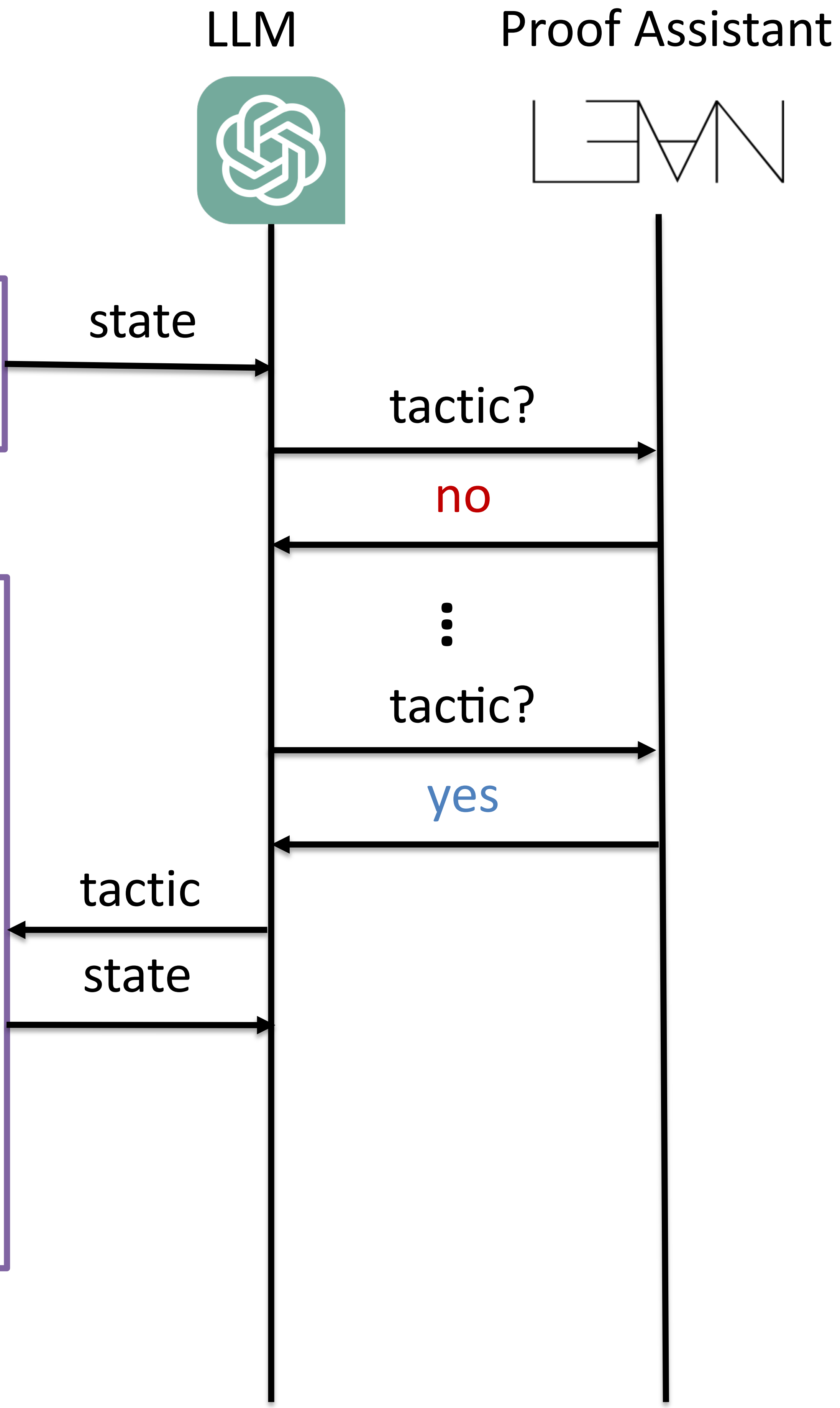
Eg. Markov inequality $P\{x \mid g(x) - f(x) \geq \varepsilon\} \leq \frac{1}{\varepsilon} E[g(X) - f(X)]$

Input (theorem)

```
theorem lintegral_add_mul_meas_add_le_le_lintegral {f g : α → ℝ≥0∞} (hle : f ≤ᵐ[μ] g)
  (hg : AEMeasurable g μ) (ε : ℝ≥0∞) :
  ∫⁻ a, f a ∂μ + ε * μ { x | f x + ε ≤ g x } ≤ ∫⁻ a, g a ∂μ := by
```

Sequential Output (proof)

```
rcases exists_measurable_le_lintegral_eq μ f with ⟨ϕ, hϕm, hϕ_le, hϕ_eq⟩
 calc
  ∫⁻ x, f x ∂μ + ε * μ { x | f x + ε ≤ g x } = ∫⁻ x, ϕ x ∂μ + ε * μ { x | f x + ε ≤ g x } := by
    rw [hϕ_eq]
  _ ≤ ∫⁻ x, ϕ x ∂μ + ε * μ { x | ϕ x + ε ≤ g x } := by
    gcongr
    exact fun x => (add_le_add_right (hϕ_le _) _).trans
  _ = ∫⁻ x, ϕ x + indicator { x | ϕ x + ε ≤ g x } (fun _ => ε) x ∂μ := by
    rw [lintegral_add_left hϕm, lintegral_indicator₀, setLIntegral_const]
    exact measurableSet_le (hϕm.nullMeasurable.measurable'.add_const _) hg.nullMeasurable
  _ ≤ ∫⁻ x, g x ∂μ := lintegral_mono_ae (hle.mono fun x hx₁ => ?_)
 simp only [indicator_apply]; split_ifs with hx₂
 exacts [hx₂, (add_zero _).trans_le <| (hϕ_le x).trans hx₁]
```



