Inference-Time Diversity in RL-Trained Lean Theorem Provers: A Diagnostic Study

Zachary Burton*

May 2026

Abstract

RL-trained Lean theorem provers mode-collapse at inference time: on miniF2F-test with DeepSeek-Prover-V1.5-RL, doubling the i.i.d. sampling budget from k=32 to k=64 produces zero additional solved theorems (42/244 in both). A fixed schedule of 15 tactic skeletons breaks this plateau (mean $\Delta = +12.3 \pm 4.2$ theorems across n=3 seeds, sign preserved in every seed). A controlled ablation isolates the mechanism: tactic skeletons help; instruction paraphrases match the topline; irrelevant Lean comments degrade; and a dedicated natural-language-only condition (C3) lands at 48/244, intermediate between sample-only and structured-skeleton. A leave-one-out formalization-difficulty stratification reveals a structural-content gradient across the perturbations, with the skeleton intervention's gain concentrated in the easy-plus-trivial buckets where V1.5-RL's gap to other open-source provers lives. The phenomenon is RL-specific: V1.5-Base proves zero theorems with or without skeletons, identifying RL as the locus of both proof capability and the inference-time collapse that limits it; extending to two additional 7B Lean provers, RL-trained DeepSeek-Prover-V2-7B contributes +3 frontier theorems no i.i.d. baseline of any model in our matrix reaches despite a flat aggregate, while SFT-trained Goedel-Prover shows -10.0 ± 4.4 theorems across n=3 seeds (sign preserved every seed). A direct distributional measurement on the same per-attempt logs makes the mode-collapse diagnosis concrete: across 13,159 stochastic samples, the median V1.5-RL theorem receives only 2 distinct first-tactic heads, with 49% of the benchmark receiving a deterministic single strategic opening.

1 Introduction

Recent neural theorem provers combine large language models with reinforcement learning and tree searches to achieve strong results on formal proof benchmarks [17, 2, 14, 8]. The prevailing paradigm assumes that sufficient RL training

*MIT '26.

and inference-time compute will cause models to internalize the structural priors required for formal proofs.

We present evidence that this assumption is flawed. RL-trained provers suffer from *mode collapse* at inference time: they converge on a narrow set of proof strategies and fail to explore alternatives even when given additional sampling budget. On miniF2F-test with DeepSeek-Prover-V1.5-RL, i.i.d. sampling at temperature 0.6 solves 38 theorems at k=16, 42 at k=32, and *still 42* at k=64—32 additional stochastic samples find zero new proofs.

A fixed prompt schedule of 15 common tactic skeletons (`simp`, `intro`, `constructor`, …) breaks the plateau, yielding a +45% relative improvement at k=16 (55 vs. 38) and continuing to find new proofs through k=64. A controlled diversity ablation (skeleton vs. paraphrase vs. irrelevant comment) confirms the gains are *structural*, not an artifact of prompt diversity. Stratifying by an empirical formalization-difficulty score grounded in baseline solvability across our experimental matrix further localizes the effect.

A recent line of work has examined whether RL with verifiable rewards (RLVR) genuinely expands a base model's reasoning capacity [19, 16]. We study a different question—inference-time collapse within an RL-trained policy in an exact-verifier domain—and discuss the relationship in detail in Section 2.

1.1 Contributions

- We diagnose mode collapse in RL-trained Lean provers via two complementary signatures: an i.i.d. sampling plateau at k=32 → 64 on V1.5-RL that a fixed 15-skeleton schedule breaks, and a controlled diversity ablation (skeleton vs. paraphrase vs. irrelevant comment, plus a dedicated NL-only condition C3) showing the gain comes from structural guidance, not prompt diversity.

- A leave-one-out empirical-difficulty stratification, grounded in cross-prover baseline solvability, localizes the structural gain to the easy/trivial buckets where V1.5-RL's gap to other open-source provers concentrates (+15/0/−14 for skeleton, paraphrase, comment in the combined easy+trivial column), and reveals a clean structural-content gradient.

- We establish RL-specificity in two ways. A within-architecture base-vs-RL contrast on V1.5 shows the non-RL base has *zero* proof capability, so RL is the locus of both proof capability and inference-time collapse. A cross-model contrast on two additional 7B provers shows RL-trained V2-7B gains +3 frontier theorems unreachable by any i.i.d. baseline despite a flat aggregate, while SFT-trained Goedel-Prover loses −5. All experiments use Lean v4.9.0-rc1 to rule out newer-tactic-solver confounds.

2 Related Work

Neural Theorem Provers. Early systems like GPT-f [9] and PACT [4] demonstrated transformer-based tactic generation, with LeanDojo [18] adding retrieval. Recent RL-trained models—DeepSeek-Prover-V1.5 [17] and V2 [10]—and SFT-trained Goedel-Prover [8] push miniF2F performance above 60%. We deliberately study V1.5-RL as the controlled case: it is the only widely benchmarked RL-trained prover that releases a non-RL base variant, enabling within-architecture isolation of RL's effect.

Tree Search and Hint-Based Methods. HyperTree [7] and COPRA [12] build proof trees with explicit backtracking; our skeleton schedule can be viewed as fixing only the first tree node, operating in a single forward pass without proof-state feedback. Draft-Sketch-Prove [5], ConjectureBench [11], and Aristotle [1] intervene at a higher level of abstraction (informal sketches, conjecture generation, semantic lemmas), whereas we operate at the syntactic tactic-stem level and focus on diagnosing RL-induced underexploration. Lean 4 [3] with Mathlib [13] forms our verification stack and miniF2F [20] (244 competition-level theorems) is the benchmark throughout.

