THE WEAKEST LINK: DISTILLING LLM REASONING WITH WORST-CASE CONSTRAINED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Distilling the reasoning capabilities of large language models (LLMs) into smaller students is a central challenge for efficient deployment. Current approaches face a fundamental tension: optimizing purely for verifiable task rewards (e.g., via GRPO) leads to *reward hacking*, where students arrive at correct final answers through flawed intermediate logic, while regularizing with soft divergence penalties against a teacher (e.g., KL-based distillation) dilutes task performance and, critically, allows the student to *compensate* for severe logical violations at one step with high teacher agreement at others. We argue that this averaging is fundamentally misaligned with the nature of reasoning: a chain-of-thought is only as valid as its *weakest link*. Motivated by this observation, we formulate reasoning distillation as a constrained reinforcement learning problem in which the task reward is maximized subject to a worst-case constraint on the teacher log-likelihood along every prefix of the trajectory. To avoid the prohibitive cost of dual Lagrangian solvers and the test-time teacher dependence of state-augmented methods such as Saute, we derive an unaugmented constrained MDP whose reward transformation preserves the hard-constraint semantics, admits a low-variance policy gradient decomposition into single-step and long-term terms, and provably satisfies the worst-case constraint almost surely in the penalty limit. Through extensive experiments on mathematical reasoning and code generation tasks, we demonstrate that our method significantly expands the accuracy-fidelity Pareto front. By matching the high Final Answer Correctness of pure RL and drastically reducing teacher constraint violations, we ultimately achieve the highest rigorous Reasoning Success Rate across all evaluated settings.

1 INTRODUCTION

Large Language Models (LLMs) have achieved remarkable success in tasks requiring complex reasoning (Trinh et al., 2024; Chervonyi et al., 2025; Guo et al., 2025), but their sheer size makes them impractical for resource-constrained deployment. Distillation (Hinton et al., 2015; Xiong et al., 2024) has traditionally been used to transfer this expertise to smaller models by minimizing output divergence (e.g., Kullback-Leibler divergence) between the student and the teacher (Gu et al., 2024; Ko et al., 2024). However, focusing solely on divergence forces students to blindly mimic complex reasoning paths that may exceed their representational capacity, while entirely ignoring the ultimate objective of the task. To close this gap, recent advancements reframe distillation as a reinforcement learning (RL) problem (Agarwal et al., 2024; Shao et al., 2024), where the student maximizes a verifiable task-specific reward (e.g., reaching the correct final math answer).

However, optimizing the task reward alone leads to a well-documented failure mode: **reward hacking**. The student discovers shortcuts that produce the correct *final answer* through *flawed intermediate reasoning*. Figure 1 illustrates this vividly: a GRPO-trained student computes a wrong numerical value (4.76%) and then applies a learned "rounding trick" to recover the correct answer (5%). The final answer is right, but the underlying logic is broken: it will not generalize.

The standard solution in the literature is to apply a soft regularization penalty (Agarwal et al., 2024), balancing the RL reward with a step-wise divergence penalty against the teacher using a hyper-

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.

Question:
Leilani saw a 300-foot whale with 4 45-inch remoras attached to it. What percentage of the whale's body length is the combined length of the remoras?

Ours:

To determine what percentage of the whale's body length the combined length of the remoras is, we need to follow these steps:

1. Convert the length of the remoras from inches to feet:

$$45 \text{ inches} \times \frac{1 \text{ foot}}{12 \text{ inches}} = 3.75 \text{ feet}$$

So, each remora is 3.75 feet long.

2. Calculate the total length of the remoras:

$$4 \text{ remoras} \times 3.75 \text{ feet per remora} = 15$$

3. Determine the total length of the whale:

$$300 \text{ feet}$$

4. Calculate the percentage of the whale's body length that the combined length of the remoras represents:

$$\left(\frac{15 \text{ feet}}{300 \text{ feet}}\right) \times 100\%$$

5. Simplify the fraction:

$$\frac{15}{300} = 0.05$$

6. Convert the decimal to a percentage:

$$0.05 \times 100\% = 5\%$$

Therefore, the combined length of the remoras is $\boxed{5}$ percent of the whale's body length.

GRPO (without teacher):

Let's break down the problem step by step to find the percentage of the whale's body length that the combined length of the remoras represents. First, we need to convert all units to the same unit. Let's use feet for this calculation.

The length of each remora is given as 45 inches. Since there are 12 inches in a foot, the length of each remora in feet is:

$$\frac{45}{12} = 3.75 \text{ feet}$$

Since there are 4 remoras, the total length of the remoras is:

$$4 \times 3.75 = 15 \text{ feet}$$

Next, we add the length of the whale to the combined length of the remoras to get the total length of the whale:

$$300 + 15 = 315 \text{ feet}$$

Now, we need to find what percentage the combined length of the remoras is of the whale's body length. This can be calculated using the formula for percentage:

$$\left(\frac{15}{315}\right) \times 100$$

Simplifying the fraction inside the percentage formula:

$$\frac{15}{315} = \frac{1}{21}$$

$$\left(\frac{1}{21}\right) \times 100 = \frac{100}{21} \approx 4.7619$$

Rounding to the nearest whole number, we get: $\boxed{5}$

Figure 1: GRPO (right) reaches a wrong answer (4.7619) but applies a learned rounding trick to recover the correct answer (5).

parameter λ. Unfortunately, this soft relaxation evaluates fidelity on *average*. We argue that this is fundamentally misaligned with the nature of step-by-step reasoning. Because soft penalties compute an expected cost over the entire sequence, a student can easily learn to *compensate* for a catastrophic logical error (which incurs a high penalty) by generating highly probable, boilerplate text elsewhere in the response. In logical reasoning, however, a chain-of-thought is only as strong as its **weakest link**. A single invalid step invalidates the entire path, regardless of how faithful the rest of the generation is to the teacher's stylistic manifold.

Motivated by this observation, we formulate reasoning distillation as a constrained Markov Decision Process (MDP) with a worst-case average constraint on the teacher log-likelihood. We require this constraint to be satisfied *almost surely* during training. This acts as a strict guardrail: if the student generates any prefix that severely violates the teacher's manifold, the exploratory trajectory is immediately deemed infeasible and penalized. By transforming this threshold into a hard boundary rather than a soft preference, we prevent the optimizer from exploring and exploiting compensatory reasoning hacks.

Solving such constrained problems in the standard constrained-RL framework typically relies on dual Lagrangian methods (Achiam et al., 2017; Boyd & Vandenberghe, 2004; Altman, 1999), which are often impractical at LLM scale due to the max–min inner-loop over the teacher. Alternatively, state-augmentation methods such as Saute (Sootla et al., 2022b;a) avoid duality but require tracking a remaining budget variable at every step, which would force the student to query the teacher *at test time*, therefore defeating the purpose of distillation. We address both limitations by leveraging the history-conditioned structure of LLM policies: the remaining budget is a deterministic function of the generated history and therefore need not be exposed as an augmented policy state. We further demonstrate that this unaugmented objective admits a low-variance policy gradient decomposition, splitting the update into a *single-step* term (exploiting the token-level structure of the reward) and a *long-term* credit-assignment term. This ensures that the student can be trained efficiently and deployed completely independently of the teacher while maintaining theoretical guarantees of constraint satisfaction in the penalty limit.

Our contributions are summarized as follows:

- We identify the "weakest link" flaw in soft-regularized distillation, where students mask severe logical errors by maximizing average sequence likelihood.

- We formulate reasoning distillation as a worst-case Constrained RL problem, introducing an unaugmented MDP that enforces trajectory-level hard constraints without requiring dual optimization or test-time teacher access.
- We detail a low-variance policy gradient decomposition that provides stable, intermediate token-level signals to guide the student along authorized reasoning paths.
- We demonstrate empirically that our method breaks the trade-off between final answer correctness and teacher compliance, achieving superior reasoning success rates across multiple LLM families on coding and mathematical benchmarks.

2 BACKGROUND

2.1 DISTILLATION AS POLICY OPTIMIZATION

Knowledge distillation typically transfers capabilities from a vast teacher model μ to a compact student model π by minimizing a divergence metric, often the Kullback-Leibler (KL) divergence, forcing the student to imitate the teacher's output distribution (Hinton et al., 2015; Gu et al., 2024). While this ensures fidelity, it ignores task-specific performance: a student might try to mimic a highly complex reasoning chain it lacks the capacity to complete, failing the fundamental task (Zhang et al., 2025).

To remedy this, recent frameworks merge distillation with Reinforcement Learning (RL), recasting the problem as policy optimization. The student is trained to maximize an expected, verifiable task reward R (e.g., matching the ground-truth final answer in a math problem) while being regularized by a sequence-level divergence $D(\pi, \mu)$ against the teacher:

$$\max_{\pi} \mathbb{E}_{\pi} \left[R - \lambda D(\pi, \mu) \right], \tag{1}$$

where $\lambda > 0$ acts as a soft penalty controlling the trade-off.

The Weakest Link problem: In practice, equation 1 is fundamentally ill-suited for reasoning tasks. The divergence term $D(\pi, \mu)$ is typically evaluated as an expected sum over the trajectory. Because it is a *soft average*, the optimization landscape permits **compensation**. A student model can undertake a catastrophic logical leap, incurring a large localized penalty, but offset this cost by generating long strings of highly probable, "safe" tokens that strictly adhere to the teacher's manifold. In step-by-step reasoning, this averaging effect allows reward hacking to survive: a chain of thought is only as correct as its weakest logical link. Resolving this requires a shift from soft expectations to hard, sequence-level constraints.

2.2 CONSTRAINED REINFORCEMENT LEARNING

Constrained Reinforcement Learning (CRL) offers a mathematical framework for optimizing a primary reward while strictly forbidding certain behaviors (Achiam et al., 2017; Ji et al., 2025). We formally define a Constrained MDP as a tuple $\mathcal{M}_d = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, R, C, \gamma, d \rangle$. In the context of LLM distillation, the state $\mathbf{s}_t$ corresponds to the prompt and the generated prefix, the action $\mathbf{a}_t$ is the next token generated by the student, $\mathcal{P}$ encodes the deterministic transition kernel mapping $(\mathbf{s}_t, \mathbf{a}_t) \rightarrow \mathbf{s}_{t+1}$, and R is the task-specific reward (typically sparse, given at the end of the sequence). We define $C(\mathbf{s}_t)$ as a strictly positive per-step cost (e.g., the negative log-likelihood of the generation under the teacher μ).

Rather than softly regularizing the expected cost, a strict CRL formulation enforces that the accumulated cost remains below a predefined budget d for every valid trajectory:

$$\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t, \mathbf{a}_t) \right] \text{s.t.} \quad \forall T, \quad \sum_{t=0}^{T-1} C(\mathbf{s}_t) \leq d \quad \textbf{almost surely}. \tag{2}$$

Traditional methods for solving Eq.2 rely on Lagrangian duality, effectively dynamically tuning a time-dependent λ parameter in Eq.1. However, at LLM scale, maintaining and updating dual variables through inner-loop optimization is notoriously unstable and computationally prohibitive.