- This proof is written by human (obtained from Mathlib4)
- This “proof” is a sequence of tactics

Design Policy

- **Task:**
  - **Theorem Proving by LLM in Lean**

- **Why Lean?**
  - **The Lean math library, Mathlib, is well-developed and rapidly developing**
  - **supports practical math objects such as**
    - **concentration inequalities, and stochastic processes on $\mathbb{R}^n$**

- **Major Technologies:**
  - **Hyper-Tree Proof Search (HTPS)**
    - **Monte-Carlo Tree Search (MCTS)**
    - **+ Reinforcement Learning of networks**
  - **Premise Selection by Retrieval-Augmented Generator (RAG)**
    - **Vector search with machine-learned high-dim vector-embedding**

https://github.com/auto-res/HTPS-RAG

Hyper-Tree Proof Search (HTPS): AlphaZero-like formulation for Theorem Proving

	AlphaZero	HTPS
Task	Shogi, Go, …	Theorem Proving
State	Board state	Subgoals, or Proof States
Action	Putting stones	Tactics, or Proof Steps
Policy NW	suggests where to put the stone	suggests tactics
Critic NW	estimates the probability which player wins	estimates the provability of the given state

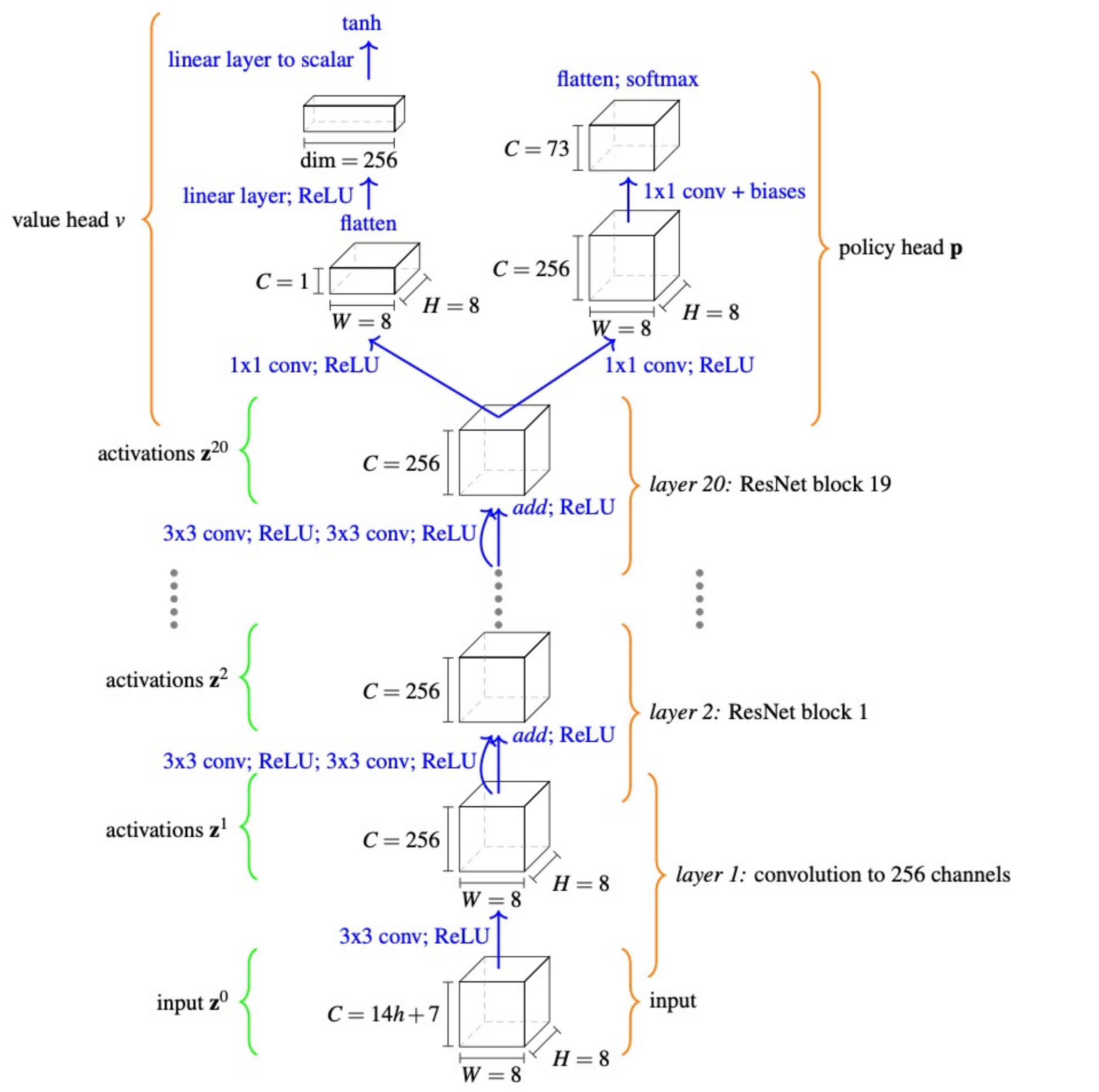


Figure 1. The AlphaZero network. Each 3×3 convolution indicates the application of 256 filters of kernel size 3×3 with stride 1. A ResNet block contains two rectified batch-normalized convolutional layers with a skip connection. In the input $\mathbf{z}^0$, a history length of $h = 8$ plies is used, encoding the current board position and those of the seven preceding plies. The input is a $8 \times 8 \times 119$-dimensional tensor.