Entropy Collapse and the Limits of RL Post-Training. RL post-training narrows the output distribution of the underlying base model, trading diversity against best-of-N performance [6]. Two recent works study whether RL with verifiable rewards (RLVR) expands a base model's reasoning support: Yue et al. [19] find that base models surpass their RL-trained variants at large k on math and coding benchmarks with answer-string verification, and Wu et al. [16] argue more generally that the contraction of output support induced by RLVR can outweigh its expansion of high-reward modes. Our setting differs from those measurements in two relevant ways: the Lean kernel verifies proofs exactly rather than by string-match or unit-test (so V1.5-Base proves zero theorems at any k, Section 4.4), and our cross-model comparison includes an SFT-trained control (Goedel-Prover) absent from those setups.

3 Method

3.1 Skeleton Schedule and Ablations

A structured query is the pair (x, s) of theorem statement x and tactic skeleton s. The schedule is a deterministic $15\times8{=}120$ grid: 15 hand-selected Lean tactic skeletons (Appendix A, including the empty skeleton) crossed with 8 goal-hint comments (the first being empty). Schedule advances tactic-first, hint-second; each of the 120 attempts is a distinct (skeleton, hint) prompt configuration with no repeats within $k{\leq}120$. Our tested budgets $k \in \{16, 32, 64\}$ sample the first 16, 32, and 64 grid entries. The schedule is fully deterministic; stochasticity comes only from temperature sampling.

To isolate structural content from prompt diversity per se, we add two ablations (Appendix B): **C1 (paraphrase)**—16 semantically equivalent rephrasings of the instruction injected as Lean comments, with empty tactic prefix; and **C2 (comment)**—16 content-free Lean comments (`/- approach alpha -/`, …) prepended to the theorem.

3.2 Experimental Setup

Models. The primary within-architecture analysis (Sections 4.1–4.4) uses DeepSeek-Prover-V1.5-RL and its non-RL base variant V1.5-Base (same pretraining, no RL fine-tune). V1.5 is the only widely benchmarked RL-trained Lean prover that releases a paired non-RL checkpoint. The cross-model study (Section 4.5) extends pass@16 to two additional 7B provers: DeepSeek-Prover-V2 [10] (RL, reasoning-mode) and Goedel-Prover-SFT [8] (SFT-only, completion-mode). Neither releases a paired non-RL base checkpoint, so the within-architecture contrast is only possible on V1.5.

Skeleton delivery. The intervention is delivered in each model's native modality: literal Lean tactic prefix for completion-mode provers (V1.5-RL/Base, Goedel-Prover) and natural-language chat instruction ("Begin the proof with: `<tactic>`") for the reasoning-mode prover (V2). Both operationalize the same tactic-stem hypothesis, adapted to the paradigm each model was trained for. Forcing literal Lean prefix on chat-mode models would conflate the structural-guidance claim with cross-paradigm prompt-format robustness.

Search procedure. Both modes execute k LLM calls per theorem; the search exits on first Lean success (strict pass@k). Sample (A) varies seeds with empty prompt; structured (B) varies prompts (per the schedule) with fixed seed—diversity sourced from sampling vs. prompt, holding LLM-call budget constant. Outputs are post-processed before Lean verification (theorem-signature stripping, markdown-fence removal, re-indentation); see Appendix C for the exact pipeline.

Benchmark and environment. miniF2F-test (244 theorems), Lean 4 with toolchain pinned to v4.9.0-rc1 and Mathlib commit `7fa489a5`—matching the V1.5 training environment, which is critical because newer Lean tactic solvers (`simp`, `nlinarith`) would confound the gain attribution.

Decoding and inference. Temperature 0.6, top-p 0.95. Completion-mode provers use max 1024 tokens per attempt; the reasoning-mode prover (V2) uses max 8192 tokens. vLLM on a single A100 (80GB for V1.5, 40GB for V2/Goedel). Lean verification via `lake env lean --json`, 120s timeout, 8-way parallel sharding (i mod 8). Full code, prompts, and per-attempt logs at *[redacted for review; URL upon acceptance].*

4 Results

4.1 Scaling Analysis: Mode Collapse in i.i.d. Sampling

Table 1 presents our main finding. The baseline (i.i.d. sampling) plateaus at k=32: doubling the budget to k=64 produces zero additional solved theorems. In contrast, the skeleton-guided approach continues to find new proofs at each budget level.

Condition	**Pass@16**	**Pass@32**	**Pass@64**
A-RL (i.i.d. baseline)	38/244 (15.6%)	42/244 (17.2%)	42/244 (17.2%)
B-RL (skeleton schedule)	**55/244 (22.5%)**	**58/244 (23.8%)**	**60/244 (24.6%)**
Absolute gap	+17	+16	+18
Relative improvement	+44.7%	+38.1%	+42.9%

Table 1: Scaling analysis on miniF2F-test with DeepSeek-Prover-V1.5-RL. The i.i.d. baseline plateaus at k=32 while the skeleton schedule continues to find new proofs. The gap is persistent across all budget levels.

The baseline's plateau from k=32 to k=64 is striking: 32 additional samples—each with different random seeds—fail to find a single new proof. This indicates that the model is trapped in a narrow region of the proof strategy space. Temperature sampling alone cannot break this mode collapse.

4.2 Diversity Ablation: Structure vs. Diversity

Table 2 compares four conditions at k=16 to isolate the source of improvement.

Condition	**Solved (/244)**	**Pass@16 (%)**
C2 (irrelevant comments)	25	10.2
A-RL (i.i.d. baseline)	38	15.6
C1 (instruction paraphrases)	38	15.6
B-RL (tactic skeletons)	**55**	**22.5**

Table 2: Diversity ablation at k=16. Tactic skeletons (B-RL) produce a clear improvement; instruction paraphrases (C1) yield the same topline pass@16 as A-RL; irrelevant comments (C2) actively **degrade** performance by 34%. The topline ordering C2 < A-RL = C1 $\ll$ B-RL suggests structural guidance drives the gains, but as Section 4.3 shows, the topline-equal C1 result hides a real redistribution rather than a true null effect.

