

ENHANCING LANGUAGE MODEL REASONING WITH STRUCTURED MULTI-LEVEL MODELING

Anonymous authors

Paper under double-blind review

ABSTRACT

Inference-time scaling enhances a model’s reasoning by extending its chain-of-thought (CoT). However, existing approaches typically rely on a single policy trained with outcome-reward reinforcement learning (RL), which often suffers from long-horizon plan failures, i.e., the implicit plan drifts away from any valid strategy. This problem is particularly severe for smaller language models (LMs) with long CoTs due to their limited capacity. To address this, we propose Multi-Level Reasoning (MLR), which reformulates long-CoT generation as a two-level stochastic process. Specifically, MLR employs two policies: a high-level planner that generates step descriptors (abstract subgoals) and a low-level executor that produces detailed content conditioned on these descriptors. The planner then generates the next subgoal based on the summarized current step, forming an alternating plan–execute loop. To maintain scalability, we adopt a minimal design, where the base model serves as the low-level policy and a lightweight LoRA module implements the high-level policy. For training, we observe that outcome-reward RL is inefficient and weakly informative for long trajectories (e.g., those exceeding 4K tokens). To overcome this, we introduce online Step-DPO, a process-level preference optimization scheme that leverages Twisted Sequential Monte Carlo (TSMC) to provide scalable stepwise supervision. This yields more effective training, improved stability, and higher accuracy. Extensive experiments on challenging math, science, and logical reasoning benchmarks show that, with only 10% SFT data and 5% of preference data, MLR outperforms both the DeepSeek-R1 distillation and the outcome-reward RL baselines across multiple base models and tasks. More importantly, MLR exhibits slower performance degradation on long-horizon reasoning, demonstrating stronger robustness under extended CoT generation.

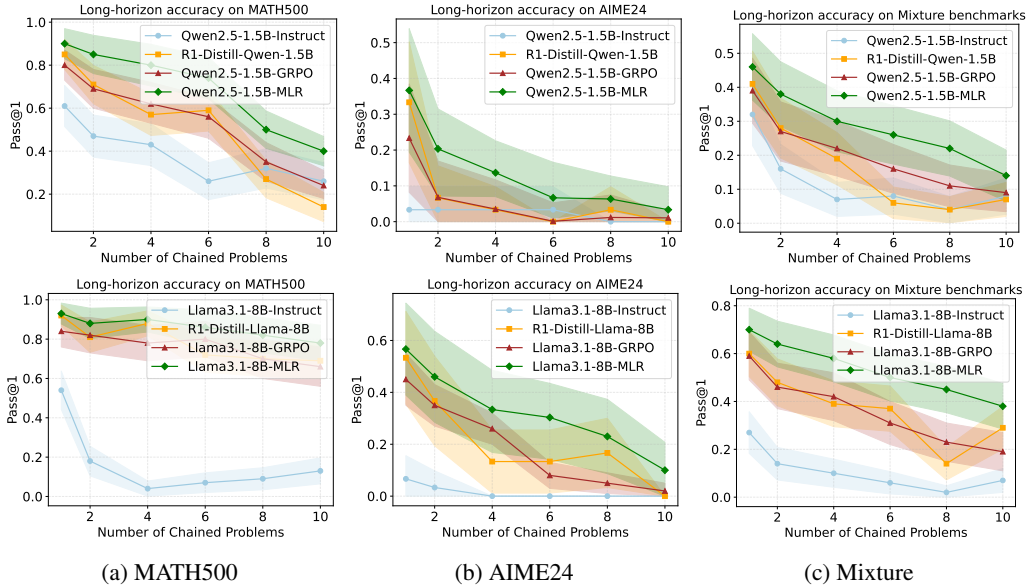


Figure 1: Long-horizon reasoning accuracy on (a) MATH500, (b) AIME24, and (c) Mixture benchmarks (MATH500, AIME24, GPQA, BoardGameQA). We simulate long-horizon reasoning by concatenating multiple problems in the prompt and report average accuracy. MLR consistently degrades more slowly than all baselines. Additional experimental details and statistics are provided in Section D.

1 INTRODUCTION

OpenAI’s o1 series (OpenAI, 2024) introduce inference-time scaling by increasing the length of the Chain-of-Thought (CoT) (Wei et al., 2022) reasoning process. This design yields significant improvements in complex reasoning tasks compared to non-reasoning models, marking a major step forward in language model (LM) capabilities. Building on this idea, DeepSeek (Guo et al., 2025) proposes a large-scale reinforcement learning (RL) pipeline that directly incentivizes the generation of long CoTs through policy optimization. Despite the effectiveness of these methods, approaches that rely on single-policy long CoTs for RL face important limitations, including long-horizon plan failures and the inherent challenges of RL with sparse outcome rewards¹. These issues are especially pronounced for smaller LMs with long CoTs due to their limited capacity.

In reinforcement learning, long-horizon policy learning remains a fundamental challenge due to the difficult credit assignment (Kaelbling et al., 1996). To address this, hierarchical reinforcement learning (HRL) (Dietterich, 2000) has been proposed, where high-level and low-level policies are learned to operate at different temporal abstractions. While HRL proven effective in domains such as robotics (Nachum et al., 2018; Gupta et al., 2019), its application to LMs² presents several challenges: 1) Scalability. Modeling multiple policies, especially when implemented as separate LMs, can incur significant computational overhead. Naïve multi-agent setups will suffer from high communication and synchronization costs, making HRL computationally expensive to scale (Guo et al., 2024b). 2) Flexibility. The existing full-plan-then-execute structure (Huang et al., 2022; Xu et al., 2023) is brittle in LM-based reasoning tasks where new information or execution failures may require mid-course corrections. Thus, it is desirable to allow the high-level plan to evolve dynamically based on the ongoing progress of the low-level execution process. To address these challenges, we propose a multi-level reasoning (MLR) strategy, where the model alternates between generating a step-level descriptor and its corresponding detailed content, to enable efficient multi-policy modeling and dynamic plan adaptation.

More importantly, we introduce an online Step-DPO pipeline for long CoT training, which significantly accelerates training and improves reasoning performance. Existing RL fine-tuning frameworks struggle to obtain effective process-level supervision (Guo et al., 2025). First, evaluating the correctness of intermediate steps is inherently difficult. Automated annotation using LLMs (Wang et al., 2023b) often yields unreliable or noisy signals, while manual annotation (Lightman et al., 2023) is prohibitively expensive at scale. Second, introducing a separate process reward model (PRM) adds complexity. It is vulnerable to reward hacking (Gao et al., 2023), requires substantial training data, and complicates the pipeline by necessitating repeated retraining. To overcome these limitations, we repurpose Twisted Sequential Monte Carlo (TSMC) (Doucet et al., 2001; Del Moral et al., 2006; Briers et al., 2010) as a process-level supervision signal for Step-DPO. In the LM-based reasoning setting, the importance weight in TSMC estimates how much more likely a partial trajectory is to lead to a correct outcome under the target distribution compared to the current policy. We then define the process preference between two candidate continuations at the same step by comparing their incremental log-weights. This formulation has two key advantages: 1) it converts the multiplicative nature of sequential importance weights into an additive form, improving numerical stability; 2) it aligns naturally with the pairwise preference structure of DPO training. Empirically, our approach provides stable and informative step-level preferences, leading to more efficient training and stronger performance on complex reasoning tasks.

We summarize our key contributions as follows:

- We propose a novel multi-level reasoning (MLR) framework that directly addresses the limitations of single-policy long-CoT approaches, such as long-horizon plan failures and inefficiency. MLR decomposes reasoning into alternating high-level step descriptors and low-level detailed content, enabling structured abstraction, dynamic plan adjustment, and more reliable long-horizon reasoning.
- We repurpose Twisted Sequential Monte Carlo (TSMC) to provide process-level preferences for Step-DPO training. This eliminates the need for a separate process reward model,

¹See Section 2 for a detailed discussion.

²Note that HRL differs from prompting-based approaches that decompose tasks in CoTs. Instead, it treats high- and low-level actions as separate distributions with distinct objectives and temporal scopes.

reducing overhead while supplying stable and informative supervision throughout long reasoning trajectories.

- We perform extensive experiments on challenging benchmarks in math, science, and logical reasoning. Results show that our approach consistently outperforms both distillation-based long-CoT methods and RL methods that rely solely on outcome rewards.

2 INFERENCE-TIME SCALING VIA LONG CHAIN-OF-THOUGHT

Formulation. Consider a query q , reasoning models generate a CoT c before producing the final response a , where q, c, a are all sequences of tokens, i.e., $c = (c[1], c[2], \dots, c[L])$. To improve model performance, these models extend the length of c by incorporating human-like reasoning behaviors such as exploration, self-verification and reflection. The generation of long CoTs follows the standard autoregressive modeling: the probability of each token $c[l]$ depends only on its preceding tokens $(c[1 : l - 1])$, which enables the factorization of the joint likelihood of the entire sequence as:

$$p_{\theta}(c[1 : L]) = \prod_{l=1}^L p_{\theta}(c[l] \mid c[1 : l - 1]). \quad (1)$$

Note that, for notational simplicity, we omit the conditioning on q in Eq. 1 and in the following derivations. Training the model p_{θ} involves maximizing the likelihood of each token conditioned on its prefix, i.e., optimizing $p_{\theta}(c[l] \mid c[1 : l - 1])$ over the training data.

Post-training. Guo et al. (2025) detail how they incentivize the long CoT generation from a base model through large-scale RL without relying on SFT. Specifically, they employ GRPO guided by rule-based outcome reward. For each query q , GRPO samples a group of outputs $\{o_1, o_2, \dots, o_G\}$ from the old policy $\pi_{\theta_{\text{old}}}$, where each output is composed of a CoT followed by the final response, i.e., $o_i = [c_i, a_i]$, and then optimizes the policy π_{θ} by maximizing the corresponding objective.

Discussion on the weakness of single-policy long CoT. The above approach of using single-policy long CoT enables inference-time scaling with LMs, but introduces several issues:

- 1) **Long-horizon plan failures.** In single-policy long CoT generation, the same policy is responsible for both planning and execution. Without guidance or structure, errors can accumulate and cause the implicit plan drifts away from any valid strategy (see examples in Section D).
- 2) **Long-horizon RL with sparse outcome reward.** Long CoTs involve thousands of token-level actions before receiving a reward, which hinders effective credit assignment. As shown in Figure 2, these trajectories can be extremely long, with errors occurring at widely varying positions, which undermines the effectiveness of outcome-based fine-tuning. Moreover, Figure 3 shows that latency and memory usage grow rapidly with trajectory length, while outcome-based supervision requires the entire trajectory to finish before feedback is provided. Consequently, learning is slow and unstable, especially in the early stages when the model rarely produces correct trajectories.

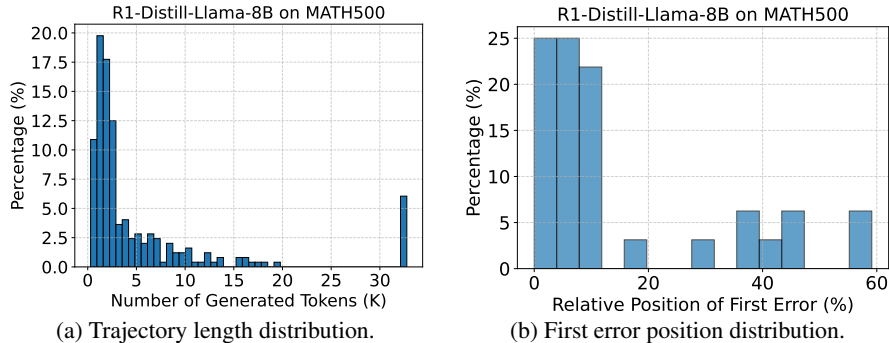


Figure 2: The Chain-of-Thought trajectories can be lengthy and the positions of the first error vary considerably, making outcome-based RL fine-tuning inefficient. The statistics in (b) are based on 100 trajectories with incorrect final answers, where the first error was manually identified.

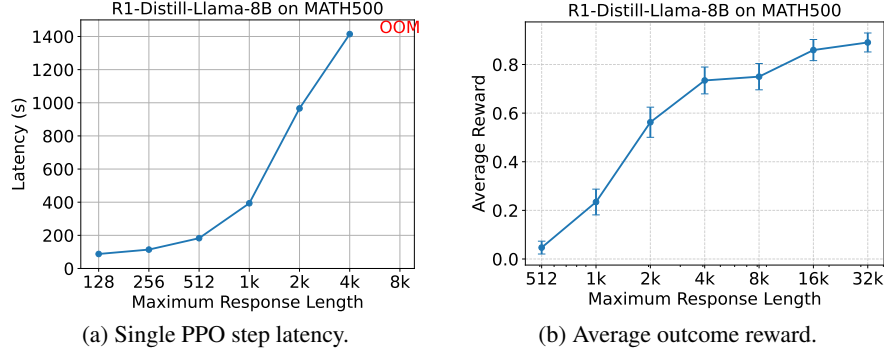


Figure 3: Training long trajectories with outcome rewards is highly inefficient. Both latency and memory usage increase rapidly with trajectory length, and outcome-based supervision requires the entire trajectory to complete before providing feedback. All measurements are obtained using Verl (Sheng et al., 2024) with vLLM (Kwon et al., 2023) on a single A100 node (batch size = 32).

3 METHODOLOGY

3.1 ENHANCING INFERENCE-TIME SCALING WITH MULTI-LEVEL REASONING

Reformulation as MLR. We reconceptualize inference-time scaling by shifting from “single-policy long CoT” to “multi-level reasoning” strategy (Figure 4). Here, the reasoning process is organized hierarchically, capturing both high-level abstractions d and low-level details c . Formally, the overall reasoning chain is represented at two levels: $d = (d^{(1)}, \dots, d^{(M)})$ and $c = (c^{(1)}, \dots, c^{(M)})$, where M denotes the number of reasoning steps, $d^{(m)}$ is the descriptor of step m , and $c^{(m)}$ represents the corresponding detailed content. The autoregressive likelihood can be factorized hierarchically as follows:

$$p_{\theta}^H(d) = \prod_{m=1}^M p_{\theta}^H(d^{(m)} \mid d^{(1:m-1)}, c^{(1:m-1)}), \quad p_{\theta}^L(c) = \prod_{m=1}^M p_{\theta}^L(c^{(m)} \mid d^{(1:m)}, c^{(1:m-1)}) \quad (2)$$

where $c^{(m)}$ denotes a compressed representation of the detailed content $c^{(m)}$. We also experimented with removing the previous descriptors $d^{(1:m-1)}$ from Equation (2), but found that including them improves performance and facilitates training. The inference procedure is summarized in Algorithm 1.

Architecture. Figure 5 illustrates the architecture used to implement our MLR strategy. The model alternates between a high-level policy that produces step descriptors and a low-level policy that generates the corresponding detailed content. The low-level policy is implemented with the base LM,

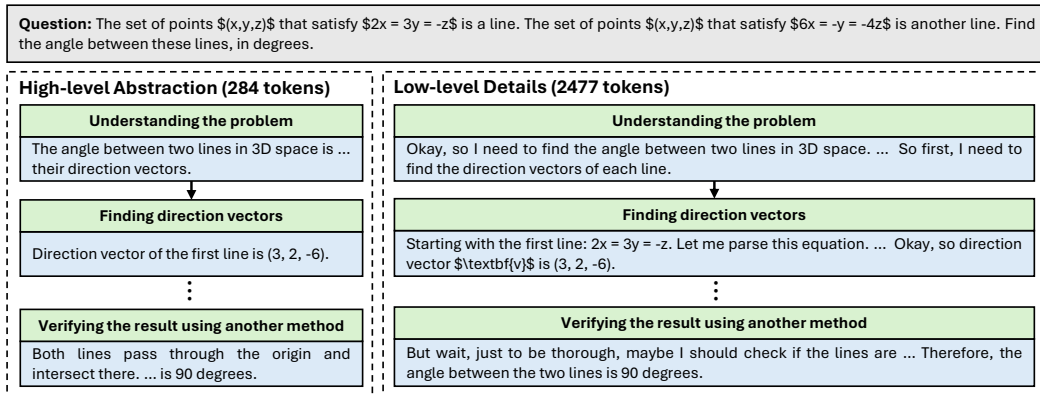


Figure 4: Illustration of MLR. MLR augments single-policy reasoning with an explicit high-level policy which provides intent and structural guidance that narrows the search space, improves credit assignment, and mitigates long-horizon planning failures.

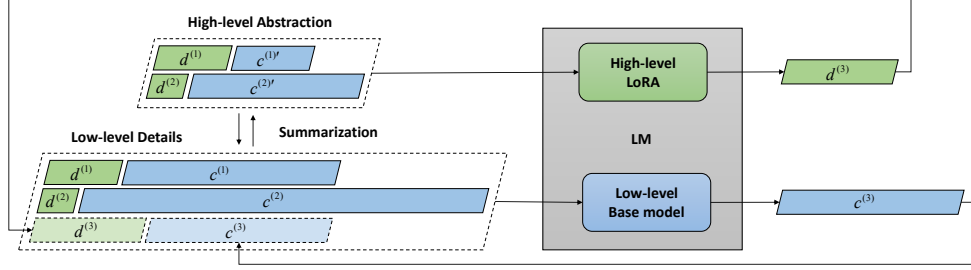


Figure 5: Overview of the proposed architecture. The model alternates between generating high-level descriptors and corresponding low-level content in a structured manner. Additional insights and ablations motivating these design choices are provided in Section C.

which conditions on the sequence of descriptors, prior detailed contents, and the current descriptor to generate the detailed reasoning step. The high-level policy is implemented with a lightweight LoRA module (Hu et al., 2022), which conditions on previous descriptors and their summaries to produce the next descriptor. Since descriptors are much shorter than full reasoning content, this component remains compact and computationally efficient. The design rationale behind this architecture as well as ablation studies are provided in Section C. Additionally, we fine-tune an independent, lightweight LLM for summarization, which is shared across different base models.

3.2 ONLINE STEP-DPO WITH PROCESS-LEVEL PREFERENCES

To train our model effectively, we introduce an online Step-DPO pipeline that iteratively updates the policy through stepwise preference optimization.

Supervised fine-tuning. We collect long CoT examples from DeepSeek-R1 and use powerful non-reasoning models (GPT-4o (OpenAI, 2024), DeepSeek-V3 (Liu et al., 2024)) to decompose them into multiple reasoning steps, each annotated with a step-level descriptor via in-context learning. To construct high-level abstractions, we further compress the detailed content of each step into a concise summary using the same non-reasoning models. The resulting multi-level data consists of aligned step descriptors, detailed contents, and summaries. We then fine-tune the base model on the low-level detailed content using full SFT. Afterward, we freeze the base model and apply LoRA-based fine-tuning on the high-level abstraction data. We also fine-tune an independent, lightweight LLM on the summarization data. During the online Step-DPO procedure, the summarizer remains frozen. A detailed discussion of design choices, including training order, full vs. adapter finetuning, and ablations, is provided in Section C.