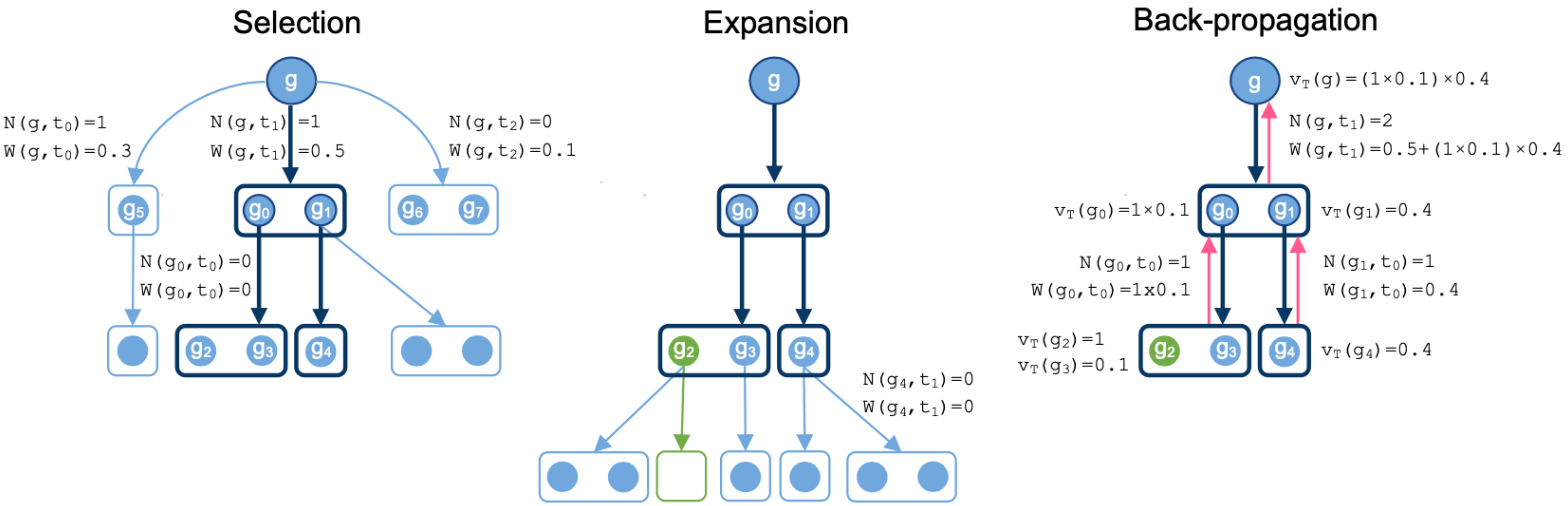
Figure from McGrath et al. pnas 119(47) 2022

[1] Lampe et al., HyperTree Proof Search for Neural Theorem Proving. NeurIPS2022

https://github.com/auto-res/HTPS-RAG

Monte-Carlo Tree Search (MCTS) --- HTPS by Lample et al. (NeurIPS2022)

- **Repeat Select-Expand-Backpropagate to grow proof tree**
  - **Select a leaf node by running the policy NW to reach the leaf**
  - **Expand the leaf by applying the tactics suggested by the policy NW**
  - **Back-propagate the node values according to critic NW**
- **We re-implement this as it is not open-sourced**