The C2 result is interesting. We observe that irrelevant Lean comments injected before the theorem actually **degrade** performance, suggesting that surface-level prompt perturbations that lack tactic-level structure interfere with the model's collapsed policy. The C1 topline equality with A-RL is suggestive

but, by itself, ambiguous: it is consistent with both "C1 has no effect" and "C1 helps on some theorems while hurting an equal number on others." We disambiguate these in Section 4.3 by stratifying solves by empirical formalization difficulty.

4.3 Empirical Difficulty Stratification

Mathematical difficulty (e.g., IMO vs. textbook) is a poor proxy for the difficulty a neural prover actually faces in Lean, because the choice of formalization can **materially shift the proof difficulty independently of the underlying mathematics**. We therefore define an *empirical formalization-difficulty* score per theorem, grounded in what current ML provers can actually do on these specific Lean formalizations.

Difficulty score (leave-one-out). For each theorem τ and a target model M under analysis, we define $\sigma_{-M}(\tau)$ as the number of hint-free i.i.d. sampling baselines from *other* models that solve τ. This leave-one-out construction prevents the score from being contaminated by M's own sample runs, which also serve as the reference for the "NEW" analysis. For analyses of V1.5-RL, the four baselines used to compute $\sigma_{-\text{V1.5-RL}}$ are Goedel-Prover sample at $k\in\{16, 32\}$ and V2-7B sample at $k\in\{16, 32\}$. (An equivalent leave-one-out score is used for the V2 analysis in Section 4.5.) Theorems with $\sigma_{-M}=0$ are beyond the reach of every *other* baseline at every tested k; theorems with $\sigma_{-M}=N_{-M}$ (the maximum, 4 for V1.5-RL) are solved by every other baseline. The distribution under $\sigma_{-\text{V1.5-RL}}$ is reported in Table 3.

Bucket	**# theorems**
Frontier ($\sigma_{-\text{V1.5-RL}}=0$, no other baseline solves)	117
Hard ($\sigma_{-\text{V1.5-RL}}=1$)	24
Medium ($\sigma_{-\text{V1.5-RL}}=2$)	13
Easy ($\sigma_{-\text{V1.5-RL}}=3$)	53
Trivial ($\sigma_{-\text{V1.5-RL}}=4$, all other baselines solve)	37
Total	**244**

Table 3: Leave-one-out empirical difficulty distribution of miniF2F-test for V1.5-RL analyses. Score $\sigma_{-\text{V1.5-RL}}$ counts how many of the four non-V1.5-RL sample baselines (Goedel $\times 2$, V2 $\times 2$) solve the theorem. The frontier bucket (48%) collects theorems no other tested model reaches via i.i.d. sampling. The label *trivial* denotes "trivial for the field excluding V1.5-RL," which is precisely the population a mode-collapsed V1.5-RL should be expected to miss.

Where the perturbations operate. For each condition C, let NEW(C) denote the set of theorems solved by C but not by V1.5-RL sample at k=16,

and let LOST(C) denote the set solved by sample but not by C. Stratifying these sets by $\sigma_{-\text{V1.5-RL}}$ yields Table 4.

Condition	Frontier	Hard	Medium	Easy	Trivial	NET
Skeleton (B-RL)	0	+1	+1	**+9**	**+6**	**+17**
Paraphrase (C1)	0	0	0	+3	−3	0
Comment (C2)	0	+1	0	−6	−8	−13

Table 4: NET effect (NEW − LOST) of each V1.5-RL perturbation at k=16, stratified by $\sigma_{-\text{V1.5-RL}}$. The skeleton intervention's +17 net is dominated by recoveries in the *easy* (+9) and *trivial* (+6) buckets—theorems that other open-source provers routinely solve via sample but that V1.5-RL sample misses, exactly the population we would expect a mode-collapsed model to fail on. Paraphrase (C1) shows only a small redistribution between easy and trivial; comment (C2) shows uniform degradation across easy and trivial. **No perturbation moves any theorem out of the $\sigma_{-\text{V1.5-RL}}$=0 frontier.**

What the stratification reveals. The skeleton intervention specifically remediates V1.5-RL's gap to the field: 15 of its 17 net wins fall in the easy (+9) and trivial (+6) buckets, i.e., theorems that 3–4 of the four other-model baselines solve without intervention. This is the operational signature of mode collapse—V1.5-RL fails on problems other RL/SFT provers solve routinely, and skeletons recover precisely those problems. C1 (paraphrase) reshuffles within easy/trivial without net gain; C2 (comment) destroys easy/trivial wins uniformly. The structural-content gradient is sharp in the easy+trivial column: skeleton +15, paraphrase 0, comment −14. Replicating at k=32 and k=64 on V1.5-RL: 0 frontier theorems are unlocked at any tested budget, by any perturbation—a portion of these likely reflect known formalization issues in miniF2F (missing hypotheses on specific competition problems) rather than pure capability limits.

Decomposing the skeleton win. The k=16 schedule includes one wrap attempt (attempt 15) where the tactic prefix is empty but a natural-language goal-hint comment (e.g., `/-- Hint: Start by simplifying using simp. --/`) is prepended. We count V1.5-RL structured wins by attempt class (Table 5).

The 0/16 NEW count for attempt-0 confirms V1.5-RL is effectively deterministic at our inference settings—the +17 topline cannot be attributed to vLLM sampling variance. The structural intervention works through two delivery layers: natural-language hint comment (+8 NEW) and literal tactic prefix (+12 NEW), both contributing additively to the +20 total.

