Prover Agent: An Agent-Based Framework for Formal Mathematical Proofs

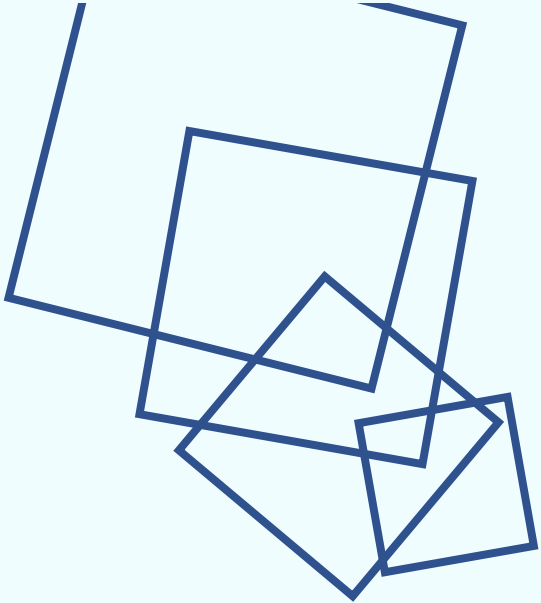
Kaito Baba[1], Chaoran Liu[2], Shuhei Kurita[3 2], Akiyoshi Sannai[4 5 6 2 7]

May. 2026
AI, Proof and Verification 2026 (AIPV 2026)
collocated with the 27th International Symposium on Formal Methods (FM 2026)

[1]The University of Tokyo, Tokyo, Japan [2]Research and Development Center for Large Language Models, National Institute of Informatics, Tokyo, Japan
[3]National Institute of Informatics, Tokyo, Japan [4]Kyoto University, Kyoto, Japan [5]Shiga University, Shiga, Japan
[6]RIKEN Center for Advanced General Intelligence for Science Program, Kobe, Japan [7]National Institute of Science Technology Policy (NISTEP), Tokyo, Japan

Section1

Overview



Motivation

- Large language models (LLMs):
  - ✓ Capable of powerful reasoning and generation
  - ✗ Prone to errors and hallucinations
- Formal proof assistants (e.g., Lean):
  - ✓ Verify mathematical correctness
  - ✗ Not generative; requires painstaking meticulous detail

➡ LLM-based formal proving is gaining attention

Yet, a large gap remains between informal reasoning and formal proving

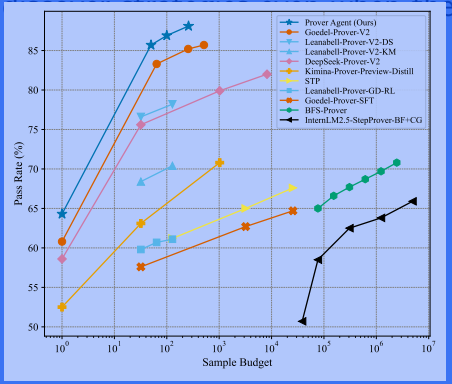
Our goal: Bridge this gap

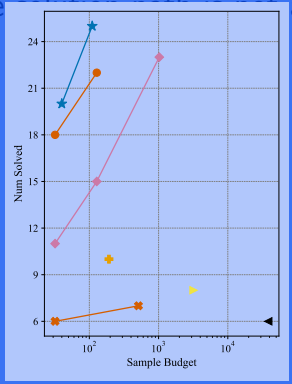
Our Contributions

- Coordination of informal and formal reasoning with Lean feedback
- Auxiliary lemma generation for strategy discovery
  - Helps discover strategies even when the solution path is not apparent at first

Our Contributions

- State-of-the-art theorem-proving performance among methods using small language models (SLMs)
- Auxiliary lemma generation for strategy discovery
  - Helps discover strategies even when the solution path is not apparent at first

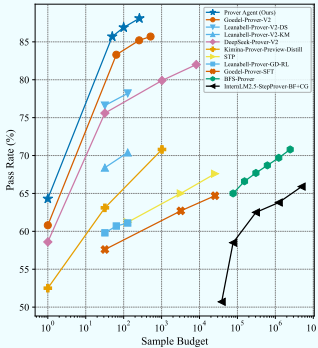




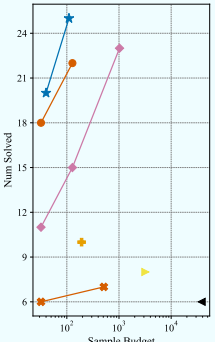
MiniF2F (Zheng et al., 2022) results PutnamBench (Tsoukalas et al., 2024) results

