How Deep Can LLMs Learn to Reason? Expressiveness Is Key

Anonymous authors
Paper under double-blind review

Abstract

Reinforcement learning (RL) has been applied to improve large language model (LLM) reasoning, yet the systematic study of how training scales with task difficulty has been hampered by the lack of controlled, scalable environments. Observed LLM shortcomings in long-horizon reasoning have raised the prospect that they are fundamental to the autoregressive transformer architecture. To address this, we introduce ScaleLogic, a synthetic logical reasoning framework that offers independent control over two axes of difficulty: the depth of the required proof planning (i.e., the *horizon*) and the expressiveness of the underlying formal system. Our proposed framework supports a wide range of logics: from simple implication-only logic ("if-then") towards more expressive first-order reasoning with conjunction ("and"), disjunction ("or"), negation ("not"), and universal quantification ("for all"). Using this framework, we show that the RL training compute T follows a power law with respect to reasoning depth D ($T \propto D^{\gamma}$, $R^2 > 0.99$), and the scaling exponent γ increases monotonically with expressiveness, from 1.05 to 2.60. On downstream mathematics and general reasoning benchmarks, more expressive training settings yield both larger performance gains (up to $+10.66$ points) and more compute-efficient transfer compared to less expressive settings, demonstrating that *what* a model is trained on, not just *how much* it is trained, shapes downstream transfer. We further show that the power-law relationship holds across multiple RL methods, and curriculum-based training substantially improves scaling efficiency. More broadly, our results suggest that LLM shortcomings in long-horizon reasoning are not fundamental to the underlying architecture, and could be addressed by improved training methodology and data.

1 Introduction

Recent advances in reinforcement learning (RL) for large language model (LLM) reasoning have shown that training on verifiable domains such as mathematics and coding can elicit enhanced reasoning behavior and improve benchmark performance (Guo et al., 2025a; Jaech et al., 2024). However, these gains do not yet translate into robust long-horizon[1] reasoning, as the same model's performance can degrade sharply once tasks require many sequential reasoning steps, even when the corresponding subproblems are individually manageable (Rameshkumar et al., 2025; Motwani et al., 2025; Lu et al., 2025; Zhou et al., 2025). This has led to the suggestion that limitations in long-horizon reasoning may be fundamental to the autoregressive transformer architecture (Saparov et al., 2025; Bachmann & Nagarajan, 2024), raising a central question about the feasibility of extending a model's reasoning horizon through reinforcement learning:

As reasoning depth increases, can RL keep extending a model's reasoning horizon, or does training eventually become ineffective? If it can, how does the required training effort scale with depth?

Answering this question requires a training environment with three properties simultaneously: exact verifiability, fine-grained control over reasoning difficulty, and data available at scale. The domains currently driving RL progress, such as mathematics and coding, often provide reliable verification, but high-quality problems are expensive to curate and offer only limited control over reasoning horizon and task structure (Hendrycks et al., 2021; Jain et al., 2024). Existing alternatives, including

[1]Throughout, we use "horizon" in the sense of the reasoning horizon, i.e., the number of sequential inference steps composed before the final answer, following the usage in prior work (Motwani et al., 2025; Lu et al., 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: **Comparison of training data sources for RL post-training of LLM reasoning.** Here, × denotes no direct control, ✓ denotes direct support, and ○ denotes coarse or indirect control. E.g., the logic task SAT can vary the number of variables and clauses, but only indirectly controls the reasoning horizon.

Data Source	Verifiable	Scalable	Controllable Horizon	Controllable Expressiveness
Math / Code (e.g., Hendrycks et al., 2021)	✓	✗	✗	✗
Game-based (e.g., Wu et al., 2024)	○	✓	✗	✗
Self-Evolving (e.g., Zhao et al., 2025)	✓	✓	✗	✗
Logic Puzzles (e.g., Liu et al., 2025a)	✓	✓	○	✗
Benchmark Suites (e.g., Liu et al., 2025b)	✓	✓	○	✗
SCALELOGIC	✓	✓	✓	✓

synthetic tasks (Lan et al., 2025; Xie et al., 2025; Liu et al., 2025a) and self-evolving pipelines (Zhao et al., 2025; Huang et al., 2025), reduce the cost of data collection through automatic generation and verification, but often do not provide a sufficiently clean and interpretable set of control axes for analyzing scaling behavior. Together, these limitations make it difficult to systematically characterize how reasoning complexity shapes RL scaling (Table 1).

To address this gap, we propose SCALELOGIC, a synthetic logical reasoning environment with explicit control over two key dimensions of difficulty: the depth of the required proof planning (i.e., the *horizon*) and the logical expressiveness[2] of the reasoning problems. As illustrated in Figure 1, in each example, the model is given a set of facts and a handful of candidate conclusions. The model is then asked to identify which candidate conclusion is logically derivable from those facts, yielding an exact and easily verifiable multiple-choice objective. To find the correct multiple-choice option, the model must search for a proof of each candidate conclusion. Our environment requires it to expend comparable reasoning effort to determine the provability of each candidate, which substantially limits the model's ability to exploit simple heuristics or shortcuts during reasoning. Beyond scaling the depth of the proof tree, our environment allows us to vary the expressive structure of the underlying formal system, including the logical operators and structural properties involved in each example. Combined with automatic low-cost generation and verification, this provides a controlled testbed for systematically studying RL scaling in long-horizon reasoning.

Experiments in this controlled environment show that over the observed depth range, the RL training steps T required to reach 90% validation accuracy follow a power law in the proof-tree depth D ($T \propto D^{\gamma}$, $R^2 > 0.99$), where the scaling exponent γ increases monotonically with expressiveness from 1.05 to 2.60. This indicates that more logically expressive settings require disproportionately more training to solve problems of the same depth. We also find that the power-law relationship holds across different RL methods and models, and these dynamics depend strongly on the training distribution: a carefully designed curriculum improves training efficiency and stability.

We also demonstrate that training a model in our synthetic reasoning environment improves performance on downstream real-world mathematics and general reasoning benchmarks, with the most expressive setting improving mean accuracy across eight benchmarks by up to $+10.66$ percentage points over the base model. Critically, training on highly expressive data yields a monotonically increasing downstream performance curve, while less expressive settings plateau early. This indicates that *what* the model is trained on, not just *how much* it is trained, shapes downstream transfer. Taken together, these results suggest that the expressiveness of the training data influences both RL scaling on synthetic tasks and transferable gains in reasoning performance for real-world applications. More broadly, LLM shortcomings in long-horizon reasoning could be surmountable through improved training methodology and data.

In summary, we introduce a scalable synthetic reasoning framework for studying and improving long-horizon reasoning in RL post-training. Our main contributions are as follows:

- We propose SCALELOGIC, a controlled and scalable synthetic logical reasoning testbed for RL post-training, with exact verifiability, low-cost automatic generation, and explicit control over reasoning horizon and expressiveness.
- We show that RL training effort follows a power law in reasoning depth, with the scaling exponent γ increasing monotonically with expressiveness. The power-law relationship also holds across multiple RL methods and models, while curriculum training improves scaling efficiency.

[2]In this work, we use *expressiveness* to refer to the set of proofs that can be expressed using a progressively larger set of inference rules, each associated with a logical operator.

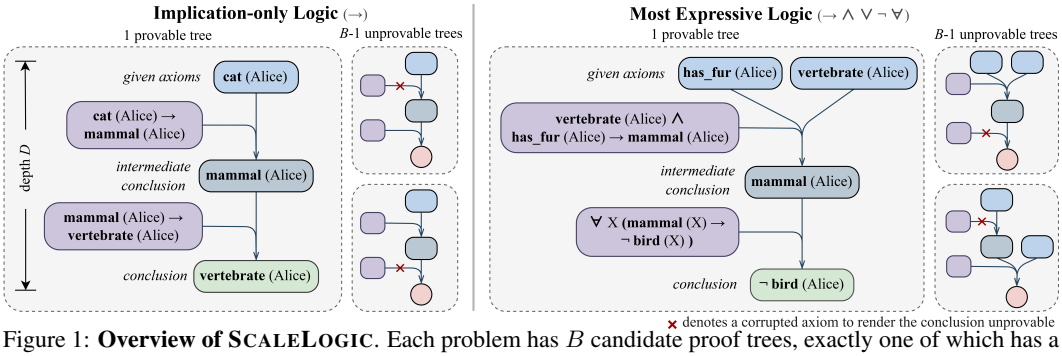


Figure 1: **Overview of SCALELOGIC.** Each problem has B candidate proof trees, exactly one of which has a provable conclusion; the others are made unprovable by corrupting one axiom. The depth D controls proof depth. **Left:** Implication-only reasoning. **Right:** The most expressive logic setting (referred to as + *Quantification* in Section 3.2) combines conjunction, disjunction, negation, and universal quantification.

- We show that synthetic reasoning training improves downstream reasoning performance by up to $+10.66$ points, and these gains depend strongly on the expressiveness of the training environment, with more expressive settings yielding larger and more compute-efficient transfer.

2 RELATED WORK

Long-horizon Reasoning Limitations. Recent work reveals that LLMs often exhibit sharp performance degradation as the reasoning horizon increases. Rameshkumar et al. (2025) show that reasoning models can perform well on graph-based reasoning tasks within a limited complexity regime, but their performance drops abruptly once the reasoning horizon is sufficiently large. SeqBench (Ramezanali et al., 2025) and GSM-Infinite (Zhou et al., 2025) also report steep performance degradation with exponential and sigmoid-like decay patterns. Saparov et al. (2025); Bachmann & Nagarajan (2024) found that teacher forcing alone is unable to teach long-horizon reasoning to transformers, suggesting these limitations may be more fundamental, perhaps architectural. Further studies, such as R-Horizon (Lu et al., 2025) and h1 (Motwani et al., 2025), compose individually solvable mathematical problems into multi-step dependency chains and investigate whether RL training can mitigate long-horizon failures. Building on this line of work, we characterize how RL training behaves as reasoning structures are systematically scaled, and how these scaling dynamics shape downstream transfer.

Scaling in LLMs. Early scaling-law studies found regular power laws relating pre-training performance to model scale, data volume, and training compute (Kaplan et al., 2020; Henighan et al., 2020; Hoffmann et al., 2022). Later, test-time scaling emerged to improve reasoning by allocating additional computation during decoding (Wei et al., 2022; Wang et al., 2023; Yao et al., 2023; Muennighoff et al., 2025). Recent work has extended scaling-law analysis to RL post-training, showing regular scaling behavior with model scale, data, and compute (Khatri et al., 2025; Tan et al., 2025). However, existing RL scaling studies primarily vary the volume of training data, while providing limited control over the reasoning complexity of individual problems. In contrast, our work enables a cleaner analysis through explicit and interpretable control of reasoning complexity.

RL for LLM Reasoning. RL with verifiable rewards (RLVR) has become a promising paradigm for reasoning-oriented post-training (Luong et al., 2024; Guo et al., 2025a). With practical optimizers such as GRPO (Shao et al., 2024) and later variants (Yu et al., 2025; Zheng et al., 2025), it has enabled large-scale RL training and long chain-of-thought reasoning (Jaech et al., 2024; Guo et al., 2025a). However, most existing RL reasoning work centers on mathematics and programming, where high-quality training problems are limited, often depend on human-curated solutions or test cases, and offer only coarse difficulty control (Liu et al., 2025a). As a result, RL performance is limited by the collected data, making sustained improvement difficult as training scales. In contrast, SCALELOGIC offers explicit complexity control, verifiable solutions, and unlimited low-cost data generation, enabling a cleaner and more scalable framework for reasoning-oriented RL.

Synthetic Data for Post-training. Synthetic data is a natural fit for RLVR, motivating recent work on automatically generated reasoning problems with verifiable rewards. Early efforts studied task-specific synthetic settings for logical reasoning, such as Knights and Knaves (Xie et al., 2025; Lin et al., 2025). More recent work shifted toward task families with difficulty control such as SAT (Liu

et al., 2025a), graph reasoning tasks such as G1 (Guo et al., 2025b), and benchmark-style synthetic reasoning suites (Liu et al., 2025b; Chen et al., 2025; Stojanovski et al., 2025; He et al., 2026). While these studies show the promise of synthetic data for post-training, their tasks are typically worst-case NP-hard search problems (e.g., SAT and Hamiltonian path in G1) with limited control over underlying expressiveness, making it difficult to isolate how task complexity affects RL training compute and downstream reasoning. In contrast, SCALELOGIC independently controls proof depth and expressiveness with each instance oracle-solvable in time linear in the proof size.

3 METHOD

We present SCALELOGIC, a framework for generating synthetic logical reasoning problems with fine-grained control over task difficulty. Section 3.1 describes how reasoning problems are constructed, Section 3.2 explains how the difficulty of the generated problems is systematically controlled through expressiveness, and Section 3.3 describes the reinforcement learning setup for post-training.

3.1 GENERATION OF LOGICAL REASONING PROBLEMS

Our training environment is built on a pipeline for generating synthetic logical reasoning problems. We refer to a grounded predicate applied to a specific entity, possibly negated, such as "Alice is a cat" (written in logic as `cat(Alice)`) or "Alice is not a cat" (`¬cat(Alice)`), as a *literal*. Each instance consists of a collection of axioms, which include literals such as "Alice is a cat" (`cat(Alice)`) and rules such as "If Alice is a cat, then Alice is a mammal" (`cat(Alice)` $\rightarrow$ `mammal(Alice)`), which together determine the set of conclusions that can be logically derived, such as "Alice is a mammal" (`mammal(Alice)`). The underlying task is to identify, among a set of candidate conclusions (each a single literal), which one is logically derivable from the given axioms.

Each instance is presented as a single-answer multiple-choice problem. To construct it, we first sample B literals, each serving as the root of a proof tree. Starting from each root, we recursively expand the tree by adding child literals to its leaves. Each parent node, along with its set of children, defines a proof step: the children serve as the premises, while the parent is the conclusion. The conclusion of one proof step may in turn serve as a premise for another step higher in the tree. Thus, expanding a leaf amounts to generating its supporting premises, which become new leaves of the proof tree. We repeat this process until a target proof depth D is reached, at which point the leaves are treated as the literal axioms from which the proof begins (see Algorithm 1). This procedure generates proofs "backwards": it starts from the conclusion and progressively constructs the premises needed to derive it. To avoid introducing alternative derivations, each expansion adds premise literals with fresh predicates that do not appear elsewhere in any proof trees. This ensures that every node has a unique derivation from the axioms, corresponding to the subtree rooted at that node.[3] Applying the procedure to the B roots yields one proof tree for each candidate conclusion.

We keep one proof tree intact so that its root conclusion remains derivable. For each of the remaining $B - 1$ candidates, we uniformly sample one axiom from the proof and corrupt it in one of two ways: (i) *removing* the axiom, or (ii) *flipping* the polarity of one literal within the axiom, e.g., changing "Alice is a cat" (`cat(Alice)`) to "Alice is not a cat" (`¬cat(Alice)`). Note that option (ii) is available only when negation is included in the underlying formal system; it is disabled in less expressive settings that do not support negation (see Appendix C). The uniform sampling process prevents the model from exploiting positional shortcuts. Since our backward construction yields a unique proof for each candidate, corrupting a single axiom severs the proof path to the root and makes the corresponding conclusion non-derivable. We additionally insert a small number of distracting rules to increase local ambiguity without changing the derivability of any candidate conclusion; their generation is detailed in Appendix B.2.

This construction yields two interpretable structural variables, B and D, that directly govern instance complexity. Increasing B introduces more plausible candidate conclusions that the model must distinguish, while increasing D requires reasoning over longer proof chains. At the same time, the formulation allows exact verification of the final answer without supervising the entire proof, making it a natural fit for reinforcement learning with verifiable rewards.

[3] Similar to the generation procedure in Opedal et al. (2025) which also guarantees the uniqueness of generated proofs.

After construction, we convert the axioms and candidate conclusions into natural language via predefined templates, instantiated with randomly sampled entity names and fake predicate words (see Appendix B.3). An illustrative example is provided below and algorithms are given in Appendix B.

Natural Language Example

Suppose we have the following facts:
Alice has fur. Alice is a vertebrate. If Alice has fur and is a vertebrate, then Alice is a mammal. If anyone is a mammal, then they are not a bird. ...
Here are some candidate statements: (A) Alice is not a bird. (B) Alice is cold-blooded. (C) ...
Which one of the above statements can be logically derived from the given facts?
Answer: Alice is not a bird.

3.2 Control of Expressiveness

Beyond depth and candidate count, we vary the expressiveness of the underlying formal system while preserving the same task format. We construct a hierarchy of five settings by progressively expanding the set of available inference rules, with each added logical operator enabling a broader range of proof structures; further details are provided in Appendix C.

Implication-only. We start with a simple logic that contains only the implication operator (i.e., "if-then", written $\rightarrow$). Each axiom is either a grounded literal, such as "Alice is a cat" (`cat(Alice)`), or a simple implication rule such as "If Alice is a cat, then Alice is a mammal" (`cat(Alice) → mammal(Alice)`).[4] A statement is considered valid if it can be derived from the grounded literals by repeatedly applying rules whose antecedents are satisfied.

+ Conjunction. We extend the simplest logic with the conjunction operator (i.e., "and", written $\wedge$), which allows a rule to depend on multiple premises simultaneously.[5] Under this logic, a rule may take the form of a conjunction of grounded literals entailing a single conclusion, such as "If Alice is a vertebrate and has fur, then Alice is a mammal" (`vertebrate(Alice) ∧ has_fur(Alice) → mammal(Alice)`). During inference, the model must coordinate multiple satisfied supporting literals before applying each rule, rather than relying on single-premise inference.