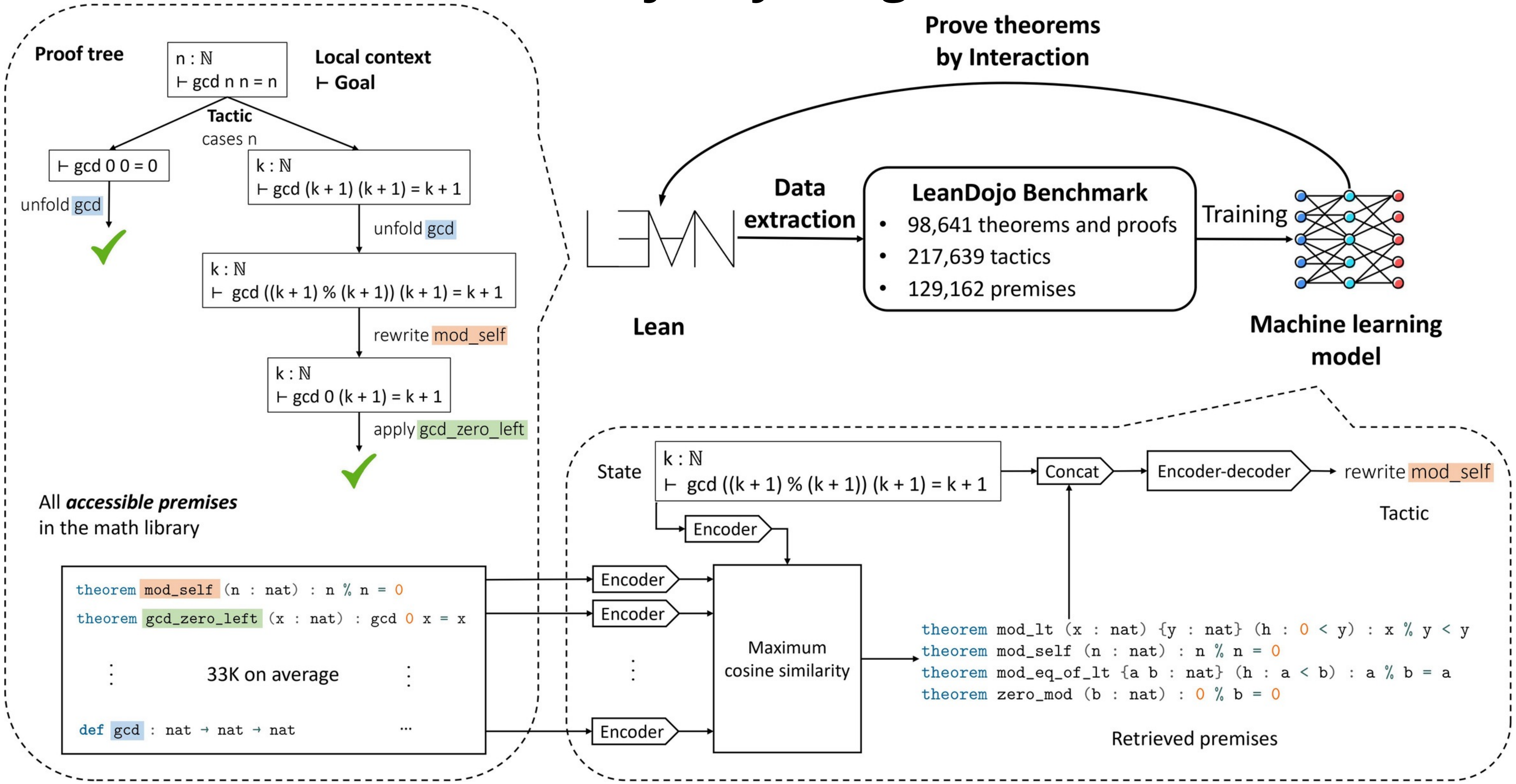


[1] Lample et al., HyperTree Proof Search for Neural Theorem Proving. NeurIPS2022

https://github.com/auto-res/HTPS-RAG

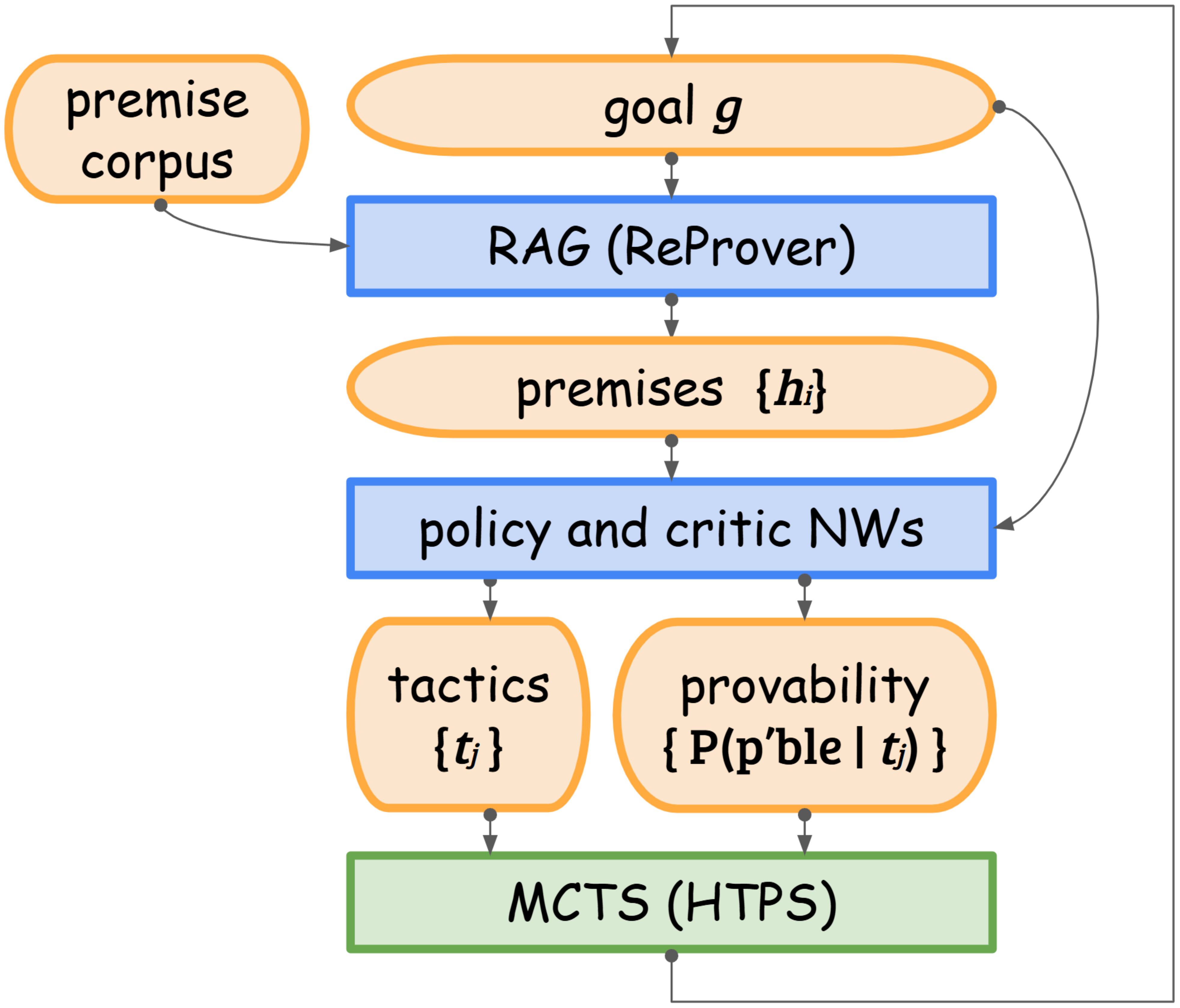
Premise Selection by Retrieval-Augmented Generator (RAG)

- **= vector search with machine-learned high-dim vector-embedding**
- **We use ReProver (from LeanDojo) by Yang et al. (NeurIPS2023dt)**