Our Contributions

- Coordination of informal and formal reasoning with Lean feedback
- Auxiliary lemma generation for strategy discovery
  - Helps discover strategies even when the solution path is not apparent at first
- State-of-the-art theorem-proving performance among methods using small language models (SLMs)
- Efficiency in inference-time cost
  - Small model with much smaller sample budget than prior work



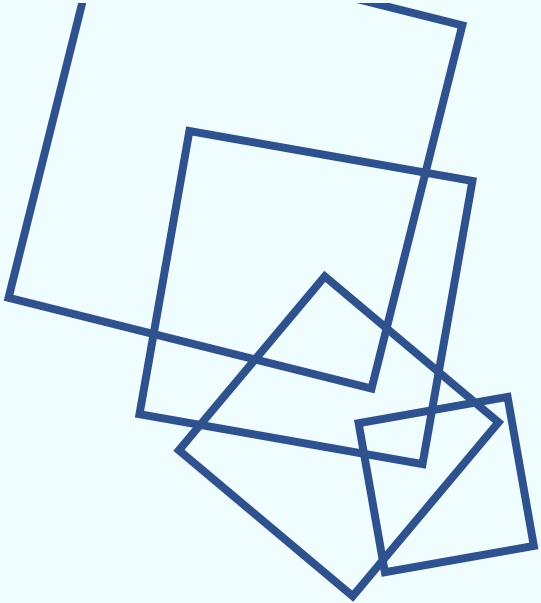
MiniF2F results



PutnamBench results

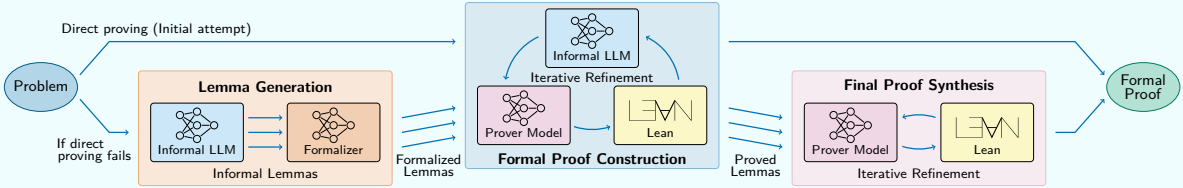
Section2

Prover Agent



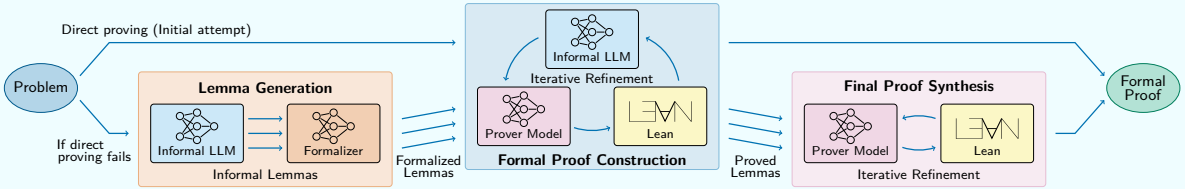
Overall Workflows

- Prover Agent coordinates:
  - Informal reasoning LLM
  - Formal prover model
  - Lean feedback



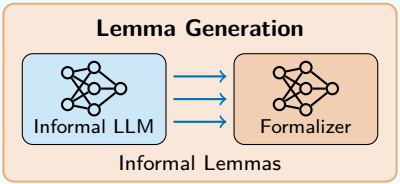
3 Key Components of Prover Agent

- Lemma generation via informal reasoning
- Formal proof construction guided by informal reasoning and iterative feedback
- Final proof synthesis guided by verified lemmas and iterative feedback



① Lemma Generation via Informal Reasoning

- Generate auxiliary lemmas
  - Specific cases
  - Potentially useful intermediate facts
- Not limited to subgoals of predefined proof sketch
  - Key difference from prior approachs