+ Negation. We further extend the logic with the negation operator (i.e., "not", written $\neg$), which allows premises and conclusions to involve negated literals. Under this logic, rules may condition on negated properties or derive negated literals as a consequence of other literals, such as "If Alice is a mammal, then Alice is not a bird" (`mammal(Alice) → ¬bird(Alice)`). With negation available, each literal in a proof tree has a well-defined negated counterpart, so the model must track not only whether a predicate has been established but also its polarity.

+ Disjunction. We further extend the logic with the disjunction operator (i.e., "or", written $\vee$), which allows a rule to produce multiple possible consequents from the same premises. Under this logic, a rule may take the form of the antecedent entailing a disjunction of grounded literals, such as "If Alice is a pet, then Alice is a cat or a dog" (`pet(Alice) → cat(Alice) ∨ dog(Alice)`). With disjunction, the model must reason over multiple possible conclusions, determining which alternatives are eliminated and which support the target statement.

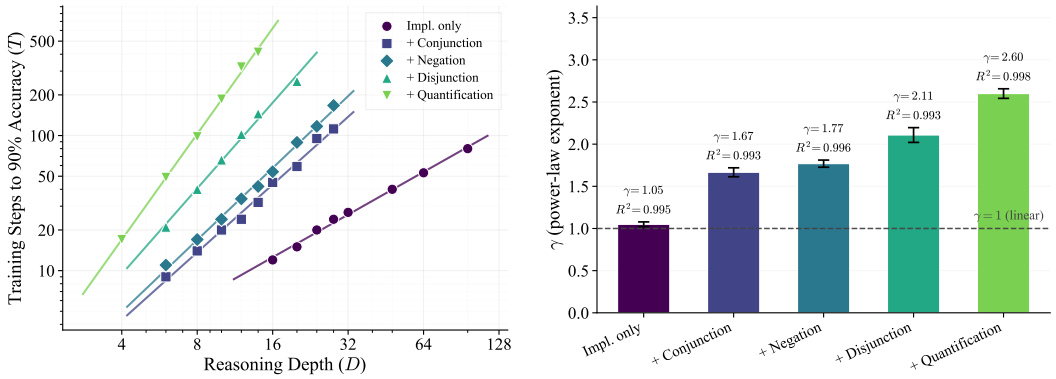
+ Quantification. The most expressive logic we consider extends the earlier logic from purely propositional reasoning towards first-order reasoning, through *universal quantification* (i.e., "for all", written $\forall$), which enables the definition of rules that apply to *any* entity rather than a specific one. For example, "Anyone who is a cat is a mammal" (`∀X(cat(X) → mammal(X))`). At inference time, applying such quantified rules requires the model to instantiate them with a concrete entity in the current context, and to verify that the instantiated antecedents hold before deriving the consequents.

3.3 Reinforcement Learning Framework

We use DAPO (Yu et al., 2025), an extension of GRPO (Shao et al., 2024), as our default RL algorithm. For each prompt q, we sample G completions $\{o_i\}_{i=1}^{G}$ and optimize the policy objective

[4] This logic is also known as *implicational propositional logic* or *implicational propositional calculus*.

[5] We only permit conjunctions in the antecedent of if-then statements, since `A → B ∧ C` is equivalent to `A → B` and `A → C`. Similarly, we only permit disjunctions in the consequent.



(a) Log-log scaling curves with power-law fits. (b) Fitted exponent γ vs. Expressiveness of the logic.

Figure 2: **Training cost exhibits power-law scaling with reasoning depth, with exponent γ increasing systematically along the expressiveness hierarchy. (a)** Scatter points show the training steps T required to reach the 90% accuracy threshold as a function of reasoning depth D. Solid lines show power-law fits $T \propto D^{\gamma}$. **(b)** Fitted exponent γ increases monotonically when expressiveness increases from *Implication-only* ($\gamma = 1.05$) to + *Quantification* ($\gamma = 2.60$). Error bars denote ± 1 standard error of the fitted exponent.

$$\mathcal{L}_{\text{DAPO}}(\theta) = \mathbb{E}_{q,\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}} \left[\frac{1}{\sum_{i=1}^{G} |o_i|} \sum_{i=1}^{G} \sum_{t=1}^{|o_i|} \min \left(r_{i,t}(\theta)\, \hat{A}_i,\ \text{clip}(r_{i,t}(\theta),\, 1-\epsilon_l,\, 1+\epsilon_h)\, \hat{A}_i \right) \right],$$

$$\text{where} \qquad r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})}, \qquad \hat{A}_i = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^{G})}{\text{std}(\{R_j\}_{j=1}^{G})}.$$

Here, $r_{i,t}(\theta)$ is the token-level policy ratio, and $\hat{A}_i$ is the group-normalized advantage over completion rewards $\{R_i\}_{i=1}^{G}$. Following DAPO, we use dynamic sampling and asymmetric clipping with $(\epsilon_l, \epsilon_h) = (0.20, 0.28)$. For reward design, we adopt a simple and verifiable binary reward, following common practice in reasoning-oriented RL (Guo et al., 2025a). Specifically, we require the model to place its final answer within `<answer>...</answer>`. During evaluation, the verifier extracts the predicted answer from this span and compares it with the ground-truth answer via exact match. If the output violates the required format or the extracted answer does not match the ground truth, we set $R_i = 0$; otherwise, $R_i = 1$. Full prompt templates are provided in Appendix F.

4 Experiments

In this section, we aim to answer the following research questions.

RQ1 Scaling with complexity (§4.2). How does the cost of training a model with RL to reach a target accuracy scale with reasoning depth and expressiveness?

RQ2 Downstream transfer (§4.3). Does training on synthetic reasoning tasks improve performance on real-world benchmarks, and how does expressiveness affect transfer?

RQ3 Training distribution (§4.4). How does the training distribution affect scaling efficiency?

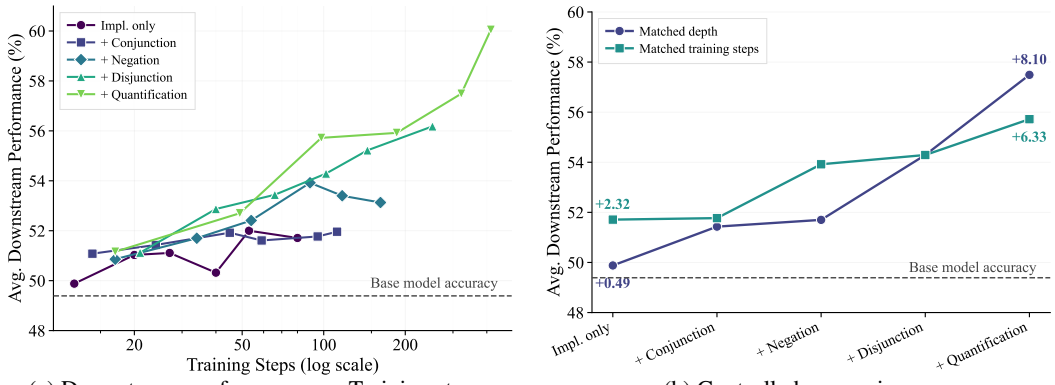
RQ4 Generalization across RL algorithms (§4.5). Is the observed scaling behavior algorithm-specific, or does it reflect a broader phenomenon across RL methods?

RQ5 OOD generalization (§4.6). Does training generalize to more difficult (unseen) depths?

4.1 Experimental Setup

Models and implementation. We perform RL post-training on the non-thinking Qwen3-4B (Yang et al., 2025) using the `verl` library (Sheng et al., 2024). To assess generality across model scales and families, we replicate a subset of experiments on Qwen3-8B and Ministral3-3B (Liu et al., 2026) in Appendix H.5. Full implementation details and hyperparameters are provided in Appendix E.

Task configurations. We vary task difficulty along two axes: *reasoning depth* and *expressiveness*. Unless otherwise specified, we fix the number of candidates to $B = 4$ throughout, and study the effect of varying B separately in Appendix H.1. For each target depth D, we construct a training set by uniformly sampling instances from depths $\{1, \ldots, D\}$, choosing a dataset size (typically 100,000) sufficient for convergence within a single epoch, with each instance seen only once. Evaluation is



(a) Downstream performance vs. Training steps. (b) Controlled comparisons.

Figure 3: **Downstream transfer from synthetic reasoning training. (a)** All settings outperform the base model, with richer logical settings producing larger and more sustained gains. **(b)** Average downstream performance across logical settings under *matched depth* ($D = 12$) and *matched training steps* (~ 100 steps). Both controls exhibit the same monotone trend: More expressive training settings yield stronger downstream performance.

performed on a held-out validation set of 1,000 instances from the same configuration. We study five levels of expressiveness, ranging from implication-only reasoning towards increasingly expressive first-order reasoning, as summarized in Table 2. To confirm task difficulty, we evaluate six frontier LLMs on SCALELOGIC; all models degrade substantially at large depths (see Appendix J).

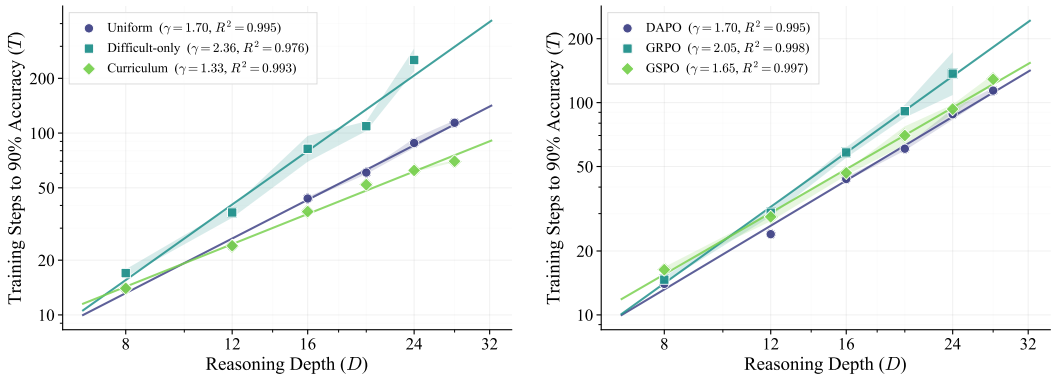
Training protocol and compute metric. For each configuration, we train a separate model and evaluate it on the corresponding held-out validation set after every RL training step. Our primary measure of training compute, denoted by T, is the number of RL training steps required to reach a held-out `Pass@1` threshold of $\mu = 90\%$. We use this metric throughout the main text and discuss additional compute-related quantities, including generated tokens and FLOPs, in Appendix G.3.

Downstream benchmarks. To evaluate transfer beyond the synthetic environment, we test the trained models on a diverse set of downstream reasoning benchmarks, including AIME 2024 and 2025, AMC 2023, MATH-500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), OlympiadBench text-only subset (He et al., 2024), GPQA-Diamond (Rein et al., 2023), and a STEM subset of MMLU-Pro (Wang et al., 2024). Our main downstream metric is the mean `Avg@8` across the eight benchmarks. Detailed results for each benchmark are reported in Appendix H.2.

4.2 POWER-LAW SCALING WITH DEPTH AND EXPRESSIVENESS

We first study how training effort scales with reasoning depth across different levels of expressiveness. As shown in Figure 2a, over the observed depth range, training compute T follows a power law with respect to the reasoning depth D, namely $T = a \cdot D^{\gamma}$, across all five expressiveness levels, with $R^2 > 0.99$ in every case. We fit this relationship via ordinary least squares in log-log space. The power-law model consistently outperforms an exponential fit across all settings ($\Delta\text{AIC} \geq +13.9$; see Appendix G.1), suggesting that training cost grows polynomially rather than exponentially with reasoning depth in the studied regime.

While all settings follow a power-law form, the fitted exponent γ increases monotonically with expressiveness (Figure 2b), from 1.05 for *Implication-only* to 2.60 for *+ Quantification*. Thus, doubling reasoning depth increases training cost by roughly $2\times$ in the simplest setting, but about $6\times$ in the most expressive setting. The near-linear exponent for *Implication-only* suggests that, when the logical structure is sufficiently simple, each additional unit of depth imposes an approximately constant marginal learning cost: the model primarily learns to "chain one more step." By contrast, richer logical operators introduce additional combinatorial structure, such as jointly verifying multiple premises under conjunction, and exhibit more rapidly compounding depth-related costs. Notably, *+ Conjunction* ($\gamma = 1.67 \pm 0.05$) and *+ Negation* ($\gamma = 1.77 \pm 0.04$) exhibit the smallest increase in scaling exponent between adjacent levels ($\Delta\gamma = 0.10$). This is consistent with negation primarily requiring the model to track literal polarity rather than introducing additional combinatorial structure, particularly as our setting excludes proof by contradiction. Appendix G further shows that the monotone expressiveness ordering is robust to alternative analysis choices and potential confounders in task construction.



(a) Training distribution comparison. (b) Cross-algorithm comparison.

Figure 4: **Effect of training distribution and RL algorithm on scaling efficiency in the + *Conjunction* setting.** Each curve aggregates three independent seeds; shading denotes ± 1 standard deviation in log-space across seeds. **(a)** Curriculum training yields the lowest exponent ($\gamma = 1.33$), whereas difficult-only training yields the highest ($\gamma = 2.36$). **(b)** All RL algorithms follow power-law scaling ($R^2 > 0.99$), with exponents from 1.65 to 2.05.

4.3 Downstream Transfer from Synthetic Reasoning Tasks

We next study whether learning on synthetic reasoning tasks transfers to real-world downstream benchmarks. Figure 3a reports average downstream performance as a function of RL training steps across all five expressiveness levels. All settings improve over the base model (49.39%), confirming that the reasoning skills learned in our synthetic environment are not confined to the training domain. However, the size and persistence of these gains differ markedly depending on the expressiveness of the training examples. Simpler settings (*Implication-only* and + *Conjunction*) plateau early around 52%, whereas the + *Quantification* setting continues to improve throughout training, reaching 60.05% at 414 steps—an absolute gain of 10.66 percentage points.

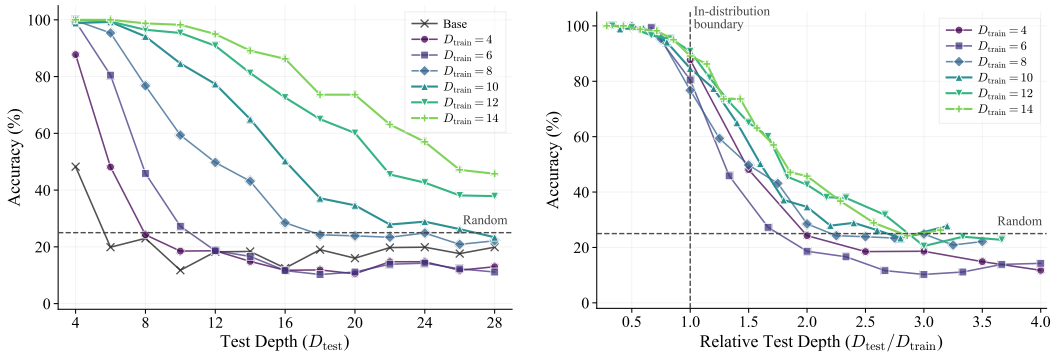
To disentangle the contributions of expressiveness and training compute, Figure 3b compares two controlled conditions across the five settings. The *matched depth* line holds the target reasoning depth at $D = 12$, isolating the effect of expressiveness at a comparable task horizon.[6] The *matched training steps* line selects the checkpoint closest to 100 training steps for each setting, controlling for training budget. Under matched depth, downstream gains increase from $+0.49$ points for *Implication-only* to $+8.10$ points for + *Quantification*, indicating a strong association between expressiveness and downstream transfer. Under matched training steps, gains follow the same monotone trend, rising from $+2.32$ to $+6.33$ points. Together, these controls suggest that downstream reasoning transfer depends strongly on *what* the model is trained on, not merely on *how much* training it receives. Additional analyses in Appendix H show that neither SFT nor correctness-independent RL reproduces these downstream gains, while external logical-reasoning evaluations and comparison with an alternative synthetic framework further support the effectiveness and generality of ScaleLogic.

4.4 Effect of Training Distribution on Scaling Efficiency

We next study how the distribution of training difficulty affects the efficiency and stability of RL scaling. Under the + *Conjunction* setting, we compare three strategies for a target depth D: **uniform** sampling over depths $\{1, \ldots, D\}$ as our default setting, **curriculum** sampling that gradually increases the depth during training, and **difficult-only** training that uses only depth-D instances. We run each setting with three seeds per depth. The implementation details are provided in Appendix E.2.

As shown in Figure 4a, all three strategies retain power-law scaling, but the fitted exponents differ substantially: curriculum training yields the lowest exponent ($\gamma = 1.33$), followed by uniform training ($\gamma = 1.70$), while difficult-only training scales most steeply ($\gamma = 2.36$) and exhibits the largest variance. Appendix H.3 shows the same benefit under + *Quantification*, reducing γ from 2.60 to 2.30. To better understand this effect, we analyze training dynamics in Appendix H.4. Under curriculum training, long chain-of-thought behavior emerges at broadly similar stages across depths; under uniform or difficult-only training, however, its emergence is increasingly delayed and often

[6] For the *Implication-only* setting, we use $D = 16$ because smaller depths are close to saturation (see Appendix E).



(a) Accuracy vs. Absolute test depth. (b) Accuracy vs. Normalized test depth by train depth.