[2] Yang et al., LeanDojo: Theorem Proving with Retrieval-Augmented Language Models, NeurIPS2023 dataset track

Total Network Architecture



ReProver
ByT5 (encoder-only)

Policy NW:
ByT5 (encoder-decoder)
[g, h1, …, hn] \mapsto [t_1,…,t_m]
(fixed-dimension input)
300M params

Ciritic NW:
ByT5 (endoder-decoder)
g \maptso value
300M params

Experimental Details and Results

- ITP: Lean3
- NVIDIA A100-SXM4-80GB for 24 hours
  - Timeout for each run: 150 seconds
  - The number of premise selection: 20

- Training Dataset: Mathlib3
- Benchmark Datasets: MiniF2F and ProofNet
  - MiniF2F is high-school level
  - ProofNet is undergrad level

- Our model marked
  - 26.2% on miniF2F by pass@1
  - 10.0% on ProofNet by pass@1

https://github.com/auto-res/HTPS-RAG

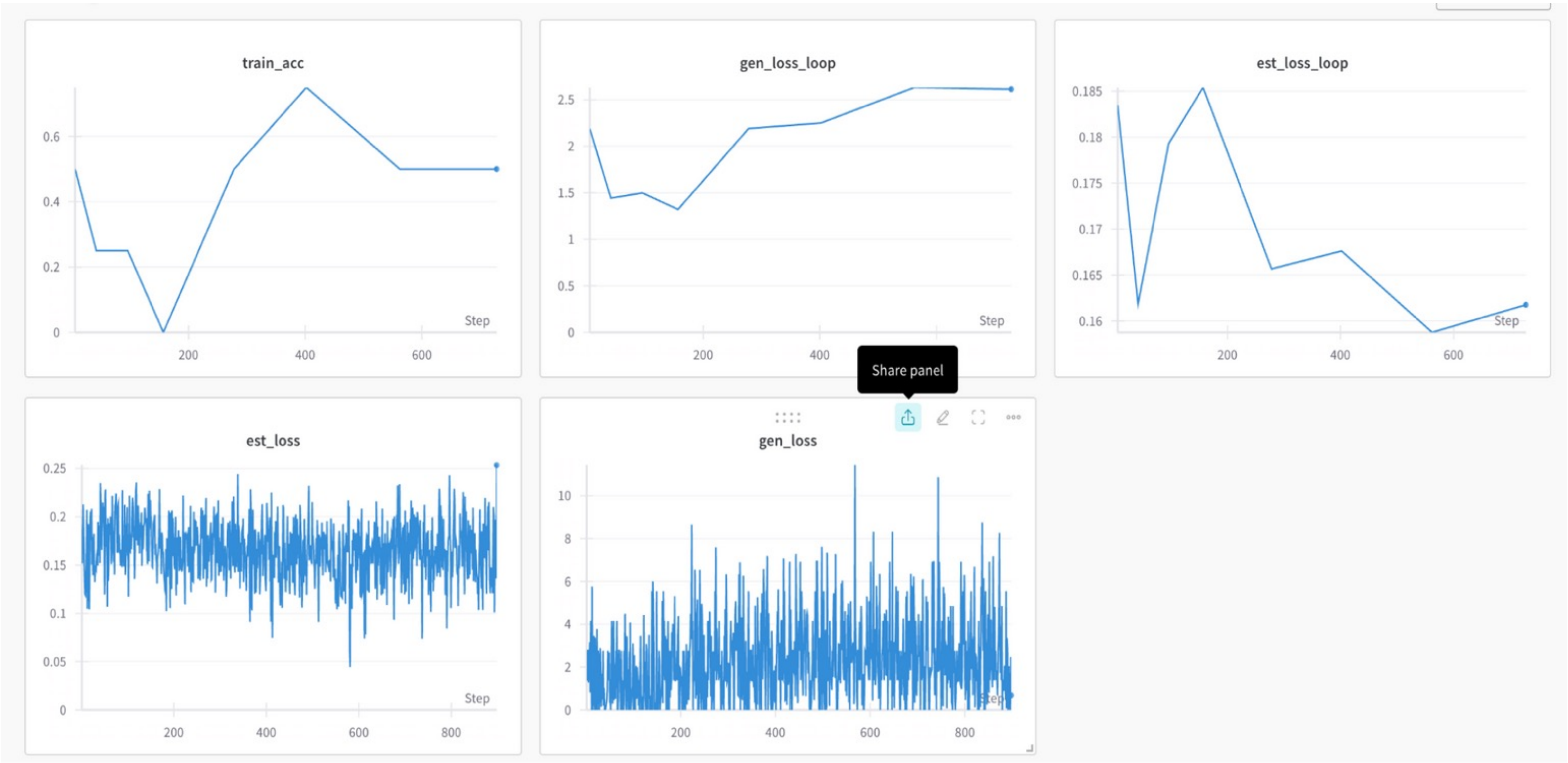
Score Boards

model	ITP	miniF2F	ProofNet
HTPS (2022)	Lean3	41.0%(pass@64)	-
LEGO Prover. (2023)	Isabelle	47.1%(pass@100)	-
ReProver (2023)	Lean3	26.5%	13.8%
Deep-Seek Prover V1.5 (2024.08.15)	Lean4	63.5% (SOTA)	23.5% (SOTA)
HTPS + RAG (**proposed**)	Lean3	26.2%(pass@1)	10.0%(pass@1)

Experimental Results

After a 24-hours of training,

- **The training loss for the critic model has decreased,**
- **but the training accuracy did not significantly improved**
- **Suggesting that we need a more machine power**