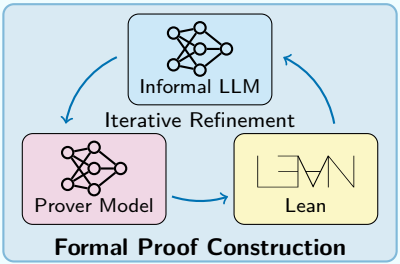
- e.g. Problem: Show that $n^2 + an$ is even ($n \in \mathbb{N}$, a: even)
  - Consider $n^2 + n$ or $n^2 + 3n$

➡ Help discover overall proof strategy

✓ Mirrors how human mathematicians typically work

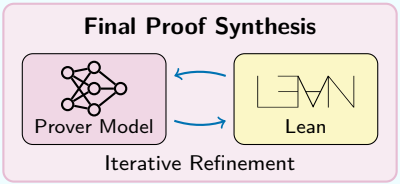
② Formal Proof Construction

- Leverage the stronger mathematical ability of the informal LLM
- Construct a formal proof using an informal proof as a guide
- Iteratively refine the proof based on Lean feedback
  - Can be seen as self-correction through in-context learning
  - Akin to how humans improve their understanding based on feedback



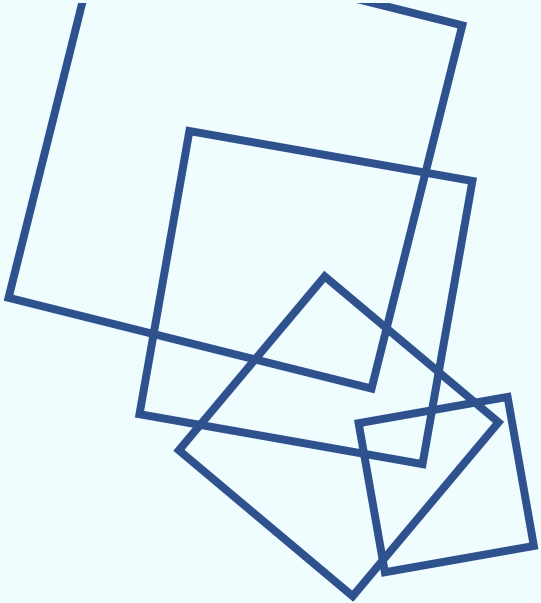
③ Final Proof Synthesis

- Consider overall proof using the lemmas
  - Use only the verified lemmas
- Allows bottom-up strategy construction even when the full plan isn't initially clear
  - Prior work: top-down approach requiring the full plan upfront
- Iteratively refine the proof based on Lean feedback



Section3

Theoretical Analysis



Theoretical Analysis

- The use of lemmas serves two key purposes:
  - Decomposing proof steps to make them more manageable
  - Helping discover proof strategies

Benefits of Lemmas for Proof Decomposition (1/3)

- Suppose the theorem involves m subgoals $G_1, \ldots, G_m$
  - Typically appear as `have` statements in Lean

Assumption 4.1. The probability p_i that the model correctly solves each G_i in a single attempt is independent across i within one proof attempt.

  - Subgoals do not interfere with each other

Assumption 4.2. Given a set of completed subgoals $\{G_i\}_{i \in S}$ with $S \subseteq [m]$[*1], the probability of proving their composition G_S (e.g., simply concatenating them) is higher than the probability of proving G_S without being given those facts: $\mathbb{P}(G_S \mid \{G_i\}_{i \in S}) > \mathbb{P}(G_S)$.

  - Providing solved subgoals does not degrade the model's original ability

[*1] $[m]$ denotes the set $\{1, 2, \ldots, m\}$

Benefits of Lemmas for Proof Decomposition (2/3)

Theorem 4.3 (Required Number of Trials). *Let N_{dir} denote the number of trials required to directly prove a problem T with probability at least $1-\delta$. Let N_{lem} denote the total number of trials required to prove the problem T with probability at least $1-\delta$, when lemmas $L_1, \ldots, L_n$ are introduced with an allowed failure probability δ_{lem}. Suppose each lemma L_i contains a subset of the subgoals $\{G_i\}_{i \in S_i}$ with $S_i \subseteq [m]$. Then the following holds:*

$$N_{\mathrm{dir}} = \Theta(p^{-m}), \qquad \mathbb{E}[N_{\mathrm{lem}}] = \tilde{\Theta}(p^{-s}),$$

where $s := \max\{\max_i |S_i|, |R_0|\} \leq m$, $R_0 := [m] \setminus \bigcup_{i=1}^{n} S_i$, and $\tilde{\Theta}$ indicates asymptotic order ignoring higher-order terms in δ_{lem}, which vanish when δ_{lem} is sufficiently small.