Figure 5: **Out-of-distribution generalization across reasoning depths. (a)** Increasing the training depth consistently extends the range of solvable test depths. **(b)** OOD generalization remains bounded: even the models trained at the largest depths fall to random at $D_{\text{test}}/D_{\text{train}} \approx 3$.

irregular (e.g., note the large gap between depths 20 and 24). These patterns suggest that curriculum training improves scaling efficiency by accelerating long-CoT emergence across depths, possibly because shallower instances provide a smoother bootstrap for learning useful reasoning patterns. We leave a thorough investigation of this mechanism to future work.

4.5 Cross-Algorithm Robustness

To assess whether the observed scaling behavior is specific to DAPO, we repeat the scaling experiment in the + *Conjunction* setting with base GRPO (Shao et al., 2024) and GSPO (Zheng et al., 2025), using three seeds per depth. As shown in Figure 4b, all three algorithms exhibit clear power-law scaling ($R^2 > 0.99$), confirming that the relationship $T \propto D^\gamma$ is not an artifact of a particular optimization method. DAPO and GSPO have similar fitted exponents ($\gamma = 1.70$ and 1.65, respectively), while GRPO exhibits a steeper scaling exponent ($\gamma = 2.05$) and greater variance at larger depths, suggesting lower sample efficiency in the long-horizon regime.

4.6 Out-of-Distribution Generalization

We examine whether training at a target reasoning depth generalizes to higher unseen depths under + *Quantification*. Figure 5a shows that larger training depths consistently extend the range of strong performance, suggesting that training expands the model's effective reasoning horizon rather than merely improving performance at the target depth. However, generalization remains bounded: all models eventually degrade at sufficiently large depths. After normalizing test depth by training depth (Figure 5b), the two deepest models ($D_{\text{train}} = 12, 14$) collapse onto a frontier curve and approach random performance at roughly $3D_{\text{train}}$. Thus, deeper training extends the solvable range approximately linearly but does not eliminate the horizon limit.

5 Conclusion

We introduced ScaleLogic, a synthetic logical reasoning environment with independent control over reasoning depth and expressiveness, and used it to study how RL post-training scales with task complexity. Training compute T follows a power law in proof depth D ($T \propto D^\gamma$, $R^2 > 0.99$) across settings, with the scaling exponent γ increasing monotonically from 1.05 to 2.60 as the logic grows richer. Crucially, expressiveness influences not only training dynamics but also downstream transfer: more expressive settings yield larger and more compute-efficient gains on real-world reasoning tasks, indicating that downstream transfer is shaped by *what* a model is trained on, not merely by *how much* it is trained. Our results suggest that improved training methodology and data could be used to overcome LLM limitations in long-horizon reasoning. Several directions remain open for future work. First, it remains important to test whether these scaling trends persist for larger models and broader RL regimes. Second, richer logical fragments, including equality, higher-order and non-monotonic reasoning, may reveal new scaling regimes. Finally, explaining why different logical operators induce different scaling exponents is an important direction for future theoretical work.

AI USE STATEMENT

In this work, we did not use generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, or interpret results. We only used generative AI tools to polish the manuscript for clarity, readability, and concision. We reviewed all AI-assisted work and take responsibility for the final content of this work.

ETHICS STATEMENT

This work studies RL scaling behavior in a controlled synthetic setting using abstract logical literals with randomly sampled entities and predicates; it does not involve human subjects, real-world sensitive data, or content that could be directly misused. A better understanding of how training complexity and data expressiveness govern RL scaling may help the community design more compute-efficient post-training pipelines. We acknowledge that general improvements in reasoning ability could in principle enable both beneficial and harmful downstream applications, but these risks are not specific to the methods introduced in this work.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the task generation procedure and logical formalisms in Appendices B, C, and D, including pseudocode and logical soundness verification. Appendix E specifies the training and evaluation setup, implementation details, and hyperparameters, while Appendix F provides the prompt templates. Appendices G, H, and I report robustness analyses, detailed experimental results, and training dynamics to facilitate verification and reproduction of our findings. We further provide anonymous source code for the SCALELOGIC pipeline in the supplementary material.

REFERENCES

Gregor Bachmann and Vaishnavh Nagarajan. The pitfalls of next-token prediction. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pp. 2296–2318. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/bachmann24a.html.

Jiangjie Chen, Qianyu He, Siyu Yuan, Aili Chen, Zhicheng Cai, Weinan Dai, Hongli Yu, Qiying Yu, Xuefeng Li, Jiaze Chen, Hao Zhou, and Mingxuan Wang. Enigmata: Scaling logical reasoning in large language models with synthetic verifiable puzzles. *arXiv preprint arXiv:2505.19914*, 2025.

Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 4963 of *Lecture Notes in Computer Science*, pp. 337–340. Springer, 2008. doi: 10.1007/978-3-540-78800-3_24.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Xiaojun Guo, Ang Li, Yifei Wang, Stefanie Jegelka, and Yisen Wang. G1: Teaching llms to reason on graphs with reinforcement learning. *arXiv preprint arXiv:2505.18499*, 2025b.

Andre He, Nathaniel Weir, Kaj Bostrom, Allen Nie, Darion Cassel, Sam Bayless, and Huzefa Rangwala. Resyn: Autonomously scaling synthetic environments for reasoning models. *arXiv preprint arXiv:2602.20117*, 2026.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiad-

bench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B. Brown, Prafulla Dhariwal, Scott Gray, Chris Hallacy, Benjamin Mann, Alec Radford, Aditya Ramesh, Nick Ryder, Daniel M. Ziegler, John Schulman, Dario Amodei, and Sam McCandlish. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004*, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S. Dhillon, David Brandfonbrener, and Rishabh Agarwal. The art of scaling reinforcement learning compute for llms. *arXiv preprint arXiv:2510.13786*, 2025.

Guangchen Lan, Huseyin A Inan, Sahar Abdelnabi, Janardhan Kulkarni, Lukas Wutschitz, Reza Shokri, Christopher Brinton, and Robert Sim. Contextual integrity in LLMs via reasoning and reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. Zebralogic: On the scaling limits of llms for logical reasoning. In *International Conference on Machine Learning*, 2025.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

Huanyu Liu, Ge Li, Jia Li, Hao Zhu, Kechi Zhang, and Yihong Dong. Saturn: Sat-based reinforcement learning to unleash llms reasoning. *arXiv preprint arXiv:2505.16368*, 2025a.

Junteng Liu, Yuanxiang Fan, Zhuo Jiang, Han Ding, Yongyi Hu, Chi Zhang, Yiqi Shi, Shitong Weng, Aili Chen, Shiqi Chen, Yunan Huang, Mozhi Zhang, Pengyu Zhao, Junjie Yan, and Junxian He. Synlogic: Synthesizing verifiable reasoning data at scale for learning logical reasoning and beyond. *arXiv preprint arXiv:2505.19641*, 2025b.

Elita Lobo, Chirag Agarwal, and Himabindu Lakkaraju. On the impact of fine-tuning on chain-of-thought reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 11679–11698, 2025.

Yi Lu, Jianing Wang, Linsen Guo, Wei He, Hongyin Tang, Tao Gui, Xuanjing Huang, Xuezhi Cao, Wei Wang, and Xunliang Cai. R-horizon: How far can your large reasoning model really go in breadth and depth? *arXiv preprint arXiv:2510.08189*, 2025.

Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. *arXiv preprint arXiv:2401.08967*, 2024.

Sumeet Ramesh Motwani, Alesia Ivanova, Ziyang Cai, Philip Torr, Riashat Islam, Shital Shah, Christian Schroeder de Witt, and Charles London. h1: Bootstrapping llms to reason over longer horizons via reinforcement learning. *arXiv preprint arXiv:2510.07312*, 2025.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Andreas Opedal, Yanick Zengaffinen, Haruki Shirakami, Clemente Pasti, Mrinmaya Sachan, Abulhair Saparov, Ryan Cotterell, and Bernhard Schölkopf. Are language models efficient reasoners? a perspective from logic programming. In *Advances in Neural Information Processing Systems*, 2025.

Revanth Rameshkumar, Jimson Huang, Yunxin Sun, Fei Xia, and Abulhair Saparov. Reasoning models reason well, until they don't. *arXiv preprint arXiv:2510.22371*, 2025.

Mohammad Ramezanali, Mo Vazifeh, and Paolo Santi. seqbench: A tunable benchmark to quantify sequential reasoning limits of llms, 2025. URL https://arxiv.org/abs/2509.16866.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Abulhair Saparov, Srushti Ajay Pawar, Shreyas Pimpalgaonkar, Nitish Joshi, Richard Yuanzhe Pang, Vishakh Padmakumar, Seyed Mehran Kazemi, Najoung Kim, and He He. Transformers struggle to learn to search. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=qFVVBzXxR2V.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2505.24760*, 2025.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Zelin Tan, Hejia Geng, Xiaohang Yu, Mulei Zhang, Guancheng Wan, Yifan Zhou, Qiang He, Xiangyuan Xue, Heng Zhou, Yutao Fan, et al. Scaling behaviors of llm reinforcement learning post-training: An empirical study in mathematical reasoning. *arXiv preprint arXiv:2509.25300*, 2025.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Qwen Team. Qwen3.5: Accelerating productivity with native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2023.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

Shuang Wu, Liwen Zhu, Tao Yang, Shiwei Xu, Qiang Fu, Yang Wei, and Haobo Fu. Enhance reasoning for large language models in the game werewolf. *arXiv preprint arXiv:2402.02330*, 2024.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Yang Zhou, Hongyi Liu, Zhuoming Chen, Yuandong Tian, and Beidi Chen. Gsm-infinite: How do your llms behave over infinitely increasing context length and reasoning complexity? In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Appendix

A Limitations **16**

B Task Generation Procedure **16**

B.1 Backward Construction of Proof Trees 17

B.2 Multi-Choice Instance Assembly 18

B.3 Natural Language Conversion 18

C Detailed Description of Expressiveness **19**

C.1 Implication-only 19

C.2 + Conjunction 19

C.3 + Negation 19

C.4 + Disjunction 20

C.5 + Quantification 20

D Validity of the Construction **20**

D.1 Shortcut Controls 20

D.2 Logical Soundness via Z3 Verification 21

E Full Implementation Details **21**

E.1 Experimental Details 21

E.2 Curriculum Implementation 22

F Prompt Templates **22**

G Scaling Robustness **22**

G.1 Power-Law vs. Exponential Fit Comparison 23

G.2 Sensitivity to the Accuracy Threshold 23

G.3 Different Metrics of Training Compute 24

G.4 Variability across Random Seeds 25

G.5 Multi-Entity Reasoning as a Potential Confounder 26

G.6 Prompt Length as a Potential Confounder 26

G.7 Controlling for Surface-Form Differences 27

G.8 Operator-Swap Ablation 28

H Additional Experimental Results **28**

H.1 Candidate Count Scaling 28

H.2 Detailed Downstream Results 29

H.3 Curriculum at the Most Expressive Setting 29

H.4 Data Distribution Shapes Long-CoT Emergence 29

H.5 Cross-Scale and Cross-Model Replication 31
H.6 Matched-Budget Alternative-Training Controls 32
H.7 Proof-Faithfulness Audit 34
H.8 Evaluation on External Logical-Reasoning Benchmarks 34
H.9 Comparison with an Alternative Synthetic-Data Baseline 35

I Training Dynamics of the Main Experiments **35**

J Evaluation of Frontier LLMs on SCALELOGIC **36**

K Qualitative Examples **36**

Algorithm 1: Backward Construction of a Proof Tree

Input: target depth D, rule budget E, expressiveness flags $\Phi \subseteq \{\wedge, \neg, \vee, \forall\}$
Output: axiom-goal pair $(\mathcal{S}, goal)$

```
goal ← sample a fresh literal ;          // root conclusion of the proof tree
Q ← [(0, goal)] ;                        // initial depth and root literal
S ← [];
T ← [] ;                                 // reusable quantified-rule templates
while Q ≠ ∅ do
    (d, u) ← POP(Q) ;                                  // pop max-depth leaf
    if d ≥ D or rules(S) ≥ E then
        S ← S ∪ {u} ;                                  // add a literal axiom
        continue;
    scope ← CHOOSEENTITYSCOPE(∀ ∈ Φ) ;  // entity-specific or universal
    if scope = ∀ and T ≠ ∅ and BERNOULLI(p_reuse) then
        r ← instantiate a template from T with a fresh entity;
        Bind one conclusion of r to u, update S; enqueue other literals of r into Q;
        Remove the template from T if it has been instantiated for all entities;
        continue;
    (n_p, n_c) ← SAMPLEARITY(Φ) ;   // n_p ≥ 2 needs ∧ ∈ Φ, n_c ≥ 2 needs ∨ ∈ Φ
    P ← BUILDPREMISES(d + 1, n_p);
    C ← BUILDCONCLUSIONS(d, n_c, u) ;          // u should be one of the
     conclusions
    enqueue newly introduced literals from P and C into Q;
    S ← S ∪ {P → C} ;                                  // adds a rule axiom
    if scope = ∀ then
        T ← T ∪ {P → C};
if ¬ ∈ Φ then
    foreach literal u in S do
        if BERNOULLI(p_¬) then
            negate all occurrences of u in S ;     // random polarity flip for ¬
return (S, goal);
```

A LIMITATIONS

The main experiments are conducted on Qwen3-4B, with a subset replicated on Qwen3-8B and Ministral3-3B (Appendix H.5) to assess cross-scale and cross-model consistency. However, it remains unclear whether the observed scaling laws generalize to substantially larger models or different architectures like MoE. The power-law relationship $T = a \cdot D^{\gamma}$ is fitted using limited depth values for each expressiveness level. Due to computational constraints, we cannot further extend the depth range. Although the fits achieve $R^2 > 0.99$ and consistently outperform exponential alternatives (Table 4), this limited range means that the power-law characterization should be interpreted as a high-fidelity empirical description of the observed regime, rather than a robust asymptotic law. In addition, the main scaling curves are fit from single-seed runs; our multi-seed analysis at + *Conjunction* shows similar fitted exponents (Appendix G.4), though broader multi-seed validation is still needed. Both the training-distribution comparison and the cross-algorithm analysis are conducted at selected expressiveness levels; whether these conclusions generalize across all expressiveness levels is not verified. Finally, our analysis is empirical. While we observe a clean power-law dependence and a monotone relationship between expressiveness and the scaling exponent γ, we provide no theoretical derivation of these relationships. Establishing the theoretical foundation is left to future work.

B TASK GENERATION PROCEDURE

This appendix provides the full generation procedure for SCALELOGIC. We decompose the pipeline into three stages: backward construction of candidate proof trees, assembly of a multiple-choice

Algorithm 2: Multiple-Choice Instance Assembly

Input: B axiom-goal pairs $\{(\mathcal{S}_i, goal_i)\}_{i=1}^{B}$, maximum depth $D_{\max}$, distractor budget M, expressiveness flags Φ
Output: combined axiom set $\mathcal{S}$, candidate conclusions $\mathcal{C}$

```
S ← S_1;
C ← {goal_1} ;                                  // goal 1 is fixed as provable
for i = 2 to B do
    a* ← a randomly selected axiom from S_i;
    if ¬ ∉ Φ or BERNOULLI(p_remove) then
        remove a* from S_i;
    else
        flip the polarity of one literal in a* ;        // available only when ¬ ∈ Φ
    S ← S ∪ S_i;
    C ← C ∪ {goal_i};
m ← UNIFORM{0, ..., M};
k ← 0;
while k < m and rules(S) < B D_max do
    randomly choose to introduce fresh distractor literals for P_d, C_d, or both;
    S ← S ∪ {P_d → C_d}; k ← k + 1 ;                // add distractor rules
return (S, C);
```

instance with exactly one provable candidate, and conversion from symbolic formulas to natural language prompts. Throughout the pipeline, the expressiveness flags $\Phi \subseteq \{\wedge, \neg, \vee, \forall\}$ determine which logical operators are available, corresponding to the settings described in Section 3.2. The soundness of the resulting construction is audited in Appendix D.

B.1 Backward Construction of Proof Trees

Algorithm 1 constructs a single candidate proof tree. The generator initializes the work queue as $Q = [(0, goal)]$, where $goal$ is a randomly sampled root literal at depth 0. During construction, Q stores the current open leaves of the partially generated proof tree, each paired with its depth. In practice, Q is processed in a depth-first order: `POP` always selects a leaf literal at the largest current depth. Once a popped literal has depth $d \geq D$ or the rule budget E is exhausted ($E \geq D$), it is added to the axiom set as a given literal and is no longer expanded. This depth-first expansion guarantees that the proof tree is grown until it reaches the target proof depth D.

For each popped leaf literal u with $d < D$, the generator assigns a rule whose conclusion contains u. The expressiveness flags Φ restrict the admissible rule forms: when conjunction or disjunction is enabled, we allow rules with multiple premises or multiple conclusions, respectively, with the arity set to 2 in our experiments. All newly introduced literals in the premise set P and conclusion set C, except for the already fixed literal u, are assigned fresh predicates to avoid ambiguity, and are treated as new leaf literals added to Q. For details on how disjunctive conclusions are resolved to preserve a unique derivation, see Appendix C. When universal quantification is enabled, the generator may reuse an existing universal rule template from the template set $\mathcal{T}$ with probability $p_{\text{reuse}} = \frac{|\mathcal{T}|}{|\mathcal{T}|+1}$. It instantiates the template with an entity for which the template has not yet been used, and binds one of the instantiated conclusions to the current leaf literal u. All remaining literals introduced by the instantiated rule are then handled in the same way as in ordinary rule expansion. In our experiments, each instance contains at most two entities, so a reused universal template is removed from $\mathcal{T}$ once it has been instantiated for all available entities. After the proof tree is built, if negation is enabled, each literal is independently polarity-flipped with probability $p_{\neg} = 0.5$, and the flip is applied consistently to all of its occurrences in the axiom set. This final polarity randomization changes the surface polarity distribution while preserving the derivability of the constructed proof.