An elegant alternative that avoids dual optimization is the method of state augmentation, known as Saute (Sootla et al., 2022b). Saute resolves the problem by formulating a new augmented MDP

$\widetilde{\mathcal{M}}_d^n = \langle \tilde{\mathcal{S}}, \mathcal{A}, \tilde{\mathcal{P}}, \tilde{R}_n, \gamma, d \rangle$. It introduces an auxiliary tracking variable $\mathbf{z}_t$ that depletes the initial budget d at each step ($\mathbf{z}_0 = d$, $\mathbf{z}_{t+1} = \mathbf{z}_t - C(\mathbf{s}_t)$), extending the state space to $\tilde{\mathcal{S}} = \mathcal{S} \times \mathcal{Z}$. The optimization then seamlessly proceeds over an unconstrained surrogate objective with a reshaped reward:

$$\max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \tilde{R}_n(\mathbf{s}_t, \mathbf{z}_t, \mathbf{a}_t) \right], \tilde{R}_n(\mathbf{s}_T, \mathbf{z}_T, \mathbf{a}_T) = \begin{cases} R(\mathbf{s}_T, \mathbf{a}_T) & \text{if } \mathbf{z}_T \geq 0, \\ -n & \text{if } \mathbf{z}_T < 0. \end{cases} \tag{3}$$

where $n \gg R_{\max}$ provides a harsh, terminal penalty for any trajectory that exhausts the budget. As $n \to \infty$, this penalty guarantees feasibility without saddle-point continuous optimization.

Limitations for Distillation: While Saute is an elegant approach, it is ill-suited for reasoning distillation for three reasons. First, its cumulative-cost constraint Eq.2 is poorly calibrated for sequence generation: because $C(\mathbf{s}_t) > 0$ at each step, longer trajectories accumulate larger budgets by construction, making the constraint length-dependent and biased against longer reasoning chains. Second, maintaining the augmented state $\mathbf{z}_t$ online requires evaluating C at every step, which means querying the teacher *at test time* and therefore fundamentally defeats the purpose of distillation. Third, its fixed penalty $-n$ for constraint violations provides no informative signal about *which* token caused the violation, making credit assignment difficult. In the next section, we address all three issues by (i) reformulating the constraint as a worst-case trajectory-level bound, (ii) showing that the augmented state can be eliminated under history-conditioned policies, and (iii) deriving a low-variance policy gradient that exploits the token-level structure of the reward.

3 METHOD

To address the fundamentally sequential nature of reasoning, we formulate LLM distillation as a Constrained MDP. Unlike previous methods, our framework enforces a strict "weakest-link" boundary on teacher adherence, operates entirely without teacher access at test time, and introduces a theoretically grounded variance-reduction mechanism for stable optimization.

3.1 THE WEAKEST LINK: WORST-CASE CONSTRAINED RL

We begin by defining the per-step cost as the negative log-likelihood of the student's output under the teacher's distribution: $C(\mathbf{s}_t, \mathbf{a}_t) = -\log \mu(\mathbf{a}_t|\mathbf{s}_t)$. We validate this proxy in Appendix E, worst-case teacher likelihood correlates positively with judge-assessed reasoning validity in every run. Standard CRL frameworks, such as Saute (Sootla et al., 2022b), constrain the cumulative sum of costs $\sum_t C_t \leq d$. However, because C is strictly positive, a cumulative budget inherently penalizes longer trajectories, artificially discouraging the student from generating the extended chains of thought necessary for complex problem-solving. Furthermore, as discussed in Section 2, soft-regularization methods constrain the *expected average* sequence cost, which permits reward-hacking via "compensation" (e.g., masking a severe logical error with highly probable surrounding text).

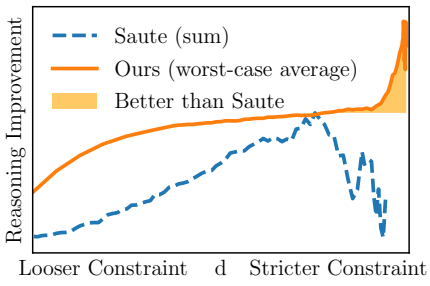


Figure 2: Reasoning Improvement across budget bounds. Constraining the worst-case average prevents local hallucination effectively, yielding strictly higher peak reasoning performance across optimal budget thresholds compared to a cumulative sum budget.

To prevent future compensation after a prefix has already crossed the boundary while remaining length-invariant, we propose optimizing the task reward subject to a **worst-case average constraint** over every prefix of the trajectory. Specifically, we require that at no point during the generation does the average cost exceed the threshold d:

$$\max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{s}_t, \mathbf{a}_t) \right] \quad \text{s.t.} \quad \max_{T \geq 1} \frac{1}{T} \sum_{t=0}^{T-1} C(\mathbf{s}_t, \mathbf{a}_t) \leq d \quad \textbf{almost surely}. \tag{4}$$

Empirically, restricting the worst-case trajectory effectively preserves reasoning integrity. As shown in Figure 2, maximizing reward under our worst-case average yields strictly higher reasoning im-

provement than optimal bounded cumulative sums. We formalize this advantage, and explicitly distinguish our approach from standard Lagrangian relaxations, in the following proposition:

Proposition 3.1 (Anti-Compensation and Non-Equivalence)**.** *Let τ be a trajectory of length T with non-negative costs C.*

(a) ***Anti-Compensation:*** *If τ satisfies an expected average threshold $\frac{1}{T}\sum_{t=0}^{T-1} C_t \leq d$ (as induced by soft penalties), the maximum single-step error $C_{\max} = \max_t C_t$ can grow as $\mathcal{O}(T \cdot d)$, allowing unbounded local hallucinations. Conversely, under the worst-case average constraint $\max_{\tau \leq T} \frac{1}{\tau}\sum_{t=0}^{\tau-1} C_t \leq d$, the severity of a hallucination precisely at step k is bounded by $(k+1) \cdot d$.*

(b) ***Non-Equivalence:*** *There exists no constant scalar penalty $\lambda \geq 0$ such that the optimal policy of the unconditionally soft problem $\max_\pi \mathbb{E}_\pi[\sum \gamma^t (R - \lambda C)]$ matches the optimal policy of the worst-case constrained MDP in Eq. 4 for all (R, C).*

See Appendix A.1 for full proof.

3.2 Eliminating State Augmentation via Absorbing Infeasibility

Although state-augmented methods like Saute can theoretically enforce trajectory constraints without inner-loop Lagrangian optimization, they rely on appending a remaining budget variable $\mathbf{z}_T$ to the MDP state. For distillation, tracking $\mathbf{z}_T$ online requires computing the teacher's likelihood $C(\mathbf{s}_t, \mathbf{a}_t)$ at every autoregressive decoding step. This mandates test-time access to the massive teacher model, completely neutralizing the computational benefits of distillation.

We solve this by directly modifying the reward function *without* state augmentation. We construct a history-based MDP $\widehat{\mathcal{M}}_d^n = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \hat{R}_n, \gamma, d \rangle$, where the constrained reward fuses the task reward with a terminal feasibility penalty n:

$$\hat{R}_n(\mathbf{s}_T, \mathbf{a}_T) = \begin{cases} R(\mathbf{s}_T, \mathbf{a}_T) & \text{if } \max_{\tau \leq T} \frac{1}{\tau}\sum_{t=0}^{\tau-1} C(\mathbf{s}_t, \mathbf{a}_t) \leq d, \\ -n & \text{otherwise.} \end{cases} \tag{5}$$

Why does this avoid the need for budget tracking? Because our worst-case constraint is inherently prefix-wise. We formalize this critical structural property:

Lemma 3.2 (Absorbing Infeasibility)**.** *Let the infeasible state space be defined as $\mathcal{S}_{inf} = \{\mathbf{s}_T \mid \max_{\tau \leq T} \frac{1}{\tau}\sum_{t=0}^{\tau-1} C > d\}$. For any generation step $\mathbf{a}_T \in \mathcal{A}$, the deterministic transition to the new prefix $\mathbf{s}_{T+1} = \mathbf{s}_T \oplus \mathbf{a}_T$ ensures that if $\mathbf{s}_T \in \mathcal{S}_{inf}$, then sequentially $\mathbf{s}_{T+1} \in \mathcal{S}_{inf}$.*

By Lemma 3.2, a single constraint violation irreversibly "kills" the trajectory. Because an LLM explicitly autoregressively conditions on the entire prefix $\mathbf{s}_T$, the control process is fully observable. The viability of the state is perfectly encoded within the history, eliminating the need to explicitly pass or compute an augmented state $\mathbf{z}_T$. Consequently, at test time, the student model operates entirely independently of the teacher. The teacher is merely queried *a posteriori* during RL training rollouts to compute $\hat{R}_n$. Under standard assumptions, this unaugmented objective preserves the core theoretical guarantees of constrained RL (Theorem 3.3; proofs in Appendix A.2):

Theorem 3.3. *As the penalty $n \to \infty$, the optimal value function for $\widehat{\mathcal{M}}_d^n$ converges monotonically. Furthermore, if the limiting MDP admits an optimal policy with finite value, that policy satisfies the worst-case averaged constraint almost surely.*

3.3 Policy Gradient Optimization and Variance Reduction

To optimize Eq. 5, we maximize the expected discounted return $J_n(\theta) = \mathbb{E}_{\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t)]$. Under the standard REINFORCE formulation (Sutton et al., 1999), the gradient depends on the aggregate future return $R_t^n \triangleq \sum_{k=t}^{\infty} \gamma^{k-t} \hat{R}_n(\mathbf{s}_k, \mathbf{a}_k)$. However, this standard estimator suffers from high variance, especially in our constrained case, where a single token's choice to violate the boundary violently swings the entire future trajectory return to $-n$.

To stabilize optimization, we exploit the fact that evaluating our immediate penalized reward $\hat{R}_n(\mathbf{s}_t, \mathbf{a})$ does not require environment interaction, but just a forward pass of the frozen teacher

over the current vocabulary. Using the recursion $R_t^n = \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) + \gamma R_{t+1}^n$, we elegantly decompose the gradient into a *single-step* term evaluated precisely over the action space, and a *long-term* Monte Carlo term (see Appendix A.3 for the full derivation):

$$\nabla_\theta J_n(\theta) = \underbrace{\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \nabla_\theta \Big(\sum_{\mathbf{a}} \pi_\theta(\mathbf{a} \mid \mathbf{s}_t)\, \hat{R}_n(\mathbf{s}_t, \mathbf{a}) \Big)\right]}_{(\nabla_\theta J_n)_{\text{SINGLE}}} + \underbrace{\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^{t+1} R_{t+1}^n \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)\right]}_{(\nabla_\theta J_n)_{\text{LONG}}}.$$

This decomposition provides intermediate token-level signals to guide the student carefully along authorized reasoning prefixes. Crucially, while this algebraic decomposition is universally valid, its practical variance-reduction benefits are uniquely impactful and tractable by our worst-case constraint formulation. Under standard trajectory-level constraints (such as global sums or expected averages), a sequence's feasibility is typically only determined at the terminal step. Consequently, the immediate reward at step t contains no constraint information, and the massive variance of the constraint penalty remains entirely trapped in the stochastic long-term return R_{t+1}^n.