- Exponential improvement in the order of required trials

Benefits of Lemmas for Proof Decomposition (3/3)

Theorem 4.4 (Threshold Condition for Lemma Efficiency). *There exists a threshold* $\tau \in [0, 1]$ *such that if* $p \leq \tau$*, then* $\mathbb{E}[N_{\mathrm{lem}}] \leq N_{\mathrm{dir}}$ *holds for any* $\delta, \delta_{\mathrm{lem}} \in (0, 1)$.

- Justifies our approach of generating lemmas only for difficult problems

Theorem 4.5 (Optimal Partition of Lemma Coverage). *Under the fixed lemma coverage* $U \coloneqq \bigcup_{i=1}^{n} S_i \subseteq [m]$*,* $\mathbb{E}[N_{\mathrm{lem}}]$ *is minimized when* $|S_i| = \lceil |U|/n \rceil$ *or* $\lfloor |U|/n \rfloor$ *for all* $i \in [n]$.

- Optimal lemmas are those that decompose the problem with roughly equal difficulty

Benefits of Lemmas for Discovering Proof Strategies (1/2)

- $\mathcal{S}$: the set of possible proof strategies
- π_0: the prior distribution over strategies that the model possesses
- $L_1, \ldots, L_n$: the generated lemmas
- $Y_1, \ldots, Y_n$: the observations for each lemma
- $\pi_n(\cdot) \coloneqq \pi(\cdot \mid Y_{1:n})$: the posterior distribution over strategies
- $p(z) \in [0, 1]$: the model's success probability under a given strategy $z \in \mathcal{S}$
- $r \coloneqq \inf_z p(z)$: the worst-case success probability across strategies
- $H_0 \coloneqq H(Z) = -\sum_{z \in \mathcal{S}} \pi_0(z) \log \pi_0(z)$: the entropy of the prior distribution

Benefits of Lemmas for Discovering Proof Strategies (2/2)

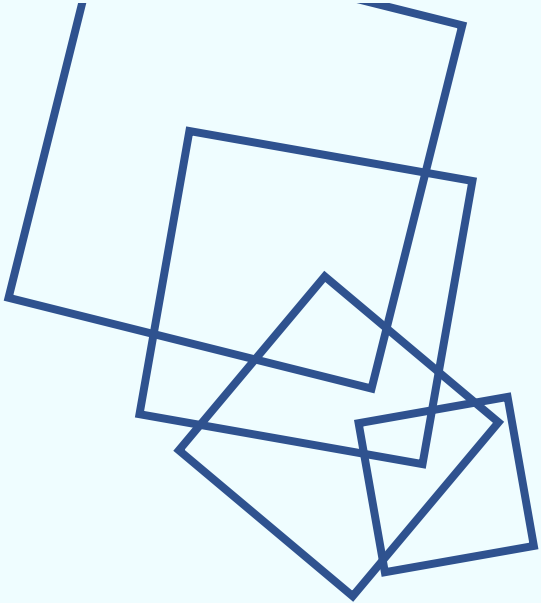
Theorem 4.5 (Success Probability Improvement by Lemmas). *The success probability of performing one trial of final proving by sampling a strategy from the posterior distribution π_n is bounded as follows:*

$$\mathbb{E}[\mathbb{P}(\mathsf{succ@1})] \;\geq\; r\exp(-H_0 + I(Z; Y_{1:n})).$$

- It improves exponentially in the mutual information contributed by lemmas
- In particular, it exceeds the no-lemma case, where $I(Z; Y_{1:n}) = 0$
- This result can also apply to natural language proofs and Lean feedback