Algorithm 3: Natural Language Conversion

Input: axiom set $\mathcal{S}$, candidate conclusions $\mathcal{C}$
Output: prompt x, answer y

```
φ_e ← random bijection from entity IDs to names ;        // e.g., e_0 ↦ "Alice"
φ_p ← random bijection from literal IDs to predicates ;  // e.g., u_3 ↦ "furry"
F ← [] ;                                                 // natural-language facts
foreach s ∈ SHUFFLE(S) do
    if s is a given literal then
        F.APPEND(TOTEXT(s, φ_e, φ_p)) ;                  // e.g., "Alice is furry."
    else
        f ← RULETOTEXT(s, φ_e, φ_p);
        if s involves a universal entity then
            f ← QUANTIFY(f) ;                            // e.g., "If anyone is X, they are Y."
        F.APPEND(f);
C_nl ← [];                                               // natural-language conclusions
foreach c ∈ SHUFFLE(C) do
    C_nl.APPEND(TOTEXT(c, φ_e, φ_p));
y ← the unique provable candidate in C_nl;
x ← FORMATPROMPT(F, C_nl);
return (x, y);
```

B.2 Multi-Choice Instance Assembly

Algorithm 2 assembles a B-choice problem from B independently generated axiom-goal pairs $\{(S_i, goal_i)\}_{i=1}^{B}$. Without loss of generality, the first proof tree is kept intact, so $goal_1$ remains provable and serves as the unique correct candidate (the candidate order is randomized later in Appendix B.3). For each remaining proof tree $i = 2, \ldots, B$, the generator selects an axiom $a^\star$ uniformly at random from S_i and corrupts it. If negation is not enabled, the selected axiom is removed. If negation is enabled, the axiom is either removed with probability $p_{\text{remove}} = 0.5$ or corrupted by uniformly sampling one literal in the axiom and flipping its polarity. Because Algorithm 1 constructs each candidate with a unique derivation, corrupting one selected axiom breaks the only derivation of $goal_i$, making it unprovable. The corrupted axiom set is then merged into the global axiom set S, and $goal_i$ is added to the candidate list.

After merging all B proof trees, the generator optionally adds distractor rules to increase local ambiguity. In our implementation, the total number of rule applications is capped at $BD_{\max}$, where $D_{\max}$ denotes the maximum depth of the corresponding training configuration. We sample a distractor count $m \sim \text{Uniform}\{0, \ldots, M\}$ with $M = 5$, and then add at most m distractor rules without exceeding the $BD_{\max}$ edge cap. Each distractor follows the rule forms permitted by Φ, but is constructed so that at least one side of the rule, either the premise set P_d, the conclusion set C_d, or both, contains only fresh predicates not used elsewhere in the instance. If only one side is fresh, the remaining side is sampled from the literals already present in the current instance. For example, a distractor might state "If Alice is a cat, then Alice is small" (`cat(Alice) → small(Alice)`) where "small" is a fresh predicate absent from all proof trees. Consequently, each distractor can interact with the existing derivation graph on at most one side: it may be triggered by existing literals but only derive fresh literals, or it may require fresh literals and therefore cannot be triggered by the existing axioms. This increases local ambiguity without changing the unique provable answer.

B.3 Natural Language Conversion

Algorithm 3 converts the symbolic instance into a natural-language multiple-choice prompt. For each instance, we first sample a bijection from internal entity IDs to a fixed pool of 26 names, shown below.

Name pool. Alice, Bob, Carol, David, Emma, Frank, Grace, Henry, Irene, Jack, Kate, Leo, Mona, Nick, Olivia, Paul, Quincy, Rachel, Sam, Tina, Ulysses, Victoria, Wendy, Xavier, Yvonne, and Zach.

We also sample a separate bijection from predicate IDs to random 5-letter predicate strings. These mappings are resampled independently for every instance, preventing the model from relying on fixed entity identities or real-world predicate semantics.

Each symbolic axiom is then rendered with predefined templates: literal axioms become factual statements (e.g., Alice is abcde), grounded rules become if-then statements (e.g., If Alice is abcde, Alice is bcdef), and universal rules are expressed with quantified templates such as "If anyone is abcde, then they are bcdef". The rendered facts are shuffled before being placed in the prompt, and the candidate conclusions are also shuffled to remove ordering cues. The final answer y is the natural-language rendering of the unique provable candidate, while the prompt x consists of the rendered facts with the shuffled candidate list using the templates in Appendix F.

C Detailed Description of Expressiveness

This section provides the full description of the five expressiveness levels summarized in the main text (Section 3.2). Table 2 summarizes these levels and their features.

Table 2: The five levels of expressiveness and their features.

Setting	$\rightarrow$	$\wedge$	$\neg$	$\vee$	$\forall$
Implication-only	✓	–	–	–	–
+ Conjunction	✓	✓	–	–	–
+ Negation	✓	✓	✓	–	–
+ Disjunction	✓	✓	✓	✓	–
+ Quantification	✓	✓	✓	✓	✓

C.1 Implication-only

In the implication-only setting, each axiom is either a grounded literal or a simple implication rule. Starting from the literal axioms, reasoning proceeds by repeatedly applying rules whose antecedents are satisfied, thereby deriving new literals. A statement is considered valid if it can be derived from the axioms through a sequence of such rule applications. Performing deductive reasoning under this logic is equivalent to path-finding in simple directed graphs.

C.2 + Conjunction

In this setting, a rule may take the form of a conjunction of grounded literals entailing a single conclusion. Accordingly, at inference time, the model must jointly track and verify multiple supporting literals at each reasoning step. Compared to implication-only reasoning, this shifts the task from single-premise inference to coordinating multiple supporting literals before each rule application. Performing deductive reasoning under this logic is equivalent to path-finding in simple directed hypergraphs where each hyperedge may have more than one source node.

C.3 + Negation

In this setting, rules may condition on negated properties or derive negated consequents. Compared to the conjunction setting, negation further requires the model to maintain polarity information throughout the proof, introducing a qualitatively different form of reasoning.[7] As a byproduct, negation provides a natural mechanism for constructing non-target candidate conclusions: flipping the polarity of an intermediate literal within a proof-tree axiom breaks the derivation and renders the root conclusion unprovable. For example, `mammal(Alice)` $\rightarrow$ `¬bird(Alice)` can be corrupted into

[7] We note that our proofs do not contain hypothetical derivations, such as proofs by contradiction.

either $\neg$mammal(Alice) $\rightarrow$ $\neg$bird(Alice) or mammal(Alice) $\rightarrow$ bird(Alice), depending on whether the sampled literal is the premise or the conclusion (cf. Section 3.1; algorithm is provided in Appendix B.2).

C.4 + DISJUNCTION

In this setting, a rule may take the form of the antecedents entailing a disjunction of grounded literals. Applying such a rule yields a set of candidate conclusions rather than a single deterministic outcome. A consequence of this is that the simple proof tree analogy described in Section 3.1 must be extended: With disjunction, each proof step may have more than one literal as the conclusion. As such, each edge in the proof tree must be generalized as a hyperedge, which can have more than one parent node and more than one child node. This creates an ambiguity that is absent in ordinary proof trees: a disjunctive rule establishes a disjunction, but not any individual disjunct by itself. For example, suppose the current root literal is mammal(Alice), and the backward construction introduces the rule cat(Alice) $\rightarrow$ mammal(Alice), making cat(Alice) a literal that must be supported. If we then try to support this literal using pet(Alice) $\rightarrow$ cat(Alice) $\vee$ dog(Alice) together with pet(Alice), we can derive cat(Alice) $\vee$ dog(Alice), but not cat(Alice) itself. Therefore, the original root literal mammal(Alice) is not guaranteed to be derivable unless the alternative disjunct dog(Alice) is resolved.

To keep the proof well-defined during our "backwards" proof generation procedure, we resolve each alternative disjunct in one of two standard ways: (i) it is ruled out by additional axioms that contradict it (e.g., a separate axiom states $\neg$dog(Alice)), or (ii) it eventually leads to a common downstream conclusion shared with the chosen disjunct (e.g., add a rule dog(Alice) $\rightarrow$ mammal(Alice) so that both cat(Alice) and dog(Alice) lead to mammal(Alice)). At inference time, the model must therefore examine which disjuncts are eliminated and which converge on a shared conclusion, shifting the task from committing to a single outcome at each step to reasoning over a set of possible conclusions.

C.5 + QUANTIFICATION

In this setting, universal quantification enables defining rules that apply to *any* entity rather than to a specific one. We also consider the number of entities in the problem as an additional axis of complexity. In many of our experiments, we consider the simplest case where all generated literals apply to a single entity (e.g., Alice). We also consider a multi-entity setting, where axioms may involve several distinct entities within the same problem (e.g., Alice and Bob); by default, we use two entities. The use of multiple entities is particularly important when rules are universally quantified: without multiple entities, quantified rules reduce to entity-specific ones and offer no additional reasoning challenge. With multiple entities present, we additionally enable *predicate and rule reuse*: the same predicate may apply to different entities, and a single quantified rule may be instantiated multiple times within the same proof, creating compositional overlap that requires the model to disambiguate shared structure rather than treating each occurrence independently. Together, these extensions move the environment from basic propositional inference towards the more expressive regime of first-order reasoning.

D VALIDITY OF THE CONSTRUCTION

We provide two complementary checks to ensure that our synthetic problems are free of surface-form shortcuts and are logically sound.

D.1 SHORTCUT CONTROLS

We control surface-form shortcuts through the generation and rendering procedure described in Appendix B. Predicates are sampled as fresh random strings and entity-name mappings are resampled independently for each instance, preventing stable lexical or world-knowledge cues. For unprovable candidates, the corrupted axiom is selected uniformly from the corresponding proof tree, preventing the position of the corrupted axiom from revealing the answer. When negation is enabled, polarity-flip corruptions are also randomized, preventing polarity statistics from serving as a shortcut. Finally,

Table 3: Default RL Training Hyperparameters

Parameter	Value
Generate batch size	384 prompts
Training batch size	256 prompts
Mini batch size	64 prompts
Samples per prompt	8
Max response length	8192
Clip ratio	(0.20, 0.28)
Temperature	1.0
Top p	1.0
Optimizer	AdamW
KL coefficient	0
Actor learning rate	1×10^{-6}
LR warmup steps	10
LR scheduler	constant

both the axiom order and candidate order are shuffled before rendering, preventing ordering cues in the prompt or answer options. These controls are designed to ensure that the correct answer is determined by logical derivability rather than by surface statistics.

D.2 Logical Soundness via Z3 Verification

Although our backward-construction procedure is designed to produce correct labels, implementation imperfections may still introduce unwanted label noise. To further verify the correctness of the labels, we audit a random subset of generated problems using the Z3 SMT solver (de Moura & Bjørner, 2008). For each sampled problem, we encode the axiom set as a first-order theory in Z3 and test entailment for every candidate literal. An instance passes the audit only if Z3 confirms that the marked-provable candidate is entailed and all $B - 1$ marked-unprovable candidates are not entailed. We sample $N = 1000$ problems from every configuration used in our experiments, and all sampled instances pass the audit, confirming that the labels used during RL training are logically reliable.

E Full Implementation Details

E.1 Experimental Details

Training. We conduct all RL post-training with the `verl` library (Sheng et al., 2024) on the non-thinking versions of Qwen3-4B and Qwen3-8B (Yang et al., 2025) and Ministral3-3B-Instruct-2512 (Liu et al., 2026). Most runs are performed on 8×B200 180G GPUs (see Figure 8). After completing the main experiments, we additionally ran a small number of configurations on 8×H100 GPUs using the same training recipe to increase coverage in overlapping depth ranges in the scaling analysis of Figure 2. These additional runs are used only for the scaling analyses. When selecting depth values, we exclude configurations for which the base model is already near saturation, defined as $\geq 85\%$ validation accuracy before RL, as they provide little measurable learning range. Unless otherwise specified, all experiments share the training recipe described in Section 3.3, with hyperparameters summarized in Table 3. Full training dynamics are provided in Appendix I.

Evaluation. For all downstream benchmarks, we follow the official decoding parameters recommended by Qwen (Yang et al., 2025): temperature $T = 0.7$, top-$p = 0.8$, and top-$k = 20$. The maximum response length is set to 8192 tokens, matching the training budget. We record `Avg@8` and `Pass@8` for each benchmark and report the mean accuracy across benchmarks (i.e., the mean of per-benchmark `Avg@8` scores) as our primary metric in Section 4.3.

For the out-of-distribution generalization experiments in Section 4.6, where evaluation depths can substantially exceed the training depth, we increase the maximum response length to 32,768 tokens. We find this is necessary because solving deeper OOD problems requires substantially longer reasoning than seen during training, and a tighter budget would truncate completions before they

reach an answer, conflating reasoning capability with generation-length limits. All other decoding settings remain unchanged; we also report `Avg@8` as the main metric.

E.2 Curriculum Implementation

We implement the curriculum strategy used in Section 4.4 following the threshold-triggered scheme of Stojanovski et al. (2025). At each training step, tasks are sampled uniformly from proof depths $\{1, \ldots, D_{\text{cur}}\}$, where D_{cur} is the current curriculum depth; instances beyond D_{cur} are filtered out. Whenever the rolling training accuracy reaches 70%, we increase D_{cur} by a fixed step size Δ, until it reaches the maximum training depth $D_{\max}$. We use the curriculum hyperparameters in the following table. The initial D_{cur} is chosen as the smallest depth at which the base model is not yet saturated; shallower depths are already solved at or near the threshold and would trigger immediate curriculum advancement. We focus on demonstrating the effectiveness of curriculum-based training; designing more efficient curricula and selecting their hyperparameters more systematically is left to future work.

Setting	D_{cur}	Δ
+ Conjunction	8	4
+ Quantification	4	2

For comparison, the default uniform setting is trained on depths uniformly distributed over $1, \ldots, D_{\max}$ throughout, while *difficult-only* training is trained exclusively on depth $D_{\max}$. All three strategies use identical training hyperparameters (Table 3).

F Prompt Templates

We provide the prompt templates used for both training and downstream evaluation, ensuring consistency and reproducibility across tasks.

Prompt Template for Logical Reasoning Task

Suppose we have the following facts:

{List of shuffled axioms and rules}

Here are some candidate statements:

{List of shuffled candidate statements}

Exactly one of the above statements can be logically derived from the given facts.
Identify that statement and return it within <answer> </answer> tags. Example: <answer> Alice is abcde </answer>.

Prompt Template for Math Benchmarks

{Problem}

Please reason step by step, and put your final answer within \boxed{}.

Prompt Template for GPQA-Diamond and MMLU-Pro

{Problem}

Please reason step by step, and put your final answer within \boxed{}. Example: \boxed{A}.

G Scaling Robustness

We conduct eight complementary checks to verify that our main scaling claims are robust to key design and analysis choices.

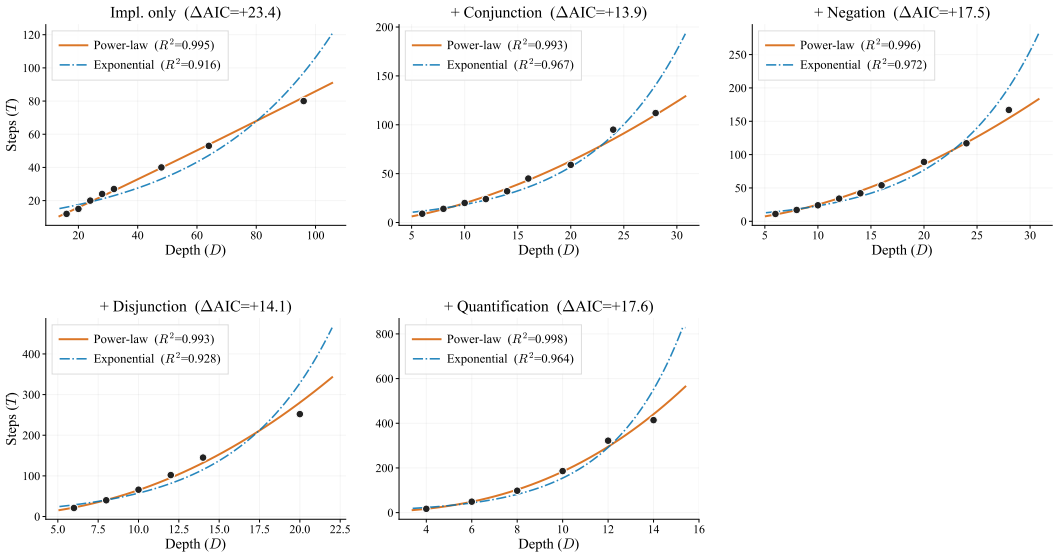


Figure 6: Power-law (orange) vs. exponential (blue) fits for each expressiveness setting. ΔAIC values consistently favor the power-law model.

G.1 Power-Law vs. Exponential Fit Comparison

To validate the power-law scaling relationship reported in Section 4.2, we compare power-law and exponential fits for each expressiveness setting. Both models are fit via ordinary least squares in log space with two free parameters each, enabling direct comparison through the Akaike Information Criterion (AIC).