In contrast, as established by Lemma 3.2, our prefix-wise constraint definitively triggers the $-n$ penalty the exact moment the boundary is crossed. By shifting the feasibility penalty into the immediate reward $\hat{R}_n(\mathbf{s}_t, \mathbf{a})$, we can analytically integrate it out over the vocabulary using a single forward pass of the teacher. Thus, this decomposition provides a mathematically guaranteed variance reduction for the immediate reward evaluation at every step (formalized in Appendix A.4 in Theorem A.1).

Theorem A.1 demonstrates that resolving the immediate step using the teacher distribution analytically removes the sampling variance associated with that specific token step. The long-term term $(\nabla_\theta J_n)_{\text{LONG}}$ is preserved to allow proper credit assignment, penalizing early tokens that steer the prefix toward an unavoidable constraint violation later in the trajectory.

4 Experiments

We design our experiments to answer three core research questions:

- **RQ1 (The Trade-off):** Does enforcing a worst-case constraint prevent reward-hacking without sacrificing task performance?
- **RQ2 (Reasoning Quality):** Does eliminating compensatory reward-hacking lead to tangibly superior logical reasoning?
- **RQ3 (Anti-Compensation):** Can we empirically validate that soft baselines suffer from severe localized violations, and that our worst-case metric predicts reasoning failure?

4.1 Experimental Setup

Tasks & Models. We evaluate on several mathematical reasoning datasets: MATH (Hendrycks et al., 2021), Apple/GSM-Symbolic (main) (Mirzadeh et al., 2025) and GSM8K (Cobbe et al., 2021). We consider three distinct distillation settings to ensure generalization across model families and task difficulties: (1) *Qwen2.5-1.5B* distilled from *Qwen2.5-Math-7B-Instruct* on MATH; (2) *Llama-3.2-3B* distilled from *Llama-3.2-11B-Vision-Instruct* on MATH; and (3) *Qwen2.5-1.5B* distilled from *Qwen2.5-Math-7B-Instruct* on GSM8K and evaluated on GSM-Symbolic. We also performed experiments on code generation with different coding models that we report in Appendix C.

Baselines. We compare our unaugmented constrained RL approach against: (1) **GKD** (Agarwal et al., 2024) and **Mini-LLM** (Gu et al., 2024) for pure divergence minimization; (2) **GRPO** (Shao et al., 2024) for pure task-reward maximization; and (3) **GKD-GRPO**, the standard soft Lagrangian relaxation (Agarwal et al., 2024) tuned across a wide spectrum of constraint penalties $\lambda \in \{0.001, 0.01, 0.1, 1.0, 10\}$. For our method, we evaluate several worst-case average budget thresholds $d \in \{-\ln(1/4), -\ln(1/2), -\ln(3/4)\}$. For an exhaustive description of all baselines, prompt templates, training hyperparameters, and the exact evaluation protocols, please refer to Appendix B.

Metrics. We report: 1) **Final Answer Correctness (FAC):** Verifies if the extracted `\boxed{}` answer precisely matches the ground truth. 2) **Constraint Satisfaction (CS):** The percentage

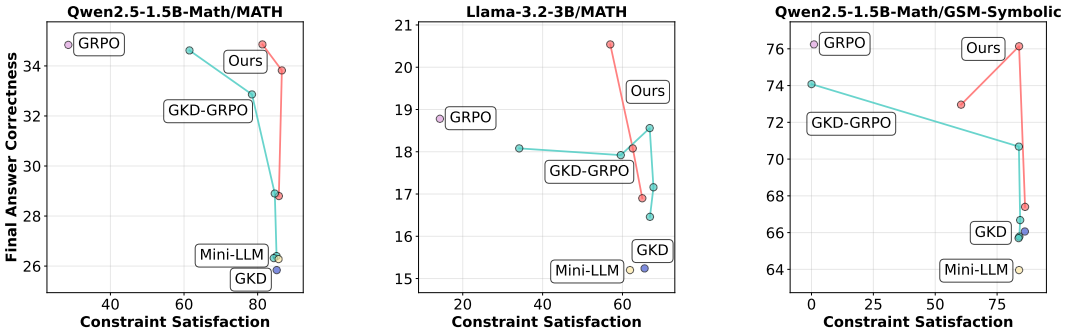


Figure 3: **Accuracy-Fidelity Pareto Front on the 3 distillation settings.** X-axis: Constraint Satisfaction (higher is better). Y-axis: Final Answer Correctness (higher is better). Soft-regularized baselines (GKD-GRPO line) force a harsh trade-off between fidelity and accuracy. Our worst-case constrained method matches or outperforms pure RL accuracy while retaining high fidelity.

Table 1: Main distillation results across three settings. All metrics are reported in percentages (%) and higher is better. Our method ($d = -\ln(1/4)$) achieves the highest Reasoning Success Rate (RSR) across all settings by breaking the trade-off between Final Answer Correctness (FAC) and Constraint Satisfaction (CS). Best results are in **bold**, second best are underlined.

	Qwen2.5 (MATH)			Llama-3.2 (MATH)			Qwen2.5 (GSM-Sym.)		
Method	**RSR**	**FAC**	**CS**	**RSR**	**FAC**	**CS**	**RSR**	**FAC**	**CS**
SFT on Teacher CoT	24.66	27.02	78.54	13.64	17.48	57.72	55.78	65.9	83.46
GKD	24.00	25.84	85.18	9.34	15.24	65.54	57.58	66.06	86.22
MiniLLM	24.22	26.28	85.68	11.08	15.20	61.88	55.28	63.96	83.88
GKD-GRPO $\lambda = 10.0$	24.58	26.32	84.38	7.44	16.46	66.88	56.56	65.70	83.70
GKD-GRPO $\lambda = 1.0$	24.32	26.40	85.14	9.60	17.16	**67.76**	57.18	65.78	83.98
GKD-GRPO $\lambda = 0.1$	26.56	28.90	84.66	11.98	18.56	66.82	58.66	66.68	84.32
GKD-GRPO $\lambda = 0.01$	29.42	32.86	78.46	12.48	17.92	59.58	61.04	70.68	83.80
GKD-GRPO $\lambda = 0.001$	29.48	34.62	61.48	11.44	18.08	34.14	64.02	74.08	0.00
GRPO	27.54	34.84	28.56	12.90	18.78	14.28	64.36	**76.24**	1.00
Ours $d = -\ln(1/4)$	**30.00**	**34.86**	81.30	**14.12**	**20.54**	56.94	**65.82**	76.14	83.88
Ours $d = -\ln(1/2)$	29.76	33.82	**86.58**	13.34	18.08	62.58	63.06	72.96	60.44
Ours $d = -\ln(3/4)$	26.36	28.80	85.76	13.92	16.9	64.96	58.96	67.40	**86.30**
Student		0.54			0.08			0	
Teacher		34.46			32.08			82.12	

of test trajectories where the worst-case prefix average remains below the nominal threshold $d = -\ln(1/2)$. This measures fidelity. 3) **Reasoning Success Rate (RSR):** Beyond the final answer, we assess the absolute logical validity of the intermediate steps using an LLM-as-a-judge (Zheng et al., 2023). We also verified the judge labels on a small subset (see Appendix F).

4.2 RQ1: Bridging the Accuracy-Fidelity Trade-off

A core premise of our work is that soft divergence constraints (like GKD-GRPO) invoke a fundamental compromise: to get high accuracy, one must abandon fidelity, leading to reward hacking. Figure 3 and Table 1 empirically support this.

In Table 1, looking at pure GRPO on Qwen+GSM-Symbolic, the model achieves a high FAC of 76.24%, but suffers an almost zero Constraint Satisfaction rate (CS = 1%). It has learned to game the final reward. The soft GKD-GRPO baseline attempts to fix this by interpolating with λ. However, reducing λ to 0.001 to recover accuracy (FAC 74.08%) violently decreases CS to 0%. Conversely, raising λ to 0.1 fixes CS (84.32%) but cripples task performance (FAC 66.68%).

Our method ($d = -\ln(1/4)$) achieves a remarkable trade-off on the Pareto front on the 3 settings. By enforcing the constraint as a hard worst-case boundary during training, we guide the optimizer to valid regions even on the testing sets. As a result, our method matches the peak FAC of GRPO (76.14% vs 76.24%) while simultaneously holding CS high at 83.88%. This remarkable dominance holds across settings: on Llama+MATH, we reach a peak FAC of 20.54% (outperforming GRPO's 18.78%) while quadrupling CS (56.94% vs 14.28%).

It is worth noting the sensitivity of the d boundary. Table 1 shows that an overly strict constraint ($d = -\ln(3/4)$) actively harms both FAC and RSR. Because the student has limited representational capacity compared to the teacher, forcing it to mimic the teacher perfectly prevents it from discovering simpler logic paths it can actually execute, establishing $d = -\ln(1/4)$ as the empirical sweet spot. Furthermore, we observe an expected train-test generalization gap: during training, our method satisfies the constraint almost strictly (CS $\approx$ 100%), but domain shifts at test time reduce this to $\sim$80%. Nonetheless, this inductive bias clearly prevents the catastrophic 0% CS collapse seen in soft baselines optimized for accuracy.

4.3 RQ2: Superior Logical Reasoning Quality

Does suppressing constraint violations actually yield better reasoning? To validate this rigorously, we conduct a strict Pairwise Judge evaluation. The judge prompt explicitly instructs the evaluating LLM to search for *logical gaps and false leaps, even if the final answers are identical*. Figure 4 displays the overall win rate for each method across the datasets.

The results starkly confirm the "weakest link" compensation failure. Pure GRPO collapses under scrutiny: despite achieving high final answer correctness, its reasoning win rate plummets to 25.5% on Qwen-MATH and 30.9% on Qwen-GSM because the judge heavily penalizes its learned reward-hacks. Meanwhile, standard soft-regularization (GKD-GRPO $\lambda = 0.01$) proves incredibly brittle: while it spikes to 67.7% on Qwen-MATH, it completely breaks down on Llama-MATH (40.5% win rate), showing that soft penalties are fundamentally sensitive to architecture and dataset dynamics.

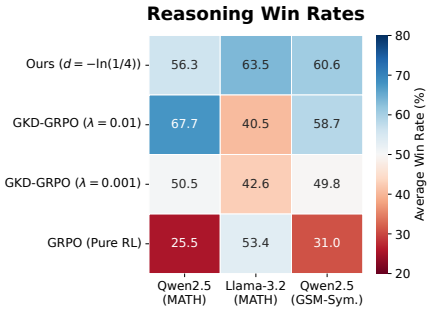


Figure 4: Average win rate (%) evaluated by an LLM-as-a-Judge across the three distillation settings.