Section4

Experiments



Experimental Setup

- Informal LLM:
  - `DeepSeek-R1-0528-Qwen3-8B` (DeepSeek-AI, 2025)
- Formal prover model:
  - `DeepSeek-Prover-V2-7B` (Ren et al., 2025)
  - `Goedel-Prover-V2-8B` (Lin et al., 2025b)
- Formalizer:
  - `Kimina-Autoformalizer-7B` (Wang et al., 2025)
  - `Goedel-Formalizer-V2-8B` (Lin et al., 2025b)

Comparison with the Previous State-of-the-Art (MiniF2F)

Prover System			Method	Model Size	Sample Budget	miniF2F test
DeepSeek-Prover-V1.5-RL + RMaxTS (Xin et al., 2025a)			Tree search	7B	32 × 16 × 400	63.5%
InternLM2.5-StepProver + BFS + CG (Wu et al., 2024)			Tree search	7B	256 × 32 × 600	65.9%
HunyuanProver v16 + BFS + DC (Li et al., 2025)			Tree search	7B	600 × 8 × 400	68.4%
BFS-Prover (Xin et al., 2025b)			Tree search	7B	2048 × 2 × 600	70.8%
Leanabell-Prover-GD-RL (Zhang et al., 2025)			Whole-proof	7B	128	61.1%
Goedel-Prover-SFT (Lin et al., 2025a)			Whole-proof	7B	25600	64.7%
STP (Dong & Ma, 2025)			Whole-proof	7B	25600	67.6%
Kimina-Prover-Preview-Distill (Wang et al., 2025)			Whole-proof	7B	1	52.5%
					32	63.1%
					1024	70.8%
DeepSeek-Prover-V2 (non-CoT) (Ren et al., 2025)			Whole-proof	7B	1	55.5%
					32	68.0%
					1024	73.2%
					8192	75.0%
DeepSeek-Prover-V2 (CoT) (Ren et al., 2025)			Whole-proof	7B	1	58.6%
					32	75.6%
					1024	79.9%
					8192	82.0%
Prover Agent (Ours)	w/ DeepSeek-Prover-V2	(Direct proving w/o iterative refinement)	Agent	8B	1	61.5%
		(Direct proving w/o iterative refinement)			50	79.9%
		(Direct proving w/ iterative refinement)			100	82.0%
		(Final proof synthesis w/ lemma)			260	82.8%
	w/ Goedel-Prover-V2	(Direct proving w/o iterative refinement)			1	64.3%
		(Direct proving w/o iterative refinement)			50	84.4%
		(Direct proving w/ iterative refinement)			100	85.7%
		(Final proof synthesis w/ lemma)			260	86.5%
	w/ Ensemble of Goedel-Prover-V2 and DeepSeek-Prover-V2	(Direct proving w/o iterative refinement)			1	64.3%
		(Direct proving w/o iterative refinement)			50	85.7%
		(Direct proving w/ iterative refinement)			100	86.9%
		(Final proof synthesis w/ lemma)			260	**88.1%**

Comparison with the Previous State-of-the-Art (MiniF2F)

- State-of-the-art performance among methods using SLMs
- High success rate under low sample budget
- Modular and Scalable Design