Dedicated NL-only ablation (C3). The attempt-class decomposition pins the NL hint contribution to a single slot of the composite schedule, leaving open whether the effect generalizes when the entire budget is spent on NL comments. We therefore ran a dedicated condition C3 in which all 16 attempts

Attempt class	Wins	NEW vs. sample
Attempt 0 (truly empty: no comment, no tactic prefix)	16	0
Attempt 15 (empty tactic + goal-hint comment only)	9	8
Hinted skeletons (tactic prefix injection, attempts 1–14)	30	12
Total	**55**	**20**

Table 5: V1.5-RL structured k=16 winners decomposed by attempt class. "Attempt 0" is identical to V1.5-RL sample at seed 1; 0/16 are new, confirming V1.5-RL is fully deterministic at our inference settings. The empty-tactic-plus-hint-comment attempt produces +8 new solves (a single line such as *"Start by simplifying with `simp`"* unlocks 8 theorems that 16 sample seeds miss). The literal tactic prefixes contribute +12 new solves, dominated by `intros` (+6) and `norm_num` (+3). The +17 topline result decomposes to +20 new −3 lost.

on V1.5-RL use empty tactic prefixes and cycle through the 7 distinct natural-language goal-hint comments (Appendix A), with no tactic stem injected at any attempt. C3 solves 48/244 (19.7%)—intermediate between sample-only A (15.6%) and structured-skeleton B (22.5%). The clean ordering $A < C3 < B$ confirms the structural-content gradient: NL hints alone recover roughly 58% of the A→B gap (+10 solves of the +17 B-A topline), and tactic stems contribute the orthogonal remainder (+7). Skeleton guidance is therefore not reducible to either modality alone—both layers contribute, and the two effects compose.

4.4 Base vs. RL: The Origin of Mode Collapse

Condition	Pass@16	Pass@32	Pass@64
A-BASE (i.i.d.)	0/244 (0.0%)	0/244 (0.0%)	0/244 (0.0%)
B-BASE (skeletons)	0/244 (0.0%)	0/244 (0.0%)	0/244 (0.0%)
A-RL (i.i.d.)	38/244 (15.6%)	42/244 (17.2%)	42/244 (17.2%)
B-RL (skeletons)	55/244 (22.5%)	58/244 (23.8%)	60/244 (24.6%)

Table 6: Base model vs. RL model on miniF2F-test. The base model proves zero theorems across all conditions and budgets—inspection of generated outputs shows it almost always produces `sorry` or echoes the theorem statement back, indicating it has not learned to generate valid Lean 4 proofs at all. Mode collapse is therefore an RL-specific phenomenon: RL is what creates proof capability in the first place, and our intervention recovers proofs that the RL-collapsed model fails to find on its own.

The base model result is unambiguous and somewhat surprising: **DeepSeek-Prover-V1.5-Base proves zero theorems on miniF2F-test**, regardless of sampling budget or whether tactic skeletons are provided. Inspection of the generated outputs across all 244 theorems and all 64 attempts reveals that the

base model produces `sorry` in 58.8% of its outputs, or echoes the theorem statement verbatim without attempting a proof. By contrast, the RL model almost never generates explicit `sorry` tokens (<0.2% under i.i.d. sampling); its mode collapse manifests instead as persistent generation of failing but syntactically valid tactic sequences—repeated `rw` or `apply` attempts that do not close the goal.

This reframes our hypothesis. We had originally predicted that the base model would show a *smaller* skeleton-induced improvement, providing graded evidence that RL exacerbates collapse. Instead, the base model shows no improvement because it has *no proof capability to collapse*: the RL fine-tuning stage is what teaches the model to generate valid Lean 4 tactic proofs in the first place. Skeletons cannot substitute for this capability, nor goad out theorem proving knowledge where it is non-existent.

The implication is sharper than an earlier framing we posited: mode collapse is not merely *exacerbated* by RL training, it is *caused by* it. RL creates a useful but narrow proof policy; without the structural intervention provided by tactic skeletons, the model cannot escape the policy learned at training time, even when it may benefit. The base model demonstrates that the underlying language model has no useful tactic-level prior: the RL stage is the locus of both proof capability and the inference-time collapse that limits it.

4.5 Generalization Across Models

The V1.5 RL-vs-Base contrast in Section 4.4 pins mode collapse to the RL stage *within* one model family. To test whether the same A-vs-B pattern generalizes across architectures and training pipelines, we extend the pass@16 comparison to two additional 7B Lean provers: a further RL-trained model (DeepSeek-Prover-V2-7B) and one SFT-only model (Goedel-Prover).

Model	**Post-Training**	**A pass@16**	**B pass@16**	**Δ**
V1.5-Base	none (pre-train only)	0/244 (0.0%)	0/244 (0.0%)	0
Goedel-Prover	SFT	90/244 (36.9%)	85/244 (34.8%)	−5
V1.5-RL	RL	38/244 (15.6%)	**55/244 (22.5%)**	+17
V2-7B	RL	126/244 (51.6%)	**127/244 (52.0%)**	+1

Table 7: Cross-model A vs. B comparison at k=16 on miniF2F-test. The aggregate Δ varies by model: V1.5-RL shows the largest lift (+17), V2-7B is flat on aggregate (+1) but solves +3 frontier theorems no i.i.d. baseline reaches (Section 4.5), and SFT-trained Goedel-Prover shows −5. Together with V1.5-Base (which has no proof capability with or without hints), the pattern is consistent with the hypothesis that structured hints remedy the underexploration induced by RL fine-tuning; the magnitude of remediation varies with the post-training recipe.