In contrast, our worst-case constraint acts as a robust, invariant guardrail. Our method is the only approach to maintain a dominant, consistent win rate across all three discrete settings (56.2%, 63.5%, and 60.6% respectively). The qualitative example in Figure 1 perfectly encapsulates this dynamic: GRPO learns an artificial rounding trick to secure the reward, which the judge flags, whereas our constraint forces the student to derive the mathematically sound, generalized path.

4.4 RQ3: Empirical Validation of Anti-Compensation

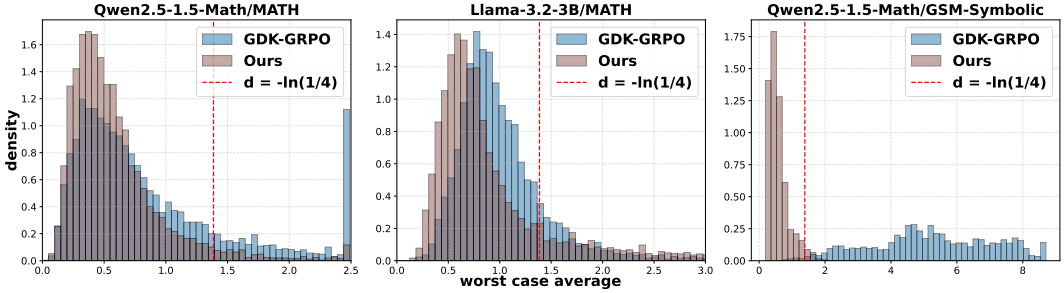


Figure 5: Distribution of the worst-case prefix average cost at test time. Soft baselines display extended "tails" of severe localized violations, empirically validating the compensation problem.

We empirically validate this by analyzing the trained models at test time. For each generated test trajectory, we compute its specific worst-case prefix average cost $\max_T \frac{1}{T} \sum_{t=0}^{T-1} C_t$. Figure 5 plots the distribution of this metric (for legibility, we isolate our best reasoning variant $d = -\ln(1/4)$ against the highest-accuracy baseline GKD-GRPO $\lambda = 0.001$). As predicted, the soft baseline exhibits a right-skewed "tail" (especially with the Qwen model). While their *mean* sequence cost might be low, they can generate trajectories containing localized violations. In contrast, our method imposes a sharper cutoff at the boundary d, proving our method actively suppresses these compensations.

5 Related Work

RL-Based Distillation and Reward Hacking. The traditional paradigm of LLM distillation aligns the student's output distribution with the teacher's via a divergence metric, typically reverse KL (Hinton et al., 2015; Sanh et al., 2020; Gu et al., 2024; Zimmer et al., 2025). However, strictly mimicking the teacher ignores the downstream objective and can force the student into modeling complex distributions beyond its capacity (Zhang et al., 2025). Consequently, modern distillation increasingly leverages RL to incorporate verifiable task rewards (e.g., reaching the correct final answer) (Ko et al., 2024; 2025). While pure RL dramatically improves final task performance, it reliably induces *reward hacking*: the student exploits logical shortcuts or hallucinatory leaps to reach the correct final answer without sound intermediate reasoning (Gudibande et al., 2024). To combat this, standard practices composite the RL reward with a soft KL regularizer against the teacher (Agarwal et al., 2024; Ouyang et al., 2022; Stiennon et al., 2022; Yang et al., 2024).

Enhancing Reasoning Distillation. Because soft regularizers often fail to ensure rigorous reasoning, much of the recent literature relies on injecting dense, intermediate supervision. *Process-aware* distillation supervises the student directly on the teacher's intermediate logical steps, transferring causal structure rather than just final outcomes (Hsieh et al., 2023; Adarsh et al., 2024; Chen et al., 2025). Other approaches include *logit-aware* distillation, which utilizes attention or Bayesian heuristics to dynamically weight the distillation loss on "pivotal" tokens (Li et al., 2025; 2024; Saadi & Wang, 2025), and *knowledge-augmented* strategies that retrieve external information to patch knowledge gaps (Kang et al., 2023; Tian et al., 2025). While effective, these methods introduce significant friction, requiring fine-grained step-level annotations, external process-reward models, or complex retrieval overhead.

Constrained RL and State Augmentation. Preventing catastrophic deviations is the primary focus of Constrained RL (Achiam et al., 2017; Schulman et al., 2015). Framing LLM alignment as a strictly constrained problem offers strong theoretical guarantees but is computationally hostile. Standard continuous Constrained MDPs are solved via dual Lagrangian descents, which, at the scale of modern LLMs, require managing unstable dual variables across deep exploratory rollouts, incurring prohibitive compute, memory, and variance costs (Dasgupta et al., 2023). Recent advances in safe RL avoid Lagrangian duality via *state augmentation*, introducing frameworks like Saute (Sootla et al., 2022b;a; Calvo-Fullana et al., 2024), which append a remaining safety budget to the MDP state and enforce constraints via a reshaped hard-penalty reward.

6 Conclusion

In this work, we identified a fundamental flaw in RL-based LLM distillation: soft divergence penalties allow student models to mask severe logical errors through compensation. Recognizing that a reasoning chain is only as valid as its weakest link, we introduced a worst-case Constrained RL framework that enforces a strict boundary on the teacher log-likelihood along every prefix of the trajectory. To make this tractable at scale, we formulated an unaugmented MDP that strictly penalizes constraint violations without requiring test-time teacher access or unstable dual Lagrangian optimization. Extensive evaluations across multiple model families, coding and mathematical benchmarks demonstrate that our approach successfully breaks the accuracy-fidelity Pareto front. By strictly pruning invalid reasoning paths during exploration, our method matches the high final answer correctness of pure RL while drastically reducing constraint violations, ultimately yielding superior, trustworthy reasoning capabilities in compact models.

AI USE STATEMENT

In this work, we used generative AI tools for writing assistance. We have not used generative AI tools to run any of the experiments. The algorithm code was written by the authors. We reviewed all AI-assisted work notably all the sentence it suggested to edit. The LLM-as-a-judge evaluation protocol is a standard evaluation technique described in detail in Appendix B. No LLMs were used as a novel or non-standard component of the core method development itself. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work aims to improve the reliability of distilled reasoning LLMs. By enforcing that student models' logical steps remain closely aligned with a trusted teacher, our method mitigates a known failure mode of RL-trained models, reward hacking via shallow shortcuts, which could otherwise produce confidently-presented but logically invalid solutions in high-stakes deployment settings (e.g., educational tools, scientific reasoning assistants). We do not foresee negative societal impacts beyond those inherent to LLM distillation more broadly.

REPRODUCIBILITY STATEMENT

The code will be fully open-sourced upon acceptance. All datasets used (MATH, GSM8K, GSM-Symbolic) are publicly available, and all models (Qwen2.5, Llama-3.2) are publicly accessible. Sufficient implementation details are provided for reproducibility. Section 4 specifies all models, datasets, and evaluation metrics. Appendix B provides full training details including hyperparameters, optimizer choice, batch size, training epochs, precision, and baseline configurations. The exact LLM-as-a-judge prompts are reproduced in full.

All theoretical results (Proposition 3.1, Lemma 3.2, Theorem 3.3, Theorem A.1) are formally stated with their assumptions in the main text and the appendix, with proof sketches provided inline for the proposition. Complete formal proofs are given in Appendices A.1, A.2, A.3, and A.4. Theorem 3.3 explicitly states its assumptions (bounded rewards, discrete setting, positive costs) and references standard conditions (B1–B3 from Hernández-Lerma & Muñoz de Ozak (1992)).

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 22–31. JMLR.org, 2017.

Shivam Adarsh, Kumar Shridhar, Caglar Gulcehre, Nicholas Monath, and Mrinmaya Sachan. Siked: Self-guided iterative knowledge distillation for mathematical reasoning, 2024. URL `https://arxiv.org/abs/2410.18574`.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

Eitan Altman. *Constrained Markov Decision Processes*. Stochastic Modeling Series. CRC Press, 1999.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004.

Miguel Calvo-Fullana, Santiago Paternain, Luiz F. O. Chamon, and Alejandro Ribeiro. State augmented constrained reinforcement learning: Overcoming the limitations of learning with rewards. *IEEE Transactions on Automatic Control*, 69(7):4275–4290, 2024. doi: 10.1109/TAC.2023.3319070.

Jack Chen, Fazhong Liu, Naruto Liu, Yuhan Luo, Erqu Qin, Harry Zheng, Tian Dong, Haojin Zhu, Yan Meng, and Xiao Wang. Step-wise adaptive integration of supervised fine-tuning and reinforcement learning for task-specific llms. *arXiv preprint arXiv:2505.13026*, 2025.

Yuri Chervonyi, Trieu H. Trinh, Miroslav Olšák, Xiaomeng Yang, Hoang Nguyen, Marcelo Menegali, Junehyuk Jung, Vikas Verma, Quoc V. Le, and Thang Luong. Gold-medalist performance in solving olympiad geometry with alphageometry2, 2025. URL https://arxiv.org/abs/2502.03544.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Sayantan Dasgupta, Trevor Cohn, and Timothy Baldwin. Cost-effective distillation of large language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 7346–7354, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.463. URL https://aclanthology.org/2023.findings-acl.463/.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models, 2024. URL https://arxiv.org/abs/2306.08543.

Arnav Gudibande, Eric Wallace, Charlie Victor Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. The false promise of imitating proprietary language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Kz3yckpCN5.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

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. *NeurIPS*, 2021.

Onésimo Hernández-Lerma and Myriam Muñoz de Ozak. Discrete-time markov control processes with discounted unbounded costs: optimality criteria. *Kybernetika*, 28(3):191–212, 1992.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes, 2023. URL https://arxiv.org/abs/2305.02301.

Xiaotong Ji, Shyam Sundhar Ramesh, Matthieu Zimmer, Ilija Bogunovic, Jun Wang, and Haitham Bou Ammar. Almost surely safe alignment of large language models at inference-time. *arXiv preprint arXiv:2502.01208*, 2025.

Minki Kang, Seanie Lee, Jinheon Baek, Kenji Kawaguchi, and Sung Ju Hwang. Knowledge-augmented reasoning distillation for small language models in knowledge-intensive tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models, 2024. URL https://arxiv.org/abs/2402.03898.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. Distillm-2: A contrastive approach boosts the distillation of llms, 2025. URL https://arxiv.org/abs/2503.07067.

Chenglin Li, Qianglong Chen, Liangyue Li, Caiyu Wang, Feng Tao, Yicheng Li, Zulong Chen, and Yin Zhang. Mixed distillation helps smaller language models reason better. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (Findings)*, pp. 1–12, 2024. URL https://aclanthology.org/2024.findings-emnlp.91.