Prover System			Method	Model Size	Sample Budget	miniF2F test
DeepSeek-Prover-V1.5-RL + RMaxTS (Xin et al., 2025a)			Tree search	7B	32 × 16 × 400	63.5%
InternLM2.5-StepProver + BFS + CG (Wu et al., 2024)			Tree search	7B	256 × 32 × 600	65.9%
HunyuanProver v16 + BFS + DC (Li et al., 2025)			Tree search	7B	600 × 8 × 400	68.4%
BFS-Prover (Xin et al., 2025)			Tree search	7B	2048 × 2 × 600	70.8%
Geanabel-Prover-GS-RL (Zhang et al., 2025)			Whole-proof	7B	128	61.1%
Goedel-Prover-SFT (Lin et al., 2025a)			Whole-proof	7B	25600	64.7%
STP (Dong & Ma, 2025)			Whole-proof	7B	25600	67.6%
Kimina-Prover-Preview-Distill (Wang et al., 2025)			Whole-proof	7B	1	52.5%
					32	63.1%
					1024	70.8%
DeepSeek-Prover-V2 (non-CoT) (Ren et al., 2025)			Whole-proof	7B	1	55.5%
					32	68.0%
					1024	73.2%
					8192	75.0%
DeepSeek-Prover-V2 (CoT) (Ren et al., 2025)			Whole-proof	7B	1	58.6%
					32	75.6%
					1024	79.9%
					8192	82.0%
Prover Agent (Ours)	w/ DeepSeek-Prover-V2	(Direct proving w/o iterative refinement)	Agent	8B	1	61.5%
		(Direct proving w/o iterative refinement)			50	79.9%
		(Direct proving w/ iterative refinement)			100	82.0%
		(Final proof synthesis w/ lemma)			260	82.8%
	w/ Goedel-Prover-V2	(Direct proving w/o iterative refinement)			1	64.3%
		(Direct proving w/o iterative refinement)			50	84.4%
		(Direct proving w/ iterative refinement)			100	85.7%
		(Final proof synthesis w/ lemma)			260	86.5%
	w/ Ensemble of Goedel-Prover-V2 and DeepSeek-Prover-V2	(Direct proving w/o iterative refinement)			1	64.3%
		(Direct proving w/o iterative refinement)			50	85.7%
		(Direct proving w/ iterative refinement)			100	86.9%
		(Final proof synthesis w/ lemma)			260	**88.1%**

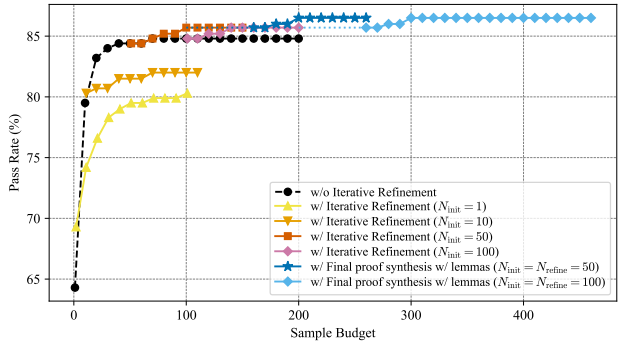
Comparison with the Previous State-of-the-Art (PutnamBench)

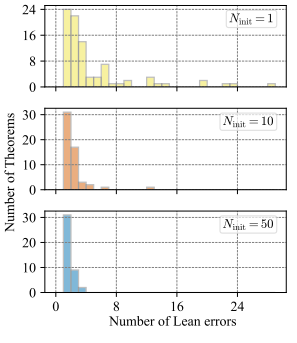
Prover System	Method	Model Size	Sample Budget	# Solved
InternLM2.5-StepProver-BF + CG (Wu et al., 2024)	Tree search	7B	$2 \times 32 \times 600$	6/640
STP (Dong & Ma, 2025)	Whole-proof	7B	3200	8/644
Goedel-Prover-SFT (Lin et al., 2025a)	Whole-proof	7B	32	6/644
			512	7/644
Kimina-Prover-Preview-Distill (Wang et al., 2025)	Whole-proof	7B	192	10/644
DeepSeek-Prover-V2 (Ren et al., 2025)	Whole-proof	7B	32	9/658
			128	10/658
			1024	11/658
Goedel-Prover-V2 (Lin et al., 2025b)	Whole-proof	8B	32	18/659
			128	22/659
Prover Agent (Ours) (Direct proving w/ iterative refinement)	Agent	8B	40	20/659
w/ Goedel-Prover-V2 (Final proof synthesis w/ lemma)			110	**25/659**

✓ State-of-the-art performance among methods using SLMs

Ablation Studies

- Importance of the quality of the initial draft used to start refinement
- The effectiveness of selecting the proof with the fewest Lean errors
- Overcome saturation from the model's inherent limitations by iterative refinement and lemmas





Case Study: Success with Lemma-Guided Proofs

- Problem:

```
theorem induction_nfactltnexpnm1ngt3
  (n : ℕ)
  (h₀ : 3 ≤ n) :
  (n)! < n^(n - 1) := by sorry
```