Table 4: Power-law ($T = a \cdot D^{\gamma}$) vs. exponential ($T = a \cdot e^{bD}$) fit comparison across expressiveness levels. Both models use two free parameters and are fit via OLS in log space. ΔAIC = $\text{AIC}_{\text{exp}} - \text{AIC}_{\text{pow}}$; positive values favor the power-law model.

Setting	**Power-law**		**Exponential**		ΔAIC
	$\gamma \pm$ SE	R^2	$b \pm$ SE	R^2	
Implication-only	1.05 ± 0.03	0.995	0.024 ± 0.003	0.899	+23.4
+ Conjunction	1.67 ± 0.05	0.993	0.113 ± 0.008	0.967	+13.9
+ Negation	1.77 ± 0.04	0.996	0.120 ± 0.008	0.972	+17.5
+ Disjunction	2.11 ± 0.09	0.993	0.174 ± 0.024	0.929	+14.1
+ Quantification	2.60 ± 0.06	0.998	0.318 ± 0.031	0.964	+17.6

As shown in Table 4 and Figure 6, the power-law model achieves higher R^2 and lower AIC than the exponential model across all five settings ($\Delta\text{AIC} \geq +13.9$). Figure 6 visualizes both fits against the observed data, showing that the exponential model systematically overshoots at large depths while the power-law fit remains accurate throughout.

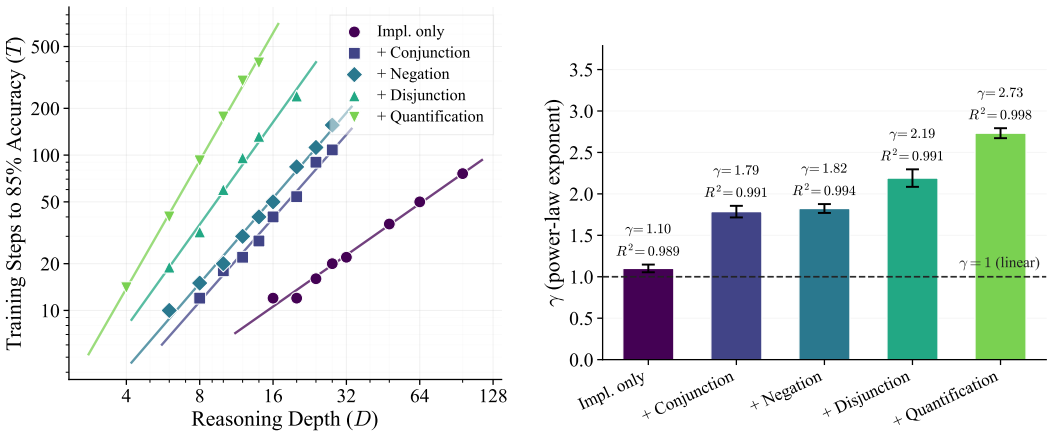
G.2 Sensitivity to the Accuracy Threshold

To verify that our scaling claims are not artifacts of the specific 90% accuracy threshold used in the main text, we refit $T \propto D^{\gamma}$ using a lower threshold of 85% (Figure 7). As shown in Table 5, the fitted exponents shift only modestly across all five settings.

All five settings continue to admit a clean power-law fit ($R^2 \geq 0.989$), and γ remains monotonically increasing in expressiveness ($1.10 \rightarrow 2.73$). The relative ordering of the five exponents is preserved exactly. Our main conclusions are therefore not sensitive to the specific accuracy threshold used.

Table 5: Power-law fits at the main-text accuracy threshold ($\mu = 90\%$) versus a relaxed threshold ($\mu = 85\%$). All five expressiveness settings preserve a clean power-law fit under both thresholds, and the relative ordering of the fitted exponents is preserved exactly.

	$\mu = 90\%$		$\mu = 85\%$	
Setting	$\gamma \pm$ SE	R^2	$\gamma \pm$ SE	R^2
Implication-only	1.05 ± 0.03	0.995	1.10 ± 0.05	0.989
+ Conjunction	1.67 ± 0.05	0.993	1.79 ± 0.07	0.991
+ Negation	1.77 ± 0.04	0.996	1.82 ± 0.05	0.994
+ Disjunction	2.11 ± 0.09	0.993	2.19 ± 0.11	0.991
+ Quantification	2.60 ± 0.06	0.998	2.73 ± 0.06	0.998



(a) Log-log scaling curves at 85% threshold.

(b) Fitted γ vs. expressiveness at 85% threshold.

Figure 7: Power-law scaling under 85% accuracy threshold (cf. main-text Figure 2 for 90%). The qualitative pattern is unchanged: all five settings follow a power law ($R^2 \geq 0.989$), and γ increases monotonically with expressiveness.

G.3 Different Metrics of Training Compute

The main text uses RL training steps as the measure of training effort T. To check that our scaling claims do not depend on this choice, we refit $T \propto D^\gamma$ under four alternative compute measures: total generated tokens across all rollouts including rejected samples $T_{\text{gen-tok}}$; prompt and response tokens in the kept batch $T_{\text{upd-tok}}$; training FLOPs by the Kaplan rule (Kaplan et al., 2020), $T_{\text{FLOPs}} = 2N\,T_{\text{gen-tok}} + 6N\,T_{\text{upd-tok}}$ with N the model's parameter count; and wall-clock GPU-hours on 8×B200.

Table 6: Power-law fits $T(D) = a \cdot D^\gamma$ for four compute-effort measures. $T_{\text{gen-tok}}$ counts generated tokens over all rollouts (including rejected samples); $T_{\text{upd-tok}}$ counts the prompt and response tokens of the kept batch only; $T_{\text{FLOPs}} = 2N\,T_{\text{gen-tok}} + 6N\,T_{\text{upd-tok}}$ follows the Kaplan rule; GPU-hrs is wall-clock $\times$ 8 B200 GPUs. Because wall-clock time is not comparable across hardware, the GPU-hrs column is fit only on the runs executed on this machine, a subset of the depth points used in the token- and FLOP-based columns.

Setting	$T_{\text{gen-tok}}$		$T_{\text{upd-tok}}$		T_{FLOPs}		GPU-hrs	
	$\gamma \pm$ SE	R^2	$\gamma \pm$ SE	R^2	$\gamma \pm$ SE	R^2	$\gamma \pm$ SE	R^2
Implication-only	1.05 ± 0.03	0.995	1.55 ± 0.04	0.997	1.44 ± 0.03	0.995	1.38 ± 0.05	0.994
+ Conjunction	2.06 ± 0.11	0.981	2.35 ± 0.07	0.994	2.26 ± 0.08	0.991	2.40 ± 0.13	0.989
+ Negation	2.12 ± 0.05	0.996	2.32 ± 0.02	0.999	2.26 ± 0.03	0.999	2.33 ± 0.06	0.998
+ Disjunction	2.42 ± 0.15	0.982	2.72 ± 0.11	0.992	2.62 ± 0.12	0.990	2.61 ± 0.15	0.984
+ Quantification	2.97 ± 0.17	0.987	3.19 ± 0.13	0.994	3.12 ± 0.14	0.992	3.09 ± 0.16	0.990

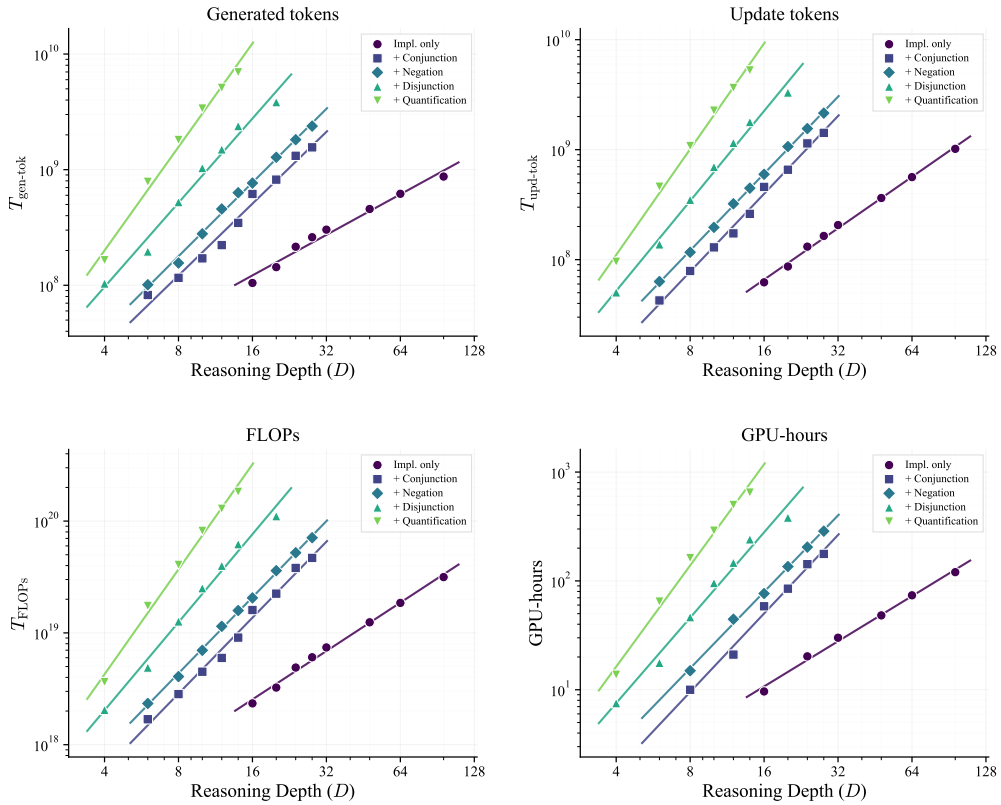


Figure 8: Power-law fits $T(D) = a \cdot D^{\gamma}$ under four alternative compute measures, complementing Table 6. **Top-left:** generated tokens $T_{\text{gen-tok}}$. **Top-right:** kept-batch prompt and response tokens $T_{\text{upd-tok}}$. **Bottom-left:** training FLOPs $T_{\text{FLOPs}} = 2N\,T_{\text{gen-tok}} + 6N\,T_{\text{upd-tok}}$. **Bottom-right:** wall-clock GPU-hours on $8\times$B200 (fit on the runs from this machine only; see Table 6). All four measures preserve the power-law form ($R^2 \geq 0.97$).

Table 6 and Figure 8 report the fits. The power-law form holds in every cell ($R^2 \geq 0.97$), and the overall ranking, *Implication-only* $<$ + *Conjunction*, + *Negation* $<$ + *Disjunction* $<$ + *Quantification*, is preserved across all four measures. Exponents shift upward relative to step counts (e.g., + *Quantification* shifts from $\gamma = 2.60$ to $\gamma = 3.12$ under FLOPs), reflecting that token- and FLOP-based measures aggregate per-instance length on top of the number of gradient updates: prompt length grows with depth and expressiveness, and response length grows further during training. + *Conjunction* and + *Negation* remain within each other's standard-error ranges across all metrics, consistent with the main-text finding that the two impose comparable training burdens.

We retain training steps as our primary measure because they isolate the convergence rate from confounds in per-instance length. Token- and FLOP-based measures aggregate two such confounds: prompts grow structurally with reasoning depth and expressiveness (more axioms, longer rules, quantified templates), and within-training response length grows with both training progress and task difficulty (Figure 15). Conflating these with training effort would mix structural and dynamical properties of the training environment into the scaling measurement.

G.4 Variability across Random Seeds

The main scaling experiment in Section 4.2 uses a single random seed per configuration due to computational constraints. To verify that this does not bias the fitted exponent, we leverage the multi-seed runs from the training-distribution and cross-algorithm experiments (Sections 4.4 and 4.5), which use three seeds per depth on the + *Conjunction* setting. Pooling all three seeds and refitting yields $\gamma = 1.70$ ($R^2 = 0.995$), within 0.02 of the single-seed fit on the same depth points ($\gamma = 1.72$, $R^2 = 0.991$). The marginally higher R^2 under pooling likely reflects per-depth variance being averaged out before fitting. This confirms that our single-seed scaling estimates are reliable.

G.5 Multi-Entity Reasoning as a Potential Confounder

The most expressive setting (+ *Quantification*) introduces multiple design elements simultaneously (Appendix C). A natural concern is that the elevated γ at + *Quantification* could be partially driven by the multi-entity dimension rather than by quantification itself.

To isolate the multi-entity contribution, we run an ablation that varies only the number of entities while keeping $\forall$ and reuse disabled (i.e., starting from the + *Disjunction* setting). As shown in Table 7, training steps to 90% accuracy are essentially identical between the single-entity (1 person) and multi-entity (2 persons) variants across $D \in \{6, 8, 10\}$, with absolute differences of at most 2 steps and no systematic trend with depth.

Table 7: Training steps to 90% accuracy under the + *Disjunction* setting with single vs. multiple entities. Differences are within ± 2 steps and show no systematic trend with depth, indicating that the multi-entity dimension by itself does not measurably increase training difficulty.

D	Single entity (1 person)	Multiple entities (2 persons)
6	21	21
8	39	40
10	68	66

We conclude that the multi-entity design by itself contributes negligibly to training difficulty in this depth range, and that the elevated γ at + *Quantification* therefore reflects the additional difficulty imposed by universal quantification and rule reuse rather than by the multi-entity dimension.

G.6 Prompt Length as a Potential Confounder

Richer expressiveness settings tend to produce longer prompts because their rules contain more structural elements, such as multiple premises, multiple conclusions, and quantified templates. This raises a potential confound: the exponent ordering in Section 4.2 might reflect increasing prompt length rather than the reasoning structure induced by the logical operators. To examine this possibility, we reparameterize training cost in terms of the mean tokenized prompt length L of each training set and fit $T \propto L^{\beta}$ separately for each expressiveness setting. If the observed ordering were primarily an artifact of prompt length, this reparameterization should substantially reduce the systematic separation across expressiveness levels.

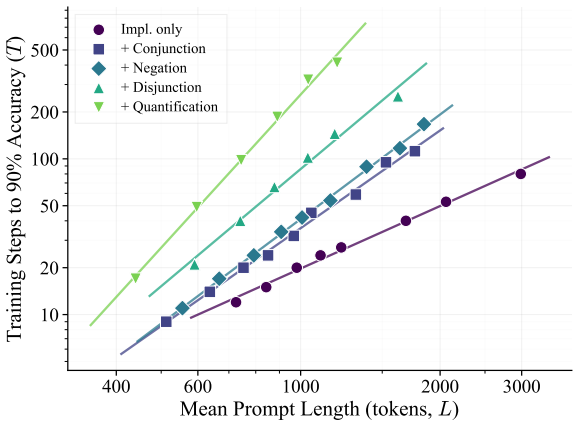


Figure 9: Training cost as a function of mean prompt length L on log–log axes. The fitted exponent β increases monotonically with expressiveness (Table 8).

Figure 9 and Table 8 show that the separation persists after reparameterizing by prompt length. More directly, prompt length does not induce a consistent ordering of training difficulty across settings: *Implication-only* at $D = 96$ has an approximately 3,000-token prompt but reaches the target accuracy in 80 steps, whereas + *Quantification* at $D = 8$ needs only 744 tokens yet requires 98 steps. This

Table 8: Power-law fits of training steps against mean prompt length, $T \propto L^{\beta}$, for each expressiveness setting.

Setting	$\beta \pm$ SE	R^2
Implication-only	1.33 ± 0.04	0.994
+ Conjunction	2.09 ± 0.06	0.994
+ Negation	2.23 ± 0.03	0.998
+ Disjunction	2.50 ± 0.14	0.988
+ Quantification	3.27 ± 0.10	0.996

separation is not driven by disjoint prompt-length regimes: most runs across settings overlap within approximately 500 to 3,000 tokens, while their training costs remain systematically different.

Consistent with this observation, the fitted length exponent β remains monotonically ordered across expressiveness levels, increasing from 1.33 for *Implication-only* to 3.27 for + *Quantification*, in lockstep with the depth exponents γ. The monotone β ordering also replicates on Qwen3-8B ($\beta = 1.27, 1.86, 1.93, 2.24, 3.18$). Together, these results indicate that the observed exponent ordering is not explained by prompt length.

G.7 Controlling for Surface-Form Differences

The five expressiveness settings share the same prompt scaffolding, question format, and entity set (Appendices B and F); their main surface-level difference is the appearance of additional operator words as the rule language becomes richer: *and* for conjunction, *not* for negation, *or* for disjunction, and *for any* for quantification. These operator words cannot simply be removed from the higher-expressiveness settings without changing the underlying formal system. We therefore construct the reverse control: enriching the surface form of the *Implication-only* setting while preserving its original semantics and proof structure. Specifically, each `Implication-only` rule is randomly rendered using one of two "dressed" templates that incorporate the surface form of the richer expressiveness settings:

> **plain:** "If [entity] is [attribute$_1$], then [entity] is [attribute$_2$]."
> **dressed 1:** "If [entity] is [attribute$_1$] and True, then [entity] is [attribute$_2$] or False."
> **dressed 2:** "For any X, if X is [attribute$_1$] and True, then X is [attribute$_2$] or False."

Here, "and True" and "or False" use logical constants that require minimal verification and introduce no additional proof node or inference step. This differs from the genuine + *Conjunction* setting, where each conjunct must itself be established from the facts or preceding rules. Likewise, under the single-entity construction, the "For any X" wrapper is semantically vacuous and does not introduce rule reuse. We do not add *not*, because we found no natural way to introduce negation into the surface form while preserving the original semantics.