Wei Li, Lujun Li, Mark Lee, Shengjie Sun, Lei Zhang, Wei Xue, and Yike Guo. Bayeskd: Bayesian knowledge distillation for compact llms in constrained fine-tuning scenarios. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 138–152, 2025.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AjXkRZIvjB.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020. URL http://jmlr.org/papers/v21/19-346.html.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Khouloud Saadi and Di Wang. TASKD-LLM: Task-aware selective knowledge distillation for LLMs. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2025. URL https://openreview.net/forum?id=QQBfoVJWY2.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020. URL https://arxiv.org/abs/1910.01108.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schulman15.html.

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, 2024. URL https://arxiv.org/abs/2402.03300.

Aivar Sootla, Alexander Cowen-Rivers, Jun Wang, and Haitham Bou Ammar. Enhancing safe exploration using safety state augmentation. *Advances in Neural Information Processing Systems*, 35:34464–34477, 2022a.

Aivar Sootla, Alexander I Cowen-Rivers, Taher Jafferjee, Ziyan Wang, David H Mguni, Jun Wang, and Haitham Ammar. Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *International Conference on Machine Learning*, pp. 20423–20443. PMLR, 2022b.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, 2022. URL https://arxiv.org/abs/2009.01325.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Yunhao Tang and Rémi Munos. On a few pitfalls in kl divergence gradient estimation for rl. *arXiv preprint arXiv:2506.09477*, 2025.

Yijun Tian, Yikun Han, Xiusi Chen, Wei Wang, and Nitesh V. Chawla. Beyond answers: Transferring reasoning capabilities to smaller llms using multi-teacher knowledge distillation. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, WSDM '25, pp. 251–260, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400713293. doi: 10.1145/3701551.3703577. URL https://doi.org/10.1145/3701551.3703577.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

Zheng Xiong, Risto Vuorio, Jacob Beck, Matthieu Zimmer, Kun Shao, and Shimon Whiteson. Distilling morphology-conditioned hypernetworks for efficient universal morphology control, 2024. URL https://arxiv.org/abs/2402.06570.

Joy Qiping Yang, Salman Salamatian, Ziteng Sun, Ananda Theertha Suresh, and Ahmad Beirami. Asymptotics of language model alignment, 2024. URL https://arxiv.org/abs/2404.01730.

Chen Zhang, Qiuchi Li, Dawei Song, Zheyu Ye, Yan Gao, and Yao Hu. Towards the law of capacity gap in distilling language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22504–22528, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1097. URL https://aclanthology.org/2025.acl-long.1097/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

Matthieu Zimmer, Milan Gritta, Gerasimos Lampouras, Haitham Bou Ammar, and Jun Wang. Mixture of attentions for speculative decoding. In *The Thirteenth International Conference on Learning Representations*, 2025.

A Derivations and Proofs

A.1 Proof of Proposition 3.1

Part (a): Anti-Compensation
Consider a generation trajectory $\tau = (\mathbf{s}_0, \mathbf{a}_0, \ldots, \mathbf{s}_{T-1}, \mathbf{a}_{T-1})$ of length T, evaluated under a non-negative per-step cost $C_t \triangleq C(\mathbf{s}_t, \mathbf{a}_t) \geq 0$. Suppose the sequence satisfies a traditional average threshold (such as those induced by soft penalties evaluating the whole-sequence divergence), namely $\frac{1}{T}\sum_{t=0}^{T-1} C_t \leq d$.

To isolate the magnitude of a single-step hallucination, consider an adversarial scenario where the student perfectly mimics the teacher at all steps except for an arbitrary step k. Specifically, let $C_t = 0$ for all $t \neq k$. To satisfy the soft average constraint, it is strictly required that:

$$\frac{1}{T}\left(0 + \cdots + C_k + \cdots + 0\right) \leq d \implies C_k \leq T \cdot d.$$

As T grows (longer reasoning chains), the tolerated single-step deviation C_k scales linearly with T. Thus, the maximum possible maximum single-step error $C_{\max} = \max_t C_t$ is bounded by $\mathcal{O}(T \cdot d)$, representing an unbounded localized hallucination completely masked by surrounding safe text.

Conversely, our worst-case average constraint requires that for *all* prefix lengths $\tau \in \{1, \ldots, T\}$, $\frac{1}{\tau}\sum_{t=0}^{\tau-1} C_t \leq d$. At step k (representing a prefix length of $k+1$ in 0-based indexing), evaluating the prefix sum up to k yields:

$$\frac{1}{k+1}\sum_{t=0}^{k} C_t \leq d.$$

Since all $C_t \geq 0$, removing the terms prior to k establishes a strict upper bound:

$$\frac{1}{k+1}C_k \leq \frac{1}{k+1}\sum_{t=0}^{k} C_t \leq d \implies C_k \leq (k+1) \cdot d.$$

Thus, the magnitude of a reasoning hallucination at step k is tightly bounded by the prefix length $(k+1) \cdot d$, decisively preventing global averaging and the "hiding" of arbitrary magnitude logic breaks within long sequences.

Part (b): Non-Equivalence to Soft Lagrangian Regularization
We are to prove there exists no constant scalar $\lambda \geq 0$ such that the unconstrained soft objective $J_{\text{soft}}(\pi) = \mathbb{E}_\pi[\sum_t \gamma^t (R_t - \lambda C_t)]$ shares an optimal policy with our constrained objective for all combinations of rewards and costs. We proceed by constructing a counterexample.

Consider a deterministic environment providing a final task reward at terminal step $T = 2$. Let the budget $d = 1$, with $\gamma = 1$. Let the optimization act over two possible sequences, τ_1 and τ_2.

- **Sequence 1 (Faithful):** Task reward $R(\tau_1) = 1$. The costs are evenly distributed: $C_1 = 1, C_2 = 1$.
- **Sequence 2 (Reward Hack):** Task reward $R(\tau_2) = 1 + \epsilon$ (where $\epsilon > 0$). The costs are heavily skewed: $C_1 = 2, C_2 = 0$.

Evaluating under our worst-case constrained MDP ($n \to \infty$): For τ_1, prefixes average to $\frac{1}{1} \leq 1$ and $\frac{1+1}{2} \leq 1$. The trajectory is feasible, so its value is $V_{\text{con}}(\tau_1) = 1$. For τ_2, the first prefix averages to $\frac{3}{1} > 1$. The trajectory violates the constraint and earns the penalty, so $V_{\text{con}}(\tau_2) = -n < 0$. The optimal constrained policy deterministically selects the faithful trajectory τ_1 in our framework.

Now, evaluating under the generic soft Lagrangian objective: The expected sum of costs for both trajectories is identical ($\sum C_t = 2$). Thus:

$$V_{\text{soft}}(\tau_1) = 1 - \lambda(2)$$
$$V_{\text{soft}}(\tau_2) = 1 + \epsilon - \lambda(2)$$

For *any* choice of $\lambda \geq 0$, we have $V_{\text{soft}}(\tau_2) > V_{\text{soft}}(\tau_1)$ by exactly ϵ. The soft regularized objective will *always* prefer the reward-hacked, locally hallucinatory trajectory τ_2. Therefore, no scalar λ can replicate the exact constraint boundary of our formulation, Q.E.D.

A.2 Proof of Theorem 3.3

For this proof, we assume that the reward function is bounded, that we are in a discrete LLM distillation setting and the cost function is always positive. We first prove the Bellman equation holds for $\widehat{\mathcal{M}}_d^n$ by verifying conditions B1–B3 (Hernández-Lerma & Muñoz de Ozak, 1992).

- **B1 (Bounded and Measurable):** On feasible histories, $\hat{R}_n(\mathbf{s}, \mathbf{a}) = R(\mathbf{s}, \mathbf{a})$, which is uniformly bounded by assumption $R \in [0, R_{\max}]$. On infeasible histories (which by Lemma 3.2 is an absorbing state), $\hat{R}_n(\mathbf{s}, \mathbf{a}) = -n$. The reward is piecewise bounded by $[-n, R_{\max}]$. In the discrete token mapping topology, all mappings are measurable.
- **B2 (Weak Continuity):** Since the joint space $\mathcal{S} \times \mathcal{A}$ of auto-regressive prefix sequences and discrete tokens is countable, any metric mapping naturally satisfies sequence convergence (all convergent sequences are eventually constant), thereby securing topological weak continuity.
- **B3 (Compactness):** The token vocabulary $\mathcal{A}$ is finite, meaning the action space is inherently bounded, closed, and thus compact.

Under these standard conditions and given $\gamma \in (0, 1]$, standard dynamic programming theory ensures the existence of a stationary Markov policy optimizing the Bellman equation for $\widehat{\mathcal{M}}_d^n$. The state $\mathbf{s}_t$ encodes the entire token history precisely; therefore, evaluating the feasibility bound $\max_{\tau \le T} \frac{1}{\tau} \sum C_t \le d$ preserves the Markov property.

Monotone Value Convergence & Almost Sure Safety:
By construction in Eq. 5, the penalty branch maps to $-n$. Thus, for any two penalty parameters $m > n$, we analytically hold $\hat{R}_m(\mathbf{s}, \mathbf{a}) \le \hat{R}_n(\mathbf{s}, \mathbf{a})$ everywhere. Evaluating the discounted return over time, this injects a strict inequality $V_m^*(\mathbf{s}) \le V_n^*(\mathbf{s})$. As $\{\hat{V}_n^*(\mathbf{s})\}_{n \ge 1}$ is monotone non-increasing and trivially lower-bounded below on valid subsets, it must converge to a point-wise limit $\hat{V}_\infty^*$.

In the limiting target $\widehat{\mathcal{M}}_d^\infty$ (where $n \to \infty$), any trajectory passing through the absorbing infeasible boundary $\mathcal{S}_{\text{inf}}$ obtains a return of $-\infty$. If it is given that the optimum policy π^* for $\widehat{\mathcal{M}}_d^\infty$ achieves a strictly finite value, its probability mass entering the penalty region must be uniformly zero. Put simply:

$$\mathbb{P}_{\pi^*}\Big(\exists\, T \ge 1 \text{ s.t. } \frac{1}{T} \sum_{t=0}^{T-1} C_t > d\Big) = 0.$$

The complementary probability dictates that the sequence remains feasible:

$$\mathbb{P}_{\pi^*}\Big(\max_{T \ge 1} \frac{1}{T} \sum_{t=0}^{T-1} C(\mathbf{s}_t, \mathbf{a}_t) \le d\Big) = 1,$$

meaning it satisfies the constraint *almost surely*.

Remark on the finite-value assumption. Theorem 3.3 assumes the limiting MDP $\widehat{\mathcal{M}}_d^\infty$ admits an optimal policy with finite value; we argue this is a mild and verifiable condition rather than a technicality. A sufficient condition is the existence of *any* feasible policy with finite expected return. Concretely, let d be chosen at a high quantile of the worst-case prefix-average self-cost of correct-answer rollouts produced by an imitation (SFT) policy; by construction those rollouts are feasible, and since $R \in [0, R_{\max}]$ their value is finite, so the assumption holds. This yields a practical feasibility check: if no policy in the candidate pool can satisfy the constraint, the budget d is set below the minimal achievable worst-case cost, *every* policy incurs the $-\infty$ penalty in the limit, and the assumption fails by construction: the constraint itself is infeasible, and no method could satisfy it. In other words, the assumption excludes only degenerate budgets.