- Reasoning trace w/ lemmas:
  - ✓ Consider the specific cases for $n = 3, 4, 5$
  - ➡ Clearly identify the use of mathematical induction
  - ✓ Employ proof techniques used in the lemmas
- Reasoning trace w/o lemmas:
  - ✗ Proof strategy is unclear
  - ✗ The details cannot be worked out sufficiently

- Generated lemmas:

```
theorem base_case_3 : (3 : ℕ)! < 3^(3 - 1) := by
  have h_main : (3 : ℕ)! < 3^(3 - 1) := by
    -- Calculate the factorial and the power step-by-step
    norm_num [Nat.factorial, Nat.pow_succ, Nat.mul_assoc]
    -- Use 'decide' to confirm the inequality
    <;> decide
  exact h_main
```

```
theorem exponent_inequality {n : ℕ} (hn : 2 ≤ n) :
    n^(n - 1) < (n + 1)^(n - 1) := by
  have h_main : n^(n - 1) < (n + 1)^(n - 1) := by
    have h₀ : n - 1 + 1 = n := by
      omega
    have h₁ : n ≥ 2 := by omega
    have h₂ : n - 1 ≥ 1 := by omega
    have h₃ : n < n + 1 := by omega
    -- Use the fact that if a < b and n > 0, then a^n < b^n
    exact calc
      n^(n - 1) < (n + 1)^(n - 1) := by
        -- Apply the lemma that if a < b and n > 0, then a^n < b^n
        exact Nat.pow_lt_pow_of_lt_left h₃ (by omega)
      _ = (n + 1)^(n - 1) := by rfl
  exact h_main
```

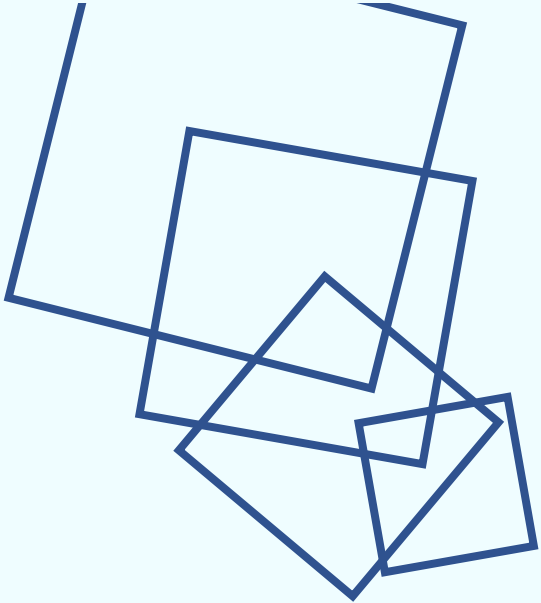
Performance on Olympiad-Level Problems

- Show strong performance on Olympiad-level problems
  - ➡ Suggest that coordination with informal reasoning may be the key
    - Olympiad-level problems require a high degree of mathematical reasoning
- Consistent gap in MATH and Custom
  - ➡ Suggests that model size and sample budget may play a more significant role here
    - Prover model also possesses a certain level of mathematical ability

		Model Size	Sample Budget	Olympiad				MATH			Custom			
				IMO	AIME	AMC	Sum	Algebra	Number Theory	Sum	Algebra	Number Theory	Induction	Sum
Number of Problems				20	15	45	80	70	60	130	18	8	8	34
Prover Agent (Ours)	(Direct proving w/o iterative refinement)	8B	1	40.0	53.3	62.2	55.0	71.4	60.0	66.2	55.6	75.0	50.0	58.8
	(Direct proving w/o iterative refinement)		100	70.0	80.0	82.2	78.8	82.9	88.3	85.4	66.7	75.0	62.5	67.6
	(Direct proving w/ iterative refinement)		400	80.0	80.0	88.9	85.0	84.3	91.7	87.7	66.7	75.0	62.5	67.6
	(Final Proof Synthesis w/ lemma)		2000	**80.0**	80.0	**91.1**	**86.3**	85.7	91.7	88.5	72.2	87.5	75.0	76.5
DeepSeek-Prover-V2 (Ren et al., 2025)		617B	8192	50.0	**93.3**	77.8	73.8	**100.0**	**96.7**	**98.5**	**83.3**	**87.5**	**100.0**	**88.2**

Section5

Conclusion



Conclusion

Summary

- Coordination of informal and formal reasoning with Lean feedback
- Auxiliary lemma generation for strategy discovery
- State-of-the-art performance among methods using SLMs

Future Work

- Extending to other domains, such as software verification

Our paper

Our code

References

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dong, K. and Ma, T. STP: Self-play llm theorem provers with iterative conjecturing and proving. *arXiv preprint arXiv:2502.00212*, 2025.