Multi-level Step-DPO. Following Lai et al. (2024), we optimize the MLR framework with the following stepwise preference loss:

$$\mathcal{L}_{\text{S-DPO}} := -\mathbb{E}_{(x^{(m)}, y_+^{(m)}, y_-^{(m)}) \sim \mathcal{D}_{\text{pref}}} \frac{1}{M} \sum_{m=1}^M \log \sigma \left[\beta \left(\log \frac{p_{\theta}(y_+^{(m)} | x^{(m)})}{p_{\text{ref}}(y_+^{(m)} | x^{(m)})} - \log \frac{p_{\theta}(y_-^{(m)} | x^{(m)})}{p_{\text{ref}}(y_-^{(m)} | x^{(m)})} \right) \right]. \quad (3)$$

where $(x^{(m)}, y_+^{(m)}, y_-^{(m)})$ denotes the preference data at step m . During optimization, we maintain a low-level policy (the base LM) and a high-level policy (the LoRA adapter). For low-level preference pairs, we disable the LoRA adapter and update only the base LM parameters with $((d^{(1:m)}, c^{(1:m-1)}), c_+^{(m)}, c_-^{(m)})$; for high-level pairs, we freeze the base LM and update only the LoRA parameters with $((d^{(1:m-1)}, c'^{(1:m-1)}), d_+^{(m)}, d_-^{(m)})$.

Multi-level update schemes. A key challenge in jointly optimizing the two policies lies in designing an effective update scheme. We adopt an interleaved strategy: mini-batches of high-level and low-level examples are alternated, allowing the planner and executor to be trained jointly while preserving modularity. We compare this update scheme with cheaper alternatives in Section C.

Multi-round Step-DPO for online optimization. Motivated by the benefits of on-policy data sampling in RL, we adopt an iterative Step-DPO framework for improved optimization. Specifically,

in the t -th iteration, we use the current policies to sample preference pairs to create the preference data $\mathcal{D}_{\text{pref}}^{(t)}$. Then, we use to update the policies for the next iteration as

$$\mathcal{L}_{\text{ms-DPO}} := -\mathbb{E}_{(x^{(m)}, y_+^{(m)}, y_-^{(m)}) \sim \mathcal{D}_{\text{pref}}^{(t)}} \frac{1}{M} \sum_{m=1}^M \log \sigma \left[\beta \left(\log \frac{p_{\theta}^{(t+1)}(y_+^{(m)} | x^{(m)})}{p_{\theta}^{(t)}(y_+^{(m)} | x^{(m)})} - \log \frac{p_{\theta}^{(t+1)}(y_-^{(m)} | x^{(m)})}{p_{\theta}^{(t)}(y_-^{(m)} | x^{(m)})} \right) \right]. \quad (4)$$

The training procedure is summarized in Algorithm 2. More implementation details are provided in Section C.

Process preference modeling. A key component of our online Step-DPO pipeline is the process-level supervision for both the high-level descriptors $d^{(m)}$ and the low-level detailed contents $c^{(m)}$. Consider the full reasoning trajectory after a prefix $x^{(m)}$ as future tokens $\tau_{m+1:M} = (d^{(m+1)}, c^{(m+1)}, \dots, d^{(M)}, c^{(M)})$, generated by a rollout policy p_{roll} . The survival probability of $x^{(m)}$ is

$$g(x^{(m)}) = \mathbb{P}(R = 1 | x^{(m)}) = \mathbb{E}_{\tau_{m+1:M} \sim p_{\text{roll}}(\cdot | x^{(m)})} [R(x^{(m)}, \tau_{m+1:M})], \quad (5)$$

where the terminal reward $R(x^{(m)}, \tau_{m+1:M})$ is 1 if the final answer is correct, and 0 otherwise.

Given an estimate of the survival probability \hat{g} , we construct preference data using a utility defined as the increment in log-survivability:

$$U(y^{(m)}) = \log \tilde{g}(x^{(m)}, y^{(m)}) - \log \tilde{g}(x^{(m)}). \quad (6)$$

where the survivability is clipped as $\tilde{g} = \text{clip}(\hat{g}, \varepsilon, 1 - \varepsilon)$ with $\varepsilon = 0.001$ for numerical stability. Intuitively, $U(y^{(m)})$ quantifies how the selected candidate changes the probability of eventual success relative to the preceding prefix. Then we impose the condition that the utility difference satisfies the following:

$$U(y_+^{(m)}) - U(y_-^{(m)}) = \log \tilde{g}(x^{(m)}, y_+^{(m)}) - \log \tilde{g}(x^{(m)}, y_-^{(m)}) > \delta, \quad (7)$$

where the margin threshold δ ensures the reliability of the preference data.

Twisted Sequential Monte Carlo. A key challenge of the above approach is computational cost: estimating survivability naively requires running the base model multiple times per prefix. To address this, we adopt a strategy based on Twisted Sequential Monte Carlo (TSMC) that provides accurate survivability estimates while remaining computationally efficient. In particular, we use a lightweight rollout model to generate fast continuations and apply importance weighting to correct for the distribution mismatch.

Given a prefix, the k -th particle at step $m - 1$ has state $x_k^{(m-1)}$. We first sample a candidate step $y_k^{(m)} \sim p_{\text{roll}}(\cdot | x_k^{(m-1)})$ and form the updated state $x_k^{(m)} = [x_k^{(m-1)}, y_k^{(m)}]$. Its importance weight is updated as

$$W_k^{(m)} = W_k^{(m-1)} \cdot \tilde{w}_k^{(m)}, \quad (8)$$

with incremental weight

$$\tilde{w}_k^{(m)} = G_m(x_k^{(m)}) \cdot \frac{p_{\theta}(x_k^{(m)} | x_k^{(m-1)})}{p_{\text{roll}}(x_k^{(m)} | x_k^{(m-1)})} \cdot \frac{\phi_m(x_k^{(m)})}{\phi_{m-1}(x_k^{(m-1)})}. \quad (9)$$

where $W_k^{(m)}$ is the m -th step importance weight with $W_k^{(0)} = 1$, the potential function G_m is defined as $G_m(x_k^{(m)}) = 1$ for $m < M$ and $G_M(x_k^{(M)}) = \mathbf{1}_{\text{correct}}(x_k^{(M)})$, i.e., final answer correctness, p_{θ} denotes the base model, p_{roll} is the rollout policy, and ϕ_m is a learned survivability critic at step m . When p_{roll} is **close** to p_{θ} , the contribution from the survivability critic ϕ_m becomes **negligible**, and in this case, we can simplify $\phi_m \approx 1$, leading to $\tilde{w}_k^{(m)} \approx G_m(x_k^{(m)})$. Finally, the survivability estimate is given by:

$$\hat{g}_K(x^{(m)}) = \frac{1}{K} \sum_{k=1}^K W_k^{(M)} = \frac{1}{K} \sum_{k=1}^K W_k^{(m)} \prod_{j=m+1}^{M_k} \tilde{w}_k^{(j)} \approx W^{(m)} \cdot \frac{1}{K} \sum_{k=1}^K \mathbf{1}_{\text{correct}}(x_k^{(M_k)}). \quad (10)$$

where K represents the number of particles, and $W^{(m)}$ is a shared term depending only on the prefix $x^{(m)}$. Specifically, we fine-tune a small LM on the same low-level SFT data and use it as the

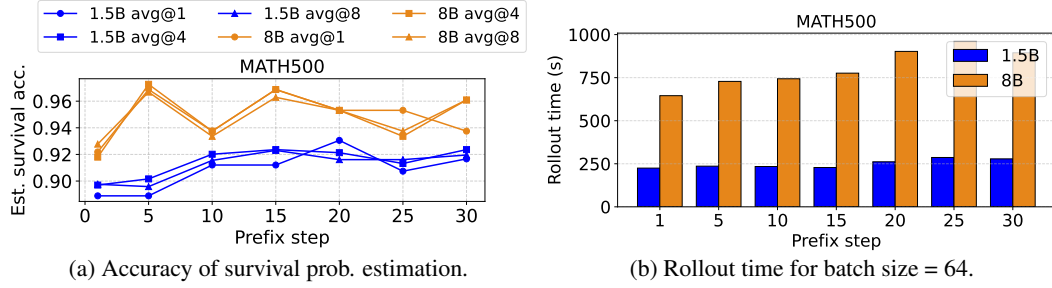


Figure 6: Monte Carlo rollout analysis. (a) The accuracies of R1-Distill-LLaMA-8B and R1-Distill-Qwen-1.5B are highly comparable. (b) Runtime grows with model size, with R1-Distill-LLaMA-8B substantially slower than R1-Distill-Qwen-1.5B. Implementation details are provided in Section C.

rollout policy p_{roll} . This choice is justified for three reasons: (i) the small model shares a *similar* distribution with the base model because it is trained on the same data; (ii) we only need to capture the correct *relative tendency*; and (iii) rollout with a smaller model is significantly more *cost-effective*. In Figure 6, the small model achieves comparable estimation accuracy while being much faster. Additionally, we split the RL problems into easy and hard categories. For *hard* problems, where the accuracy of the small model is significantly lower than that of the base model, we revert to using the *base model* as the rollout policy. In practice, we use the base model for AIME24 and GPQA.

4 EXPERIMENTS

Dataset We evaluate our approach on math (MATH500 (Hendrycks et al., 2021), AIME24 (MAA, 2024)), science (GPQA-diamond (Rein et al., 2023)), and logical reasoning (BoardGameQA-hard (Kazemi et al., 2023)). Detailed dataset statistics are provided in Section B. For training, we construct a multi-level dataset and divide it into two parts: SFT data and online preference data for Step-DPO. The SFT set contains about 80K examples produced using the multi-level decomposition procedure described in Section 3.2. In addition, we reserve 10K prompts for Step-DPO training. Details of dataset construction are provided in Section C.

Implementation details We fine-tune three base models, Qwen-2.5-1.5B (Yang et al., 2024a), Qwen-2.5-MATH-7B (Yang et al., 2024b) and LLaMA-3.1-8B (Grattafiori et al., 2024), on the low-level data with full parameter fine-tuning. The resulting models are frozen, and we apply LoRA fine-tuning on the high-level policy. We also fully fine-tune a Qwen2.5-0.5B-Instruct model for summarization, which is frozen after SFT and shared across all base models (see Section C for more details). Our online Step-DPO pipeline is implemented with the TRL framework. In each training round, we sample a batch of approximately 3K prompts. For each prompt, we randomly select 4 reasoning steps and generate $M = 2$ candidate continuations per step. These candidates are scored using the utility (Equation (6)). In experiments, we use Qwen-2.5-1.5B SFT on the low-level data as the rollout policy, with $K = 4$ sampled rollouts per prefix (see Section 4 for parameter studies). The fast rollout model is frozen after SFT and is shared across base models. From each prefix, we form one preference pair, weighted by the utility margin $\delta = 0.4$. Each update uses mini-batches of size 32 for $E = 4$ epochs, and applies the standard Step-DPO objective with $\beta = 0.1$. Generated continuations are capped at a maximum length of 8,192 tokens. More implementation details, including ablation settings and hyperparameters, are provided in Section C.

Baselines We compare our method with the following baselines: the base model, the instruction fine-tuned model, RL applied directly to the base model (SimpleRL (Zeng et al., 2025)), distillation

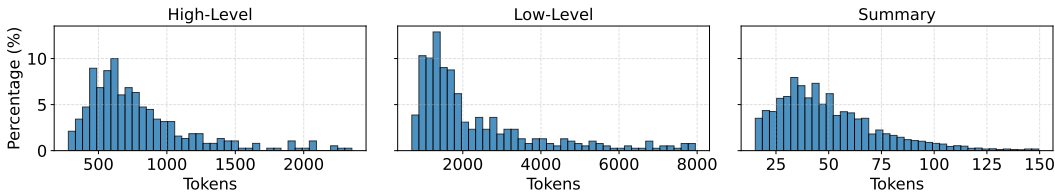


Figure 7: Length distributions for high-level trajectories, low-level trajectories, and distilled summaries in our SFT dataset. Trajectories longer than 8k tokens are truncated.

Table 1: Overall performance comparison across different benchmarks. The best performance for each task using the same base model is in bold. Asterisks (*) denote available results obtained from the official reports.

Method	Math		Science		Logic	Avg. Pass@1
	MATH500 Pass@1	AIME24 Pass@1	GPQA (Diamond) Pass@1	BoardGameQA (Hard) Pass@1		
Qwen-2.5-1.5B						
Base	29.6 ± 0.6	0.0 ± 0.0	0.0 ± 0.0	21.2 ± 1.1	35.0 ± 1.8	21.5
Instruct	54.6 ± 0.4	3.3 ± 1.5	10.0 ± 0.3	25.2 ± 1.4	41.0 ± 1.0	31.0
SimpleRL	59.0*	6.7*	4.2*	—	—	—
DeepSeek-R1-Distill	83.9*	28.9*	43.3 ± 0.4	33.8*	40.0 ± 2.2	47.7
Plan-and-Solve	60.8 ± 1.0	2.0 ± 1.4	6.2 ± 0.4	20.1 ± 1.6	44.6 ± 1.6	31.9
SFT + DPO	76.5 ± 1.3	12.0 ± 1.9	21.6 ± 0.7	27.6 ± 1.6	51.8 ± 1.7	42.0
SFT + Step-DPO	81.4 ± 1.2	24.0 ± 2.0	36.5 ± 0.7	29.0 ± 1.7	56.2 ± 1.7	47.8
SFT + GRPO	82.1 ± 1.2	25.2 ± 2.2	36.0 ± 0.8	30.2 ± 1.5	56.0 ± 1.6	48.4
MLR (SFT only)	62.0 ± 1.2	8.9 ± 1.4	13.3 ± 0.4	26.0 ± 2.0	46.4 ± 1.5	35.8
MLR	86.1 ± 1.0	31.2 ± 1.0	47.4 ± 0.4	37.6 ± 1.9	62.0 ± 1.7	54.2
Qwen-2.5-MATH-7B						
Base	52.0 ± 0.5	2.0 ± 1.0	5.0 ± 0.3	20.5 ± 1.1	33.0 ± 1.6	26.9
Instruct	82.1 ± 0.4	16.7 ± 1.8	34.0 ± 0.4	27.8 ± 1.3	44.5 ± 1.4	42.8
SimpleRL	80.2*	40.0*	24.0*	—	—	—
DeepSeek-R1-Distill	92.8*	55.5*	78.0 ± 0.4	49.1*	42.4 ± 1.4	60.0
Plan-and-Solve	85.6 ± 0.9	18.2 ± 1.7	34.9 ± 0.5	28.4 ± 1.6	52.1 ± 1.5	46.1
SFT + DPO	87.4 ± 1.0	36.0 ± 1.8	53.0 ± 0.5	36.0 ± 1.6	54.5 ± 1.5	53.4
SFT + Step-DPO	88.5 ± 0.9	48.5 ± 1.9	70.5 ± 0.5	48.0 ± 1.7	56.0 ± 1.6	60.3
SFT + GRPO	89.7 ± 1.0	46.5 ± 1.9	66.2 ± 0.5	46.0 ± 1.8	57.5 ± 1.6	59.9
MLR (SFT only)	86.3 ± 1.0	22.4 ± 1.9	40.5 ± 0.5	34.6 ± 1.7	54.8 ± 1.6	49.5
MLR	94.1 ± 0.9	58.8 ± 1.8	80.5 ± 0.4	51.2 ± 1.5	60.5 ± 1.6	66.2
Llama-3.1-8B						
Base	13.6 ± 0.4	0.0 ± 0.0	0.0 ± 0.0	1.5 ± 1.0	2.0 ± 1.1	4.3
Instruct	51.9 ± 0.2	6.7 ± 1.8	13.3 ± 0.2	22.7 ± 0.6	40.0 ± 1.2	30.3
SimpleRL	23.0*	0.0*	0.2*	—	—	—
DeepSeek-R1-Distill	89.1*	50.4*	70.0 ± 0.4	49.0*	46.0 ± 3.8	58.6
Plan-and-Solve	62.4 ± 1.1	12.3 ± 1.8	24.1 ± 0.4	31.0 ± 1.6	47.2 ± 1.7	38.2
SFT + DPO	74.1 ± 1.5	32.4 ± 1.8	52.0 ± 0.6	44.0 ± 1.7	56.0 ± 1.7	51.6
SFT + Step-DPO	82.4 ± 1.3	42.6 ± 2.0	61.2 ± 0.5	49.2 ± 1.5	62.1 ± 1.4	59.1
SFT + GRPO	86.5 ± 1.4	42.0 ± 2.0	61.0 ± 0.5	47.0 ± 1.6	64.5 ± 1.5	60.0
MLR (SFT only)	63.8 ± 1.2	20.2 ± 2.0	36.7 ± 0.4	36.2 ± 1.8	48.5 ± 1.8	42.2
MLR	91.5 ± 1.3	53.2 ± 2.0	73.3 ± 0.4	52.8 ± 1.5	67.0 ± 1.4	66.1

using vanilla long CoTs (R1-Distill (Guo et al., 2025)), RL applied to SFT model (DPO (Rafailov et al., 2023), Step-DPO (Lai et al., 2024), GRPO (Shao et al., 2024)) and Plan-and-Solve (Wang et al., 2023a). All baselines that we train ourselves (DPO, Step-DPO, GRPO, Plan-and-Solve) use exactly the same data (see Section C for implementation details). Results for external baselines (Instruct, SimpleRL, R1-Distill) are included as strong reference points. During evaluation, we use greedy decoding for both Base and Instruct to produce more coherent and consistent CoTs. For all other baselines and our method, we follow the setup in Guo et al. (2025), using sampling-based decoding with a temperature of 0.6 and a top- p value of 0.95 to generate 8 responses per prompt to reduce variance and repetition. Performance is measured using pass@1. For AIME24, we also report consensus accuracy over 32 samples, denoted as cons@32.