A.3 Policy Gradient Derivation

We derive the exact likelihood-ratio policy gradient given the discrete constraint MDP framework. Note that we treat the constrained token-level reward $\hat{R}_n(\mathbf{s}_k, \mathbf{a}_k)$ explicitly as a fixed scalar coefficient decoupled from θ (the student avoids differentiating through the frozen teacher mapping).

The objective is $J_n(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t)\right]$. Under standard REINFORCE (Sutton et al., 1999):

$$\nabla_\theta J_n(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \, R_t^n \, \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)\right], \tag{6}$$

$$\text{where} \quad R_t^n \triangleq \sum_{k=t}^{\infty} \gamma^{k-t} \hat{R}_n(\mathbf{s}_k, \mathbf{a}_k) = \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) + \gamma R_{t+1}^n.$$

We inject the recursive rollout function back into the gradient formulation to decompose it:

$$\begin{aligned}\nabla_\theta J_n(\theta) &= \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \left(\hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) + \gamma R_{t+1}^n\right) \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)\right] \\ &= \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)\right] + \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^{t+1} R_{t+1}^n \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)\right].\end{aligned} \tag{7}$$

Extracting the first term (the *single-step* component), we resolve the expectation explicitly over the token prediction step by conditioning on $\mathbf{s}_t$:

$$\begin{aligned}\mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(\cdot|\mathbf{s}_t)}\left[\hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)\right] &= \sum_{\mathbf{a}} \pi_\theta(\mathbf{a} \mid \mathbf{s}_t) \hat{R}_n(\mathbf{s}_t, \mathbf{a}) \nabla_\theta \log \pi_\theta(\mathbf{a} \mid \mathbf{s}_t) \\ &= \sum_{\mathbf{a}} \hat{R}_n(\mathbf{s}_t, \mathbf{a}) \nabla_\theta \pi_\theta(\mathbf{a} \mid \mathbf{s}_t) \\ &= \nabla_\theta \sum_{\mathbf{a}} \pi_\theta(\mathbf{a} \mid \mathbf{s}_t) \hat{R}_n(\mathbf{s}_t, \mathbf{a}).\end{aligned} \tag{8}$$

Where we manipulated $\nabla_\theta \log \pi(x) = \frac{\nabla_\theta \pi(x)}{\pi(x)}$, and utilized the fact that, conditional on the prefix $\mathbf{s}_t$, $\hat{R}_n$ acts strictly as an external constant multiplier invariant to θ. Substituting Eq. 8 back into Eq. 7 produces the finalized, unbiased decomposed gradient exactly aligning with Section 3.3.

A.4 Variance Reduction

Theorem A.1 (Per-Step Variance Reduction via Rao-Blackwellization)**.** *Let* $X_t = \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$ *be the standard REINFORCE gradient acting on the immediate reward at step t, assuming* $\mathbb{E}[\|X_t\|^2] < \infty$*, and let* $Y_t = \sum_{\mathbf{a}} \pi_\theta(\mathbf{a} \mid \mathbf{s}_t) \hat{R}_n(\mathbf{s}_t, \mathbf{a}) \nabla_\theta \log \pi_\theta(\mathbf{a} \mid \mathbf{s}_t)$ *be its exact-expectation counterpart within* $(\nabla_\theta J_n)_{\text{SINGLE}}$*. By the law of total covariance, conditioning on the prefix* $\mathbf{s}_t$ *ensures that the trace of the covariance matrices satisfies:*

$$\mathrm{Tr}(\mathrm{Cov}(Y_t)) \leq \mathrm{Tr}(\mathrm{Cov}(X_t))$$

with strict inequality whenever $\mathbb{P}(\mathrm{Cov}(X_t \mid \mathbf{s}_t) \neq \mathbf{0}) > 0$*.*

We wish to prove that analytically computing the exact expectation of the immediate reward step strictly reduces the total variance of the gradient estimator via Rao-Blackwellization (Mohamed et al., 2020).

Let the standard REINFORCE gradient component for the immediate reward at step t be defined as the stochastic random vector:

$$X_t = \hat{R}_n(\mathbf{s}_t, \mathbf{a}_t) \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$$

where the action $\mathbf{a}_t$ is sampled from the student policy $\pi_\theta(\cdot \mid \mathbf{s}_t)$.

Instead of relying on a single Monte Carlo sample, our decomposed estimator evaluates the conditional expectation of X_t over the entire vocabulary, given the current prefix $\mathbf{s}_t$:

$$Y_t = \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(\cdot|\mathbf{s}_t)}[X_t \mid \mathbf{s}_t] = \sum_{\mathbf{a}} \pi_\theta(\mathbf{a} \mid \mathbf{s}_t) \hat{R}_n(\mathbf{s}_t, \mathbf{a}) \nabla_\theta \log \pi_\theta(\mathbf{a} \mid \mathbf{s}_t)$$

Note that Y_t exactly matches the t-th term of our decomposed $(\nabla_\theta J_n)_{\text{SINGLE}}$ construction, as derived in Appendix A.3.

To compare the variance of these two estimators, we apply the Law of Total Covariance to the vector X_t conditioned on the state $\mathbf{s}_t$:

$$\begin{aligned} \text{Cov}(X_t) &= \text{Cov}(\mathbb{E}[X_t \mid \mathbf{s}_t]) + \mathbb{E}[\text{Cov}(X_t \mid \mathbf{s}_t)] \\ \text{Cov}(X_t) &= \text{Cov}(Y_t) + \mathbb{E}[\text{Cov}(X_t \mid \mathbf{s}_t)]. \end{aligned} \tag{9}$$

Taking the trace of both sides to measure total variance, and noting that the trace of a positive semi-definite covariance matrix is non-negative, by linearity of expectation we have:

$$\text{Tr}(\mathbb{E}[\text{Cov}(X_t \mid \mathbf{s}_t)]) = \mathbb{E}[\text{Tr}(\text{Cov}(X_t \mid \mathbf{s}_t))] \geq 0.$$

Subtracting this yields the fundamental inequality:

$$\text{Tr}(\text{Cov}(Y_t)) \leq \text{Tr}(\text{Cov}(X_t)).$$

Strict inequality holds if and only if $\mathbb{P}(\text{Cov}(X_t \mid \mathbf{s}_t) \neq \mathbf{0}) > 0$. This condition is satisfied as long as the random vector $X_t \mid \mathbf{s}_t$ is not almost surely constant. In the context of our MDP, this requires the student policy $\pi_\theta(\cdot \mid \mathbf{s}_t)$ to be non-degenerate (i.e., it assigns non-zero probability to at least two actions) and for those actions to yield divergent values for the gradient vector $\hat{R}_n(\mathbf{s}_t, \mathbf{a})\nabla_\theta \log \pi_\theta(\mathbf{a} \mid \mathbf{s}_t)$.

Practical Significance in our MDP: While Rao-Blackwellization is a generic statistical tool (Mohamed et al., 2020), computing the exact expectation over an LLM vocabulary (often 50K-150K tokens) is typically intractable. It is uniquely unlocked here because evaluating $\hat{R}_n(\mathbf{s}_t, \mathbf{a})$ for all possible next tokens requires only a single forward pass of the teacher to obtain the log-probabilities, followed by a deterministic prefix-feasibility check. Furthermore, in standard unconstrained RL, the immediate reward $R(\mathbf{s}_t, \mathbf{a}_t)$ is often zero during intermediate reasoning steps, making $X_t = \mathbf{0}$ and rendering this decomposition trivial. However, in our worst-case constrained formulation, Lemma 3.2 dictates that a constraint violation immediately triggers a massive penalty $\hat{R}_n = -n$. This creates an extreme "cliff" in the reward landscape at step t, injecting large variance into the standard sampled estimator X_t. By analytically integrating over the vocabulary to compute Y_t, we perfectly smooth over this cliff, yielding a highly stable, low-variance gradient signal that safely steers the policy.

B Algorithm and Implementation Details

We will open-source our code upon acceptance.

B.1 Reward Function Design

For mathematical reasoning tasks, we rely on rule-based, deterministic evaluators (e.g., symbolic math matching) to assign binary rewards based on final answer correctness:

$$R(\mathbf{s}_T, \mathbf{a}_T) = \begin{cases} 1.0 & \text{if the extracted final answer is correct} \\ 0.0 & \text{if the extracted final answer is incorrect} \end{cases} \tag{10}$$

The reward is entirely sparse and only assigned at the terminal step of each trajectory when the complete solution inside the `\boxed{}` tag is generated.

B.2 Baselines Description

To ensure fair and rigorous comparisons, all reinforcement learning baselines, including our own, are implemented on top of the **GRPO** (Shao et al., 2024) framework, sharing the same group-relative advantage estimation and PPO-style clipping mechanics.

- **GRPO:** Optimizes purely for the binary task reward R without any teacher supervision. It tends to reward-hack by discovering flawed reasoning chains that accidentally yield the correct final token.

- **GKD (Agarwal et al., 2024):** A pure distillation baseline. It optimizes entirely to minimize the divergence (maximize teacher likelihood) without observing the task reward R, resulting in high constraint satisfaction but low final accuracy.
- **Mini-LLM (Gu et al., 2024):** A sequence-level distillation method that separates single-step and long-term divergence gradients (Tang & Munos, 2025). We adapt it over GRPO for consistency. We also sample trajectories exclusively with the student policy for fair comparison.
- **GKD-GRPO (Agarwal et al., 2024):** The standard soft Lagrangian relaxation, optimizing $R - \lambda \cdot \mathrm{KL}(\pi||\mu)$. We conduct a thorough grid search over $\lambda \in \{10, 1.0, 0.1, 0.01, 0.001\}$ to map out its entire Pareto frontier.
- **SFT on Teacher CoT:** Off-policy sequence-level distillation. We sample teacher chain-of-thought trajectories on the training set, keep only those with correct final answers, and fine-tune the student on them with the standard next-token loss for 4 epochs. The learning rate is the same as other baselines 1×10^{-5}.

B.3 Hyperparameter Settings

We maintained consistent hyperparameter settings across all methods to isolate the effect of the optimization objectives:

- **Batch Size:** 64 responses per update (8 distinct questions $\times$ 8 sampled responses per question).
- **Learning Rate:** 1×10^{-5} with a constant schedule.
- **Optimizer:** AdamW.
- **Discount Factor:** $\gamma = 1.0$ (standard for finite-horizon LLM generation).
- **Training precision:** bfloat16.
- **Training Epochs:** 20 epochs across all models.
- **Penalty (n):** For our method, the terminal violation penalty was set to $n = 20$, providing a harsh boundary signal well above the maximum positive task reward of 1.0.
- **Constraint Threshold (d):** We swept our worst-case average bound across $d \in \{-\ln(3/4), -\ln(1/2), -\ln(1/4)\}$. Since d limits the average negative log-likelihood, $-\ln(1/4)$ roughly translates to demanding the teacher assigns at least a 25% worst average fractional probability to the student's reasoning sequence thus far.