Li, Y., Du, D., Song, L., Li, C., Wang, W., Yang, T., and Mi, H. HunyuanProver: A scalable data synthesis framework and guided tree search for automated theorem proving. *arXiv preprint arXiv:2412.20735*, 2025.

Lin, Y., Tang, S., Lyu, B., Wu, J., Lin, H., Yang, K., Li, J., Xia, M., Chen, D., Arora, S., and Jin, C. Goedel-Prover: A frontier model for open-source automated theorem proving. *arXiv preprint arXiv:2502.07640*, 2025a.

Lin, Y., Tang, S., Lyu, B., Yang, Z., Chung, J.-H., Zhao, H., Jiang, L., Geng, Y., Ge, J., Sun, J., Wu, J., Gesi, J., Lu, X., Acuna, D., Yang, K., Lin, H., Choi, Y., Chen, D., Arora, S., and

References

Jin, C. Goedel-Prover-V2: Scaling formal theorem proving with scaffolded data synthesis and self-correction. *arXiv preprint arXiv:2508.03613*, 2025b.

Ren, Z. Z., Shao, Z., Song, J., Xin, H., Wang, H., Zhao, W., Zhang, L., Fu, Z., Zhu, Q., Yang, D., Wu, Z. F., Gou, Z., Ma, S., Tang, H., Liu, Y., Gao, W., Guo, D., and Ruan, C. DeepSeek-Prover-V2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. 2025.

Tsoukalas, G., Lee, J., Jennings, J., Xin, J., Ding, M., Jennings, M., Thakur, A., and Chaudhuri, S. Putnambench: Evaluating neural theorem-provers on the putnam mathematical competition. In *Advances in Neural Information Processing Systems*, volume 37, pp. 11545–11569, 2024.

Wang, H., Unsal, M., Lin, X., Baksys, M., Liu, J., Santos, M. D., Sung, F., Vinyes, M., Ying, Z., Zhu, Z., Lu, J., de Saxcé, H., Bailey, B., Song, C., Xiao, C., Zhang, D., Zhang, E., Pu, F.,

References

Zhu, H., Liu, J., Bayer, J., Michel, J., Yu, L., Dreyfus-Schmidt, L., Tunstall, L., Pagani, L., Machado, M., Bourigault, P., Wang, R., Polu, S., Barroyer, T., Li, W.-D., Niu, Y., Fleureau, Y., Hu, Y., Yu, Z., Wang, Z., Yang, Z., Liu, Z., and Li, J. Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.11354*, 2025.

Wu, Z., Huang, S., Zhou, Z., Ying, H., Wang, J., Lin, D., and Chen, K. InternLM2.5-StepProver: Advancing automated theorem proving via expert iteration on large-scale lean problems. 2024.

Xin, H., Ren, Z. Z., Song, J., Shao, Z., Zhao, W., Wang, H., Liu, B., Zhang, L., Lu, X., Du, Q., Gao, W., Zhu, Q., Yang, D., Gou, Z., Wu, Z. F., Luo, F., and Ruan, C. DeepSeek-Prover-V1.5: Harnessing proof assistant feedback for reinforcement learning and monte-carlo tree search, 2025a.

Xin, R., Xi, C., Yang, J., Chen, F., Wu, H., Xiao, X., Sun, Y., Zheng, S., and Shen, K.

References

BFS-Prover: Scalable best-first tree search for llm-based automatic theorem proving. 2025b.

Zhang, J., Wang, Q., Ji, X., Liu, Y., Yue, Y., Zhang, F., Zhang, D., Zhou, G., and Gai, K. Leanabell-Prover: Posttraining scaling in formal reasoning. *arXiv preprint arXiv:2504.06122*, 2025.

Zheng, K., Han, J. M., and Polu, S. miniF2F: a cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations*, 2022.