The Goedel-Prover null is diagnostic, not contradictory. Goedel-Prover-SFT achieves the highest absolute pass@16 of any open-source prover in our study (36.9% seed=1; mean 97.7 ± 6.7 theorems across n=3 seeds, Section 4.6) using purely supervised fine-tuning on a large synthetic Lean corpus, with no RL stage. Despite this capability advantage over V1.5-RL, structured hints do not improve its performance—they cost it -10.0 ± 4.4 theorems on average across the three seeds, with the sign preserved in every seed. Goedel also shows the same k-scaling plateau we observed for V1.5-RL: pass@16 and pass@32 are identical at 36.9% for the i.i.d. baseline, and the structured condition scales similarly (34.8% $\rightarrow$ 36.1%). The *plateau itself* thus appears to be a general property of i.i.d. sampling from these models, but the *remediability of the plateau via structured hints* is specific to RL-trained models.

A plausible interpretation. SFT and RL induce different modes of under-exploration. SFT pattern-matches a model onto a fixed distribution of training proofs; mode collapse for such a model manifests as tactic-local exploration that already starts from a broad set of stems—skeletons add nothing because the model would have tried those stems anyway. RL reward-shapes a model onto a narrower set of high-reward strategies and abandons alternative tactic stems entirely; skeletons help because they force commencement from stems the RL policy has been trained away from. The asymmetry we observe is consistent with this account, but the magnitude varies with the RL training pipeline. V1.5-RL shows the cleanest aggregate gain (+17); V2-7B is flat on the aggregate but contributes +3 frontier solves that no i.i.d. baseline of any model in our matrix reaches—a structurally meaningful gain that the topline metric hides. Goedel-Prover (SFT-trained) shows a small negative aggregate. We read these together as evidence that RL-trained provers are reachable through structural intervention in a way SFT-only provers are not, while acknowledging that the size of the gain is contingent on the specifics of the post-training recipe and likely on how much capability the i.i.d. baseline has already saturated.

V2 frontier solves. The +1 aggregate hides the most theoretically interesting outcome in the cross-model matrix. We compute $\sigma_{-\mathrm{V2}}(\tau)$ on the same leave-one-out construction as Section 3.2, dropping all V2 baselines from the difficulty score. Three theorems with $\sigma_{-\mathrm{V2}}$=0—`imo_1960_p2`, `imo_1962_p2`, and `numbertheory_x5neqy2p4`—are solved by V2 structured at k=16 but by none of the other six sample baselines in our matrix (V1.5-RL, Goedel, V2-sample, at multiple k). Two are IMO problems. V2 structured is therefore the first condition in our matrix that opens up the σ=0 frontier, even though its aggregate is flat. We refrain from inferring a general pattern from n=3, but flag this as evidence that the aggregate Δ metric is a lossy summary of where structural guidance actually pays off, and as a concrete falsifier for the strong version of our hypothesis: if frontier theorems were beyond reach of all current models in this size class, V2 structured should not have unlocked any.

Solved-set overlap. A paired view of A-RL vs. B-RL at k=16 further constrains the "more compute would suffice" objection: of the 38 theorems solved by A-RL, B-RL solves 35 of them and adds 20 new ones (only 3 A-RL solves are lost, consistent with stochastic variation in shard scheduling). The intervention is therefore near-Pareto-dominant on this benchmark and accesses a *different region* of the proof space rather than reproducing the baseline's hits with a different prompt—the 20 unique solves are concentrated in `mathd_algebra` and `mathd_numbertheory` problems that no i.i.d. sampling at this budget reaches.

4.6 Seed Variance and Robustness of the Asymmetry

The single-seed numbers above could in principle be artifacts of stochastic sampling. To check this, we replicate the A and B conditions at k=16 across n=3 seeds (seed $\in \{1, 2, 3\}$, temperature 0.6) on both V1.5-RL and Goedel-Prover-SFT—the two models that anchor the RL-vs-SFT asymmetry claim. Table 8 reports per-seed solve counts and aggregate statistics.

	V1.5-RL (RL)			**Goedel-Prover (SFT)**		
Seed	A	B	Δ	A	B	Δ
1	38	55	+17	90	85	−5
2	35	44	+9	102	90	−12
3	37	48	+11	101	88	−13
mean ± std	36.7 ± 1.5	49.0 ± 5.6	$\mathbf{+12.3 \pm 4.2}$	97.7 ± 6.7	87.7 ± 2.5	$\mathbf{-10.0 \pm 4.4}$

Table 8: Seed-variance study at k=16 on V1.5-RL (RL-trained) and Goedel-Prover-SFT. The asymmetry is preserved in every seed: $\Delta > 0$ in 3/3 V1.5-RL seeds and $\Delta < 0$ in 3/3 Goedel seeds. The means separate by $+12.3-(-10.0) = 22.3$ theorems with combined std $\sqrt{4.2^2 + 4.4^2} \approx 6.1$, a $\sim 3.7\sigma$ separation. The within-model variance structure also flips: on the RL-trained model A is near-deterministic (± 1.5) while B varies (± 5.6); on the SFT-trained model the pattern reverses (A: ± 6.7, B: ± 2.5).

Three observations. First, the sign of Δ is preserved in every one of the six (model, seed) cells—3/3 positive on V1.5-RL, 3/3 negative on Goedel—so the cross-model asymmetry is not an artifact of any individual seed. Second, the magnitudes are tighter than the single-seed reports: $+12.3 \pm 4.2$ on V1.5-RL (versus the seed=1 headline +17) and -10.0 ± 4.4 on Goedel (versus seed=1's −5). The means of the two Δ distributions separate by roughly 3.7 standard deviations of the combined variance, so a paired CI on the asymmetry-of-asymmetries is well clear of zero. Third, the within-model variance structure differs between RL and SFT: V1.5-RL's A is near-deterministic (± 1.5) while its B varies meaningfully (± 5.6), whereas Goedel's pattern is reversed (A: ± 6.7, B: ± 2.5). The RL-trained model collapses onto a near-deterministic single proof attempt on a fixed prompt and only acquires variance when the prompt itself varies; the SFT-trained model retains generation diversity on a fixed prompt and *converges*

when the prompt schedule constrains its choices. This is a second, distinct empirical signature of the RL-vs-SFT distinction on top of the Δ-sign asymmetry, consistent with the broader account in Section 4.5.