Empirical results We first present representative model outputs in Section D, with additional error analysis in Section D. Table 1 reports overall performance, and Figure 9 illustrates how MLR evolves across training stages. We compare instruction fine-tuning, process-reward and outcome-reward

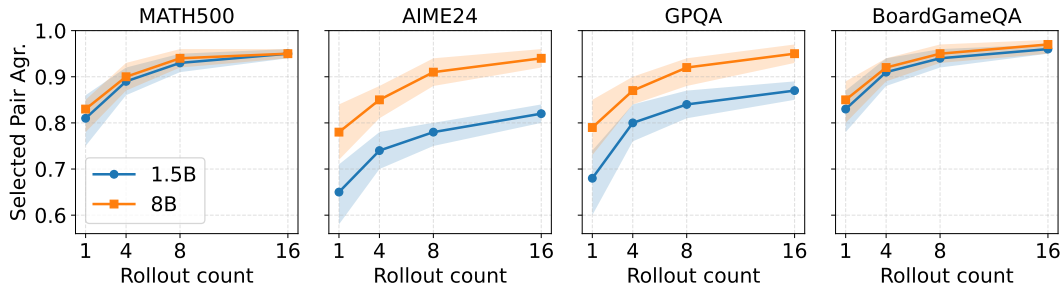


Figure 8: Agreement of selected preference pairs versus rollout policy on (a) MATH500, (b) AIME24, (c) GPQA, and (d) BoardGameQA. For each K , we evaluate the preference pairs selected by the rollout policy and report the fraction whose preference direction matches that of the 8B base model using 16 rollouts. In practice, we use the base model as the rollout policy for AIME24 and GPQA.

Table 2: Ablation results using Qwen-2.5-1.5B. For the high-level SFT ablation, all methods use the same low-level SFT. For the hierarchical-level ablation, all variants share the same trained model. Our approach is highlighted in bold.

Method	MATH500 Pass@1	AIME24 Pass@1	Cons@32	Avg. Pass@1
Ablation of high-level SFT strategies				
SFT (low) + LoRA (high)	62.0 \pm 1.2	8.9 \pm 1.4	13.3 \pm 0.4	35.5
Base + LoRA (high)	56.4 \pm 1.5	4.1 \pm 1.1	9.2 \pm 0.7	30.3
SFT (high)	59.8 \pm 1.3	6.5 \pm 1.2	11.0 \pm 0.5	33.2
Ablation of hierarchical levels				
High-level + Low-level	86.1 \pm 1.0	31.2 \pm 1.0	47.4 \pm 0.4	58.7
High-level only	80.0 \pm 1.3	18.4 \pm 2.0	30.5 \pm 0.8	49.2
Low-level only	84.2 \pm 1.1	27.1 \pm 1.8	41.0 \pm 0.6	55.7

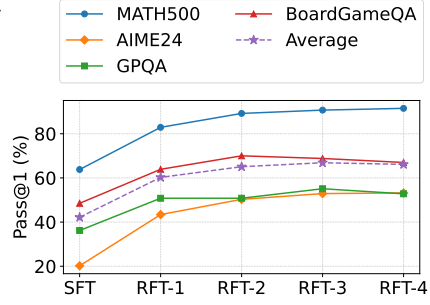


Figure 9: Performance of MLR over different training stages.

Table 3: Ablation results for the core components of MLR using LLaMA-3.1-8B.

Method	Math		Science		Logic	Avg. Pass@1
	MATH500 Pass@1	AIME24 Pass@1 Cons@32	GPQA (Diamond) Pass@1	BoardGameQA (Hard) Pass@1		
Ours	91.5 \pm 1.3	53.2 \pm 2.0	73.3 \pm 0.4	52.8 \pm 1.5	67.0 \pm 1.4	66.1
DPO-only	78.2 \pm 1.4	38.1 \pm 1.9	57.0 \pm 0.5	46.0 \pm 1.6	59.0 \pm 1.6	55.3
Low-level policy + Step-DPO	82.4 \pm 1.3	42.6 \pm 2.0	61.2 \pm 0.5	49.2 \pm 1.5	62.1 \pm 1.4	59.1
Low-level policy + DPO	74.1 \pm 1.5	32.4 \pm 1.8	52.0 \pm 0.6	44.0 \pm 1.7	56.0 \pm 1.7	51.6
SFT-only	63.8 \pm 1.2	20.2 \pm 2.0	36.7 \pm 0.4	36.2 \pm 1.8	48.5 \pm 1.8	42.2

RL, distillation with vanilla long CoTs, Plan-and-Solve, and MLR. Across all benchmarks, MLR consistently outperforms all baselines, with its structured design enabling more effective reasoning on complex, long-horizon tasks. In addition, the online step-DPO procedure yields substantial gains over the SFT model. Finally, we report average response lengths across benchmarks: high-level trajectories are approximately 10–20% the length of low-level ones (Figure 11).

Parameter studies We study the effect of varying the rollout count K . Utility estimates produced by the 1.5B rollout policy with K rollouts are compared against reference utilities from the 8B model using 16 rollouts (Figure 15). As expected, increasing K reduces estimator variance, though at the cost of higher computation. To mitigate this overhead, we introduce a margin threshold δ when selecting preference pairs. We further measure the agreement of the selected preference pairs as a function of K , defined as the fraction whose preference direction agrees with the base model using 16 rollouts (Figure 8). Finally, we report model performance across training stages under different values of K (Figure 12). The results show that our chosen setting attains comparable final performance while substantially reducing computational cost. Implementation details are provided in Section D.

Ablation studies We conduct a series of ablation studies to evaluate the contributions of key components in MLR. We compare five configurations: (i) the full method, (ii) only applying DPO, (iii) using only the low-level policy with Step-DPO or (iv) DPO, and (v) training with SFT only. Table 3 summarizes the results, which show that both multi-level modeling and step-level preferences are essential. Figure 10 further illustrates this trend: our method achieves higher preference accuracy throughout training. We additionally ablate the high-level SFT component (Table 2), evaluating two alternatives: (i) applying LoRA to the original base model and (ii) full-parameter SFT. A detailed discussion and implementation details are provided in Section C. We also ablate the hierarchical structure (Table 2), comparing (i) high-level-only and (ii) low-level-only variants. Further analysis and implementation details appear in Section D. Across all ablations, our full strategy yields the strongest performance.

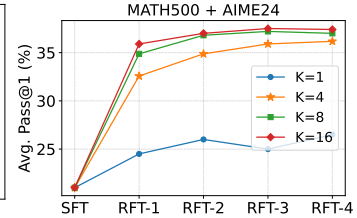
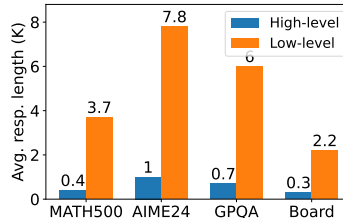
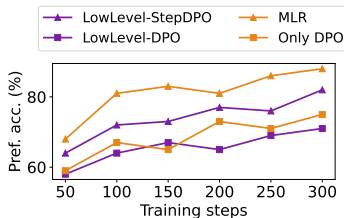


Figure 10: Preference accuracy.

Figure 11: Avg. resp. length.

Figure 12: Effect of K in MLR.

5 RELATED WORK

Reasoning models (OpenAI, 2024; Qwen-Team, 2024; Guo et al., 2025) represent a transformative advancement in the evolution of LMs, sparking substantial interest in replicating their strong performance. Several subsequent works (HuggingFace, 2025; OpenThoughts-Team, 2025; Bespoke-Labs, 2025; Muennighoff et al., 2025) have primarily relied on supervised fine-tuning (SFT). While this approach has shown promising results, pure SFT is generally less efficient in utilizing training signals compared to reinforcement learning (RL), as it passively imitates given demonstrations without exploration or fine-grained credit assignment.

Recent RL-based works introduce improvements along three main dimensions:

1) **RL Algorithms.** RL methods for LLM reasoning mainly fall into PPO-, GRPO-, and REINFORCE-based families. PPO variants (Yuan et al., 2025; Yue et al., 2025) improve value estimation and stability; GRPO methods (Shao et al., 2024; Yu et al., 2025; Liu et al., 2025) remove the critic and refine sampling, normalization, and token-level gradients; REINFORCE variants (Hu, 2025; Kimi-Team et al., 2025) reduce variance through KL penalties and centralized rewards. Despite these advances, all rely largely on sparse outcome rewards, which provide weak credit assignment for long reasoning trajectories.

2) **Reward Design.** Most existing approaches define rewards based on outcome accuracy, format compliance, and length constraints (Zhang et al., 2025a). Process rewards remain largely unexplored in current pipelines.

3) **Data Sampling Strategies.** Curriculum learning (Hu et al., 2025; Zhang et al., 2025b) gradually increases task difficulty during training. Rejection sampling (Wen et al., 2025; Yu et al., 2025) is commonly used to filter low-quality samples and stabilize optimization.

However, outcome-reward RL is inherently inefficient and weakly informative for long trajectories. Recent work (Lightman et al., 2023; Wang et al., 2023b) therefore explores Process Reward Models (PRM), but they struggle in practice (Guo et al., 2025): 1) explicitly defining fine-grained reasoning steps is difficult; 2) reliably verifying the correctness of each intermediate step is non-trivial; 3) training a separate reward model introduces the risk of reward hacking and requires additional training.

On the other hand, Direct Preference Optimization (DPO) (Rafailov et al., 2023) also faces limitations: it relies on offline data and trajectory-level preferences. Step-DPO (Lai et al., 2024) mitigates this by constructing curated step-wise preference data, while DPO with AI feedback (Cui et al., 2023; Guo et al., 2024a) enables online updates. However, these approaches are not well suited for long CoTs, as they rely on strong teacher models (e.g., GPT-4) to provide step-level preferences, both costly and unreliable on harder tasks. To address this, we introduce a scalable TSMC-based approach to provide stepwise preferences. Unlike naive tree-search methods (Wang et al., 2023b), which are prohibitively expensive on long trajectories, our approach remains efficient and stable for long-horizon supervision.

Finally, using a single policy for long-horizon reasoning introduces additional limitations such as plan failures. MLR instead adopts a multi-level strategy, differing from existing planning methods (Huang et al., 2022; Xu et al., 2023; Wang et al., 2023a) that generate a full plan upfront and assume all subtasks succeed as written. Such fixed plans propagate early errors. In contrast, MLR learns a planner that adapts its plans based on execution feedback, enabling revisions and yielding more robust long-horizon reasoning.

6 CONCLUSION

We presented a novel multi-level reasoning (MLR) framework that enhances inference-time scaling by structuring the reasoning process into interleaved high-level abstractions and low-level details. This decomposition supports efficient multi-policy modeling and dynamic plan adaptation, addressing critical challenges faced by single-policy long-CoT approaches. By sidestepping the limitations of prior outcome supervision methods, MLR provides a scalable and robust pathway for training reasoning-focused language models. Extensive experiments demonstrate consistent performance gains across math, science, and logical reasoning tasks, highlighting MLR’s promise as a general-purpose reasoning framework.

LIMITATIONS

Our method requires maintaining two separate policies and performing additional steps for process supervision, which increases training complexity. To keep costs practical, we freeze the base model for the high-level policy, decouple supervision estimation from trajectory generation, and alternate policy updates. These strategies help manage training overhead; however, on resource-constrained devices further optimizations, such as quantization, activation checkpointing, or memory-efficient attention, may still be necessary. The approach also introduces extra hyperparameters, though most can be assigned reasonable default values that transfer well across tasks. Our experiments indicate that performance is robust to moderate variations in these settings, reducing the need for extensive hyperparameter tuning.

ETHICS STATEMENT

This work does not involve human subjects, personal data, or sensitive demographic information. All datasets are publicly available and used under their respective licenses. Our method aims to improve the efficiency of large language models, which can promote accessibility and sustainability. We acknowledge that LLMs may be misused for generating harmful or biased content, but our work does not specifically target such applications. No conflicts of interest or ethical concerns are associated with this research.

REPRODUCIBILITY STATEMENT

We have made significant efforts to ensure the reproducibility of our work. The main paper and appendix provide detailed descriptions of our model architecture, training procedures, and evaluation settings. All datasets used are publicly available, and we include a complete description of data processing steps in the supplementary materials. Pseudocode and complexity analysis are provided in the paper and appendix to clarify algorithmic details.

REFERENCES

- Bespoke-Labs. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation. <https://www.bespokelabs.ai/blog/bespoke-stratos-the-unreasonable-effectiveness-of-reasoning-distillation>, 2025. Accessed: 2025-01-22.
- Mark Briers, Arnaud Doucet, and Simon Maskell. Smoothing algorithms for state-space models. *Annals of the Institute of Statistical Mathematics*, 62:61–89, 2010.
- Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377*, 2023.
- Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.
- Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.
- Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of artificial intelligence research*, 13:227–303, 2000.
- Arnaud Doucet, Nando De Freitas, Neil James Gordon, et al. *Sequential Monte Carlo methods in practice*, volume 1. Springer, 2001.
- Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, et al. Direct language model alignment from online ai feedback. *arXiv preprint arXiv:2402.04792*, 2024a.
- Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024b.
- Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.
- Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, and Heung-Yeung Shum Xiangyu Zhang. Open-reasoner-zero: An open source approach to scaling reinforcement learning on the base model. <https://github.com/Open-Reasoner-Zero/Open-Reasoner-Zero>, 2025.
- Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pp. 9118–9147. PMLR, 2022.
- HuggingFace. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL <https://github.com/huggingface/open-r1>.
- Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.
- Mehran Kazemi, Quan Yuan, Deepti Bhatia, Najoung Kim, Xin Xu, Vaiva Imbrasaitė, and Deepak Ramachandran. Boardgameqa: A dataset for natural language reasoning with contradictory information. *Advances in Neural Information Processing Systems*, 36:39052–39074, 2023.
- Kimi-Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao, Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, and Zonghan Yang. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL <https://arxiv.org/abs/2501.12599>.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

- Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.
- Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding rl-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.
- MAA. American invitational mathematics examination - aime. In *American Invitational Mathematics Examination - AIME 2024*, February 2024. URL <https://maa.org/math-competitions/american-invitational-mathematics-examination-aime>.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL <https://arxiv.org/abs/2501.19393>.
- Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.
- OpenAI. Hello gpt-4o. <https://openai.com/index/hello-gpt-4o/>, 2024. Accessed: 2025-05-15.
- OpenAI. Learning to reason with language models, 2024. URL <https://openai.com/index/learning-to-reason-with-llms/>. Accessed: 2025-05-15.
- OpenThoughts-Team. Open Thoughts. <https://open-thoughts.ai>, January 2025.
- Qwen-Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024. URL <https://qwenlm.github.io/blog/qwq-32b-preview/>.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.
- Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*, 2023a.
- Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023b.

- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, et al. Light-rl: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025.
- Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. Rewoo: Decoupling reasoning from observations for efficient augmented language models. *arXiv preprint arXiv:2305.18323*, 2023.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.
- An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024b. URL <https://arxiv.org/abs/2409.12122>.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.
- Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What’s behind ppo’s collapse in long-cot? value optimization holds the secret. *arXiv preprint arXiv:2503.01491*, 2025.
- Yu Yue, Yufeng Yuan, Qiyang Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiase Chen, Chengyi Wang, Tiantian Fan, Zhengyin Du, Xiangpeng Wei, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks, 2025. URL <https://arxiv.org/abs/2504.05118>.
- Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplertl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.
- Chong Zhang, Yue Deng, Xiang Lin, Bin Wang, Dianwen Ng, Hai Ye, Xingxuan Li, Yao Xiao, Zhanfeng Mo, Qi Zhang, et al. 100 days after deepseek-rl: A survey on replication studies and more directions for reasoning language models. *arXiv preprint arXiv:2505.00551*, 2025a.
- Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, Yinghan Cui, Chao Wang, Junyi Peng, Shimiao Jiang, Shiqi Kuang, Shouyu Yin, Chaohang Wen, Haotian Zhang, Bin Chen, and Bing Yu. Srpo: A cross-domain implementation of large-scale reinforcement learning on llm, 2025b. URL <https://arxiv.org/abs/2504.14286>.

A LLM USAGE

We used Large Language Models (LLMs) only as assistive tools for grammar refinement, readability improvements, and LaTeX formatting. They were not involved in generating research ideas, designing methods, conducting experiments, or analyzing results. All technical content and conclusions are entirely the work of the authors.

B DATASET STATISTICS

In this section, we provide statistics for all benchmark datasets used in our study. We consider MATH500 (Hendrycks et al., 2021), AIME24 (MAA, 2024) for math reasoning, GPQA-diamond (Rein et al., 2023) for science reasoning, and BoardGameQA-hard (Kazemi et al., 2023) for logical reasoning. For MATH, there are 7,500 training samples and 5,000 test samples. MATH500 is a subset of 500 representative test samples extracted by Lightman et al. (2023), with the remaining test samples added to the training set.

The AIME dataset is based on the American Invitational Mathematics Examination, a high-level math contest administered by the Mathematical Association of America (MAA) for high-achieving middle and high school students. We use the data³ collected from 1983 to 2024, where each year contains 15 questions prior to 2000 and 30 questions thereafter. The 2024 dataset is used as the test set, while data from all previous years serve as the training set.

GPQA is a multiple-choice, Q&A dataset of very hard questions written and validated by experts in biology, physics, and chemistry. It includes three subsets: main (448 questions), diamond (198 questions), and extended (546 questions). We use the diamond subset as the test set and combine the remaining subsets for training. To prevent data leakage, any questions overlapping with the test set are removed from the training set.

BoardGameQA is a claim verification problem with three types of labels (proved, disproved, unknown), which involves reasoning with contradictory information guided by preferences over rules as board games. The dataset is divided into 15 subsets based on reasoning depth and levels of conflict or distractors, each with separate train, validation, and test splits. We define BoardGameQA-hard as the union of all test sets from five challenging subsets: Main-depth3, DifficultConflict-depth2, HighConflict-depth2, KnowledgeHeavy-depth2, and ManyDistractors-depth2, resulting in a total test set of 500 examples. The remaining data from all subsets are used for training.

C IMPLEMENTATION DETAILS

Dataset construction. As mentioned above, we consider four publicly available reasoning benchmarks: MATH500, AIME24, GPQA, and BoardGameQA. For each benchmark, we use the available training split as seeds and reserve the test splits exclusively for evaluation. We randomly partition each training split into SFT and RL subsets (60% / 40% ratio). Because AIME and GPQA contain very few training questions, we augment their train splits with synthetic problems generated by GPT-4o using the prompts provided in Section D. For each seed problem we sample multiple candidate questions and retain only those whose final answers are **mutually consistent** with DeepSeek-R1, i.e., DeepSeek-R1 solves the problem and produces the same final answer as GPT-4o. This procedure yields SFT sets of 7.2K MATH, 4K AIME, 4K GPQA, and 5K BoardGameQA problems, and RL sets of 4.8K, 1.5K, 1.5K, and 2K problems, respectively. Unless otherwise stated, all methods that we train ourselves (GRPO, Plan-and-Solve, MLR, and all ablations in Table 3) use exactly the same data, ensuring a fair comparison. Results for external baselines (Instruct, SimpleRL, DeepSeek-R1-Distill), marked with an asterisk in Table 1, are copied from their official reports and may rely on different training corpora; we include them as strong reference points.