This control yields a clear prediction: if the additional operator words themselves were responsible for the increased training difficulty, the dressed *Implication-only* curve should shift toward the richer settings. Table 9 shows the contrary. Across the depth range $D \in \{16, 24, 32, 48, 64\}$, dressing introduces only a small training overhead, with no systematic increase as depth grows. The fitted exponent remains within the uncertainty of the plain *Implication-only* estimate, with $\gamma_{\text{dressed}} = 0.98 \pm 0.06$ versus $\gamma_{\text{plain}} = 1.06 \pm 0.04$, and stays far below $\gamma_{\text{+Conjunction}} = 1.67 \pm 0.05$. At $D = 24$, + *Conjunction* requires 95 updates, $3.8\times$ more training than dressed *Implication-only*, despite the latter using the additional surface markers *or*, *and*, and *for any*. Following the analysis of Appendix G.6, we also fit training cost as a function of prompt length, $T \propto L^{\beta}$. The exponent decreases from $\beta = 1.39 \pm 0.05$ for plain *Implication-only* to 1.19 ± 0.08 for the dressed variant, remaining well below the + *Conjunction* reference of 2.09 ± 0.06; this decrease is expected because dressing inflates prompt length with semantically vacuous template tokens without adding proof-relevant information. Thus, adding richer operator words without their semantic complexity does not reproduce the higher training cost. The exponent hierarchy therefore appears to track the proof-search consequences of the operators, rather than their surface-form realization alone.

Table 9: Surface-form control: *Implication-only* semantics rendered in the full-template syntax of the richer expressiveness levels. * marks reference values for + *Conjunction*, fitted on its full depth range.

Setting		D 16	24	32	48	64	Exponent ± SE	R^2
Impl. only (plain)	T	12	20	27	40	53	$\gamma = 1.06 \pm 0.04$	0.996
	L	725	981	1224	1687	2060	$\beta = 1.39 \pm 0.05$	0.996
Impl. only (dressed)	T	15	25	29	44	61	$\gamma = 0.98 \pm 0.06$	0.990
	L	901	1239	1600	2233	2778	$\beta = 1.19 \pm 0.08$	0.986
+ Conjunction (reference)	T	45	95	—	—	—	$\gamma = 1.67 \pm 0.05^*$	0.993
	L	1055	1528	—	—	—	$\beta = 2.09 \pm 0.06^*$	0.994

G.8 Operator-Swap Ablation

The main hierarchy introduces operators in a fixed cumulative order (conjunction → negation → disjunction → quantification). We further test whether the exponent ordering is an artifact of the particular cumulative path. We construct an additional *+Conjunction+Disjunction* setting that introduces disjunction directly after conjunction, without negation, and train it over the depth range $D \in \{6, 8, 10, 12, 14\}$ using the same training recipe. Table 10 compares this setting with the corresponding configurations from the main hierarchy, with all exponents refit on the same depth range.

Table 10: Operator-swap ablation on the shared depth range $D \in \{6, 8, 10, 12, 14\}$. +Conj, +Conj+Neg, and +Conj+Neg+Disj correspond to the main hierarchy's + *Conjunction*, + *Negation*, and + *Disjunction* settings.

Setting	T at depth D 6	8	10	12	14	$\gamma \pm$ SE	R^2
+Conj	9	14	20	24	32	1.47 ± 0.06	0.996
+Conj+Neg	11	17	24	34	42	1.60 ± 0.04	0.998
+Conj+Disj	11	18	27	43	56	1.95 ± 0.08	0.995
+Conj+Neg+Disj	21	40	66	102	145	2.28 ± 0.02	1.000

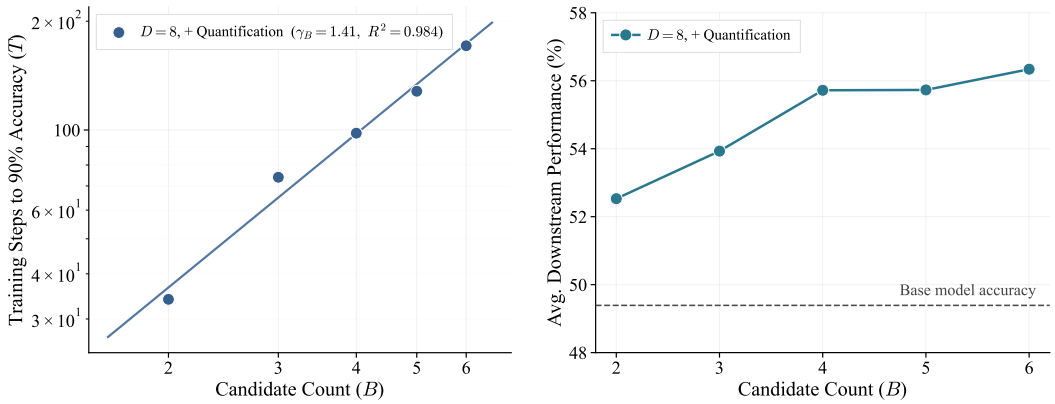
The alternative path *+Conj* → *+Conj+Disj* → *+Conj+Neg+Disj* also exhibits a strictly increasing exponent, $1.47 \to 1.95 \to 2.28$. Equivalently, adding disjunction increases the exponent both without negation ($1.47 \to 1.95$) and with negation ($1.60 \to 2.28$), while adding negation increases it both without disjunction ($1.47 \to 1.60$) and with disjunction ($1.95 \to 2.28$). Thus, this operator-swap ablation shows that the monotone scaling trend is not an artifact of the particular order in the main cumulative hierarchy for negation and disjunction.

H Additional Experimental Results

H.1 Candidate Count Scaling

We examine how the candidate count B (i.e., the number of candidate conclusions per problem) affects both training compute and downstream transfer at fixed depth $D = 8$ under the + *Quantification* setting.

As with depth scaling (Section 4.2), we fit $T = a \cdot B^{\gamma_B}$ via OLS in log-log space. As shown in Figure 10a, training compute follows a power law in B with exponent $\gamma_B = 1.41 \pm 0.10$ and $R^2 = 0.984$. The power-law model again outperforms an exponential fit (ΔAIC $= +7.0$). The sub-quadratic exponent indicates that increasing the number of candidate branches is less costly than increasing reasoning depth under the same expressiveness setting (where $\gamma = 2.60$), consistent with the intuition that deeper proofs require compositionally harder reasoning while additional branches primarily expand the search space.



(a) Log-log scaling curves with power-law fits.

(b) Performance over candidate count.

Figure 10: Effect of candidate count B at fixed depth $D = 8$ under the + *Quantification* setting. **(a)** Training steps to 90% accuracy follow a power law in B ($\gamma_B = 1.41$, $R^2 = 0.984$, $\Delta\text{AIC} = +7.0$ vs. exponential). **(b)** Average downstream performance across the eight reasoning benchmarks of Section 4.3, plotted against candidate count B. Gains saturate quickly: performance rises from 52.5% at $B=2$ to 55.7% at $B=4$ (+3.2 pp), but only an additional +0.6 pp from $B=4$ to $B=6$.

Larger B also yields stronger downstream transfer, though with diminishing returns. As shown in Figure 10b, mean downstream accuracy (averaged over the eight benchmarks of Section 4.3) rises from 52.5% at $B=2$ to 55.7% at $B=4$ (+3.2 pp), then by only an additional +0.6 pp from $B=4$ to $B=6$. This contrasts with the depth axis under the same + *Quantification* setting, where deeper training continues to yield downstream gains throughout our tested range (up to 414 training steps), suggesting that B is a less capacious axis of difficulty than depth. A small number of unprovable candidates is plausibly sufficient to force the model to evaluate each branch, after which additional candidates raise training cost without meaningfully extending the reasoning skills required. We accordingly fix $B = 4$ as the default candidate count in our main experiments.

H.2 Detailed Downstream Results

Table 11 reports the full downstream results for each benchmark and training setting. In Section 4.3, we present the corresponding averaged results across benchmarks for clarity.

H.3 Curriculum at the Most Expressive Setting

The training-distribution comparison in Section 4.4 is conducted under the + *Conjunction* setting. To verify that the benefit of curriculum training is not specific to this setting, we replicate the uniform vs. curriculum comparison at the most expressive + *Quantification* setting.

As shown in Figure 11, curriculum training continues to lower the scaling exponent at + *Quantification*, from $\gamma = 2.60$ to $\gamma = 2.30$, with $R^2 = 0.998$ in both cases. The reduction in exponent ($\Delta\gamma = -0.30$) is comparable to that observed at + *Conjunction* ($\Delta\gamma = -0.37$, from 1.70 to 1.33), and the qualitative effect is preserved: curriculum training yields sample-efficiency gains that compound with proof depth. We conclude that the benefit of curriculum training generalizes across the expressiveness hierarchy.

H.4 Data Distribution Shapes Long-CoT Emergence

The main text shows that the training distribution substantially affects scaling efficiency in the + *Conjunction* setting. Here, we provide the full training trajectories behind this comparison and analyze how the three data distributions shape models' behavior. We focus on response length, actor

Table 11: Detailed downstream results for all benchmarks and training settings. Each entry reports `Avg@8 / Pass@8` in percentage. The last column reports the mean accuracy across the eight benchmarks (i.e., the mean of per-benchmark `Avg@8` scores).

D	Math						Science		Avg
	Math500	AMC23	AIME24	AIME25	Minerva	Olympiad	GPQA-Diamond	MMLU-Pro	
Base model									
–	83.93 / 93.60	65.00 / 87.50	21.67 / 40.00	20.00 / 40.00	40.21 / 50.74	53.32 / 71.36	42.55 / 76.26	68.45 / 78.60	49.39
Implication-only									
16	84.28 / 94.00	64.69 / 87.50	22.92 / 43.33	22.50 / 43.33	40.40 / 49.63	53.43 / 71.07	42.93 / 76.26	67.85 / 87.20	49.88
24	84.85 / 94.60	68.44 / 90.00	24.58 / 46.67	21.67 / 46.67	40.85 / 50.00	54.32 / 73.15	43.62 / 75.25	69.90 / 88.60	51.03
32	84.95 / 95.60	68.44 / 90.00	24.17 / 50.00	22.50 / 46.67	40.63 / 50.00	54.78 / 71.81	44.32 / 72.73	69.05 / 87.80	51.11
48	84.73 / 94.60	67.19 / 87.50	21.67 / 40.00	20.42 / 43.33	40.81 / 51.10	55.51 / 72.70	43.18 / 77.27	69.05 / 88.40	50.32
64	86.20 / 96.00	70.31 / 97.50	28.75 / 43.33	21.25 / 46.67	41.27 / 50.37	55.14 / 73.00	42.55 / 77.78	70.55 / 89.40	52.00
96	85.53 / 95.20	70.31 / 97.50	23.75 / 53.33	24.17 / 50.00	40.40 / 51.84	55.17 / 73.00	44.32 / 77.78	70.03 / 88.80	51.71
+ Conjunction									
8	84.30 / 95.00	70.63 / 95.00	26.25 / 46.67	22.50 / 50.00	39.75 / 50.74	53.75 / 71.51	42.36 / 75.25	69.10 / 87.80	51.08
12	85.53 / 95.40	69.69 / 95.00	30.00 / 60.00	20.00 / 40.00	40.40 / 49.63	54.67 / 73.74	41.60 / 77.27	69.55 / 88.80	51.43
16	85.60 / 95.00	71.88 / 95.00	24.17 / 43.33	24.17 / 56.67	40.53 / 51.10	55.42 / 73.15	43.06 / 78.79	70.53 / 88.00	51.92
20	85.45 / 95.20	70.94 / 92.50	26.67 / 53.33	21.25 / 46.67	39.66 / 49.63	55.25 / 74.18	44.44 / 79.29	69.20 / 87.80	51.61
24	85.83 / 94.80	72.81 / 97.50	24.17 / 40.00	19.58 / 36.67	40.76 / 50.37	55.84 / 71.81	45.52 / 77.27	69.68 / 87.60	51.77
28	86.15 / 95.60	70.94 / 97.50	27.92 / 53.33	22.08 / 43.33	40.63 / 50.37	55.40 / 74.33	42.42 / 76.26	70.15 / 88.20	51.96
+ Negation									
8	84.98 / 95.20	71.88 / 95.00	22.92 / 50.00	20.42 / 43.33	39.52 / 50.00	54.56 / 73.44	43.06 / 76.77	69.55 / 89.20	50.86
12	85.83 / 96.00	71.56 / 95.00	26.25 / 53.33	20.42 / 43.33	40.53 / 50.74	55.77 / 73.44	44.00 / 76.26	69.25 / 87.40	51.70
16	86.13 / 95.60	72.19 / 90.00	29.17 / 50.00	20.42 / 36.67	40.72 / 52.21	56.64 / 73.89	43.94 / 75.76	70.10 / 89.00	52.41
20	86.85 / 95.40	74.38 / 97.50	30.00 / 53.33	22.92 / 50.00	41.73 / 52.21	56.81 / 73.59	46.72 / 80.30	71.93 / 89.60	53.92
24	87.43 / 96.40	72.81 / 92.50	30.00 / 53.33	22.50 / 50.00	40.72 / 51.47	57.07 / 74.63	44.07 / 75.76	72.63 / 89.60	53.40
28	87.38 / 97.00	73.75 / 92.50	27.08 / 46.67	20.83 / 43.33	40.95 / 51.10	57.62 / 76.41	45.39 / 78.28	72.08 / 87.60	53.13
+ Disjunction									
6	85.48 / 95.40	70.31 / 90.00	24.58 / 46.67	22.92 / 46.67	39.80 / 50.37	54.95 / 73.15	42.30 / 77.27	68.63 / 87.80	51.12
8	86.75 / 96.20	70.00 / 95.00	25.83 / 50.00	25.42 / 53.33	41.41 / 50.00	56.88 / 73.15	46.15 / 78.79	70.53 / 88.60	52.87
10	87.40 / 96.40	71.56 / 95.00	27.50 / 50.00	25.42 / 50.00	40.81 / 51.10	57.60 / 74.33	45.08 / 79.29	72.18 / 88.40	53.44
12	87.20 / 95.60	74.38 / 95.00	30.00 / 50.00	27.50 / 53.33	41.27 / 52.21	57.86 / 73.89	44.07 / 77.78	72.00 / 89.60	54.29
14	87.78 / 96.00	76.56 / 95.00	33.33 / 63.33	25.83 / 50.00	41.27 / 51.84	58.88 / 76.41	45.27 / 80.30	72.85 / 89.60	55.22
20	88.63 / 96.60	76.25 / 95.00	35.00 / 63.33	27.92 / 56.67	41.82 / 51.84	59.20 / 75.82	46.53 / 81.31	74.08 / 89.40	56.18
+ Quantification									
4	85.43 / 95.40	69.38 / 92.50	24.58 / 46.67	23.33 / 50.00	40.17 / 50.37	55.06 / 73.15	41.86 / 77.78	69.47 / 88.20	51.16
6	86.75 / 96.80	70.00 / 90.00	28.33 / 50.00	21.67 / 40.00	40.67 / 51.47	56.82 / 74.78	45.79 / 75.25	71.53 / 88.60	52.69
8	88.15 / 96.40	78.12 / 97.50	35.00 / 56.67	25.00 / 53.33	41.68 / 51.84	58.90 / 75.22	46.34 / 76.77	72.53 / 89.80	55.72
10	88.60 / 97.20	75.94 / 97.50	34.58 / 60.00	25.83 / 53.33	42.33 / 51.84	60.05 / 76.11	45.27 / 76.77	74.73 / 89.80	55.92
12	88.70 / 96.60	80.90 / 97.50	35.00 / 60.00	27.92 / 46.67	42.42 / 51.84	61.13 / 77.45	49.12 / 75.76	74.73 / 89.20	57.49
14	91.48 / 98.00	80.63 / 97.50	43.75 / 66.67	31.25 / 53.33	43.70 / 52.57	64.39 / 79.08	48.17 / 72.22	77.00 / 89.80	60.05

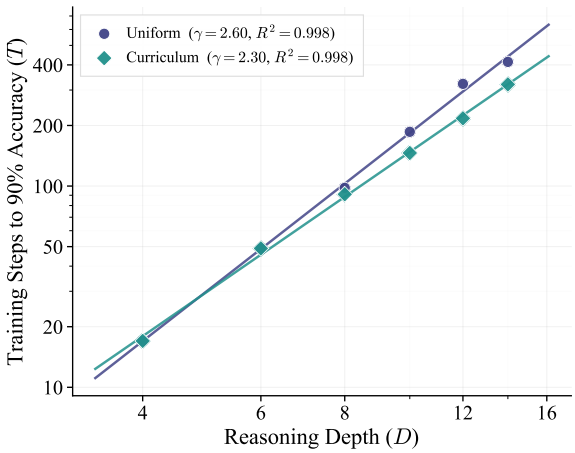
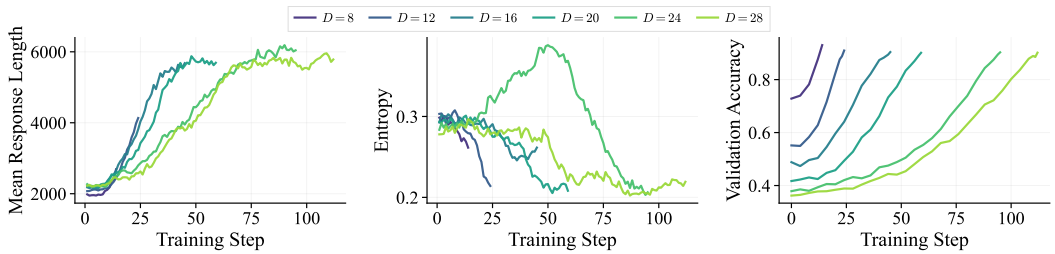


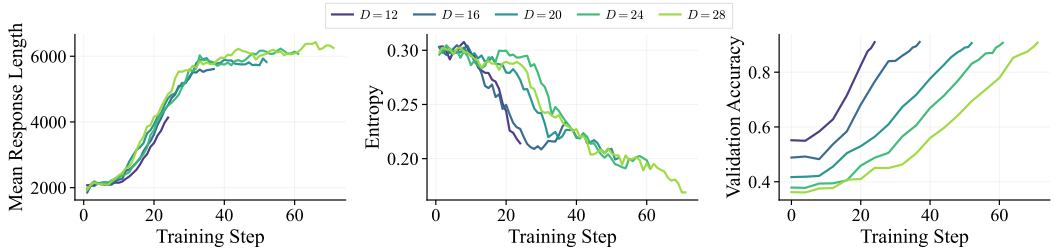
Figure 11: Uniform vs. curriculum training under the most expressive + *Quantification* setting. Curriculum training reduces the scaling exponent from $\gamma = 2.60$ to $\gamma = 2.30$ ($R^2 = 0.998$ in both cases), confirming that the benefit observed at + *Conjunction* (Section 4.4) extends to the most expressive logic in our hierarchy.

entropy, and validation accuracy, which together reveal when longer reasoning traces emerge and how this transition relates to the observed scaling behavior.