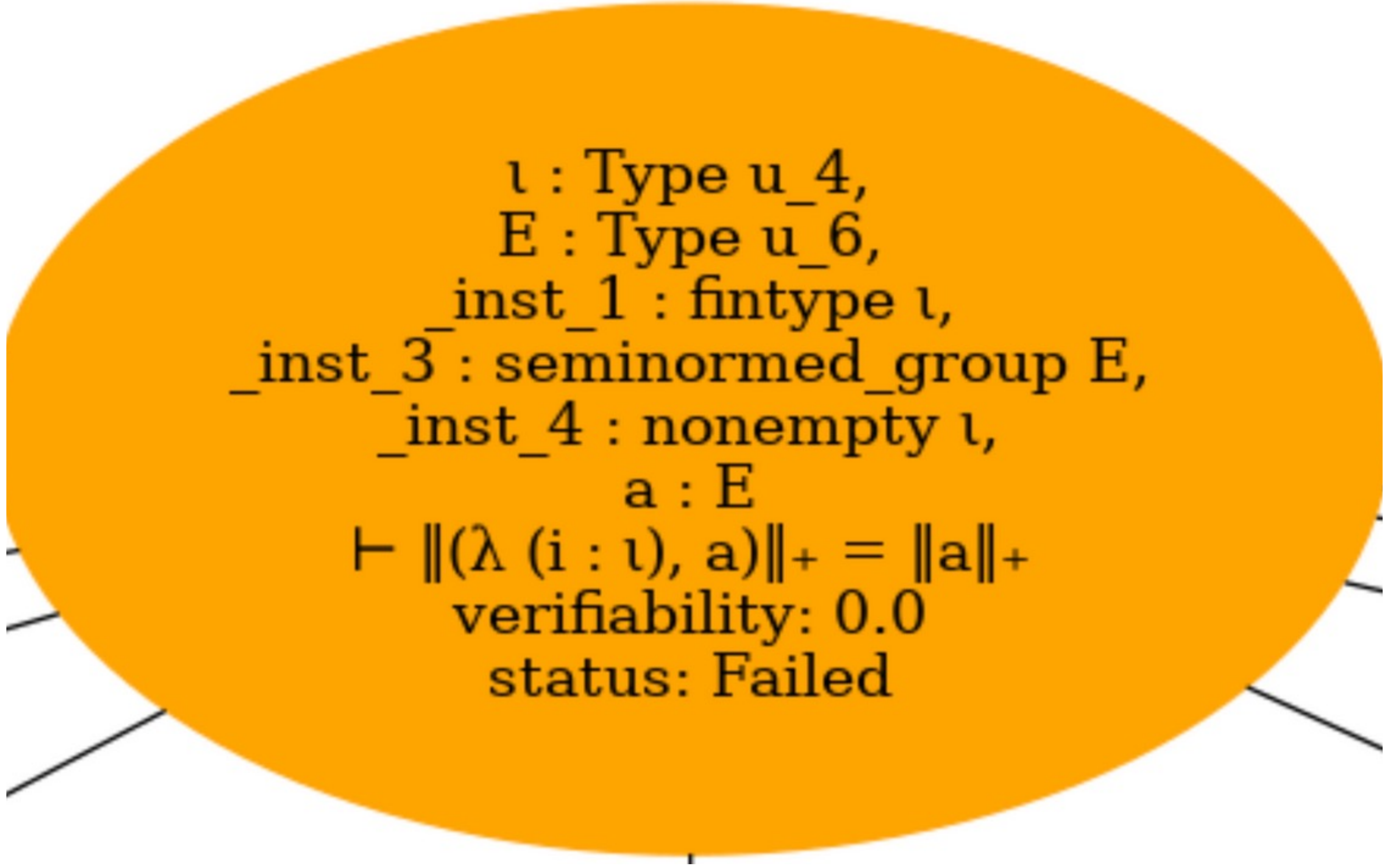
An Example of Success: Injectivity of homomorphism of fields

- **theorem exercise_3_2_7 {F : Type*} [field F] {G : Type*} [field G]**

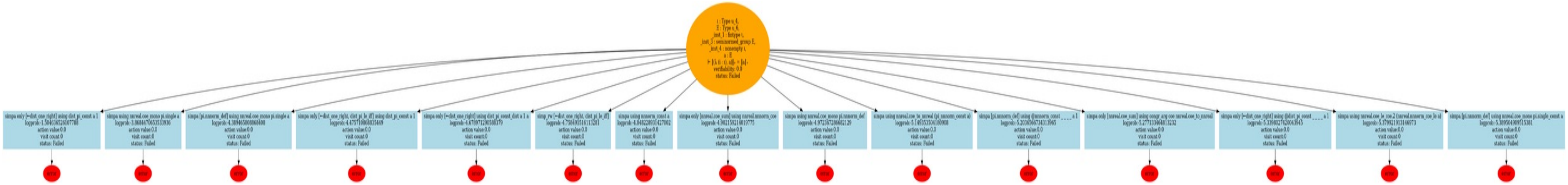
```
(φ : F →+* G) : injective φ := begin
by_cases hφ : function.injective φ
intros x y h
exact hφ h
by_contra H
apply hφ
contrapose! hφ
rw injective_iff_map_eq_zero φ
contrapose! hφ
obtain ⟨x, hx1, hx2⟩ := hφ
exact φ.injective
end
```

https://github.com/auto-res/HTPS-RAG

An Example of Failure: failed early (about 30%)