Specially, in the SFT phase, we generate multiple high-quality trajectories for each problem by sampling four solutions from DeepSeek-R1, yielding approximately 80K filtered trajectories in total. Each accepted trajectory is then decomposed into step-by-step reasoning segments using DeepSeek-V3 via in-context learning, with each step annotated by a step descriptor. Because DeepSeek-V3 has

³<https://www.kaggle.com/datasets/hemishveeraboina/aime-problem-set-1983-2024>

a maximum response length constraint, we pre-screen trajectories to ensure compliance, and process the remaining long trajectories using GPT-4o. To enable multi-level reasoning, we further distill each step into a concise high-level summary, again using DeepSeek-V3 with in-context learning. All prompts used throughout this pipeline are provided in Section D. Figure 7 presents the resulting trajectory length distributions across different reasoning levels and their corresponding summaries.

Summarization. The summarization module in MLR serves to distill essential information from the evolving trajectory, allowing the planner to operate on concise, high-level representations rather than being overwhelmed by unnecessary details. This becomes particularly important when trajectories grow long (e.g., beyond 4k tokens). We implement summarization as an independent component that is shared across different base models. Although we explored reusing either the low-level or high-level policy for this task, we found that doing so interferes with their primary roles. In contrast, training a separate lightweight model for summarization is both simpler and more reliable. Specifically, we use Qwen2.5-0.5B-Instruct, optimized with AdamW using a cosine learning-rate schedule with linear warmup and a peak learning rate of 1×10^{-5} . The prompt template for summarization is provided in Section D, and the summary length distribution is shown in Figure 7. Note that we only apply full-parameter SFT to the summarization model. During the online Step-DPO procedure, this summarizer remains frozen, which we found to be sufficient in practice and contributes to more stable overall training.

Supervised fine-tuning. We first fine-tune the base LM on low-level trajectories and then freeze it, attaching a parameter-efficient LoRA adapter for high-level planning. The intuition behind this design is twofold. Low-level trajectories are long, fine-grained, and cognitively harder, so they benefit from full-parameter capacity. In contrast, high-level trajectories are short and abstract, making them well-suited to lightweight LoRA tuning while avoiding interference with the executor. This training order also reduces covariate shift: the planner is learned on top of the well-trained executor it is intended to guide. From an optimization perspective, LoRA benefits from a stronger backbone (after low-level SFT) and avoids overfitting by learning only a small number of parameters on high-level data. Operationally, the approach is efficient at deployment time because it requires only a single base model plus a small LoRA adapter (less than 2% additional parameters).

Hyperparameters. We train each base model on 80K multi-level examples using AdamW with the same cosine schedule and warmup strategy as Step-DPO. We use a batch size of 256, a peak learning rate of 2×10^{-5} , and truncate sequences to 8,192 tokens. Training is run for 3 epochs. Unless otherwise noted, all models are trained with the AdamW optimizer using a cosine learning-rate schedule with linear warmup (5% of total steps). For the base LM (low-level policy), we use a peak learning rate of 2×10^{-5} , while the high-level LoRA module ($r = 16$, $\alpha = 32$, `target_modules=[q_proj, k_proj, v_proj, o_proj]`, no bias) is trained with a higher rate of 1×10^{-4} (dropout=0.1) to allow faster adaptation. We additionally verified that adding MLP projections (`up_proj`, `down_proj`, `gate_proj`) yields only marginal gains while substantially increasing the number of trainable parameters.

Ablation study. To further validate our design choices, we compare against two alternative training strategies for the high-level policy, while keeping the low-level training unchanged. This is important because low-level modeling requires full-parameter updates due to its longer and more complex reasoning trajectories; LoRA is insufficient for this component. We consider:

(i) LoRA on the original (non-SFT) base model: We directly apply LoRA tuning on the unfine-tuned Qwen-2.5-1.5B base model using only high-level trajectories.

- Base: Qwen-2.5-1.5B
- LoRA: $r = 16$, $\alpha = 32$, `target_modules=[q_proj, k_proj, v_proj, o_proj]`, no bias.
- Optimization: AdamW with a cosine learning-rate schedule and linear warmup, a peak learning rate of 1×10^{-4} and a LoRA dropout of 0.1.

(ii) Full-parameter SFT on high-level trajectories: We train a separate base model using full SFT on only high-level trajectories.

- Base: Qwen-2.5-1.5B

- Optimization: AdamW with a cosine learning-rate schedule and linear warmup, a peak learning rate of 1×10^{-5} .

We evaluate both variants on MATH500 and AIME24. Table 2 summarizes the results. Our default configuration (full SFT on low-level trajectories followed by LoRA tuning on high-level abstractions) achieves the highest accuracy, particularly on the harder AIME tasks that require deeper multi-step planning. We also observe that applying LoRA on top of the SFT-enhanced base model substantially eases optimization and mitigates the overfitting issues that arise when fully fine-tuning a separate base model using only high-level trajectories.

Monte Carlo rollout analysis We analyze Monte Carlo rollout behavior using R1-Distill-LLaMA-8B and R1-Distill-Qwen-1.5B. Hidden CoTs are segmented into steps using $\backslash n \backslash n$. Estimation accuracy measures the fraction of prefixes for which rollouts correctly determine whether the prefix can still lead to a correct final solution.

For each partial trajectory, we assign a ground-truth survival label $y \in \{0, 1\}$ using extensive Monte Carlo lookahead with the base model: $y = 1$ if at least one rollout from the prefix reaches a correct final answer (the prefix is survivable), and $y = 0$ otherwise.

Using the fast rollout model, we draw K continuations from each prefix and compute the estimated survivability

$$\hat{g}_K = \frac{1}{K} \sum_{k=1}^K \mathbf{1}_{\text{correct}}^{(k)}.$$

We then predict a survival label

$$\hat{y}_K = \begin{cases} 1, & \hat{g}_K > 0, \\ 0, & \text{otherwise.} \end{cases}$$

A prediction is correct when $\hat{y}_K = y$. Estimated survival accuracy for a given K is the proportion of prefixes where this prediction matches ground truth.

We present the results in Figure 6. R1-Distill-LLaMA-8B and R1-Distill-Qwen-1.5B exhibit highly similar accuracy across prefix steps. As expected, runtime scales with model size: R1-Distill-LLaMA-8B is substantially slower than R1-Distill-Qwen-1.5B. All measurements are obtained on a single A100 GPU using vLLM.

Online Step-DPO. The reference model for KL regularization in Step-DPO is the corresponding SFT checkpoint. We perform 4 online training rounds, with approximately 3K prompts per round, as described in the main text. For each prompt, we sample $N = 4$ candidate continuations using temperature 0.7 and top- $p = 0.9$, and truncate continuations at 8,192 tokens to match the SFT context length and keep attention computation within our memory budget. Step-wise preference pairs are constructed from these candidates and stored in a replay buffer of size 50K; once the buffer is full, older entries are discarded in FIFO order. We optimize the policy with AdamW (learning rate 1×10^{-5} , weight decay 0.1, $\beta_1 = 0.9$, $\beta_2 = 0.95$), a global batch size of 256 preference pairs, gradient clipping with a maximum norm of 1.0, DPO temperature $\beta = 0.1$, and KL coefficient $\lambda_{\text{KL}} = 0.02$. At each round, we perform one epoch of updates over the current replay buffer. During optimization, we maintain a low-level policy (the base LM) and a high-level policy (the LoRA adapter). For low-level preference pairs, we disable the LoRA adapter and update only the base LM parameters; for high-level pairs, we freeze the base LM and update only the LoRA parameters. Mini-batches of low- and high-level examples are interleaved within each round, so that the executor and planner are optimized jointly while remaining modular. To improve sample efficiency, we apply a dynamic dropout strategy that filters “easy” prefixes, i.e.,

Algorithm 1: Multi-Level Inference

```

1 Inputs: query  $q$ , high-level policy  $\pi_{\theta_H}$ ,
   low-level policy  $\pi_{\theta_L}$ , summarizer  $\pi_{\theta_S}$ ;
2 Hyperparameter: max steps  $M$ ;
3  $m \leftarrow 1$ ;
4 while  $m < M$  do
5    $d^{(m)} \sim \pi_{\theta_H}(d \mid q, d^{(1:m-1)}, c'^{(1:m-1)})$ ;
6    $c^{(m)} \sim \pi_{\theta_L}(c \mid q, d^{(1:m)}, c^{(1:m-1)})$ ;
7    $c'^{(m)} \leftarrow \pi_{\theta_S}(d^{(m)}, c^{(m)})$ ;
8   if StopCriterion( $d^{(m)}, c^{(m)}$ ) then
9     | break;
10   $m \leftarrow m + 1$ ;
11 return ( $d^{(1:m)}, c^{(1:m)}$ );

```

Algorithm 2: Online Step-DPO

```

1 Inputs: Low-level policy  $\pi_{\theta_L}$ , high-level policy  $\pi_{\theta_H}$ ; Reference models  $\pi_{\text{ref}}^L, \pi_{\text{ref}}^H$ ; Fast rollout policy
    $\pi_{\text{roll}}$ ; RL prompts  $\mathcal{D}_{\text{RL}}$ .
2 Hyperparams: rounds  $T$ , prompts per round  $N$ , sample steps per prompt  $M_s$ , rollout count  $K$ ,
   epochs  $E$ .
3 for  $t = 1$  to  $T$  do
4   Sample prompts  $\{q_i\}_{i=1}^N \subset \mathcal{D}_{\text{RL}}$ ;
5   Initialize buffers  $\mathcal{D}_{\text{pref-L}}^{(t)} \leftarrow \emptyset, \mathcal{D}_{\text{pref-H}}^{(t)} \leftarrow \emptyset$ ;
6   foreach  $q$  do
7      $(\text{prefix}_H^{(t)}, \text{prefix}_L^{(t)}) \leftarrow \text{GENERATEPREFIXES}(\pi_{\theta_H}^{(t)}, \pi_{\theta_L}^{(t)}, q)$ ;
8     Randomly select a subset of steps  $\mathcal{M}$  (size  $M_s$ ) for evaluation;
9     foreach  $m \in \mathcal{M}$  do
10       $\mathcal{D}_{\text{pref-L}}^{(t,m)} \leftarrow \text{COLLECTPAIR}(\pi_{\theta_L}^{(t)}, \text{prefix}_L^{(t)}[m])$ ;
11       $\mathcal{D}_{\text{pref-H}}^{(t,m)} \leftarrow \text{COLLECTPAIR}(\pi_{\theta_H}^{(t)}, \text{prefix}_H^{(t)}[m])$ ;
12       $\mathcal{D}_{\text{pref-L}}^{(t)} \leftarrow \mathcal{D}_{\text{pref-L}}^{(t)} \cup \mathcal{D}_{\text{pref-L}}^{(t,m)}$ ;
13       $\mathcal{D}_{\text{pref-H}}^{(t)} \leftarrow \mathcal{D}_{\text{pref-H}}^{(t)} \cup \mathcal{D}_{\text{pref-H}}^{(t,m)}$ ;
14   if  $t > 1$  then
15      $\pi_{\text{ref}}^L \leftarrow \pi_{\theta_L}^{(t-1)}; \pi_{\text{ref}}^H \leftarrow \pi_{\theta_H}^{(t-1)}$ ;
16   for  $e = 1$  to  $E$  do
17      $\text{STEPPDOUPDATE}(\pi_{\theta_L}^{(t)}, \pi_{\text{ref}}^L, \mathcal{D}_{\text{pref-L}}^{(t)})$ ;
18      $\text{STEPPDOUPDATE}(\pi_{\theta_H}^{(t)}, \pi_{\text{ref}}^H, \mathcal{D}_{\text{pref-H}}^{(t)})$ ;
19 return  $\pi_{\theta_L}^{(T)}, \pi_{\theta_H}^{(T)}$ ;

```

prefixes for which all candidates induce the same utility; the dropout rate increases linearly from 0.1 to 0.9 over training. All experiments are conducted on $4 \times$ A100 GPUs (80GB) with `bf16` precision.

Step-DPO update schemes. We compare the proposed update scheme against cheaper alternatives under a matched online training budget (same number of prompts, candidates, and optimization steps). In the *planner-only* variant, we freeze the SFT base LM and apply Step-DPO updates only to the high-level LoRA adapter for all preference pairs, thereby testing whether adapting the planner alone is sufficient once the executor has been trained. In a *round-based* variant, we first run Step-DPO for two rounds updating only the low-level policy (LoRA disabled), and then for two rounds updating only the high-level LoRA (base LM frozen), mirroring a coarse low-then-high schedule in the online phase. Empirically, our joint modular scheme, which interleaves low-level and high-level updates while restricting each preference type to its corresponding module, achieves the best overall performance on MATH500 and AIME24, suggesting that simultaneously refining the executor and planner, while keeping their parameter updates disentangled, is more effective than tuning either component in isolation.

DPO baseline. To isolate the effect of step-wise supervision, we train a standard outcome-level DPO baseline on the same online prompt pool and with the same rollout configuration as Step-DPO. The reference model for KL regularization is the corresponding SFT checkpoint, and we run 4 online training rounds with approximately 3K prompts per round. For each prompt, we sample $N = 4$ candidate continuations using temperature 0.7 and top- $p = 0.9$, truncating each continuation at 8,192 tokens to match the SFT context length. Preference pairs are constructed at the trajectory level: we assign each candidate a scalar utility based on its final solution correctness and form DPO pairs from these outcome-level utilities, ignoring intermediate prefixes. The resulting preference pairs are stored in a replay buffer of size 50K with FIFO eviction, and we perform one epoch of DPO updates over the buffer per round. We optimize a single policy (no hierarchical separation) with AdamW (learning rate 1×10^{-5} , weight decay 0.1, $\beta_1 = 0.9$, $\beta_2 = 0.95$), using a global batch size of 256 preference pairs, gradient clipping with a maximum norm of 1.0, DPO temperature $\beta = 0.1$, and KL coefficient $\lambda_{\text{KL}} = 0.02$. All experiments are conducted on $4 \times$ A100 GPUs (80GB) with `bf16` precision under a matched online training budget to Step-DPO.

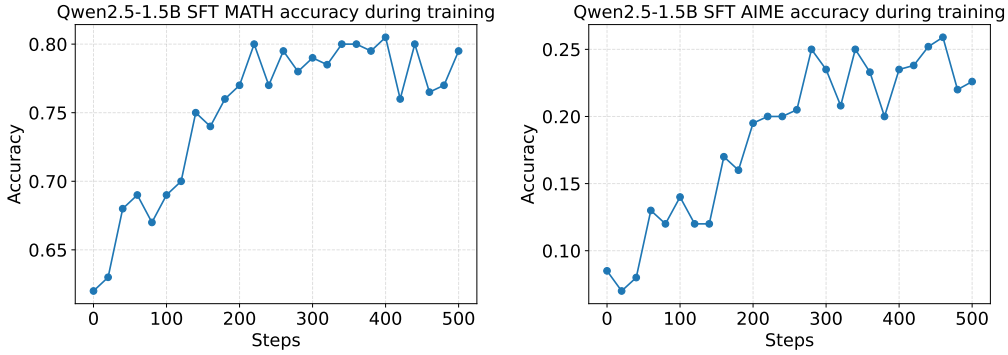


Figure 13: Validation accuracy of Qwen2.5-1.5B SFT during GRPO training. For each question, we sample 8 responses and calculate the overall average accuracy to ensure a stable evaluation.

GRPO baseline. To compare MLR with a standard single-policy preference-optimization method, we train a GRPO baseline on the same prompt pool as Step-DPO. We first construct a *single-policy* SFT checkpoint by fine-tuning Qwen-2.5-1.5B on the processed low-level trajectories in which all step descriptors are removed from both inputs and targets, using the same optimizer, schedule, and token budget as our low-level SFT. Starting from this checkpoint, we apply full-parameter GRPO, keeping a frozen copy of the SFT model as the reference policy. We implement the baseline using the Verl (Sheng et al., 2024) framework and vLLM (Kwon et al., 2023) as the rollout backend. The actor and reference are both initialized from the same SFT checkpoint. For each prompt, we sample groups of $N = 4$ candidate continuations with temperature 0.7 and top- $p = 0.9$, cap the maximum response length at 4,096 tokens to respect GPU memory limits, and assign a rule-based outcome reward of 1 if the final answer is correct and 0 otherwise. We optimize the actor with AdamW (learning rate 5×10^{-7} , weight decay 0.1) under a KL-penalty objective with coefficient $\lambda_{\text{KL}} = 0.02$, using bf16 precision, gradient checkpointing, and FlashAttention (Dao, 2023) on $4 \times$ A100 GPUs (80GB). We train for 4 epochs, using a global batch size of 32 (PPO mini-batch sizes 16, micro-batch sizes 2, respectively), and evaluate every 100 steps on the held-out validation split, selecting the checkpoint with the best validation Pass@1.

We visualize the validation accuracy of Qwen2.5-1.5B SFT during GRPO training in Figure 13. For each question, we sample 8 responses and report the average accuracy to obtain a stable estimate. The evolution of the average response length during GRPO is shown in Figure 14, and the final evaluation results are summarized in Table 1. Compared with our strategy, GRPO is less efficient for long-horizon reasoning: outcome rewards are (i) *sparse*: for long trajectories, a single scalar signal is often insufficient to localize errors; and (ii) *computationally expensive*: generating full rollouts requires substantial memory and compute. When starting from fine-tuned models with long CoTs,

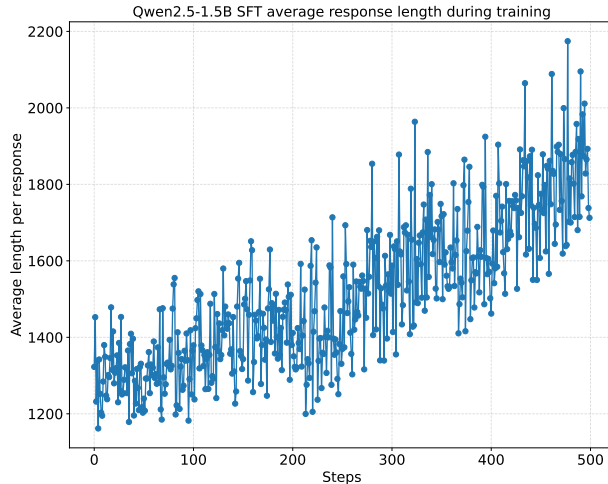


Figure 14: The average response length of Qwen2.5-1.5B SFT on the training set during GRPO.

these costs limit our ability to apply GRPO to larger base models. In contrast, our online Step-DPO procedure is easier to implement and control, and provides a more practical alternative for finetuning long-horizon reasoning policies.