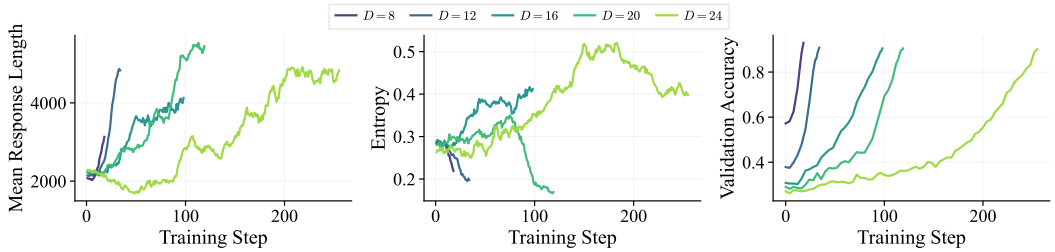
Figure 12 shows that curriculum training substantially eliminates the delay in long-CoT emergence across depths. Under the curriculum regime, runs at different target depths enter the long-CoT



(a) Uniform.



(b) Curriculum.



(c) Difficult-only.

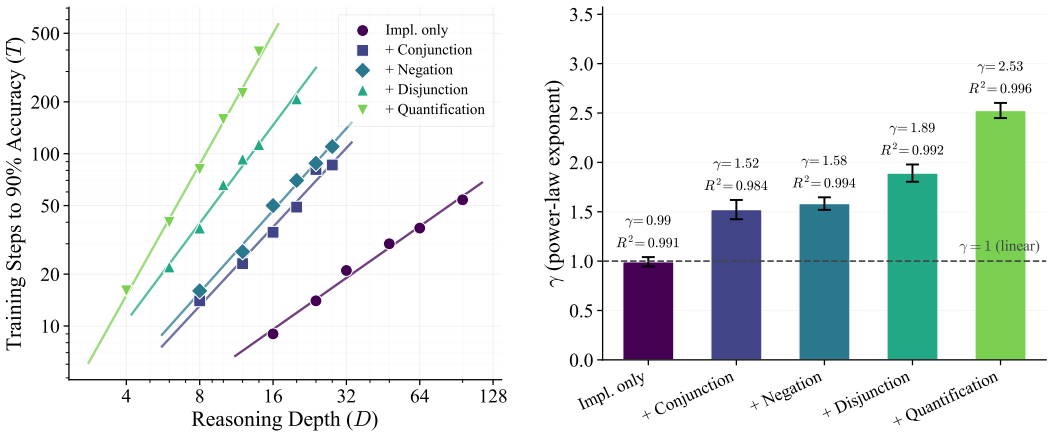
Figure 12: Training trajectories on + *Conjunction* under three data distributions. Each row: response length (left), entropy (middle), validation accuracy (right); legend (training depth D) is shared across the three sub-panels. Note that axis ranges may differ across rows to improve readability.

regime within a relatively narrow range of training steps; this length expansion is accompanied by a sharper entropy drop and earlier improvement in validation accuracy. By contrast, under the uniform distribution, long-CoT emergence becomes increasingly delayed as the target depth grows, with a particularly large gap between $D = 20$ and $D = 24$; the corresponding entropy decrease and accuracy rise are likewise shifted to much later training steps. The difficult-only regime exhibits a similar but more pronounced pattern, especially at larger depths, where response length remains short and entropy stays elevated for a longer warm-up period before accuracy begins to improve.

Together, these dynamics suggest that the distribution effect in the main text is closely tied to when long-CoT behavior emerges. Curriculum training appears to provide a smoother path into longer reasoning: the policy first learns useful reasoning patterns on shallower instances and can then extend them as deeper instances are introduced. By contrast, uniform and difficult-only training tend to remain longer in a short-response, high-entropy regime before transitioning to long-CoT behavior. This delayed transition is consistent with their larger scaling exponents, whereas the earlier and more synchronized transition under curriculum training helps explain its improved scaling efficiency.

H.5 Cross-Scale and Cross-Model Replication

Cross-scale replication. To assess whether the scaling behavior reported in the main text generalizes beyond the 4B scale, we replicate the central scaling experiment of Section 4.2 on Qwen3-8B. Figure 13 shows the log-log scaling curves and the fitted exponents. The scaling behavior at 8B closely mirrors that at 4B. All five expressiveness settings continue to follow a clean power law



(a) Log-log scaling curves with power-law fits.

(b) Fitted exponent γ vs. expressiveness of the logic.

Figure 13: Cross-scale replication on Qwen3-8B. **(a)** Training steps to convergence vs. reasoning depth across five expressiveness levels on log-log axes. Solid lines show power-law fits $T \propto D^{\gamma}$. **(b)** Fitted γ increases monotonically when expressiveness increases from *Implication-only* ($\gamma = 0.99$) to the + *Quantification* setting ($\gamma = 2.53$), mirroring the 4B picture in Figure 2 but with systematically smaller exponents. Error bars denote ± 1 standard error of the fitted exponent.

in proof depth, with $R^2 \geq 0.98$. The fitted exponent γ remains monotonically increasing with expressiveness, ranging from 0.99 for *Implication-only* to 2.53 for + *Quantification*, and the small gap between + *Conjunction* ($\gamma = 1.52$) and + *Negation* ($\gamma = 1.58$) is also preserved. Compared with the 4B model, the 8B exponents are systematically smaller across all settings (e.g., 1.52 vs. 1.67 under + *Conjunction*), while the relative ordering remains unchanged. This suggests that the qualitative scaling pattern is preserved across model scales.

As an additional downstream check, the Qwen3-8B model trained at the deepest + *Quantification* setting ($D = 14$) improves the eight-benchmark average from 50.93% to 60.53%, a +9.60 percentage-point gain, suggesting a transfer trend similar to that observed for Qwen3-4B.

Cross-model replication. We further conduct a partial replication on the non-Qwen-family Ministral3-3B-Instruct-2512 to examine whether the observed exponent ordering is specific to the Qwen family. Due to compute constraints, we evaluate three representative settings: *Implication-only*, + *Conjunction*, and + *Negation*. As shown in Figure 14, all three settings exhibit clear power-law scaling, and the fitted exponent again increases monotonically as the training setting becomes more expressive, from 0.87 ± 0.06 to 1.37 ± 0.13 and 1.39 ± 0.08, respectively. This provides complementary evidence that the qualitative exponent ordering is not unique to the Qwen family.

Together, the Qwen3-8B and Ministral3-3B results provide complementary evidence that the observed exponent hierarchy persists across both model scale and model family.

H.6 Matched-Budget Alternative-Training Controls

The downstream gains in Section 4.3 could in principle arise from factors other than learning to solve the underlying reasoning task. For example, they might arise from mere exposure to the synthetic reasoning data, or from generic RL-induced parameter drift that requires no correctness signal. We test these two controls under a matched training budget: supervised fine-tuning (SFT) on gold proof traces, and RL with correctness-independent rewards. Both controls use the + *Quantification*, $D = 8$ setting, where the RL run reaches the target accuracy in $T = 98$ steps.

Setup. (i) *SFT control.* We train Qwen3-4B on the same prompts using Z3-verified gold proof traces, obtained via breadth-first search and rendered in natural language. Each response contains the complete derivation and ends with the same answer format used in RL. We use a learning rate of 10^{-5} with cosine decay and match the RL mini-batch size. (ii) *Correctness-independent RL controls.* We use the same training data, model, and DAPO recipe as in the original RL run, replacing only the correctness reward with a format-only or random reward. The format-only variant assigns

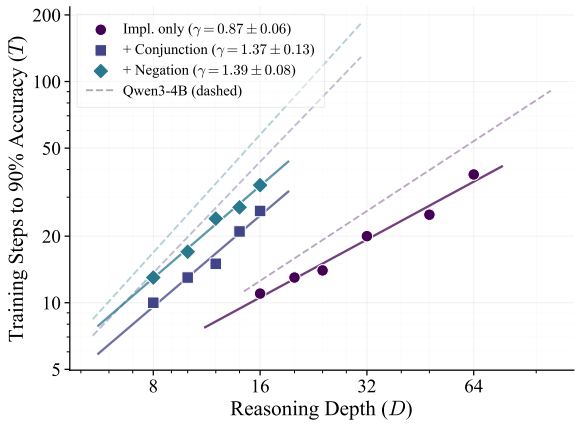


Figure 14: Partial cross-model replication on Ministral3-3B-Instruct-2512. Solid lines show power-law fits for three representative expressiveness settings; dashed lines show the corresponding Qwen3-4B fits for reference. The fitted exponent increases monotonically from *Implication-only* ($\gamma = 0.87$) to + *Conjunction* ($\gamma = 1.37$) and + *Negation* ($\gamma = 1.39$), preserving the qualitative ordering across model families.

Table 12: Matched-budget alternative-training controls (+ *Quantification*, D=8; Avg@8). Δ is the change in the eight-benchmark average relative to the base model; "In-dist" is the held-out SCALELOGIC test set.

Model	Math500	AMC23	AIME24	AIME25	Minerva	Olympiad	GPQA-D	MMLU-Pro	Avg	Δ	In-dist
Qwen3-4B (base)	83.93	65.00	21.67	20.00	40.21	53.32	42.55	68.45	49.39	—	34.35
SFT (1 ep.)	60.22	29.69	2.08	1.67	30.28	25.09	10.35	40.92	25.04	−24.35	89.84
SFT (2 ep.)	57.65	28.75	2.50	0.42	28.58	24.04	9.22	41.50	24.08	−25.31	**97.30**
Format-only RL	83.70	64.38	21.25	22.50	39.94	51.93	44.07	67.10	49.36	−0.03	37.43
Random-reward RL	83.45	60.94	20.83	17.50	38.92	52.24	43.62	66.97	48.06	−1.33	32.17
RL (ours)	**88.15**	**78.12**	**35.00**	**25.00**	**41.68**	**58.90**	**46.34**	**72.53**	**55.72**	**+6.33**	90.30

reward for agreement with the required answer format regardless of whether the selected answer is correct, whereas the random-reward variant assigns rewards independently of answer correctness. (iii) *Evaluation.* All checkpoints are evaluated with the protocol of Section 4.1: Avg@8 on the eight downstream benchmarks, together with accuracy on the held-out SCALELOGIC test set. (iv) *Budget matching.* An RL checkpoint at T steps performs gradient updates over $T \times 256$ prompts, each with 8 rollouts, corresponding to $T \times 256 \times 8$ training sequences (approximately 200k sequences at $T = 98$). We match the controls by the total number of training sequences: for SFT, the 100k gold traces correspond to two epochs; for the correctness-independent RL controls, which uses the same RL recipe, we compare models at the same training step T.

Results. As shown in Table 12, neither alternative-training control reproduces the downstream transfer of our model.

SFT successfully learns the in-distribution task but fails to transfer. At the matched budget, SFT reaches 97.30% on the held-out SCALELOGIC test set, exceeding the 90.30% achieved by RL. Nevertheless, its average downstream performance falls from 49.39 for the base model to 24.08 (-25.31 pp). This collapse is already present after one epoch: despite reaching 89.84% in-distribution accuracy, SFT scores only 25.04 downstream (-24.35 pp). The lack of transfer therefore cannot be attributed to insufficient SFT convergence or prolonged SFT training. This result indicates that SFT induces severe task specialization and forgetting of general reasoning capabilities as observed in prior work (Lobo et al., 2025).

Correctness-independent RL also fails to transfer. With the same data, model, optimizer, and training budget as our RL recipe, format-only and random rewards yield downstream averages of 49.36 (-0.03 pp) and 48.06 (-1.33 pp), respectively; both show essentially no improvement over the base model. Their in-distribution accuracies also remain near the base level (37.43% and 32.17%, versus 34.35%). Moreover, both controls shorten the model's responses: the average length decreases from 1.7k tokens initially to 1.2k under the random reward and 0.9k under the format-only reward. In

Table 13: Proof-faithfulness audit (+ *Quantification*, D=8): 500 instances $\times$ 4 rollouts per model. The audit is conditioned on rollouts with correct final answers. A symbolic theorem prover checks every extracted asserted step against the ground truth.

	RL	SFT
Correct rollouts audited	1768 / 2000	1941 / 2000
Asserted steps per rollout (mean)	50.5	6.6
Asserted steps that are entailed truths	94.6%	98.2%
Final answer explicitly derived	96.5%	97.7%
Final answer supported by prior assertions	89.4%	97.7%

contrast, correctness-guided RL increases the response length to 5.6k tokens by the end of training, consistent with the acquisition of more extensive reasoning.

Together, these controls show that, among the matched alternatives we test, only correctness-guided RL reproduces the downstream gains.

H.7 Proof-Faithfulness Audit

Our training objective uses only a final-answer exact-match reward (Section 3.3), without supervising intermediate reasoning steps. Strong outcome-level performance therefore does not by itself establish whether the generated reasoning is logically valid or whether the final answer is supported by the preceding trace. We thus conduct a direct proof-faithfulness audit.

For the + *Quantification*, D=8 checkpoint, we sample four rollouts on each of 500 fresh instances and audit every rollout with a correct final answer. A conservative rule-based parser extracts atomic assertions of the form "[entity] is (not) [attribute]", excluding hypothetical statements (markers such as "suppose" or "might"); a symbolic theorem prover then checks every extracted assertion for entailment against the ground truth. As a calibration reference, we apply the same audit to the SFT model of Appendix H.6, which is trained directly on gold proof traces.

As shown in Table 13, across 1,768 correct RL rollouts, the model produces approximately 89k asserted steps (50.5 per rollout), of which 94.6% are logically entailed truths. Moreover, 96.5% of correct rollouts explicitly derive the selected final answer, and in 89.4% the final answer is entailed by previously established assertions. The remaining 10.6% that fail our strict grounding criterion should be viewed as a conservative upper bound, since valid elimination arguments and reasoning with skipped intermediate steps can also be counted as ungrounded by this validation methodology. The SFT model, trained directly on gold proof traces, achieves 98.2% entailed assertions and 97.7% on both explicit final-answer derivation and grounding, as expected under direct reasoning supervision. These results show that, despite receiving only an outcome reward, the RL-trained model produces predominantly valid and answer-supporting reasoning traces.

H.8 Evaluation on External Logical-Reasoning Benchmarks

To evaluate whether the logical reasoning improvements induced by training on ScaleLogic transfer beyond the synthetic training distribution, we evaluate the + *Quantification*, $D = 8$ model on two established logical-reasoning benchmarks: the harder Logical Deduction-5/7 subsets of BBH (Suzgun et al., 2023), and 500 randomly sampled ZebraLogic (Lin et al., 2025) test examples in multiple-choice mode. All checkpoints are evaluated using the same protocol as in Section 4.1, and we report Avg@8.

As shown in Table 14, our model substantially outperforms the base model on both benchmarks, improving Avg@8 from 84.55 to 93.35 on BBH and from 64.70 to 75.10 on ZebraLogic. In contrast, SFT on the same data causes severe catastrophic forgetting, reducing performance to 35.85 and 4.12, respectively, while the matched-budget correctness-independent RL controls (Appendix H.6) produce only small and inconsistent changes relative to the base model (85.15/62.48 with the format reward and 84.02/61.52 with the random reward). Overall, this demonstrates the good generalization

Table 14: Performance on external logical-reasoning benchmarks (Avg@8). SFT and correctness-independent RL are the matched-budget controls from Appendix H.6.

Model	BBH	ZebraLogic
Qwen3-4B (base)	84.55	64.70
SFT	35.85	4.12
Format-only RL	85.15	62.48
Random-reward RL	84.02	61.52
RL (ours)	**93.35**	**75.10**

Table 15: Downstream comparison between SCALELOGIC and SATURN 3-SAT training (Avg@8). T is the number of training steps to reach 90% validation accuracy on the corresponding task.