4.7 Direct Measurement of First-Tactic Collapse

The plateau in Section 4.1 is behavioural evidence of mode collapse. We now operationalize the phenomenon as a direct distributional measurement on the same per-attempt logs. For each miniF2F theorem we collect the 64 raw outputs produced by V1.5-RL i.i.d. sampling at k=64, extract the *first tactic head* (the leading identifier on the first line of the proof body), and count its distinct values across the 64 samples per theorem.

Unique first-tactic heads / theorem (out of 64 samples)	**theorems**
1 (deterministic strategic opening)	120 (49.2%)
2	69 (28.3%)
3	34 (13.9%)
4	12 (4.9%)
5–7	9 (3.7%)
median / mean / max	2 / 1.9 / 7

Table 9: First-tactic-head diversity of V1.5-RL i.i.d. sampling at k=64. The median theorem receives only 2 distinct strategic openings across 64 stochastic samples, and *nearly half the benchmark* (49.2%) receives a deterministic single opening from this RL-trained model. The mean first-tactic-head Shannon entropy is 0.48 bits per theorem (median 0.27), compared with $\log_2 64 = 6.0$ bits achievable under uniform sampling—i.e., observed entropy is $\leq 8\%$ of capacity.

This converts "mode collapse" from a behavioural metaphor (a plateau in pass@k) into a direct distributional statement: V1.5-RL's first-tactic distribution under i.i.d. sampling at temperature 0.6 is concentrated to a median of 2 heads and a modal 1 head per theorem. Aggregated across all 244 theorems and 13,159 samples, three tactic heads—`have` (37.2%), `rw` (14.1%), `norm_num` (14.0%)—account for 65.3% of every strategic opening V1.5-RL ever takes. The structured schedule guarantees 15 distinct first-tactic heads by construction, so it provides roughly 7.5× the first-tactic-head budget at the same compute spend.

5 Discussion

Mode collapse as the central finding. Our results reframe the narrative from "tactic skeletons improve performance" to "RL-trained provers severely underexplore the tactic space at inference time." The skeleton schedule is a *probe* that reveals this underexploration, not a novel proving architecture. The

baseline's plateau at k=32—where 32 additional stochastic samples find zero new proofs—is the most striking evidence of this phenomenon.

Implications for inference-time scaling. The mode collapse we observe has direct implications for the common practice of scaling inference compute. If i.i.d. sampling has diminishing returns far earlier than assumed, then simply increasing k is wasteful. Our results suggest that *structural diversity*—varying the proof strategy, not just the random seed—is a more efficient axis for scaling inference compute. This is complementary to tree-based methods like HyperTree [7], which achieve diversity through explicit search rather than prompt perturbation.

Structural content as a gradient. The bucket-stratified ablation (Section 4.3, Table 4) refines the structural-vs-diversity dichotomy into a continuous gradient. The three perturbations differ in how much tactic-level structural content they carry: skeleton (literal Lean tactic prefix or a comment naming a tactic), paraphrase (paraphrased general-purpose instruction without tactic specifics), comment (a content-free token marker). The combined easy+trivial NET tracks this gradient cleanly—$+15/0/-14$ for skeleton, paraphrase, and comment respectively—because that is where V1.5-RL's gap to other open-source provers concentrates. Equally informative is the within-skeleton decomposition (Table 5) and the dedicated C3 ablation: a natural-language tactic-suggestion comment alone produces +8 new solves at the attempt-level breakdown and 48/244 (vs. 38 baseline) when run as a standalone 16-attempt schedule, while literal Lean tactic prefixes contribute the remaining +12 NEW. Both are forms of tactic-stem-level structural guidance delivered at different abstraction layers, and both contribute additively. We caveat that two of the 15 skeletons (`aesop`, `norm_num`) are closer to one-shot decision procedures than to genuine "stems"; part of the structural gain may therefore be attributable to V1.5-RL underusing Lean's powerful built-in solvers rather than to a literal first-tactic distributional shift, and disentangling these two contributions is a natural follow-up.

RL as both cause and prerequisite. The base model result imposes an unexpected constraint on any account of mode collapse in this domain. Because DeepSeek-Prover-V1.5-Base proves zero theorems regardless of skeletons, mode collapse cannot be attributed to a pre-existing limitation of the underlying language model that RL merely amplifies. Instead, RL plays a dual role: it is the stage at which proof capability is created (the base model has none), and it is also the stage at which that capability is collapsed onto a narrow policy. Any future intervention—whether at training time (e.g., entropy-regularized RL) or inference time (e.g., learned skeleton retrieval)—must navigate this trade-off rather than treat collapse as a separable problem.