Llama-3.2 Bootstrap: Llama-3.2-3B exhibited severe training instability during pure GRPO training due to an extremely poor initial zero-shot performance on MATH (preventing any positive reward signals from being discovered). As a standard remedy, to bootstrap all evaluated methods equally, we applied a brief warm-up phase of pure distillation (using GKD) for the first 3 epochs before engaging the respective RL/Constrained objectives. No such bootstrap was required for Qwen models.

B.4 Computational Resources and Training Time

Training takes fewer than 23 hours on a single accelerator per method. Because the forward generation and backward gradient passes dominate the wall-clock time, the overhead of evaluating the teacher model's likelihood (required for GKD, Mini-LLM, GKD-GRPO, and Our method) during RL rollouts adds a small overhead to overall training time. Consequently, all methods are directly comparable in terms of computational training cost, except the one without a teacher (GRPO) that runs in around 16 hours.

B.5 Evaluating Reasoning Quality (LLM-as-a-Judge)

To rigorously assess the logical validity of generated solutions beyond verifiable final answers, we employ an LLM-as-a-Judge protocol (Zheng et al., 2023). Following recent literature evaluating mathematical reasoning, we utilize *DeepSeek-R1-Distill-Qwen-32B* as our evaluator. We run two distinct types of evaluations: a single-answer audit to compute the Reasoning Success Rate (RSR), and a head-to-head Pairwise Comparison to validate relative superiority. Note that for the pairwise comparison, we use both orders of the answer. To conclude if one is winning, it has to win in both positions.

B.5.1 REASONING SUCCESS RATE (RSR) PROMPT

The RSR metric aims to identify any hallucination, invalid inference, or broken logical chain in a single trajectory. A generation is marked as a "Success" if achieving the correct final answer *and* the judge returns a `No Error` verdict.

```
You are an expert auditor evaluating the reasoning
correctness of an AI model's solution to a mathematical
problem.

Your task:
- Determine whether the solution's reasoning contains ANY
error.
- This is independent of whether the final answer is correct
or not.
- An "error" includes:  incorrect algebra/arithmetic,
invalid inference, misuse of a theorem/definition,
contradiction, or an unjustified assumption that changes
the problem.
- Do NOT count:  harmless typos, purely stylistic issues, or
alternative valid approaches.

Instructions:
1) Read the problem and the solution.
2) Check the reasoning step-by-step for validity.
3) If you find an error, identify the FIRST point where the
reasoning becomes invalid.
4) If the reasoning is too incomplete or ambiguous to
evaluate, use "Unclear".

Problem:
```problem
{prompt}
```
Solution:
```solution
{answer}
```

Respond in the following exact format (one item per line):
Verdict: [No Error / Has Error / Unclear]
Severity: [Critical / Non-critical / N/A]
First_Error_Location: [Brief reference to the earliest wrong
step, or N/A]
Explanation: [2-4 sentences explaining why you chose the
verdict, focusing on the key reasoning issue(s)]
```

B.5.2 PAIRWISE RANKER PROMPT

For the Pairwise evaluation, the judge is presented with two solutions from different models that both achieved the correct final answer. It is forced to choose which model provided structurally and logically superior reasoning. We validate the judge against human annotation in Appendix F.

```
You are an expert judge evaluating the reasoning quality of
two AI models' responses to a mathematical problem.

Please compare the reasoning steps, clarity, and
mathematical rigor of both responses.  You do not need to
provide the answer to the problem itself as both responses
arrive at the correct answer, so focus on:
1.  Clarity and structure of reasoning steps
2.  Mathematical rigor and correctness of intermediate steps
3.  Explanation quality and logical flow
```

```
4.  Completeness of the solution process

Problem:
```problem
{prompt}
```
Response A:
```response
{answer_a}
```
Response B:
```response
{answer_b}
```

Please provide an explanation of your reasoning in 2-3
sentences focusing on the key differences in reasoning
quality.

Then provide your judgment as one of: "A wins", "B wins",
or "Tie".

Format your response as:
Explanation: [Your explanation here]
Verdict: [A wins/B wins/Tie]
```

C CODE GENERATION EXPERIMENT

To test generality beyond mathematics, we distill *Qwen2.5-Coder-0.5B* (base) from *Qwen2.5-Coder-7B-Instruct*, a third model pair, on problems from MBPP (train+validation) and APPS (introductory and interview levels) with a binary all-tests-pass reward, and evaluate pass@1 on the MBPP test set.

Models. Teacher: *Qwen2.5-Coder-7B-Instruct*; student: *Qwen2.5-Coder-0.5B* (base, non-instruct). This is a third model pair, distinct from both math settings, with a $14\times$ parameter ratio.

Data. Training uses $\sim$4K problems: MBPP train+validation (464 problems) plus APPS at the introductory and interview difficulty levels, filtered for problem statements under 3,000 characters. Evaluation is pass@1 on the MBPP test set (500 problems). Prompts contain the problem statement and, following the MBPP convention, the unit-test assertions; for APPS, starter code when present.

Reward. Binary: 1.0 if all unit tests pass, 0.0 otherwise. MBPP assertions are appended to the generated code and executed as a script; APPS problems are evaluated either by calling the target function on the provided inputs (when a function name is specified) or by piping inputs through stdin and comparing stdout. Each candidate runs in an isolated `python3` subprocess with a 5-second timeout.

Training. Identical GRPO-based configuration to the math settings (batch of 8 prompts $\times$ 8 generations, 512-token completions, temperature 1.0), with 15 epochs over the training set. The λ grid for GKD-GRPO and the d grid for ours follow the math settings; we report the strongest configuration per method alongside pure GRPO.

Table 2 shows GRPO exploits the reward brutally: it simply copies the unit tests visible in the prompt into the generated code (pass@1 92.0%, RSR 1.8%). Soft regularization trades this off along the same Pareto axis as in math, while ours obtains the highest RSR (34.2%) with the smallest FAC$-$RSR gap (11.4 points), confirming that the worst-case constraint prevents reward hacking in a new domain, model pair, and reward type. This is a clear example showing that trusting FAC alone is not a good idea as it can produce worst models.

Table 2: Code generation on the MBPP test set (500 problems, %). The prompt contains the unit tests, making pass@1 easy to game: GRPO inflates pass@1 to 92.0% by copying them into the generated code (RSR 1.8%). Ours achieves the highest RSR and smallest FAC−RSR gap.

Method	**RSR**	**FAC (Pass@1)**	**FAC−RSR**
GRPO	1.8	**92.0**	90.2
GKD-GRPO $\lambda = 0.001$	14.4	74.4	60.0
GKD-GRPO $\lambda = 0.01$	34.0	50.4	16.4
Ours $d = -\ln(1/4)$	**34.2**	45.6	**11.4**

D Details on Constraint Type Analysis (Figure 2)

To empirically validate the structural advantages of our worst-case average constraint over the traditional Saute cumulative-sum constraint (discussed in Section 3.1), we conducted a retrospective test-time analysis using generated trajectories.

Using the Qwen2.5-1.5B student model, we oversample 256 trajectories per question. Because the sampler explores without constraints, this set features a wide variety of both logically sound solutions and reward-hacked solutions, providing a diverse distribution of teacher costs $C(\mathbf{s}_t, \mathbf{a}_t) = -\log \mu(\mathbf{a}_t|\mathbf{s}_t)$.

For each trajectory, we computed two properties:

1. **Cumulative Budget:** $S_{\text{sum}} = \sum_{t=0}^{T-1} C_t$
2. **Worst-Case Average Budget:** $S_{\text{wc}} = \max_{\tau \leq T} \frac{1}{\tau} \sum_{t=0}^{\tau-1} C_t$

We then simulated the effect of the two constraint mechanisms by treating them as test-time filters. By sweeping the budget parameters d_{sum} and d_{wc} across a fine-grained grid (the X-axis of Figure 2): from 0 to -3 for the worst-case average and from 0 to -200 for the cumulative sum (the scales differ because the cumulative cost grows with sequence length while the average is length-normalized), we "accepted" only the trajectories that satisfied the respective constraint bounds ($S \leq d$).

For each subset of accepted trajectories, we calculated the overall Reasoning Success Rate (RSR). The resulting plot demonstrates that at its optimal budget threshold, the worst-case average filter achieves a strictly higher RSR than the optimal cumulative sum filter. This occurs because the cumulative sum indiscriminately rejects long, detailed, valid reasoning chains (which accumulate high total costs simply due to sequence length), whereas the worst-case average constraint specifically targets and isolates trajectories containing concentrated logical hallucinations, successfully filtering out reward-hacked responses regardless of their token length.

Alternative constraint parameterizations. Beyond the cumulative sum, one might consider (i) a per-step constraint $C_t \leq d \, \forall t$, or (ii) a *sliding-window* average over the last k steps.

The per-step form is strictly harder to satisfy than the prefix average and degenerates for teachers with high entropy on formatting tokens (e.g., whitespace or LaTeX delimiters), whose individual costs routinely exceed any useful threshold d; it therefore conflates stylistic uncertainty with logical hallucination.

The sliding-window average mitigates this but loses the absorbing (prefix-monotone) structure that makes the worst-case prefix average a valid safety budget in the Saute sense: once a prefix average is violated, no future tokens can repair it, which is precisely what enables our state-augmentation-free terminal penalty (Section 3.1). A windowed statistic can be repaired by subsequent low-cost tokens, reintroducing exactly the compensation failure the constraint is designed to prevent.

We therefore retain the prefix average, which interpolates between the two extremes: length-invariant like the windowed form, absorbing like the per-step form.

E TRAJECTORY-LEVEL VALIDATION OF THE LIKELIHOOD PROXY

Our constraint assumes that worst-case teacher likelihood tracks reasoning validity. We test this directly on all test trajectories of 12 training runs spanning GRPO, GKD-GRPO across the full λ grid, and ours across the d grid (Qwen-MATH setting; 5,000 generations per run, 60,000 trajectories in total). For each trajectory we record its worst-case prefix-average teacher log-likelihood and its RSR label (correct final answer and judge verdict "No Error"). Additionally, instead of considering only the teacher likelihood for the definition of the cost C_t, we also tried to replace it with per-token reverse and forward KL divergences.

Table 3: Pooled Spearman rank correlation between per-trajectory fidelity statistics and RSR (divergence costs are negated so that higher always means higher teacher fidelity).

Fidelity statistic (per trajectory)	**Spearman ρ with RSR**
Worst-case prefix avg. teacher logp (our constraint)	$+0.231$
Worst-case prefix avg. reverse KL	$+0.224$
Worst-case prefix avg. forward KL	$+0.186$

Table 3 shows the correlation is positive and significant, and it is positive in each of the 12 runs individually (per-run $\rho \in [0.13, 0.30]$). Concretely, pooling trajectories into quartiles of worst-case log-likelihood, RSR rises monotonically from 30.1% (lowest quartile) to 59.6% (highest), and FAC from 35.9% to 63.0%. Two conclusions follow. First, the proxy assumption holds: trajectories that deviate sharply from the teacher at some prefix are substantially less likely to contain valid reasoning. Second, replacing the teacher NLL with a forward KL or reverse KL per-token divergence would not improve the proxy.