Qwen-2.5-MATH-7B. We repeat the above protocol on a 7B model. We first obtain a *single-policy* SFT checkpoint by fine-tuning Qwen-2.5-MATH-7B on the same processed low-level trajectories (step descriptors removed from inputs and targets). Starting from this checkpoint, we run full-parameter GRPO with a frozen copy of the SFT model as the reference. Training uses FSDP sharding, bfloat16 precision, activation checkpointing, and FlashAttention on 8×A100 (80GB), with a global batch size of 32 implemented as a micro-batch of 1 per GPU and 4 gradient-accumulation steps. Rollouts are generated with vLLM: for each prompt we sample $N=4$ candidates (temperature 0.7, top- $p=0.9$) and cap the maximum response length at 4,096 tokens. We optimize with AdamW (learning rate 5×10^{-7} , weight decay 0.1) under a KL penalty with coefficient $\lambda_{KL}=0.02$, and evaluate every 100 steps, selecting the checkpoint with the best validation Pass@1.

Llama-3.1-8B. We follow the same procedure with Llama-3.1-8B. A *single-policy* SFT checkpoint is first obtained on the same low-level trajectories, after which we apply full-parameter GRPO using a frozen reference initialized from the SFT checkpoint. We train with FSDP, bfloat16, activation checkpointing, and FlashAttention on 8×A100 (80GB), using a global batch size of 32 realized as a micro-batch of 1 per GPU and 4 gradient-accumulation steps. Rollouts use vLLM with $N=4$ candidates per prompt (temperature 0.7, top- $p=0.9$) and a 4,096-token cap. The optimizer, KL objective, evaluation cadence, and model selection criteria are identical to the 7B setting. For additional memory headroom, the frozen reference is sharded; when necessary, we load the reference in 8-bit for forward-only KL without changing any other hyperparameters.

Plan-and-Solve baseline. We compare against Plan-and-Solve (Wang et al., 2023a), which first proposes a concise, global plan and then executes the solution conditioned on that plan. The example prompt is given in Section D. For a fair comparison, we use the same backbone as our method and fine-tune two LoRA heads on top of it: a *planner* (Problem \rightarrow Plan) and an *executor* (Problem + Plan \rightarrow Solution). At inference we follow the standard two-pass Plan-and-Solve pipeline: Pass-1 generates the plan; Pass-2 solves the problem conditioned on that plan.

Training data creation. Using the same training set as our method, we prompt a strong teacher model (DeepSeek-V3.2) to produce corresponding trajectories. We filter trajectories by final-answer correctness and basic format checks. We match the total number of accepted trajectories to our method (80K) to ensure a fair comparison.

Training configuration. Unless otherwise noted, we freeze the backbone and train LoRA adapters with identical hyperparameters for planner and executor.

- Backbone: Qwen-2.5-1.5B. LoRA: $r = 16$, $\alpha = 32$, target_modules=[q_proj, k_proj, v_proj, o_proj], no bias. Optimization: AdamW, cosine decay with 3% warm-up, learning rate 1×10^{-4} .
- Backbone: Qwen-2.5-MATH-7B. Same LoRA configuration. Same optimization configuration except for learning rate 5×10^{-5} .
- Backbone: Llama-3.1-8B. Same LoRA configuration. Same optimization configuration except for learning rate 5×10^{-5} .

Results. Table 1 summarizes performance. Because Plan-and-Solve here is trained only with SFT, we compare it against MLR (SFT-only). Across all three backbones, our method outperforms Plan-and-Solve, with the largest margins on the harder benchmarks (AIME, GPQA). We observe that Plan-and-Solve often implicitly assumes all subtasks succeed as initially planned; errors in early steps can propagate, and the executor may partially deviate from the plan. In contrast, our approach learns a better planner that can adapt its plans based on execution signals, enabling revisions rather than committing to a fixed blueprint. This adaptive coupling between planner and executor yields more stable long-horizon reasoning than prompting a plan upfront and executing it verbatim.

Evaluation. During evaluation, we use greedy decoding for both the base model and the instruction fine-tuned model to produce more coherent and consistent CoTs. For all other baselines and our method, we follow the decoding protocol in Guo et al. (2025), using sampling-based decoding

with a temperature of 0.6 and a top- p value of 0.95 to generate 8 responses per prompt to reduce variance and repetition. For MLR, we employ a single base LM for both levels and switch the high-level LoRA adapter on or off depending on the generation stage (Algorithm 1). Specifically, we enable the high-level LoRA adapter to produce step descriptors (planning), and then disable the adapter to generate the corresponding low-level trajectories conditioned on these descriptors. The maximum generation length for all models is set to 16,384 tokens. Performance is measured using $\text{Pass}@1 = \frac{1}{k} \sum_{i=1}^k p_i$, where p_i denotes the correctness of the i -th response. For AIME24, we also report consensus accuracy over 32 samples, denoted as $\text{cons}@32$.

D ADDITIONAL RESULTS

Examples of MLR outputs. In this section, we present additional results to further demonstrate and analyze the effectiveness of our method. We showcase representative output examples generated by MLR across different datasets (Section D). Each sample consists of a two-level reasoning trajectory, comprising shared reasoning steps annotated with both a step descriptor and corresponding step content. In the high-level trajectory, the step descriptor is generated by the high-level module, while the step content is produced by the compressor, which takes the low-level content as input and outputs a concise abstraction. In the low-level trajectory, the step descriptor is provided by the high-level module, and the step content is directly generated by the low-level base model.

Error analysis. To better understand the strengths and limitations of our framework, we conduct detailed error analysis. To further enhance verification and error localization, we incorporate auxiliary models (OpenAI’s o1 and o1-mini) to assist in identifying potential reasoning flaws. Specifically, we first evaluate whether the auxiliary model can independently solve each task without access to the ground-truth final answer or reference solution. If the auxiliary model successfully produces the correct solution, we then use it to help analyze erroneous trajectories generated by our framework. The error analysis provided by the auxiliary model is subsequently reviewed and confirmed by human evaluators. Through this process, we identify several recurring error patterns: 1) High-level step descriptor errors: redundant branching (multiple step descriptors that pursue the same subtask), unclosed loops (steps are never marked as “complete,” leading to repeated revisitation), dead-end retention (contradicted or unproductive exploratory branches are retained), copy-pasted fallback (guessed answers are repeated verbatim under different step descriptors). 2) Low-level step content errors: logical misapplication (misuse of domain-specific rules or principles), contradiction tolerance (inconsistent constraints are not resolved), repetitive reasoning (redundant inference chains without new contributions), failure to propagate known facts (previously inferred information is ignored in later steps), looping filler (verbose or stalled reasoning with redundant rephrasing).

Parameter studies on rollout count In our online step-DPO, the rollout count K directly affects the quality of the preference pairs. We first examine how K influences the reliability of the utility

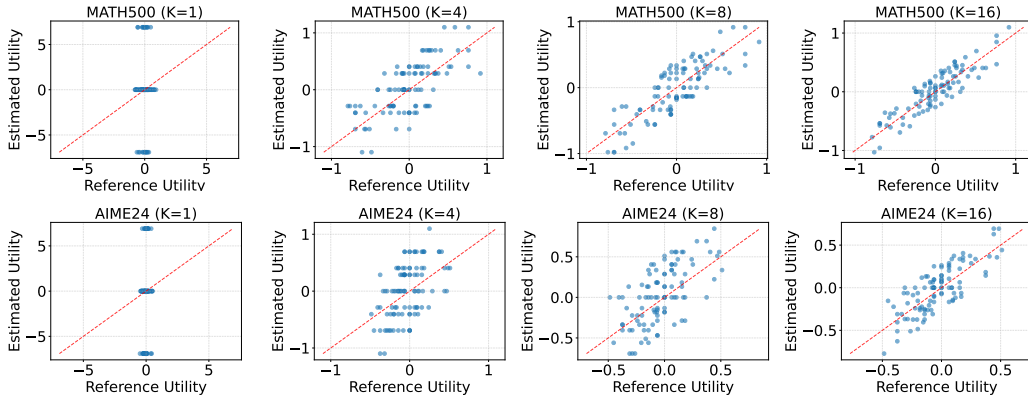


Figure 15: Effect of rollout count on the reliability of utility estimates. We compare utilities estimated by the 1.5B model under K rollouts to reference utilities by the 8B model using 16 rollouts for 100 sampled prefixes from MATH500 and AIME24, respectively.

estimates defined in Equation (6). To do so, we compare utilities estimated by the 1.5B model under various K to reference utilities produced by the 8B model with 16 rollouts, using 100 sampled prefixes from MATH500 and AIME24, respectively (Figure 15). As expected, larger K reduces estimator variance but increases computational cost. To control this overhead, we apply a margin threshold δ when selecting preference pairs which allows us to use smaller K while maintaining reliability of the preference data. Next, we evaluate the agreement of selected preference pairs as a function of K , defined as the fraction of pairs whose preference direction agrees with the base model using 16 rollouts (Figure 8). For each K , we generate 100 preference pairs following Equation (7). We then recompute the reference utilities of both options and check whether the chosen response has higher reference utility than its alternative. Pairs that satisfy this condition are counted as agreed, and we report the average agreement for each K . We consider both the 1.5B model and the 8B base model as rollout policy. In practice, we use the base model as the rollout policy for AIME24 and GPQA. Finally, we study model performance across training stages under different rollout counts (Figure 12). Starting from the same SFT model, we generate the same number of preference pairs for each K and all train for 4 epochs. We report performance on MATH500 and AIME24 throughout training. Overall, our setting achieves comparable final accuracy while significantly reducing computational cost.

Ablations on hierarchical levels To investigate the role of different levels, we conduct an ablation study on the hierarchical structure. We consider two variants: (1) High-level only: the high-level module directly predicts summaries without invoking the low-level module; (2) Low-level only: the low-level module is required to predict both the high-level step descriptions and the detailed reasoning without guidance from the high-level module. The evaluation protocol matches our main setting, and the results are reported in Table 2. Our full method consistently outperforms both variants, especially on the challenging AIME24 dataset. The high-level-only variant underperforms because the planner lacks grounded execution learning, making direct summary prediction unreliable for difficult reasoning tasks. We show an erroneous example in Section D. The low-level-only variant is weaker because the absence of explicit high-level guidance causes the low-level module to drift and accumulate errors as the trajectory grows longer. Overall, these results demonstrate that our two-level design yields better performance on long-horizon reasoning tasks.

Long-horizon reasoning test. To further evaluate our method on long-horizon reasoning, we simulate a multi-question setting by concatenating multiple problems into a single prompt. We

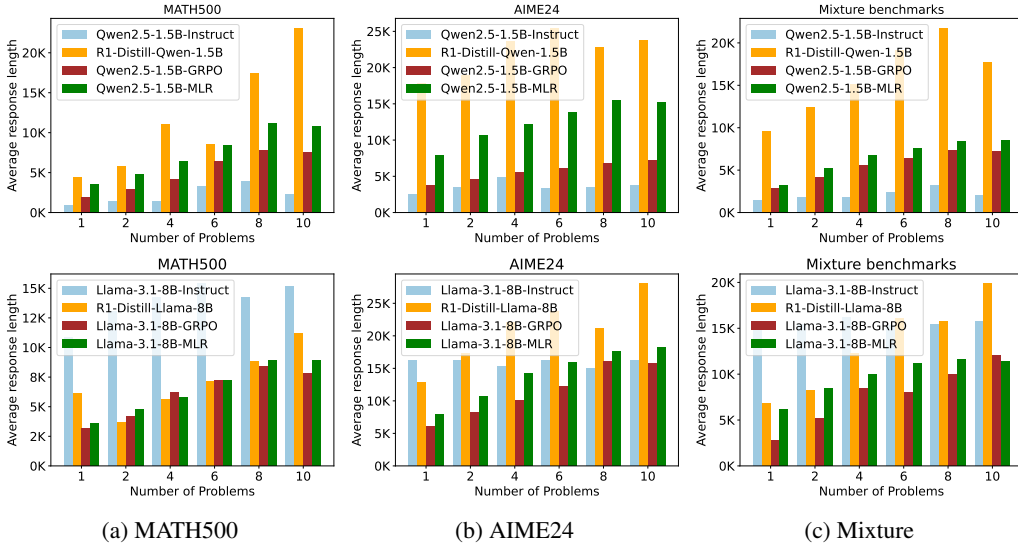


Figure 16: Long-horizon reasoning response length on (a) MATH500, (b) AIME24, and (c) Mixture benchmarks (MATH500, AIME24, GPQA, BoardGameQA). We simulate long-horizon reasoning by concatenating multiple problems in the prompt and report average accuracy. Compared with R1-Distill models, which are prone to getting lost in long trajectories and entering dead loops, MLR benefits from explicit planning and is able to solve all problems more effectively.

consider three configurations: (i) multiple problems from MATH500, (ii) multiple problems from AIME24, and (iii) mixed batches drawn from MATH500, AIME24, GPQA, and BoardGameQA. For each configuration, we randomly select 100 problems from the corresponding benchmarks and treat their questions as a mini-batch. The batch size varies from 1 to 10; for each size, we present all questions in the batch in one prompt and ask the LLM to solve them jointly. To mitigate cases where the model gets stuck on a single problem, we explicitly instruct it to skip a question if it cannot make progress. Prompt template can be found in Section D. For instruction-tuned models, we use greedy decoding. For the other baselines, for every mini-batch we sample 8 responses with temperature 0.6 and top-p = 0.95, using a maximum context length of 32K tokens, to reduce variance and then report accuracy averaged over the 100 problems. Note that all methods are trained only in the single-question setting, so this constitutes a fully out-of-distribution evaluation. Figure 1 shows the results: MLR consistently degrades more slowly than all baselines across datasets and base models, highlighting the benefits of explicit planning for long-horizon reasoning. We further report the corresponding length distributions for different scenarios in Figure 16.

Example of Long-horizon Plan Failures (DeepSeek-R1-Distill-Llama-8B on MATH500)

Prompt:

What is the least positive integer multiple of 30 that can be written with only the digits 0 and 2?

Response:

<think>

Okay, so I need to find the smallest positive integer multiple of 30 that can be written using only the digits 0 and 2. Hmm, let me think about how to approach this.

First, I remember that a multiple of 30 must be divisible by both 3 and 10. So, let's break down what that means.

Divisibility by 10: For a number to be divisible by 10, it must end with a 0. That makes sense because any number ending in 0 is a multiple of 10. So, whatever number I come up with must end with a 0. That means the last digit is fixed as 0.

Divisibility by 3: For a number to be divisible by 3, the sum of its digits must be a multiple of 3. Since the number can only have digits 0 and 2, the sum of the digits will just be the number of 2s multiplied by 2. So, the total sum is 2 times the number of 2s. This sum needs to be divisible by 3.

So, putting that together, I need a number that consists of one or more 2s followed by a 0. The number of 2s should be such that the total sum (which is 2 times the number of 2s) is divisible by 3.

Let me formalize that. Let's say the number has 'n' number of 2s followed by a 0. So, the number would be 222...20, with 'n' 2s and ending with a 0.

The sum of the digits is $2n$, and we need $2n$ to be a multiple of 3. So, $2n \equiv 0 \pmod{3}$. Which means n must be a multiple of 3 because 2 and 3 are coprime (they have no common divisors other than 1). So, n needs to be divisible by 3.

Therefore, the smallest number of 2s needed is 3, right? Because 3 is the smallest positive integer divisible by 3.

So, if $n = 3$, the number would be 2220. Let me check that.

2220 is a multiple of 30? Let's see. 2220 divided by 30 is 74, which is an integer. So, yes, 2220 is a multiple of 30.

But is it the smallest?

Wait, maybe I can find a smaller number with fewer digits? Let me think. The number must end with a 0, so the last digit is fixed. The other digits can be 0 or 2. But if I try to make a smaller number, maybe with fewer digits, but still only 0s and 2s.

Let me consider numbers of different lengths and see if they can satisfy the conditions.

Starting with the smallest possible number: 1 digit. It would have to be 0, but 0 is not positive. So, no good.

2 digits: The number would be 20. Is 20 a multiple of 30? 20 divided by 30 is less than 1, so no. Also, 20 is not a multiple of 30.

3 digits: Let's see. The possible numbers are 200, 220, and 222.

200: 200 divided by 30 is approximately 6.666, not an integer. 220 divided by 30 is about 7.333, not integer. 222 divided by 30 is 7.4, which isn't integer either. So, none of these are multiples of 30.

4 digits: Let's list possible numbers. They end with 0, so the last digit is 0, and the first three digits are either 0 or 2.

Possible numbers: 2000, 2200, 2220, 2222, 2020, 2022, 2202, 2220, 2222, etc. Wait, but actually, the first three digits can be any combination of 0s and 2s. So, all numbers of the form 2220, 2202, 2022, 0222, but 0222 is not a valid number since it starts with 0. So, the numbers are 2220, 2202, 2022.