Setting	T	Math500	AMC23	AIME24	AIME25	Minerva	Olympiad	GPQA-D	MMLU-Pro	Avg
Qwen3-4B (Base)	—	83.93	65.00	21.67	20.00	40.21	53.32	42.55	68.45	49.39
3-SAT (k=5, l=15)	19	85.58	67.19	24.17	20.00	40.95	55.29	42.23	68.85	50.53
3-SAT (k=8, l=24)	52	86.33	75.31	28.33	22.50	41.22	57.53	41.67	69.60	52.81
3-SAT (k=10, l=30)	66	86.67	76.25	30.00	24.17	41.50	58.10	45.27	71.28	54.15
3-SAT (k=12, l=36)	105	87.10	77.20	30.80	22.90	40.80	58.40	43.80	71.80	54.10
Ours (+ Quantification, D=8)	98	**88.15**	**78.12**	**35.00**	**25.00**	**41.68**	**58.90**	**46.34**	**72.53**	**55.72**

capability of our model on out-of-distribution logical reasoning tasks, providing additional evidence that the gains extend beyond the SCALELOGIC training distribution.

H.9 COMPARISON WITH AN ALTERNATIVE SYNTHETIC-DATA BASELINE

We further compare SCALELOGIC against SATURN (Liu et al., 2025a), a synthetic-data framework for Boolean satisfiability (SAT) reasoning. SATURN trains language models on synthetic 3-SAT instances with verifiable rewards, where task difficulty is controlled by the number of variables and clauses. To enable a controlled comparison, we use the same Qwen3-4B model and DAPO training recipe as in our main experiments. Starting from $(k, l) = (5, 15)$, the configuration used in the original work, we keep the clause-to-variable ratio fixed at $l/k = 3$ and progressively increase the problem size to $(8, 24)$, $(10, 30)$, and $(12, 36)$.

As shown in Table 15, increasing the SAT problem size initially improves downstream transfer, but the gains plateau at $(k, l) = (10, 30)$, where the average score reaches 54.15; further scaling to $(12, 36)$ slightly decreases the score to 54.10. Moreover, the $(12, 36)$ setting requires more training updates than our + *Quantification*, $D = 8$ setting to reach 90% validation accuracy (105 vs. 98), while still yielding lower downstream performance (54.10 vs. 55.72). Compared with the rapid saturation observed for SAT, our framework continues to benefit from increased task difficulty: raising the depth from $D = 8$ to $D = 14$ further improves the average score from 55.72 to 60.05 (Section 4.3). This controlled comparison suggests that, under the matched model and RL recipe, SCALELOGIC provides stronger downstream transfer and supports more effective scaling with task difficulty.

I TRAINING DYNAMICS OF THE MAIN EXPERIMENTS

To accompany the training-step scaling results reported in the main text, we visualize the training trajectories of the five main experimental settings. For each setting, we plot three quantities as a function of the training step: (i) the mean response length, (ii) the actor entropy, and (iii) the validation accuracy (the same quantity used to define the training compute T in the main paper).

Across the five settings, Figure 15 reveals a consistent set of training dynamics. First, validation accuracy eventually reaches the 0.9 threshold for all trained depths, but deeper problems require systematically more training steps to reach this threshold. These data points are exactly the quantities used to construct the scaling curves in the main text. Second, response length tends to grow during training, especially at larger depths and in more expressive settings, indicating that RL often induces longer reasoning traces as the task requires deeper proof search. Third, actor entropy generally decreases as training progresses, suggesting that the policy gradually concentrates on more reliable

reasoning patterns; on harder configurations, this decrease is sometimes preceded by a longer exploratory phase. Together, these trajectories show that the measured scaling behavior is not an artifact of isolated convergence points, but reflects broader changes in accuracy, reasoning length, and policy uncertainty throughout training.

J Evaluation of Frontier LLMs on ScaleLogic

To verify that our synthetic tasks present a genuine reasoning challenge even for frontier models, we evaluate six strong LLMs in the + *Quantification* setting with $B = 4$ across reasoning depths $D \in \{4, 6, \ldots, 32\}$. For each depth, we sample 30 tasks and report the number of correct answers. The evaluated models include both non-thinking LLMs (GPT-4o (Hurst et al., 2024) and DeepSeek-V3.1 (Liu et al., 2024)) and LRMs (GPT-5.4-mini (Singh et al., 2025), DeepSeek-R1 (Guo et al., 2025a), Kimi-K2.5 (Team et al., 2026), and Qwen3.5-397B-A17B (Team, 2026)). We use greedy decoding with temperature $T = 0$ whenever temperature control is supported. For GPT-5.4-mini, whose temperature cannot be modified, we use its default setting and set the reasoning effort to "high". To minimize failures due to truncated outputs, we set the maximum response length to 65,536 tokens.

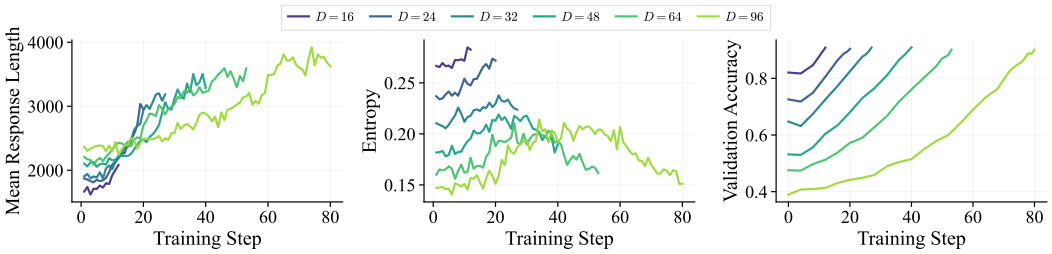
As shown in Figure 16, increasing reasoning depth leads to clear performance degradation across all evaluated models. Non-reasoning LLMs fail rapidly: GPT-4o and DeepSeek-V3.1 reach near-random accuracy by $D = 8$ and $D = 20$. Although LRMs are more robust, with Qwen3.5-397B-A17B remaining above random even at $D = 32$, they also exhibit marked degradation beyond $D = 12$. These results suggest that the most expressive tasks of ScaleLogic remain challenging even for frontier reasoning models.

Notably, Figure 5 shows that targeted RL training on ScaleLogic substantially closes the in-distribution gap between a 4B model and much larger frontier models evaluated zero-shot. At $D = 28$, the RL-trained 4B model maintains 40% accuracy, matched only by Qwen3.5-397B-A17B among the evaluated frontier models. Thus, the comparison highlights distribution-specific gains from targeted RL, while the continued degradation of all models at larger depths confirms that the depth axis captures a meaningful reasoning challenge.

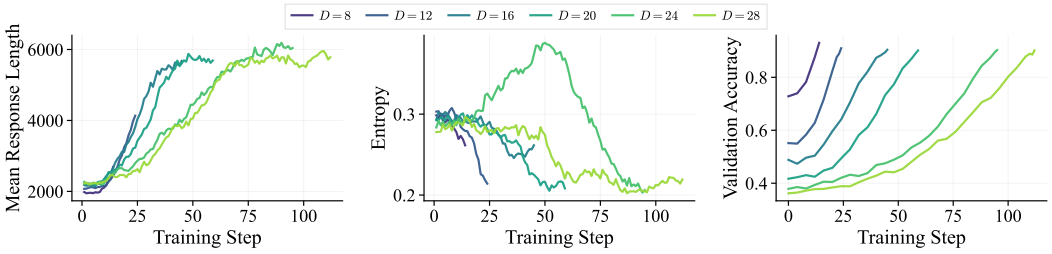
K Qualitative Examples

Beyond the aggregate accuracy gains in Section 4.3, Figure 17 illustrates how RL post-training changes the model's reasoning behavior. We compare our model with the base model on MATH-500 #80, a counting problem over a parameterized logarithmic equation whose answer is 501. Our model solves the problem in $7/8$ rollouts, whereas the base model solves it in $0/8$.

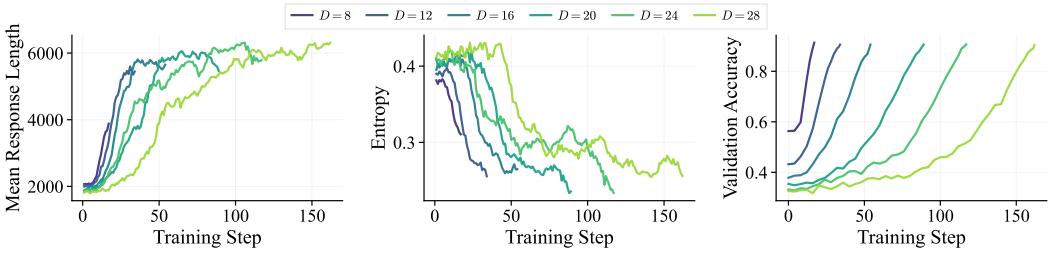
Both models reduce the equation to $x^2 + (4 - k)x + 4 = 0$ and consider the discriminant condition $\Delta = 0$, which only gives $k \in \{0, 8\}$. The base model stops there and answers 2. In contrast, our model identifies a second case: the quadratic may have two real roots, but only one lies in the logarithm domain ($kx > 0$, $x > -2$). It then splits into $k > 0$ and $k < 0$, uses Vieta's formulas and the sign of $f(-2)$ to verify the valid-domain cases, and concludes that every negative integer k also works, yielding $500 + 1 = 501$. This example suggests that training on ScaleLogic encourages a more systematic branching strategy: the model searches for hidden cases, enumerates alternatives, and verifies each branch under explicit constraints.



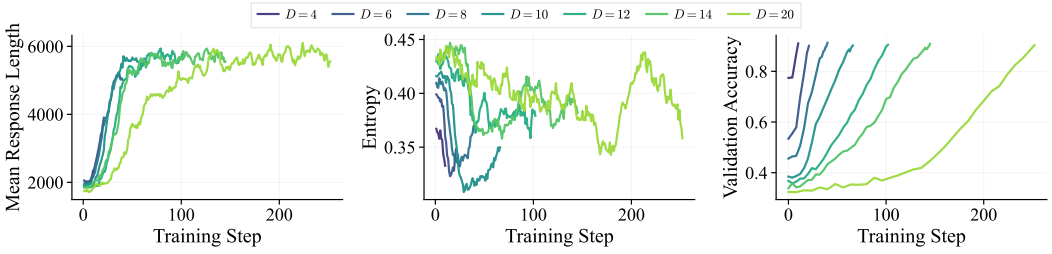
(a) Impl. only



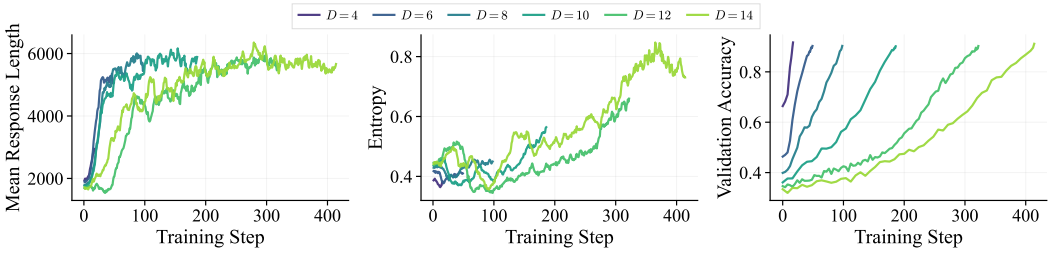
(b) + Conjunction



(c) + Negation



(d) + Disjunction



(e) + Quantification

Figure 15: Training trajectories for all five settings. Each row: response length (left), entropy (middle), validation accuracy (right); legend (training depth D) is shared across the three panels. Note that axis ranges may differ across rows to improve readability.

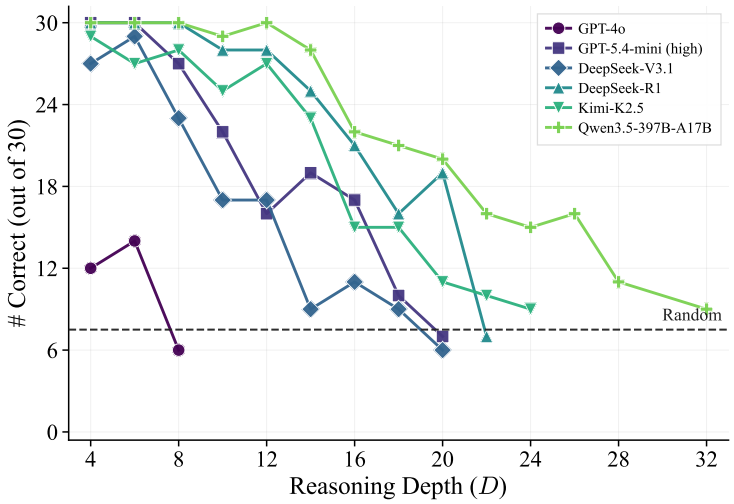


Figure 16: Number of correct answers out of 30 for frontier LLMs on SCALELOGIC (+ *Quantification* setting, $B=4$) as a function of reasoning depth. All models degrade with increasing depth.

Question (MATH-500 #80)

Find the number of integer values of k in the closed interval $[-500, 500]$ for which the equation $\log(kx) = 2\log(x+2)$ has exactly one real solution.

Gold answer: 501

Ours (Qwen3-4B + RL on synthetic logic) 7/8 correct

We are given the equation $\log(kx) = 2\log(x+2)$. We are to find the number of integer values of $k \in [-500, 500]$ for which this equation has **exactly one real solution**.

Step 1: Understand the domain
The logarithm is only defined for positive arguments, so $kx > 0$ and $x + 2 > 0 \Rightarrow x > -2$. Depending on the sign of k:

- If $k > 0$, then $x > 0$.
- If $k < 0$, then $-2 < x < 0$.

If $k = 0$, then $\log(kx) = \log 0$ is undefined, so $k = 0$ is **not allowed**.

Step 2: Solve the equation
$\log(kx) = \log((x+2)^2)$, so if both sides are defined, $kx = (x+2)^2$, giving

$$x^2 + (4-k)x + 4 = 0.$$

For "exactly one real solution" we set the discriminant to zero:

$$\Delta = (4-k)^2 - 16 = 0 \;\Rightarrow\; k \in \{0, 8\}.$$

$k = 0$ is invalid; only $k = 8$ remains. *But this seems too few.*
Wait — but maybe there's more to it. Even if the discriminant is **not zero**, the quadratic might still have **exactly one real solution in the domain**: if the quadratic has two real roots, only one of them might lie in the domain of the original equation. So we need to be more careful.

Step 3: Re-express the problem
The number of real solutions of the original equation equals the number of roots of $x^2 + (4-k)x + 4 = 0$ that lie in the domain. Two cases must be checked:

1. Discriminant $= 0$, and the unique root is in the domain.
2. Discriminant > 0, and **only one** of the two roots is in the domain.

Case 1: Discriminant is zero $\Rightarrow$ one real solution
$k = 0$: invalid. $k = 8$: $(x-2)^2 = 0 \Rightarrow x = 2$, which lies in the domain $x > 0$. So $k = 8$ is valid. [. . .]

Case 2: Quadratic has two real solutions, but only one in the domain
By Vieta's formulas, $x_1 + x_2 = k - 4$ and $x_1 x_2 = 4 > 0$, so both roots share a sign.

Subcase A: $k > 0$**.**
Both roots are positive, so both lie in the domain $x > 0 \Rightarrow$ **two solutions**, not acceptable. [. . .] Therefore no $k > 0$ works.

Subcase B: $k < 0$**.**
Sum $k - 4 < -4$, product $4 > 0$, so both roots are **negative**. The domain is $-2 < x < 0$. We want exactly one root in $(-2, 0)$ and the other ≤ -2.
Let $f(x) = x^2 + (4 - k)x + 4$. Since the parabola opens upward and both roots are negative, the desired configuration is equivalent to $f(-2) < 0$. Compute:

$$f(-2) = 4 - 2(4 - k) + 4 = 2k.$$

Hence $f(-2) < 0 \Leftrightarrow k < 0$. So **for every** $k < 0$, the equation has exactly one real solution. [. . .]

Final Answer.
The valid integers $k \in [-500, 500]$ are:

- All negative integers from -500 to -1: 500 values.
- Plus $k = 8$ from Case 1.

$$\boxed{500 + 1 = 501}$$

Base (Qwen3-4B, no RL) 0/8 correct

We are given $\log(kx) = 2\log(x + 2)$ and we are to find the number of **integer values of** k in $[-500, 500]$ for which the equation has **exactly one real solution**.

Step 1: Understand the equation
Using $\log(a) = \log(b) \Rightarrow a = b$ we obtain

$$kx = (x + 2)^2.$$

[. . .]

Step 2: Solve the equation
$x^2 + (4 - k)x + 4 = 0$. We want exactly one real solution, which happens when the **discriminant is zero**:

$$\Delta = (4 - k)^2 - 16 = 0 \;\Rightarrow\; 4 - k = \pm 4.$$

Solving: $k = 0$ or $k = 8$.
So the values of k for which the equation has exactly one real solution are

$$k = 0 \quad \text{and} \quad k = 8.$$

Step 3: Check if these values are in the interval $[-500, 500]$
Both 0 and 8 are in $[-500, 500]$, so they are valid.

Step 4: Are there any other values of k **that could make the equation have exactly one real solution?**
We need to make sure that **only** when the discriminant is zero does the equation have exactly one real solution. In general, a quadratic has two real solutions if $\Delta > 0$, one if $\Delta = 0$, none if $\Delta < 0$. So the only values of k are those for which the discriminant is zero, namely $k = 0$ and $k = 8$.

Final Answer.

$$\boxed{2}$$

Figure 17: MATH-500 #80: full reasoning trace of our model (top, $7/8$ correct) and the base model (bottom, $0/8$ correct). Wrong steps in the base model are highlighted in red; "[. . .]" marks omitted verbose passages.