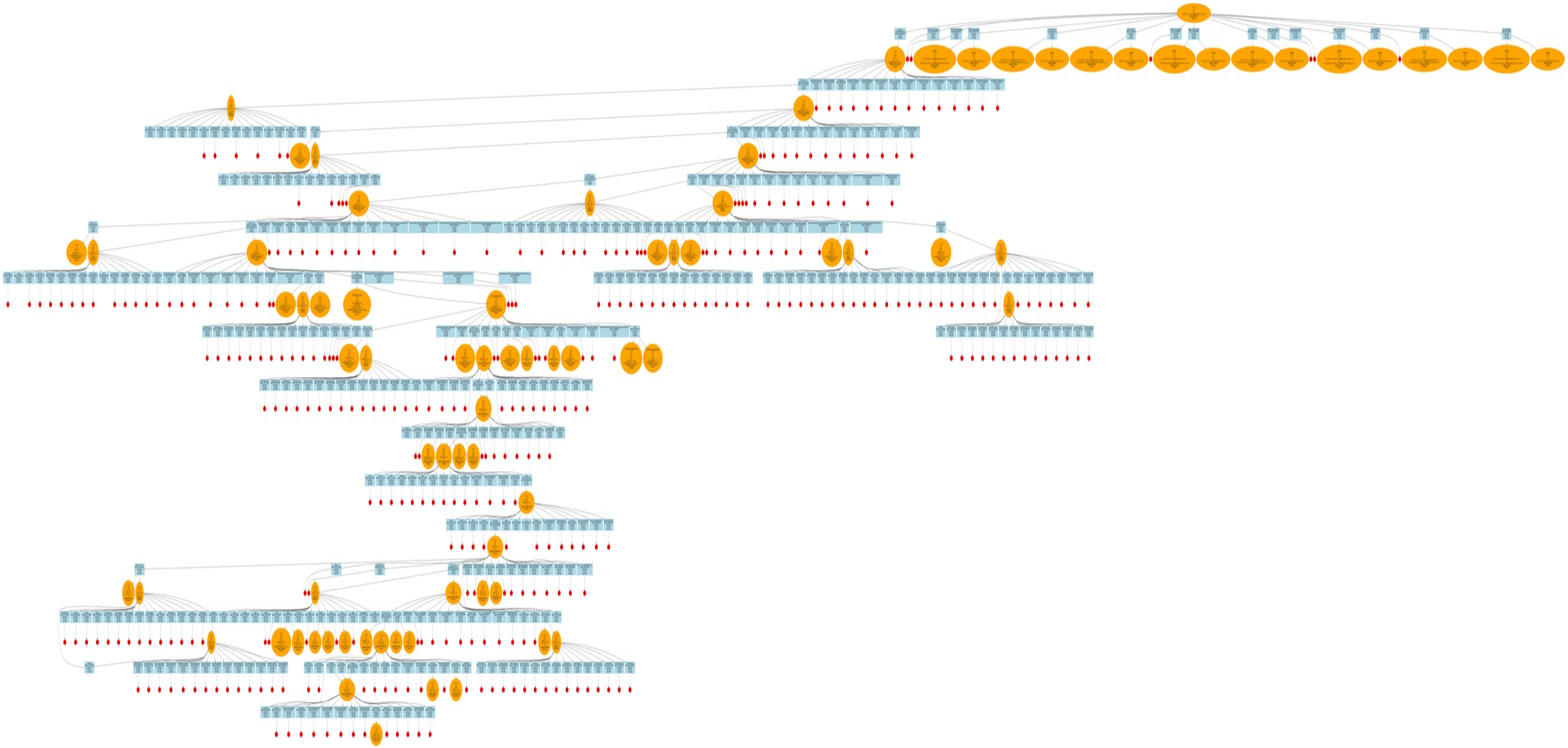


All suggested tactics failed to apply



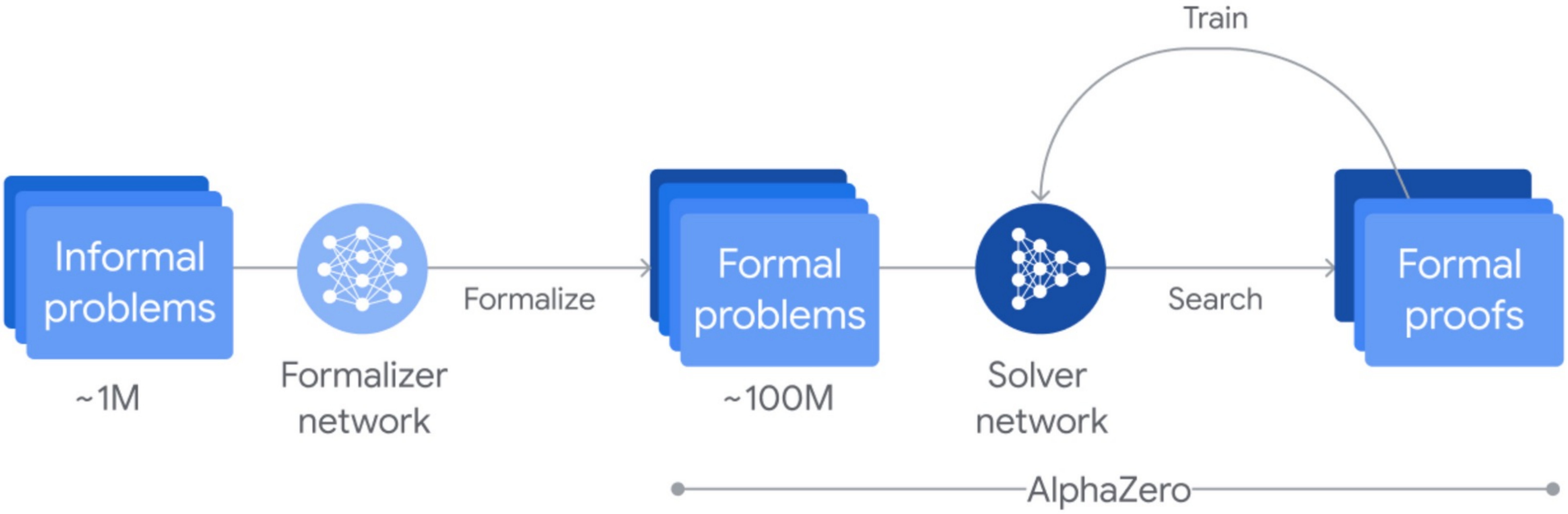
https://github.com/auto-res/HTPS-RAG

Another Example of Failure: search timed out (about 70%)