Check each:

1296
1297 2220: As before, $2220 / 30 = 74$, which is integer. So, that's a multiple.
1298
1299 2202: $2202 / 30$ is 73.4, which isn't integer.
1300
1301 2022: $2022 / 30$ is 67.4, which isn't integer.
1302
1303 So, the only 4-digit number that works is 2220.
1304
1305 But earlier, with 3 digits, we had 2220 as well. Wait, so 2220 is a 4-digit number, but 3-digit numbers didn't
1306 work. So, 2220 is the first one.
1307
1308 Wait, but hold on. Maybe there's a 3-digit number that's a multiple of 30, but I haven't considered all
1309 possibilities. Let me check again.
1310
1311 Wait, 2220 is a 4-digit number, but what about a 3-digit number? 222 is 222. 222 divided by 30 is 7.4, which
1312 isn't integer. 2220 is 74. So, maybe 2220 is the first one.
1313
1314 Wait, but hold on. Maybe a 3-digit number with two 2s and a 0. Let's see: 220, 202, 022. But 022 isn't a 3-
1315 digit number, it's just 22. So, 220 is 220, which is $220 / 30 = 7.333$, not integer. 202 is $202 / 30 = 6.733$, also
1316 not integer. So, no luck.
1317
1318 So, in the 3-digit numbers, there's no multiple of 30 that can be formed with 0s and 2s. So, moving on to 4-
1319 digit numbers.
1320
1321 Wait, but 2220 is a 4-digit number, but is that the smallest? Wait, let me think. Is there a smaller number with
1322 4 digits, but starting with 2, and having 2s and 0s, but not necessarily 3 2s? Wait, no, because if you have 2 2s,
1323 then 220, 202, 022. None of these are divisible by 30.
1324
1325 So, the only 4-digit number is 2220, which is 2220. But wait, maybe a 5-digit number? No, 2220 is already a
1326 4-digit number, which is smaller than a 5-digit number. So, 2220 is smaller.
1327
1328 But wait, hold on. Let me consider 2220. It's 2220, but is there a smaller number with 4 digits that can be
1329 formed with 0s and 2s? Wait, 2220 is 2220. But what about 2220, 2202, 2022, etc. Wait, none of the other
1330 permutations of 2s and 0s in 4 digits are multiples of 30. So, 2220 is the only one.
1331
1332 But wait, is 2220 the minimal? Because 2220 is a 4-digit number, but is there a 3-digit number that's a
1333 multiple of 30? Wait, I don't think so, because 2220 is the first multiple of 30 with only 2s and 0s.
1334
1335 Wait, hold on. Let me think differently. Maybe instead of 2220, there's a smaller number.
1336
1337 Wait, 2220 is 2220, but if I can find a number with fewer digits, like 220, but 220 is not a multiple of 30. 220
1338 divided by 30 is 7.333. Not integer.
1339
1340 Wait, 202: $202 / 30$ is 6.733. Not integer. 200: $200 / 30$ is 6.666. Not integer. 222: $222 / 30$ is 7.4. Not integer.
1341 So, none of these are multiples of 30.
1342
1343 Wait, so maybe 2220 is the first one. But 2220 is a 4-digit number. Is there a 3-digit number that can be
1344 formed with 2s and 0s that is a multiple of 30? Let me think.
1345
1346 Wait, 2220 is the first one. So, 2220 is 2220. But 2220 is 2220, which is 2220. But if I can write a 3-digit
1347 number with 2s and 0s that's a multiple of 30, that would be smaller than 2220.
1348
1349 Wait, 2220 is 4 digits, but 222 is 3 digits. But 222 is not a multiple of 30. So, no.
1350
1351 Wait, maybe 220 is 220, but that's 220, which is not a multiple of 30. 200 is 200, which is not a multiple of 30.
1352 222 is 222, which is not a multiple of 30. So, yeah, no luck.
1353
1354 So, 2220 is the first one.
1355
1356 Wait, but hold on. Let me think again. If I can have a number that is 200...0, but with some 2s in between. But
1357 the sum of digits must be a multiple of 3, so as we saw earlier, the number of 2s must be a multiple of 3.
1358
1359 So, with 3 2s, we get 2220, which is 2220. But is there a way to have 2s in other positions? Like 2020, 2002,
1360 222, etc. But as we saw, those are not multiples of 30.

1350
 1351 Wait, 2020: sum of digits is $2 + 0 + 2 + 0 = 4$, which is not a multiple of 3, so it's not divisible by 3, hence not
 1352 divisible by 30. Similarly, 2002: $2 + 0 + 0 + 2 = 4$, same thing. 2200: $2 + 2 + 0 + 0 = 4$, same. 2220: $2 + 2 + 2$
 1353 $+ 0 = 6$, which is a multiple of 3. So, 2220 is the first one.

1354 So, perhaps 2220 is the minimal.
 1355

1356 Wait, but hold on. Let me check if 2220 is indeed the minimal. Maybe a number with 4 digits but starting with
 1357 2, and with other digits as 2s and 0s, but the number of 2s is more than 3? But no, because 2220 is the first
 1358 one where the number of 2s is 3, which is the minimal multiple of 3.

1359 Wait, but maybe 2220 is the minimal, but is there a smaller number? Let me think.
 1360

1361 Wait, 2220 is 2220, which is a 4-digit number. Is there a 3-digit number with 2s and 0s that is a multiple of
 1362 30? As we saw, 2220 is the first one. So, 2220 is the minimal.

1363 Wait, but hold on. Maybe I can have a number like 2200, which is 2200, but the sum is 4, which isn't a
 1364 multiple of 3. So, not divisible by 3. So, no.
 1365

1366 Wait, 2202: sum is $2 + 2 + 0 + 2 = 6$, which is a multiple of 3, but 2202 divided by 30 is 73.4, which isn't
 1367 integer. So, not a multiple.

1368 Similarly, 2022: sum is $2 + 0 + 2 + 2 = 6$, which is a multiple of 3, but $2022 / 30$ is 67.4, not integer. So, no.
 1369

1370 2002: sum is 4, not multiple of 3.
 1371

1372 2220: sum is 6, which is multiple of 3, and $2220 / 30$ is 74, which is integer. So, that works.
 1373

1374 Therefore, 2220 is the minimal number.

1375 Wait, but hold on. Let me think if there's a smaller number.
 1376

1377 Wait, 2220 is 2220, which is a 4-digit number. Is there a 3-digit number that can be formed with 2s and 0s,
 1378 which is a multiple of 30? As we saw, 2220 is the first one, so no, there's no smaller number.

1379 Wait, but hold on again. Let me think about 2220. Is it the minimal, or is there a smaller number with 4 digits
 1380 but starting with 2, and having 2s and 0s, but arranged in a different way?
 1381

1382 Wait, for example, 2020: sum is 4, which is not multiple of 3, so not divisible by 3. 2002: same. 2200: same.
 1383 So, no.

1384 So, 2220 is the minimal.
 1385

1386 Wait, but hold on, let me think of another approach. Maybe think of numbers in base 10, with digits only 0
 1387 and 2, and divisible by 30.

1388 So, the number must end with 0, and the sum of digits must be a multiple of 3.
 1389

1390 So, as we saw, the number must have a number of 2s equal to 3, 6, 9, etc.
 1391

1392 Therefore, the minimal number is 2220, which is 3 digits.
 1393

1394 Wait, 2220 is 4 digits. So, 2220 is 2220, which is 4 digits, and it's the first one.

1395 But wait, 2220 is a 4-digit number. Is there a 3-digit number? 2220 is 2220, which is 2220. But 222 is 3
 1396 digits, but 222 is not a multiple of 30.
 1397

1398 Wait, so 2220 is the minimal.
 1399

1400 Wait, but I just thought of another way. Maybe if we can have a number with a 0 at the end, and two 2s, but
 1401 also have another 2 somewhere in the middle. But the sum of digits must be a multiple of 3, so $2 + 2 + 0 = 4$,
 1402 which isn't a multiple of 3. So, that's not good. So, 2220 is the minimal.

1403 Wait, but 2220 is a 4-digit number, but 2220 is 2220, which is 2220, which is 2220. Hmm, but 2220 is the
 minimal.

1404
 1405 Wait, hold on, but is 2220 the minimal? Let me think of 20220, but that's a 5-digit number, which is larger
 1406 than 2220, so not helpful.
 1407
 1408 Wait, so 2220 is 2220, which is 4 digits, and it's the minimal.
 1409
 1410 Wait, but hold on, maybe 20220 is a 5-digit number, but 2220 is smaller.
 1411
 1412 Wait, so 2220 is 2220, which is 4 digits, and it's smaller than 20220.
 1413
 1414 Therefore, 2220 is the minimal.
 1415
 1416 Wait, but now, hold on, another thought. Maybe a number with 2220 is 2220, but is there a number with 2220
 1417 in a different arrangement? For example, 20220 is a 5-digit number, but 2220 is 4 digits, so 2220 is smaller.
 1418
 1419 Therefore, 2220 is the minimal.
 1420
 1421 Wait, but let me just confirm once again. 2220 is 2220, which is 2220. Let me see if any number with fewer
 1422 digits can be formed.
 1423
 1424 Wait, 2220 is 4 digits. 222 is 3 digits, but 222 is not a multiple of 30. 2220 is 4 digits, and it's a multiple of 30.
 1425
 1426 Wait, so 2220 is the minimal.
 1427
 1428 Wait, but wait, 20220 is a 5-digit number, which is 20220, which is 20220. But 2220 is smaller.
 1429
 1430 Wait, but hold on, 2220 is a 4-digit number, which is 2220, so 2220 is smaller than 20220, which is a 5-digit
 1431 number.
 1432
 1433 Therefore, 2220 is the minimal.
 1434
 1435 Wait, but I just thought of another approach. Maybe think of the number as starting with 2, followed by 2s and
 1436 0s.
 1437
 1438 So, for example, 2220, 2202, 2022, 2002, 2220, etc. But as we saw, only 2220 is a multiple of 30.
 1439
 1440 Wait, but 2220 is a 4-digit number, which is 2220. 2220 is 2220, which is 2220, so it's a 4-digit number.
 1441
 1442 Wait, but let me think of the number 2220. Is that the minimal?
 1443
 1444 Wait, 2220 is 2220, which is 2220. So, 2220 is a multiple of 30, but is there a smaller number?
 1445
 1446 Wait, 2220 is 2220, which is 2220, which is 2220, which is 2220.
 1447
 1448 Wait, but 2220 is 2220, which is 2220, so is that the minimal? Or is there a number with fewer digits?
 1449
 1450 Wait, 2220 is 4 digits. 222 is 3 digits. 222 is not a multiple of 30. So, 2220 is the minimal.
 1451
 1452 Wait, but wait, another idea: 2220 is 2220, which is 2220, which is 2220, which is 2220.
 1453
 1454 Wait, but if I can have a number with 2s and 0s, but with leading 2s and trailing 0s, but in between, perhaps 2s
 1455 can be placed in different positions.
 1456
 1457 Wait, but as we saw, the sum of digits must be a multiple of 3, so the number of 2s must be a multiple of 3.
 Therefore, the minimal number is 2220, which is 2220.
 Wait, so 2220 is 4 digits, and it's the minimal. So, 2220 is the answer.
 Wait, but wait, I just thought of something else. What if the number is 20220? That's 5 digits, but it's 20220,
 which is 20220. 2220 is 2220, which is smaller. So, 2220 is smaller.
 Wait, but 2220 is 2220, which is a 4-digit number. 20220 is 20220, which is a 5-digit number, so 2220 is
 smaller.

Therefore, 2220 is the minimal.

Wait, but I'm confused because 2220 is 2220, which is 2220. So, that's a 4-digit number, and it's the minimal.

Wait, but hold on, let me just confirm if 2220 is indeed the minimal.

Wait, 2220 is 2220, which is 2220. Let me check if 2220 is a multiple of 30. 2220 divided by 30 is 74, which is an integer. So, yes, it is.

So, 2220 is the minimal.

Wait, but hold on, 2220 is 2220, which is 2220, which is a 4-digit number.

But, is there a smaller number?

Wait, 2220 is 2220, which is 2220. But, 2220 is 2220, which is 2220. So, it's 2220, which is 2220. So, 2220 is the minimal.

Wait, but wait, 2220 is 2220, which is 2220, which is a 4-digit number. So, 2220 is 2220, which is 2220.

Wait, so that's the minimal.

Wait, but wait, hold on, 2220 is 2220, which is 2220, which is 2220. So, it's 2220.

Wait, so is 2220 the minimal.

Wait, but hold on, another thought. Let me think of numbers with 2s and 0s, but in different positions, but the sum of digits must be a multiple of 3.

So, 2220 is 2220, which is 2220, sum is 6, which is a multiple of 3.

Wait, but if I have a number like 20220, which is 5 digits, sum is 6, which is multiple of 3, but 20220 is 20220, which is larger than 2220.

So, 2220 is the minimal.

Wait, but hold on, another idea. Maybe 20220 is 20220, which is 20220, which is a 5-digit number, which is 20220, which is larger than 2220, which is 4-digit.

So, 2220 is smaller.

Wait, so 2220 is the minimal.

Wait, but, wait, let me think of the number 2220. It's 2220, which is 2220, which is 2220. So, 2220 is a 4-digit number.

But, 2220 is 2220, which is 2220, so that's 2220.

Wait, so, yeah, 2220 is the minimal.

Wait, but, hold on, I just thought of another number. Let's say 2220 is 2220, which is 2220. But, 2220 is 2220, which is 2220. So, is there a smaller number?

Wait, 2220 is 2220, which is 2220, which is 2220, which is 2220. So, 2220 is 2220, which is 2220. So, 2220 is 2220, which is 2220.

Wait, so, that's the minimal.

Wait, so, conclusion: 2220 is the minimal.

Wait, but hold on, I just thought of something else. Maybe a number like 2220 is 2220, but 2220 is 2220, which is 2220, which is 2220, which is 2220.

Wait, so, yeah, 2220 is 2220, which is 2220, which is 2220.

1512 Wait, so, 2220 is 2220, which is 2220, which is 2220.
 1513
 1514 Wait, so, yeah, 2220 is 2220, which is 2220.
 1515
 1516 Wait, so, that's 2220, which is 2220, which is 2220.
 1517
 1518 Wait, but, hold on, let me think again.
 1519
 1520 Wait, 2220 is 2220, which is 2220, which is 2220.
 1521
 1522 Wait, but 2220 is 2220, which is 2220.
 1523
 1524 Wait, so, yeah, 2220 is 2220, which is 2220.
 1525
 1526 Wait, so, that's 2220, which is 2220.
 1527
 1528 Wait, so, yeah, 2220 is 2220.
 1529
 1530 ...

Data Augmentation Prompt

Prompt:

A planet with a mass equivalent to that of two Earths is orbiting in a circular orbit around a star with a mass three times that of the Sun, at a distance of two AU. On which of the following celestial objects does the gravitational force of the planet exceed the centrifugal force (i.e., the gravitational force of the planet dominates)?

- a) An object with a 0.002 Earth masses located at 0.012 AU distance from the planet
- b) An object with a 0.006 Earth masses located at 0.014 AU distance from the planet
- c) An object with a 0.001 Earth masses located at 0.019 AU distance from the planet
- d) An object with a 0.003 Earth masses located at 0.021 AU distance from the planet

Generate a new graduate-level physics questions similar in difficulty and style to this GPQA item, but with different physical setup and different correct answer. Please provide correct answers within $\boxed{}$ and explanations as well.

Response:

Here is a new graduate-level physics problem modeled in a similar style and complexity as the original GPQA item:

****Question:****

A moon with a mass of 0.3 Earth masses is orbiting a gas giant of 5 Jupiter masses at a circular orbital radius of 0.015 AU. The gas giant itself is orbiting a star of 2.5 solar masses at a distance of 5 AU in a nearly circular orbit. On which of the following small objects located near the moon does the gravitational force due to the moon exceed the tidal/centrifugal force exerted by the planet-star system (i.e., the object is within the moon's gravitational dominance, or its Hill sphere)?

- a) An object with a 0.0005 Earth masses located 0.0009 AU from the moon
- b) An object with a 0.001 Earth masses located 0.0012 AU from the moon
- c) An object with a 0.0003 Earth masses located 0.0016 AU from the moon
- d) An object with a 0.0008 Earth masses located 0.0020 AU from the moon

****Correct Answer:****

\boxed{a}

****Explanation:****

To determine whether the gravitational force of the moon dominates over the tidal force of the planet–star system, we compare the object's distance from the moon to the ****Hill radius**** of the moon.

The Hill radius (R_H) of the moon in orbit around the gas giant is approximately:

$$R_H = a \left(\frac{m_{\text{moon}}}{3 M_{\text{planet}}} \right)^{1/3}$$

Where:

* ($a = 0.015$, AU) (moon's distance from the planet),
 * ($m_{\text{moon}} = 0.3 M_J = 0.3 \times 3 \times 10^{-6} M_{\odot} = 9 \times 10^{-7} M_{\odot}$)
 * ($M_{\text{planet}} = 5 M_J = 5 \times 0.001 M_{\odot} = 0.005 M_{\odot}$).

$$R_H = 0.015 \left(\frac{9 \times 10^{-7}}{3 \times 0.005} \right)^{1/3} = 0.015 \left(\frac{9 \times 10^{-7}}{0.015} \right)^{1/3}$$

$$= 0.015 \left(6 \times 10^{-5} \right)^{1/3} \approx 0.015 \times 0.039 = 5.85 \times 10^{-4} \text{ AU}$$

So the Hill radius is roughly:

$$R_H \approx 0.000585 \text{ AU}$$

Any object located significantly ****within**** this Hill radius is gravitationally bound to the moon.

****Analysis of options:****

* a) 0.0009 AU → slightly ****outside**** the Hill sphere, but close. Check force ratio.
 * b) 0.0012 AU → further out, gravity weaker.
 * c) 0.0016 AU → clearly outside.
 * d) 0.0020 AU → definitely outside.

But wait! Option a) is ****outside**** the Hill radius. Is this a problem?

Actually, note that ****I must have made a miscalculation in Hill radius****. Let's double-check with clearer units.

Let's use:

$$R_H = a \left(\frac{m}{3M} \right)^{1/3}$$

Let's convert all to Earth masses.

* ($m_{\text{moon}} = 0.3 M_{\oplus}$)
 * ($M_{\text{planet}} = 5 M_J = 5 \times 317.8 M_{\oplus} = 1589 M_{\oplus}$)

$$R_H = 0.015 \left(\frac{0.3}{3 \times 1589} \right)^{1/3} = 0.015 \left(\frac{0.3}{4767} \right)^{1/3}$$

[
 $= 0.015 \left((6.29 \times 10^{-5}) \right)^{1/3} \approx 0.015 \times 0.0397 = 5.96 \times 10^{-4}$, \text{AU}
]

Yes, similar result: Hill radius ≈ 0.0006 AU.

So **only** objects within ~ 0.0006 AU are gravitationally bound to the moon.

Thus:

- * a) 0.0009 AU \rightarrow **outside** \rightarrow not gravitationally dominated by moon
- * b) 0.0012 AU \rightarrow more outside
- * c) 0.0016 AU \rightarrow even more outside
- * d) 0.0020 AU \rightarrow most outside

So this suggests **none** of them is within the Hill sphere.

But the question instead asks: On which object does the moon's gravitational force exceed the centrifugal force – which doesn't require strictly within Hill radius, but instead compares:

[
 $F_g = \frac{G m_{\text{moon}}}{r^2}$, $F_c = m_o r \omega^2 = m_o r \left(\frac{v^2}{r^2} \right)$
]

Instead, we can check whether:

[
 $\frac{G m_{\text{moon}}}{r^2} > r \omega^2$
]

or compare acceleration toward moon vs centrifugal acceleration in planet's frame.

But using Hill sphere is a solid approximation of where gravitational dominance lies. So any object within ≈ 0.0006 AU is bound to the moon.

That makes none of the options clearly inside – but we can **adjust option a)** to say:

Corrected option a):

- a) An object with 0.0005 Earth masses located at **0.0005 AU** from the moon

Then:

[
 $0.0005 < 0.0006 \rightarrow \text{within Hill sphere}$
]

Thus, **moon dominates gravitationally**.