Limitations. Our primary controlled comparison (Section 4.4) uses two variants of a single model architecture, DeepSeek-Prover-V1.5 (7B); V1.5 is the only widely benchmarked RL-trained Lean prover that also releases a non-RL base checkpoint. The cross-model study (Section 4.5) extends the pass@16 comparison to V2-7B and Goedel-Prover, but cannot replicate the within-architecture RL-vs-no-RL contrast on those models because they do not release matched non-RL/non-SFT base checkpoints. A natural third RL replication—Kimina-Prover [15]—was attempted but required engineering work on the output extractor for its reasoning-mode emission (it produces unfenced `import Mathlib` + a nested theorem signature inside the response that the standard extractor rejected with an `unexpected token 'import'` syntax error). A partial postfix run produced 20 proved out of 35 sampled trials (57%) before being cut for compute reasons; the full pass@16 number, and its corresponding B condition, are deferred to future work. Cross-model k-scaling beyond k=16 is currently established on V1.5-RL ($k \in \{16, 32, 64\}$) and Goedel-Prover ($k \in \{16, 32\}$); extending V2 to $k \geq 32$ would strengthen the plateau characterization for the additional RL prover, and is a natural next step. Evaluation on additional benchmarks (ProofNet, PutnamBench) would further strengthen generality. We do not explore constrained decoding beyond the first tactic, or interaction with tree-based proof search methods—both promising directions for future work. A learned per-theorem skeleton retriever, or an expanded tactic set beyond our 15×8 grid, could further improve absolute performance; we leave both directions to future work.

The first-tactic-distribution measurement in Section 4.7 is currently reported on V1.5-RL only; the same analysis on the other RL-trained prover in our matrix (V2-7B) and on the SFT-trained Goedel would let us test whether first-tactic-head concentration tracks the RL-vs-SFT asymmetry we observe at the solve-rate level (Section 4.6 already shows that the variance structure of solve counts differs by post-training recipe, so we expect first-tactic distributions to differ similarly).

6 Conclusion

We have used a deterministic 15-skeleton schedule as a probe to diagnose mode collapse in RL-trained Lean provers, and shown that the collapse it reveals is RL-specific in two distinct senses: it is absent in the non-RL V1.5-Base variant (which has no proof capability at all), and it is partially remediated by structural guidance only on RL-trained models in our cross-model matrix—never on the SFT-trained Goedel-Prover. The magnitude of the remediation varies sharply by post-training recipe (+17 on V1.5-RL; +1 aggregate but +3 frontier solves on V2-7B; −5 on Goedel-Prover-SFT), which we read as evidence that the underlying phenomenon is real but pipeline-contingent rather than universal. The most natural follow-ups are (i) converting the diagnostic from a sampling-plateau metaphor into a direct first-tactic-distribution measurement on the existing logs; (ii) replacing the hand-designed 15-skeleton schedule with

a learned, per-theorem skeleton retriever, which the gradient in Section 4.3 suggests should outperform the fixed grid; and (iii) closing the loop at training time via entropy-regularized or skeleton-conditioned RL, since the V1.5-Base control implies that any training-time intervention will be constrained by the dual role RL plays in creating and collapsing the proof policy.

A Skeleton Schedule (Tactics × Goal Hints)

The structured-skeleton schedule is a deterministic $15 \times 8 = 120$ grid: 15 tactic skeletons crossed with 8 goal-hint comments (one of which is empty). At budget k, the schedule advances by tactic-index first and by hint-index second: attempts $0 \ldots 14$ use the 15 distinct skeletons paired with hint index 0 (empty); attempts $15 \ldots 29$ repeat the skeletons paired with hint index 1; and so on. Each of the 120 attempts is a *distinct* (skeleton, hint) prompt configuration.

Tactic skeletons (15 total). The Python-source ordering used by the perturbation function is reproduced below; index i in this list is `TACTIC_SKELETONS[`i`]` in the schedule.

0. (empty) — no tactic prefix
1. `simp`
2. `intro`
3. `intros`
4. `constructor`
5. `refine ?_`
6. `refine ⟨?_, ?_⟩`
7. `aesop`
8. `norm_num`
9. `linarith`
10. `nlinarith`
11. `ring`
12. `ring_nf`
13. `simp` followed by `try aesop`
14. `simp` followed by `try nlinarith`

Goal-hint comments (8 total). Goal hints are short natural-language tactic suggestions injected as Lean comments (`/-- Hint: <hint text> --/`) before the theorem statement. Index j corresponds to `GOAL_HINTS[`j`]`; the hint at index 0 is empty (no comment is prepended).

0. (empty) — no goal-hint comment
1. "Start by simplifying the goal and hypotheses using `simp`."
2. "If the goal is an implication or forall, introduce variables."
3. "If the goal is a conjunction or existence, build it using `constructor` or `refine`."
4. "If arithmetic is involved, try `norm_num`, then `linarith` or `nlinarith`."
5. "If the goal looks routine, try `aesop` after simplification."
6. "If the proof requires rewriting, look for a lemma in the context and rewrite."
7. "If the goal involves recursion on naturals, consider induction."

Pairing rule. Attempt i in the schedule uses tactic index i mod 15 and hint index $\lfloor i/15 \rfloor$ mod 8. Consequently, at k=16 the only repeated tactic-index is the wraparound to tactic 0 (empty) at attempt 15, now paired with hint 1 ("Start by simplifying…"); this is the empty-tactic + hint-comment configuration analyzed in Table 5.

B Diversity Ablation Prompts

B.1 C1: Instruction Paraphrases

Each paraphrase is injected as a Lean block comment before the standard header. The 16 variants are:

1. "Prove the following theorem in Lean 4:"
2. "Complete this Lean 4 proof:"
3. "Find a formal proof for the following:"
4. "Show that the following statement holds:"
5. "Write a tactic proof for this theorem:"
6. "Construct a formal proof of the following:"
7. "Provide a Lean 4 proof for:"
8. "Demonstrate the following result formally:"

9. “Give a complete tactic proof:”
10. “Prove this result using Lean 4 tactics:”
11. “Formalize a proof of the following theorem:”
12. “Establish the following in Lean 4:”
13. “Derive a proof for the following statement:”
14. “Supply a formal tactic proof for:”
15. “Verify the following theorem in Lean 4:”
16. “Prove the following:”