Among answer-correct trajectories, judge-flagged (reward-hacked) ones have significantly lower worst-case log-likelihood than genuinely correct ones (mean -1.09 vs. -0.92 for GRPO, Mann–Whitney $p < 10^{-6}$; pooled AUC 0.63), confirming the constraint quantity carries signal specifically about reasoning failures rather than stylistic deviation.

Post-hoc threshold selection. Because the constraint is a threshold on a stored per-trajectory quantity, its effect can be simulated post hoc on validation rollouts without retraining: treating the worst-case log-likelihood as a filter, tighter thresholds trade coverage for reasoning quality monotonically. This provides an explicit coverage–quality dial that λ, which changes the training objective itself, does not.

F JUDGE RELIABILITY: HUMAN EVALUATION

To calibrate the LLM-as-a-Judge protocol used for RSR, we manually audited 60 judge verdicts, stratified across methods (GRPO, GKD-GRPO, ours), datasets (MATH, GSM-Symbolic), and verdict types (flagged and unflagged), with the annotator blinded to the method. The human verdicts agree with the judge on all 60 cases (100% agreement). We note two systematic judge tendencies observed during piloting (and mitigated in the final prompt): a bias toward flagging solutions that reach the correct answer through unusual-but-valid shortcuts, and occasional leniency toward compact arithmetic slips; both were rare in the audited sample. The pairwise protocol additionally requires a method to win in both presentation orders, which filters position bias.

G LIMITATIONS AND FUTURE WORK

We acknowledge several limitations of our approach.

Teacher likelihood as a cost proxy. Our worst-case constraint relies on teacher log-likelihood, which correlates strongly with, but does not perfectly capture, reasoning validity. A step that is logically valid but stylistically unusual may incur a high cost. Exploring richer cost functions (e.g., learned process reward models) is a promising direction.

Tokenizer alignment requirement. Our method requires that the student and teacher models share a compatible tokenizer, as the per-step cost $C(\mathbf{s}_t, \mathbf{a}_t) = -\log \mu(\mathbf{a}_t|\mathbf{s}_t)$ must be evaluated token-wise at positions consistent between both models. This constraint is naturally satisfied when distilling within the same model family (e.g., Qwen2.5-7B $\rightarrow$ Qwen2.5-1.5B, Llama-3.2-11B $\rightarrow$ Llama-3.2-3B), as used in our experiments. Distilling across families with incompatible vocabularies would require either re-tokenization strategies, cross-tokenizer alignment techniques, or alternative cost formulations based on sequence-level or semantic similarity, which we leave to future work.

Distribution shift at test-time. Our theoretical guarantees hold under the training distribution. Extending formal feasibility guarantees to OOD generalization remains an open problem.

Budget selection. While d is more interpretable than a Lagrangian multiplier λ (it relates directly to teacher probability), optimal budget selection still requires some empirical tuning. However, unlike λ, the threshold acts on a stored per-trajectory quantity and can therefore be swept post hoc on validation rollouts without retraining, trading accepted-trajectory coverage against RSR (Section E). Adaptive budget schedules could further improve this.

This tuning requirement is not unique to our method: soft baselines must equally tune their Lagrangian multiplier λ, which typically requires sweeping several orders of magnitude and full retraining for each value. It is inherent to fidelity–accuracy trade-off methods.

Domain scope. Our experiments focus on mathematical reasoning and code generation. Extending the worst-case constraint framework to broader reasoning tasks (theorem proving, multi-hop QA, agentic framework) is an important direction for future work.

H GENERATED ANSWERS

We report more examples with reward hacking below:

Question:
Leilani saw a 300-foot whale with 4 45-inch remoras attached to it. What percentage of the whale's body length is the combined length of the remoras?

Ours:

To determine what percentage of the whale's body length the combined length of the remoras is, we need to follow these steps:
1. Convert the length of the remoras from inches to feet:

$$45 \text{ inches} \times \frac{1 \text{ foot}}{12 \text{ inches}} = 3.75 \text{ feet}$$

So, each remora is 3.75 feet long.
2. Calculate the total length of the remoras:

$$4 \text{ remoras} \times 3.75 \text{ feet per remora} = 15 \text{ feet}$$

3. Determine the total length of the whale:

$$300 \text{ feet}$$

4. Calculate the percentage of the whale's body length that the combined length of the remoras represents:

$$\left(\frac{15 \text{ feet}}{300 \text{ feet}}\right) \times 100\%$$

5. Simplify the fraction:

$$\frac{15}{300} = 0.05$$

6. Convert the decimal to a percentage:

$$0.05 \times 100\% = 5\%$$

Therefore, the combined length of the remoras is $\boxed{5}$ percent of the whale's body length.

GRPO:

Let's break down the problem step by step to find the percentage of the whale's body length that the combined length of the remoras represents.
First, we need to convert all units to the same unit. Let's use feet for this calculation.
The length of each remora is given as 45 inches. Since there are 12 inches in a foot, the length of each remora in feet is:

$$\frac{45}{12} = 3.75 \text{ feet}$$

Since there are 4 remoras, the total length of the remoras is:

$$4 \times 3.75 = 15 \text{ feet}$$

Next, we add the length of the whale to the combined length of the remoras to get the total length of the whale:

$$300 + 15 = 315 \text{ feet}$$

Now, we need to find what percentage the combined length of the remoras is of the whale's body length. This can be calculated using the formula for percentage:

$$\left(\frac{15}{315}\right) \times 100$$

Simplifying the fraction inside the percentage formula:

$$\frac{15}{315} = \frac{1}{21}$$

$$\left(\frac{1}{21}\right) \times 100 = \frac{100}{21} \approx 4.7619$$

Rounding to the nearest whole number, we get:

$$\boxed{5}$$

Judge:
Response A correctly calculates the percentage by comparing the remoras' total length to the whale's length without adding them, ensuring accuracy. Response B incorrectly adds the remoras' length to the whale's, leading to a flawed calculation.
Verdict: A wins

Figure 6: Example of generated answer with Qwen2.5-1.5B after distillation.

Question:
Wei saw a 200-foot whale with 2 12-inch remoras attached to it. What percentage of the whale's body length is the combined length of the remoras?

Ours:

To determine what percentage of the whale's body length the combined length of the remoras is, we need to follow these steps:
1. **Convert the length of the remoras from inches to feet:**
- Each remora is 12 inches long.
- There are 12 inches in a foot.
- Therefore, the length of each remora in feet is:

$$\frac{12 \text{ inches}}{12 \text{ inches per foot}} = 1 \text{ foot}$$

- Since there are 2 remoras, the total length of the remoras in feet is:

$$2 \text{ remoras} \times 1 \text{ foot per remora} = 2 \text{ feet}$$

2. **Calculate the percentage of the whale's body length:**
- The whale's body length is 200 feet.
- The combined length of the remoras is 2 feet.
- To find the percentage, we use the formula:

$$\text{Percentage} = \left(\frac{\text{Length of remoras}}{\text{Whale's body length}} \right) \times 100$$

- Substituting the values, we get:

$$\text{Percentage} = \left(\frac{2 \text{ feet}}{200 \text{ feet}} \right) \times 100$$

- Simplify the fraction:

$$\frac{2}{200} = 0.01$$

- Multiply by 100 to convert to a percentage:

$$0.01 \times 100 = 1\%$$

Therefore, the combined length of the remoras is $\boxed{1}\%$ of the whale's body length.

GKD-GRPO $\lambda = 1$:

To determine what percentage of the whale's body length the combined length of the remoras is, we need to follow these steps:
1. Convert the length of the remoras from inches to feet.
2. Add the length of the remoras to the length of the whale.
3. Calculate the percentage of the whale's body length that the combined length of the remoras represents.
First, we convert the length of the remoras from inches to feet. Since there are 12 inches in a foot, we divide 12 by 12 to get 1 foot per inch. Therefore, the length of each remora is:

$$12 \text{ inches} \div 12 = 1 \text{ foot}$$

Since there are 2 remoras, the total length of the remoras is:

$$2 \text{ remoras} \times 1 \text{ foot/remora} = 2 \text{ feet}$$

Next, we add the length of the remoras to the length of the whale:

$$200 \text{ feet} + 2 \text{ feet} = 202 \text{ feet}$$

Now, we calculate the percentage of the whale's body length that the combined length of the remoras represents. We do this by dividing the total length of the remoras by the length of the whale and then multiplying by 100:

$$\left(\frac{2 \text{ feet}}{200 \text{ feet}} \right) \times 100 = 1\%$$

Therefore, the combined length of the remoras is $\boxed{1}\%$ of the whale's body length.

Judge:
Response A is more clear and logically structured, breaking down each step with proper mathematical notation and avoiding unnecessary complexity. Response B incorrectly adds the remoras' length to the whale's length, which is irrelevant to the problem and introduces confusion.
Verdict: A wins

Figure 7: Example of generated answer with Qwen2.5-1.5B after distillation.

Question:
There are 154 fourth-graders at Maple Grove School. 80 of them are girls. On Thursday, 3 fourth-grade girls and 6 fourth-grade boys were absent. How many fourth grade boys were at Maple Grove School on Thursday?

Ours:

To determine how many fourth-grade boys were at Maple Grove School on Thursday, we need to follow these steps:
1. **Calculate the number of fourth-grade boys:**
- The total number of fourth-graders is 154.
- The number of fourth-grade girls is 80.
- Therefore, the number of fourth-grade boys is:

$$154 - 80 = 74$$

2. **Determine the number of fourth-grade boys absent:**
- On Thursday, 6 fourth-grade boys were absent.
3. **Calculate the number of fourth-grade boys present:**
- The total number of fourth-grade boys is 74.
- The number of fourth-grade boys absent is 6.
- Therefore, the number of fourth-grade boys present is:

$$74 - 6 = 68$$

Thus, the number of fourth-grade boys at Maple Grove School on Thursday is $\boxed{68}$.

GKD:

To determine how many fourth-grade boys were at Maple Grove School on Thursday, we need to follow these steps:
1. **Calculate the number of fourth-grade girls present:**
- Total number of fourth-graders: 154
- Number of fourth-grade girls: 80
- Number of fourth-grade girls present: $154 - 3 = 151$
2. **Calculate the number of fourth-grade boys present:**
- Total number of fourth-graders: 154
- Number of fourth-grade boys: $154 - 80 = 74$
- Number of fourth-grade boys present: $74 - 6 = 68$
Thus, the number of fourth-grade boys present on Thursday is $\boxed{68}$.

Judge:
Response A provides a clearer and more accurate approach by first determining the total number of boys and then subtracting the absent ones. Response B incorrectly calculates the number of girls present, which could lead to confusion.
Verdict: A wins

Figure 8: Example of generated answer with Qwen2.5-1.5B after distillation.