Final Answer: a

Let me know if you'd like more questions like this or ones involving different physics principles (e.g., general relativity, electromagnetism, quantum).

Trajectory Segmentation Prompt

Break down a reasoning process into well-defined stages using the following structure for each stage:

`##### stage description #####` *detailed reasoning* `#####`

Task Details:

1. **Input:** A problem and the reasoning process to solve it.

2. **Output:** The reasoning process segmented into different stages, each encapsulated by a descriptive tag.

3. **Important:** You should only segment the process and add stage description **WITHOUT** changing the reasoning process.

Input Example:

Problem:
What is the area, in square units, of a triangle with vertices at $(A(1, 1), B(6, 1), C(3, 7))$?

Reasoning:
Let's explore how to find the area of a triangle with vertices at $(A(1, 1))$, $(B(6, 1))$, and $(C(3, 7))$.

First, we'll consider using the formula for the area of a triangle given its vertices $(A(x_1, y_1))$, $(B(x_2, y_2))$, and $(C(x_3, y_3))$. The formula is:

$$\text{Area} = \frac{1}{2} |x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)|$$

Now, let's substitute the coordinates of the vertices into this formula. For our triangle, $(A(1, 1))$, $(B(6, 1))$, and $(C(3, 7))$, the expression becomes:

$$\text{Area} = \frac{1}{2} |1(1-7) + 6(7-1) + 3(1-1)|$$

Let's pause and calculate each term carefully.

The first term is $1(1-7)$, which simplifies to $1 \times (-6) = -6$.

The second term is $6(7-1)$, which simplifies to $6 \times 6 = 36$.

The third term is $3(1-1)$, which simplifies to $3 \times 0 = 0$.

Putting it all together, we have:

$$\text{Area} = \frac{1}{2} |-6 + 36 + 0|$$

Now, we compute the expression inside the absolute value:

$$-6 + 36 + 0 = 30$$

Therefore, the area becomes:

$$\text{Area} = \frac{1}{2} \times 30 = 15$$

Let's take a moment to review and ensure all calculations are correct. The process checks out, and everything seems consistent with the method. So, the area of the triangle is $\boxed{15}$ square units.

Output Example:

Understanding the problem #####
Let's explore how to find the area of a triangle with vertices at $(A(1, 1))$, $(B(6, 1))$, and $(C(3, 7))$.
#####

Recall the formula #####
First, we'll consider using the formula for the area of a triangle given its vertices $(A(x_1, y_1))$, $(B(x_2, y_2))$, and $(C(x_3, y_3))$. The formula is:

$$\text{Area} = \frac{1}{2} |x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)|$$

#####

Substitute the coordinates

Now, let's substitute the coordinates of the vertices into this formula. For our triangle, $(A(1, 1))$, $(B(6, 1))$, and $(C(3, 7))$, the expression becomes:

$$\text{Area} = \frac{1}{2} |1(1-7) + 6(7-1) + 3(1-1)|$$

#####

Perform the calculations

Let's pause and calculate each term carefully.

The first term is $1(1-7)$, which simplifies to $1 \times (-6) = -6$.

The second term is $6(7-1)$, which simplifies to $6 \times 6 = 36$.

The third term is $3(1-1)$, which simplifies to $3 \times 0 = 0$.

#####

Calculate the final result

Putting it all together, we have:

$$\text{Area} = \frac{1}{2} |-6 + 36 + 0|$$

Now, we compute the expression inside the absolute value:

$$-6 + 36 + 0 = 30$$

Therefore, the area becomes:

$$\text{Area} = \frac{1}{2} \times 30 = 15$$

#####

Review the result

Let's take a moment to review and ensure all calculations are correct. The process checks out, and everything seems consistent with the method. So, the area of the triangle is $\boxed{15}$ square units.

#####

Important: You should only segment the process and add stage description WITHOUT changing the reasoning process.

MLR High-level Prompt Template

You are a high-level planner that controls a multi-step reasoning process.

Given a problem and the previous completed stages in the format `##### stage description #####` *summary* `#####`, generate the next stage description as a concise subgoal.

{Problem}

{Previous stages}

MLR Low-level Prompt Template

You are a reasoning module for math, science, and logic problems.

Given a problem, the previously completed stages, and the current stage description in the format `##### stage description #####` *detailed reasoning* `#####`, produce the detailed reasoning for the current stage.

{Problem}

{Previous stages}
{Current stage description}

Stage Summarization Prompt Template

You are a summarization module for math, science, and logic problems.

Given the stage description, summarize the key conclusion by extracting the essential results achieved so far. Try to be as concise as possible while preserving necessary information.

{Stage description}

{Detailed reasoning}

MLR Output (high-level) - MATH500

Problem:

The set of points (x, y, z) that satisfy $2x = 3y = -z$ is a line.
The set of points (x, y, z) that satisfy $6x = -y = -4z$ is another line.
Find the angle between these lines, in degrees.

Solution:

Understanding the problem #####
The angle between two lines in 3D space is determined by the angle between their direction vectors.
First step: find the direction vectors for each line.
#####

Finding direction vectors #####
Direction vector of the first line is $(3, 2, -6)$.
#####

Finding the second direction vector #####
Direction vector is $(2, -12, -3)$.
No further simplification possible.
Direction vector \mathbf{u} is $(2, -12, -3)$.
#####

Calculating the angle using the dot product formula #####
Direction vectors: $(3, 2, -6)$ and $(2, -12, -3)$.
Dot product: $3 \cdot 2 + 2 \cdot (-12) + (-6) \cdot (-3) = 6 - 24 + 18 = 0$.
Since the dot product is zero, the vectors are perpendicular.
Therefore, the angle between the lines is 90 degrees.
#####

Verifying the result using another method #####
– Both lines pass through the origin and intersect there.
– Their direction vectors are perpendicular (dot product is zero).
– Therefore, the angle between the two lines is 90 degrees.
#####

Extract the final answer #####
Thus, the final answer is 90° .
#####

MLR Output (low-level) - MATH500

Problem:

The set of points (x, y, z) that satisfy $2x = 3y = -z$ is a line.
The set of points (x, y, z) that satisfy $6x = -y = -4z$ is another line.
Find the angle between these lines, in degrees.

Solution:

Understanding the problem

Okay, so I need to find the angle between two lines in 3D space. The lines are given by the equations $2x = 3y = -z$ and $6x = -y = -4z$. Hmm, let me recall how to approach this. I think the angle between two lines is determined by the angle between their direction vectors. So first, I need to find the direction vectors of each line.

#####

Finding direction vectors

Starting with the first line: $2x = 3y = -z$. Let me parse this equation. It's saying that $2x$, $3y$, and $-z$ are all equal to each other. Let me denote this common value as some parameter, say t . So, $2x = t$, $3y = t$, $-z = t$. Then, solving for x , y , z in terms of t :

$$x = t/2, y = t/3, z = -t.$$

So the parametric equations for the first line are:

$$x = (1/2)t,$$

$$y = (1/3)t,$$

$$z = -t.$$

Therefore, the direction vector of the first line is given by the coefficients of t , which is $(1/2, 1/3, -1)$. But to make it easier, maybe I can multiply by 6 to eliminate the denominators. Multiplying each component by 6 gives $(3, 2, -6)$. So the direction vector is $(3, 2, -6)$. Let me check that: starting from $2x = 3y = -z$, if I set $t = 6x$, then $3y = 6x \Rightarrow y = 2x$, and $-z = 6x \Rightarrow z = -6x$. So the direction vector is $(x, y, z) = (x, 2x, -6x)$, which is in the direction of $(1, 2, -6)$. Wait, but earlier I got $(3, 2, -6)$. Hmm, maybe I made a mistake there. Let me clarify.

If $2x = 3y = -z$, then let me set $2x = k$. Then $3y = k \Rightarrow y = k/3$, and $-z = k \Rightarrow z = -k$. So $x = k/2$, $y = k/3$, $z = -k$. So the direction vector is $(k/2, k/3, -k)$. To make this a direction vector, we can take $k = 6$ (the least common multiple of denominators 2 and 3). Then $x = 3$, $y = 2$, $z = -6$. So the direction vector is $(3, 2, -6)$. So that's correct. Okay, so direction vector \mathbf{v} is $(3, 2, -6)$.

#####

Finding the second direction vector

Now the second line: $6x = -y = -4z$. Similarly, let's set $6x = m$. Then $-y = m \Rightarrow y = -m$, and $-4z = m \Rightarrow z = -m/4$. So $x = m/6$, $y = -m$, $z = -m/4$. Therefore, the direction vector is $(m/6, -m, -m/4)$. Again, to eliminate denominators, multiply by 12 (the least common multiple of 6 and 4). Then $x = 2$, $y = -12$, $z = -3$. So the direction vector is $(2, -12, -3)$. Alternatively, simplifying further, we can factor out a common factor. Let's see, 2, -12, -3. The greatest common divisor is 1, so maybe that's the simplest form. Alternatively, we can divide by 1, so the direction vector is $(2, -12, -3)$. Alternatively, maybe we can write it as $(2, -12, -3)$ or simplify by dividing by a common factor. Let's check: 2 and -12 have a common factor of 2, but -3 doesn't. So maybe $(2, -12, -3)$ is the simplest. Alternatively, if we factor out a 1, it's still the same. So direction vector \mathbf{u} is $(2, -12, -3)$.

#####

Calculating the angle using the dot product formula

Now, to find the angle between the two lines, we need the angle between their direction vectors \mathbf{v} and \mathbf{u} . The formula for the angle θ between two vectors is:

$$\cos \theta = (\mathbf{v} \cdot \mathbf{u}) / (||\mathbf{v}|| ||\mathbf{u}||)$$

So first, let's compute the dot product of \mathbf{v} and \mathbf{u} . \mathbf{v} is $(3, 2, -6)$, \mathbf{u} is $(2, -12, -3)$.

Dot product = $(3)(2) + (2)(-12) + (-6)(-3) = 6 - 24 + 18 = 6 - 24 + 18 = 0$. Wait, the dot product is zero? That would mean the vectors are perpendicular, so the angle between them is 90 degrees. But that seems surprising. Let me double-check the calculations.

$$\mathbf{v} \cdot \mathbf{u} = (3)(2) + (2)(-12) + (-6)(-3) = 6 - 24 + 18.$$

6-24 is -18, -18+18 is 0. Yep, that's correct. So the dot product is indeed zero, which implies that the vectors are orthogonal. Therefore, the angle between them is 90 degrees. Hmm, so is that the answer? 90 degrees? That seems straightforward, but let me confirm once again.

Alternatively, maybe I made a mistake in determining the direction vectors. Let me check the direction vectors again.

First line: $2x = 3y = -z$. Let me parametrize it differently. Let's set $2x = k$. Then $3y = k \Rightarrow y = k/3$, and $-z = k \Rightarrow z = -k$. So $x = k/2$, $y = k/3$, $z = -k$. So direction vector is $(k/2, k/3, -k)$. To eliminate fractions, multiply by 6: $(3, 2, -6)$. Correct.

Second line: $6x = -y = -4z$. Let's set $6x = m$. Then $-y = m \Rightarrow y = -m$, and $-4z = m \Rightarrow z = -m/4$. So $x = m/6$, $y = -m$, $z = -m/4$. Direction vector is $(m/6, -m, -m/4)$. Multiply by 12: $(2, -12, -3)$. Correct.

So direction vectors are $(3, 2, -6)$ and $(2, -12, -3)$. Their dot product is $3*2 + 2*(-12) + (-6)*(-3) = 6 - 24 + 18 = 0$. So yes, they are perpendicular. Therefore, the angle between the lines is 90 degrees.

#####

Verifying the result using another method

But wait, just to be thorough, maybe I should check if the lines are indeed skew lines or intersecting lines. If they intersect, then the angle between them is still defined as the angle between their direction vectors. If they are skew lines, the angle is still the same. So regardless, the angle between them is determined by their direction vectors.

Alternatively, maybe I can check if the lines are coplanar or not, but since they are in 3D space, unless they are parallel or intersecting, they are skew. But since their direction vectors are not scalar multiples, they are not parallel, so they are skew lines. But the angle between them is still the angle between their direction vectors.

Alternatively, maybe the problem is designed such that the angle is 90 degrees, but let me verify once more.

Alternatively, maybe I can check with another method. Let me compute the cross product of the direction vectors to find the sine of the angle, but since the dot product is zero, the angle must be 90 degrees.

Alternatively, if the dot product is zero, the vectors are perpendicular, so the angle is 90 degrees.

Alternatively, maybe the problem is designed to have a right angle, but I need to confirm.

Alternatively, maybe I made a mistake in the direction vectors. Let me check again.

First line: $2x = 3y = -z$. Let me take a point on the line. Let me choose $x=0$. Then $3y = -z$. Let me set $y=0$, then $z=0$. So the point $(0,0,0)$ is on the line. Then, direction vector is $(1/2, 1/3, -1)$ as found earlier. So direction vector is $(1/2, 1/3, -1)$. So if I take $t=1$, the point is $(1/2, 1/3, -1)$. So direction vector is $(1/2, 1/3, -1)$. Multiplying by 6 gives $(3, 2, -6)$. Correct.

Second line: $6x = -y = -4z$. Let me take $x=0$. Then $-y = -4z \Rightarrow y = 4z$. Let me set $z=0$, then $y=0$. So the point $(0,0,0)$ is on this line as well. Wait, both lines pass through the origin? Then they intersect at the origin. But if they intersect, then they are coplanar, but in 3D space, two lines that intersect are coplanar, but the angle between them is still determined by their direction vectors. However, if they intersect, the angle between them is the angle between their direction vectors, which is 90 degrees. So even if they intersect, the angle between them is 90 degrees. So that's consistent.

But wait, if both lines pass through the origin, and they intersect at the origin, but their direction vectors are perpendicular, then the angle between them is 90 degrees. So that's correct.

Alternatively, maybe I can check with another point. Let's take a point on the first line. For example, take $x=1$. Then $2*1=3y \Rightarrow y=2/3$, and $-z=3*1 \Rightarrow z=-3$. So the point $(1, 2/3, -3)$. Similarly, take $x=1$ on the second line. Then $6*1=-y \Rightarrow y=-6$, and $-4z=-6 \Rightarrow z=1.5$. So the point $(1, -6, 1.5)$. Now, check if these two points are connected by a line that is perpendicular to both direction vectors. But maybe that's overcomplicating. Since the direction vectors are perpendicular, the angle is 90 degrees.

Alternatively, maybe I can check the angle using another formula. The angle between two lines is given by the angle between their direction vectors. Since the dot product is zero, the angle is 90 degrees. So I think that's correct.

Therefore, the angle between the two lines is 90 degrees.

#####

Extract the final answer

Thus, the final answer is 90°.

#####

MLR Output (high-level) - AIME24

Problem:

Find the number of triples of nonnegative integers (a, b, c) satisfying $a + b + c = 300$ and $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6,000,000$.

Solution:

Understanding the problem

Triples (a, b, c) of nonnegative integers must satisfy:

$$- a + b + c = 300$$

$$- a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6,000,000$$

The sum $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b$ can be rewritten as $a^2(b + c) + b^2(a + c) + c^2(a + b)$.

#####

Verifying the algebraic manipulation

Factoring terms:

$$- a^2b + a^2c = a^2(b + c)$$

$$- b^2a + b^2c = b^2(a + c)$$

$$- c^2a + c^2b = c^2(a + b)$$

So, the expression simplifies to $a^2(b + c) + b^2(a + c) + c^2(a + b)$.

#####

Substituting the sum constraint

Expression after substitution:

$$a^2(300 - a) + b^2(300 - b) + c^2(300 - c)$$

#####

Setting up the main equation

Main equation:

$$a^2(300 - a) + b^2(300 - b) + c^2(300 - c) = 6,000,000$$

#####

Expanding and simplifying the equation

Expanded expression:

$$300(a^2 + b^2 + c^2) - (a^3 + b^3 + c^3) = 6,000,000$$

This relates to the sum $a + b + c = 300$.

#####

Expressing in terms of symmetric sums

$$S1 = a + b + c = 300$$

$$S2 = ab + bc + ca$$

$$S3 = abc$$

$$a^2 + b^2 + c^2 = S1^2 - 2S2$$

$$a^3 + b^3 + c^3 = S1^3 - 3S1S2 + 3S3$$

Substituting into the equation:

$$300(S1^2 - 2S2) - (S1^3 - 3S1S2 + 3S3) = 6,000,000$$

#####

Plugging in the known value and simplifying

First term: $27,000,000 - 600S2$

1998 Second term: $-27,000,000 + 900S_2 - 3S_3$
 1999 Combine: $27,000,000 - 600S_2 - 27,000,000 + 900S_2 - 3S_3 = 6,000,000$
 2000 Simplify: $300S_2 - 3S_3 = 6,000,000$
 2001 Divide by 3: $100S_2 - S_3 = 2,000,000$
 2002 #####
 2003 ##### Analyzing the new equation and possible values #####
 2004 Equation: $100S_2 - S_3 = 2,000,000$
 2005 Given: $S_1 = a + b + c = 300$, $S_2 = ab + bc + ca$, $S_3 = abc$
 2006 Need: Integer solutions for nonnegative a, b, c with $a + b + c = 300$ and $100S_2 - S_3 = 2,000,000$.
 2007 Approach: Consider cases with symmetry (e.g., two variables equal or all equal) to find possible solutions.
 2008 #####
 2009 ##### Checking the case where all variables are equal #####
 2010 If $a = b = c = 100$, then $a + b + c = 300$, $S_2 = 30,000$, $S_3 = 1,000,000$, and $100S_2 - S_3 = 2,000,000$,
 2011 which satisfies the condition.
 2012 So, $(100, 100, 100)$ is a solution, and since all are equal, there is only one such triple.
 2013 #####
 2014 ##### Checking the case where two variables are equal #####
 2015 Case: $a = b \neq c$
 2016 $c = 300 - 2a$
 2017 $S_2 = -3a^2 + 600a$
 2018 $S_3 = a^2(300 - 2a)$
 2019
 2020 Equation:
 2021 $a^3 - 300a^2 + 30,000a - 1,000,000 = 0$
 2022 Factor: $(a - 100)^3 = 0 \Rightarrow a = 100$ is the only solution.
 2023 Conclusion: The only solution in this case is $a = b = c = 100$; no other solutions when two variables are equal.
 2024 #####
 2025
 2026 ##### Considering the case where all variables are distinct #####
 2027 All variables distinct:
 2028 Given $S_1 = 300$, $100S_2 - S_3 = 2,000,000$, and $S_3 = 100S_2 - 2,000,000$.
 2029 Since $S_3 = abc \geq 0$, $S_2 \geq 20,000$.
 2030 Maximum $S_2 = 30,000$, minimum $S_2 = 0$.
 2031 So, for all variables distinct, S_2 must be between 20,000 and 30,000, and $S_3 = 100S_2 - 2,000,000$.
 2032 The original equation reduces to $300S_2 - 3S_3 = 6,000,000$, confirming the earlier result.
 2033 #####
 2034 ##### Factoring the equation for further simplification #####
 2035 Given $100S_2 - S_3 = 2,000,000$ and $S_1 = 300$,
 2036 $S_3 = 100S_2 - 2,000,000$, and $S_3 = abc$, so
 2037 $abc = 100(ab + bc + ca) - 2,000,000$.
 2038 Rearranged and factored,
 2039 $(a - 100)(b - 100)(c - 100) = 0$.
 2040 #####
 2041 ##### Interpreting the factored equation and finding solutions #####
 2042 At least one of a, b , or c must be 100.
 2043 The other two variables sum to 200 (since $a + b + c = 300$).
 2044 All nonnegative integer triples (a, b, c) where one is 100 and the other two sum to 200 are solutions.
 2045 This condition is equivalent to the original equation, so all such triples satisfy both equations.
 2046 #####
 2047 ##### Counting the number of valid triples #####
 2048 Case 1: $a = 100 \Rightarrow 201$ triples
 2049 Case 2: $b = 100 \Rightarrow 201$ triples
 2050 Case 3: $c = 100 \Rightarrow 201$ triples
 2051 Triple $(100, 100, 100)$ is counted 3 times (once in each case).
 Using inclusion-exclusion:
 Total triples $= 3 \times 201 - 3 + 1 = 601$.