Latest Models: AlphaProof (Jul 25, 2024)

- **Augment the dataset by using the Formalizer Network**
- **“Our teams are continuing to explore multiple AI approaches for advancing mathematical reasoning and plan to release more technical details on AlphaProof soon.”**

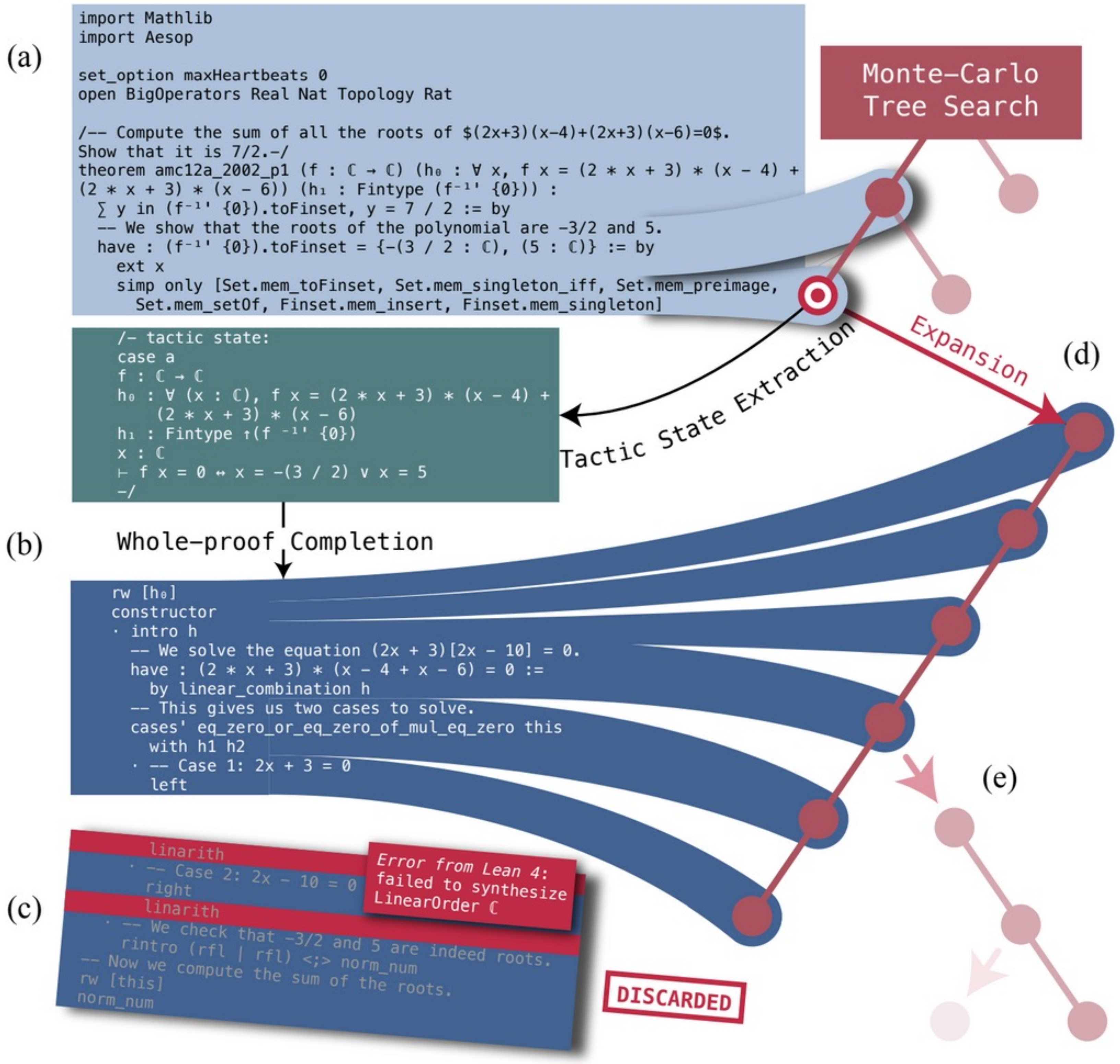


https://github.com/auto-res/HTPS-RAG

Latest Models: Deep-Seek Prover V1.5 (Aug 15, 2024)

- **Using**
  - **Lean**
  - **Reinforcement Learning**
  - **Monte-Carlo Tree Search**

- **Additionally,**
  - **Chain of Thought reasoning as a guide of proof search**
  - **Use Lean's feedback**
  - **Generates a tentative whole proof at each step for computational efficiency**



Conclusion/Limitation/Future Works

- **We need more computational resources and datasets**

- **E.g., we may increase the dataset by**
  - **auto-formalization**
  - **theorem generation**

- **Toward our final goal:**
  - **Auto-formalizer (on going)**
    - **How to verify the equivalence???**
  - **Manual dataset preparation (on going)**
    - **Particularly Hoeffding, Azuma-Hoeffding, Rademacher, Massart, …**
  - **Self-improvement by competitive game**
    - **How?**

Thank you for your attention