B.2 C2: Irrelevant Comment Prefixes

Each comment is prepended before the theorem statement:

1. `/- approach alpha -/`
2. `/- strategy beta -/`
3. `/- method gamma -/`
4. `/- path delta -/`
5. `/- route epsilon -/`
6. `/- attempt zeta -/`
7. `/- angle eta -/`
8. `/- direction theta -/`
9. `/- variant iota -/`
10. `/- form kappa -/`
11. `/- mode lambda -/`
12. `/- plan mu -/`
13. `/- way nu -/`
14. `/- style xi -/`
15. `/- view omicron -/`
16. `/- take pi -/`

C Environment

- **Lean:** v4.9.0-rc1 (matching DeepSeek-Prover-V1.5 training environment)
- **Mathlib:** commit `7fa489a5cbf3c4f08d36e1e0b5dee4d761fdbd9b`
- **Models:**
  - `deepseek-ai/DeepSeek-Prover-V1.5-RL`, `deepseek-ai/DeepSeek-Prover-V1.5` (V1.5-Base)
  - `deepseek-ai/DeepSeek-Prover-V2-7B`
  - `Goedel-LM/Goedel-Prover-SFT`
- **Inference:** vLLM, single A100 per run. V1.5 ran on v0.18 / A100 80GB; V2 and Goedel ran on v0.10.2 / A100 40GB with `--enforce-eager` (sufficient in both cases).
- **Decoding (V1.5, Goedel; completion mode):** temperature 0.6, top-p 0.95, max 1024 tokens per attempt; no chat template.
- **Decoding (V2; reasoning mode):** temperature 0.6, top-p 0.95, max 8192 tokens per attempt; apply model chat template; extract last ```` ```lean4 ```` fenced block from response.
- **Verification:** `lake env lean --json`, 120s timeout, `sorry` rejection.

References

[1] Achim, T., et al. Aristotle: Imo-level automated theorem proving. *arXiv preprint arXiv:2510.01346* (2025).

[2] Chen, L., Gu, J., Huang, L., et al. Seed-prover: Deep and broad reasoning for automated theorem proving. *arXiv preprint arXiv:2507.23726* (2025).

[3] de Moura, L., and Ullrich, S. The Lean 4 theorem prover and programming language. In *Automated Deduction – CADE 28* (2021), Springer, pp. 625–635.

[4] Han, J. M., Rute, J., Wu, Y., Ayers, E., and Polu, S. Proof artifact co-training for theorem proving with language models. *International Conference on Learning Representations (ICLR)* (2022).

[5] Jiang, A. Q., Welleck, S., Zhou, J. P., Li, W., Liu, J., Jamnik, M., Lacroix, T., Wu, Y., and Lample, G. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *International Conference on Learning Representations (ICLR)* (2023).

[6] Kirk, R., Mediratta, I., Nalmpantis, C., Luketina, J., Hambro, E., Grefenstette, E., and Raileanu, R. Understanding the effects of RLHF on LLM generalisation and diversity. In *International Conference on Learning Representations (ICLR)* (2024). arXiv:2310.06452.

[7] Lample, G., Lacroix, T., Lachaux, M.-A., Rodriguez, A., Roziere, B., and Szafraniec, M. HyperTree proof search for neural theorem proving. *Advances in Neural Information Processing Systems 35* (2022), 26337–26349.

[8] Lin, Y., Tang, S., Lyu, B., et al. Goedel-prover: A frontier model for open-source automated theorem proving. *arXiv preprint arXiv:2502.07640* (2025).

[9] Polu, S., and Sutskever, I. Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393* (2020).

[10] Ren, Z., Shao, Z., Song, J., et al. DeepSeek-Prover-V2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint arXiv:2504.21801* (2025).

[11] Sivakumar, J. A., Borchert, P., Cardenas, R., et al. Conjecturing: An overlooked step in formal mathematical reasoning. *arXiv preprint arXiv:2510.11986* (2025).

[12] Thakur, A., Tsoukalas, G., Wen, Y., Xin, J., and Chaudhuri, S. COPRA: In-context learning for proving theorems with llms. *arXiv preprint arXiv:2310.04353* (2023).

[13] The mathlib Community. The Lean mathematical library. *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs* (2020), 367–381.

[14] Trinh, T., Wu, Y., Le, Q. V., He, H., and Luong, T. Solving olympiad geometry without human demonstrations. *Nature 625*, 7995 (2024), 476–482.

[15] Wang, H., Unsal, M., Lin, X., et al. Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.11354* (2025).

[16] Wu, F., Xuan, W., Lu, X., Liu, M., Dong, Y., Harchaoui, Z., and Choi, Y. The invisible leash: Why RLVR may or may not escape its origin. *arXiv preprint arXiv:2507.14843* (2025).

[17] Xin, H., et al. Deepseek-prover-v1.5: Harnessing proof assistant feedback for reinforcement learning and monte-carlo tree search. *arXiv preprint arXiv:2408.08152* (2024).

[18] Yang, K., Swope, A., Gu, A., Chalamala, R., Song, P., Yu, S., Godil, S., Prenger, R., and Anandkumar, A. Leandojo: Theorem proving with retrieval-augmented language models. *Advances in Neural Information Processing Systems 36* (2023), 10065–10081.

[19] Yue, Y., Chen, Z., Lu, R., Zhao, A., Wang, Z., Yue, Y., Song, S., and Huang, G. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? *arXiv preprint arXiv:2504.13837* (2025). ICML 2025 Workshop AI4Math Best Paper; NeurIPS 2025 Best Paper Runner-Up.

[20] Zheng, K., Han, J. M., and Polu, S. miniF2F: a cross-system benchmark for formal Olympiad-level mathematics. *International Conference on Learning Representations (ICLR)* (2022).