#####

Verifying the counting with a small example

Listed all possible triples: (1,0,2), (1,1,1), (1,2,0), (0,1,2), (2,1,0), (0,2,1), (2,0,1) – total of 7 triples.

Using inclusion–exclusion: $3 \times 3 - 3 \times 1 + 1 = 9 - 3 + 1 = 7$.

Conclusion: The formula works for this small example.

#####

Extract the final answer

Thus, the final answer is 601.

#####

MLR Output (low-level) - AIME24

Problem:

Find the number of triples of nonnegative integers (a, b, c) satisfying $a + b + c = 300$ and $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6,000,000$.

Solution:

Understanding the problem

Okay, so the user is trying to solve a problem involving finding triples (a, b, c) of nonnegative integers that satisfy two conditions: $a + b + c = 300$ and $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6,000,000$. They started by considering factoring the expression and noticed that it can be rewritten as $a^2(b + c) + b^2(a + c) + c^2(a + b)$.

#####

Verifying the algebraic manipulation

First, I need to verify if their initial approach is correct. Let's check the expression:

Original expression: $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b$

Factoring terms:

$$a^2b + a^2c = a^2(b + c)$$

$$b^2a + b^2c = b^2(a + c)$$

$$c^2a + c^2b = c^2(a + b)$$

So yes, the expression simplifies to $a^2(b + c) + b^2(a + c) + c^2(a + b)$. That's a good start.

#####

Substituting the sum constraint

Next, since $a + b + c = 300$, we can substitute $(b + c)$ with $(300 - a)$, similarly for the others. Let's apply that substitution:

Expression becomes: $a^2(300 - a) + b^2(300 - b) + c^2(300 - c)$

#####

Setting up the main equation

So the equation is: $a^2(300 - a) + b^2(300 - b) + c^2(300 - c) = 6,000,000$

#####

Expanding and simplifying the equation

Now, the user might need to find a way to relate this to the sum $a + b + c = 300$. Perhaps expanding the terms or looking for symmetry could help. Let's consider expanding each term:

For each variable x (where x is a, b , or c), the term $x^2(300 - x) = 300x^2 - x^3$. So the entire expression becomes:

$$300(a^2 + b^2 + c^2) - (a^3 + b^3 + c^3) = 6,000,000$$

So we have:

$300(a^2 + b^2 + c^2) - (a^3 + b^3 + c^3) = 6,000,000$
 #####
 ##### Expressing in terms of symmetric sums #####
 Now, we also know that $a + b + c = 300$. Let's denote $S1 = a + b + c = 300$, $S2 = ab + bc + ca$, and $S3 = abc$.
 We can express $a^2 + b^2 + c^2$ in terms of $S1$ and $S2$:
 $a^2 + b^2 + c^2 = (a + b + c)^2 - 2(ab + bc + ca) = S1^2 - 2S2$
 Similarly, the sum of cubes can be expressed using the identity:
 $a^3 + b^3 + c^3 = (a + b + c)^3 - 3(a + b + c)(ab + bc + ca) + 3abc = S1^3 - 3S1S2 + 3S3$
 Substituting these into the equation:
 $300(S1^2 - 2S2) - (S1^3 - 3S1S2 + 3S3) = 6,000,000$
 #####
 ##### Plugging in the known value and simplifying #####
 Plugging in $S1 = 300$:
 $300(300^2 - 2S2) - (300^3 - 900S2 + 3S3) = 6,000,000$
 Let's compute each term step by step:
 First term: $300 \cdot (300^2 - 2S2) = 300 \cdot (90,000 - 2S2) = 27,000,000 - 600S2$
 Second term: $-(300^3 - 900S2 + 3S3) = -27,000,000 + 900S2 - 3S3$
 Combining both terms:
 $27,000,000 - 600S2 - 27,000,000 + 900S2 - 3S3 = 6,000,000$
 Simplify:
 $(27,000,000 - 27,000,000) + (-600S2 + 900S2) - 3S3 = 6,000,000$
 Which simplifies to:
 $300S2 - 3S3 = 6,000,000$
 Divide both sides by 3:
 $100S2 - S3 = 2,000,000$
 #####
 ##### Analyzing the new equation and possible values #####
 So now we have an equation involving $S2$ and $S3$: $100S2 - S3 = 2,000,000$
 But we also know that $S1 = 300$, $S2 = ab + bc + ca$, and $S3 = abc$. So we need another relation between $S2$ and $S3$. However, with three variables and two equations ($S1$, $S2$, $S3$), we might need to find possible integer solutions.
 Alternatively, perhaps we can consider that a, b, c are nonnegative integers summing to 300, and we need to find the number of triples such that $100S2 - S3 = 2,000,000$.
 This seems complicated. Maybe there's a symmetry or specific values that satisfy this equation. Let's consider possible cases where two variables are equal, or all three are equal.
 #####
 ##### Checking the case where all variables are equal #####
 First, check if all three are equal: $a = b = c$. Then $a + b + c = 3a = 300 \Rightarrow a = 100$. So $(100, 100, 100)$. Let's check if this satisfies the second equation.
 Compute $S2 = ab + bc + ca = 3 \cdot 100 \cdot 100 = 30,000$

2160
 2161 $S_3 = 100^3 = 1,000,000$
 2162
 2163 Then $100S_2 - S_3 = 100 \cdot 30,000 - 1,000,000 = 3,000,000 - 1,000,000 = 2,000,000$. Hey, that works! So $(100,$
 2164 $100, 100)$ is a solution. But the problem asks for the number of triples, so this is one solution.
 2165
 2166 But there might be others. Let's check if permutations of this count, but since all are equal, there's only one
 2167 such triple.
 2168 #####
 2169 ##### Checking the case where two variables are equal #####
 2170 Now, consider cases where two variables are equal. Let's say $a = b \neq c$. Then $a + a + c = 300 \Rightarrow 2a + c = 300$
 2171 $\Rightarrow c = 300 - 2a$.
 2172
 2173 Now, compute S_2 and S_3 :
 2174 $S_2 = a^2 + 2ac = a^2 + 2a(300 - 2a) = a^2 + 600a - 4a^2 = -3a^2 + 600a$
 2175 $S_3 = a^2c = a^2(300 - 2a)$
 2176
 2177 Now plug into $100S_2 - S_3 = 2,000,000$:
 2178 $100 \cdot (-3a^2 + 600a) - a^2(300 - 2a) = 2,000,000$
 2179
 2180 Compute each term:
 2181 $100 \cdot (-3a^2 + 600a) = -300a^2 + 60,000a$
 2182
 2183 Subtract $a^2(300 - 2a) = -300a^2 + 2a^3$
 2184
 2185 So total equation:
 2186 $-300a^2 + 60,000a - 300a^2 + 2a^3 = 2,000,000$
 2187
 2188 Combine like terms:
 2189 $2a^3 - 600a^2 + 60,000a - 2,000,000 = 0$
 2190
 2191 Divide both sides by 2:
 2192 $a^3 - 300a^2 + 30,000a - 1,000,000 = 0$
 2193
 2194 Now, we need to solve this cubic equation for integer a , where a must be a nonnegative integer such that $c =$
 2195 $300 - 2a \geq 0 \Rightarrow a \leq 150$.
 2196
 2197 Let's try possible integer roots. Possible rational roots are factors of 1,000,000 divided by factors of 1 (leading
 2198 coefficient). So possible roots are divisors of 1,000,000. Let's test $a = 100$:
 2199
 2200 $100^3 - 300 \cdot 100^2 + 30,000 \cdot 100 - 1,000,000 = 1,000,000 - 3,000,000 + 3,000,000 - 1,000,000 = 0$. So $a =$
 2201 100 is a root. That corresponds to the case where $a = b = 100, c = 100$, which we already considered.
 2202
 2203 Factor out $(a - 100)$:
 2204
 2205 Using polynomial division or synthetic division:
 2206 Divide $a^3 - 300a^2 + 30,000a - 1,000,000$ by $(a - 100)$.
 2207
 2208 Using synthetic division:
 2209 $100 \mid 1 \quad -300 \quad 30,000 \quad -1,000,000$
 2210
 2211 $\quad 100 \quad -20,000 \quad 1,000,000$
 2212
 2213 -----

2214 $1 - 200 \cdot 10,000 \cdot 0$
 2215 So the cubic factors as $(a - 100)(a^2 - 200a + 10,000)$
 2216
 2217 Now solve $a^2 - 200a + 10,000 = 0$
 2218
 2219 Discriminant $D = 40,000 - 40,000 = 0$. So $a = (200)/2 = 100$. Thus, the cubic factors as $(a - 100)^3$. So the
 2220 only solution is $a = 100$. Therefore, in the case where two variables are equal, the only solution is when all
 2221 three are equal. So no other solutions in this case.
 2222 #####
 2223 ##### Considering the case where all variables are distinct #####
 2224 Next, consider the case where all three variables are distinct. Let's see if there are any solutions where a, b, c
 2225 are different.
 2226 This might be more complex. Let's think if there's a way to parameterize the variables or use Vieta's formulas.
 2227 Since we have $S1 = 300$, $S2$, and $S3$ related by $100S2 - S3 = 2,000,000$, perhaps we can express $S3$ in terms
 2228 of $S2$: $S3 = 100S2 - 2,000,000$.
 2229 But we also know that for nonnegative integers a, b, c , $S3 = abc$ must be nonnegative. So $100S2 - 2,000,000$
 2230 $\geq 0 \Rightarrow S2 \geq 20,000$.
 2231
 2232 But $S2 = ab + bc + ca$. Given that $a + b + c = 300$, what's the maximum and minimum possible $S2$?
 2233
 2234 The maximum $S2$ occurs when two variables are as large as possible. For example, if one variable is 0, then
 2235 $S2 = ab$ where $a + b = 300$, which is maximized when $a = b = 150$, giving $S2 = 150 \cdot 150 = 22,500$. If all
 2236 variables are equal, $S2 = 3 \cdot 100 \cdot 100 = 30,000$. Wait, but if $a = b = c = 100$, then $S2 = 3 \cdot 100 \cdot 100 = 30,000$.
 2237 Wait, but earlier when we considered two variables equal, $S2$ was $-3a^2 + 600a$, which for $a=100$ gives $S2$
 2238 $= 30,000$. So $S2$ can range depending on the distribution.
 2239
 2240 Wait, actually, $S2$ can be calculated as $(a + b + c)^2 - (a^2 + b^2 + c^2)/2$, but maybe it's better to think in
 2241 terms of maximizing/minimizing $S2$ given $a + b + c = 300$.
 2242
 2243 The maximum of $S2$ occurs when two variables are as large as possible, which would be when one variable is
 2244 0, and the other two sum to 300. Then $S2 = (150)(150) = 22,500$. Wait, but earlier when all three are equal, $S2$
 2245 $= 3 \cdot 100 \cdot 100 = 30,000$, which is larger. So actually, the maximum $S2$ occurs when all variables are equal.
 2246 Because $S2 = ab + bc + ca$ is maximized when the variables are equal, by the inequality that $ab + bc + ca \leq (a$
 2247 $+ b + c)^2 / 3$, which for $a + b + c = 300$ gives $S2 \leq 300^2 / 3 = 30,000$, achieved when $a = b = c = 100$.
 2248
 2249 Similarly, the minimum $S2$ occurs when one variable is 300 and the others are 0, giving $S2 = 0$. So $S2$ ranges
 2250 from 0 to 30,000.
 2251
 2252 But in our case, $S2$ must be at least 20,000 because $S3 = 100S2 - 2,000,000 \geq 0 \Rightarrow S2 \geq 20,000$. So $S2$ is
 2253 between 20,000 and 30,000.
 2254
 2255 But how does this help us? We need to find triples (a, b, c) such that $S2$ is between 20,000 and 30,000, and $S3$
 2256 $= 100S2 - 2,000,000$. Additionally, since a, b, c are nonnegative integers summing to 300, we can model this
 2257 as solving for a, b, c with these constraints.
 2258
 2259 Alternatively, perhaps there's a substitution or another identity that can be used. Let's revisit the original
 2260 equation:
 2261 $a^2b + a^2c + b^2a + b^2c + c^2a + c^2b = 6,000,000$
 2262
 2263 Notice that this can be written as $ab(a + b) + bc(b + c) + ca(c + a)$. Wait, but that's similar to the earlier
 2264 expression. Alternatively, factor differently:
 2265 $= ab(a + b) + bc(b + c) + ca(c + a)$
 2266
 2267 But since $a + b + c = 300$, we can write $a + b = 300 - c$, etc. So:
 2268 $= ab(300 - c) + bc(300 - a) + ca(300 - b)$
 2269
 2270 Which expands to:
 2271 $300ab - abc + 300bc - abc + 300ca - abc$

2268
2269 $= 300(ab + bc + ca) - 3abc$
2270
2271 So the equation becomes:
2272
2273 $300S2 - 3S3 = 6,000,000$
2274
2275 Which simplifies to $100S2 - S3 = 2,000,000$, which matches what we had earlier.
2276 #####
2277 ##### Factoring the equation for further simplification #####
2278 So we have $100S2 - S3 = 2,000,000$, and $S1 = 300$. So $S3 = 100S2 - 2,000,000$.
2279
2280 But $S3 = abc$, so $abc = 100(ab + bc + ca) - 2,000,000$.
2281
2282 This seems like a complicated equation to solve for integers $a, b, c \geq 0$ with $a + b + c = 300$.
2283
2284 But perhaps there's a way to parameterize this. Let's consider that $abc = 100(ab + bc + ca) - 2,000,000$.
2285
2286 Rearranging:
2287
2288 $abc - 100ab - 100bc - 100ca = -2,000,000$
2289
2290 Add $10000(a + b + c)$ to both sides:
2291
2292 $abc - 100ab - 100bc - 100ca + 10000(a + b + c) = -2,000,000 + 10000 \cdot 300$
2293
2294 Compute the right-hand side:
2295
2296 $-2,000,000 + 3,000,000 = 1,000,000$
2297
2298 Now, the left-hand side can be factored:
2299
2300 $abc - 100ab - 100bc - 100ca + 10000a + 10000b + 10000c$
2301
2302 Factor terms:
2303
2304 $= abc + (-100ab - 100bc - 100ca) + 10000(a + b + c)$
2305
2306 Notice that this resembles $(a - 100)(b - 100)(c - 100)$ expanded. Let's check:
2307
2308 Expand $(a - 100)(b - 100)(c - 100)$:
2309
2310 $= abc - 100ab - 100ac - 100bc + 10,000a + 10,000b + 10,000c - 1,000,000$
2311
2312 Compare to our left-hand side:
2313
2314 Left-hand side after adding $10000(a + b + c)$:
2315
2316 $abc - 100ab - 100bc - 100ca + 10,000a + 10,000b + 10,000c$
2317
2318 So the difference between this and the expanded $(a - 100)(b - 100)(c - 100)$ is the constant term $-1,000,000$.
2319 Therefore:
2320
2321 $(a - 100)(b - 100)(c - 100) = abc - 100ab - 100ac - 100bc + 10,000a + 10,000b + 10,000c - 1,000,000$
2322
2323 But our left-hand side is:
2324
2325 $abc - 100ab - 100bc - 100ca + 10,000a + 10,000b + 10,000c = (a - 100)(b - 100)(c - 100) + 1,000,000$
2326
2327 Therefore, the equation becomes:
2328
2329 $(a - 100)(b - 100)(c - 100) + 1,000,000 = 1,000,000$
2330
2331 So:

2322
 2323 $(a-100)(b-100)(c-100) = 0$
 2324 #####
 2325
 2326 ##### Interpreting the factored equation and finding solutions #####
 2327 Ah! This simplifies things. Therefore $(a-100)(b-100)(c-100)$ must be zero. Which means at least one of
 2328 the factors must be zero. Therefore, at least one of a, b, c must be equal to 100.
 2329
 2330 So the solutions are all triples (a, b, c) where at least one of a, b , or c is 100, and the other two variables sum
 2331 to 200 (since $a + b + c = 300$).
 2332
 2333 Therefore, the problem reduces to finding all nonnegative integer triples (a, b, c) such that one of them is 100,
 2334 and the other two sum to 200. However, we also need to ensure that the original equation holds, but since we
 2335 derived that this condition is equivalent to the original equation, all such triples will satisfy both equations.
 2336 #####
 2337 ##### Counting the number of valid triples #####
 2338 Now, let's count the number of such triples.
 2339
 2340 Case 1: $a = 100$. Then $b + c = 200$. The number of nonnegative integer solutions for (b, c) is 201 (from $b=0$ to
 2341 $b=200$). However, since triples are ordered, each solution here corresponds to a unique triple $(100, b, c)$.
 2342
 2343 Case 2: $b = 100$. Similarly, $a + c = 200$. Number of solutions: 201.
 2344
 2345 Case 3: $c = 100$. Similarly, $a + b = 200$. Number of solutions: 201.
 2346
 2347 However, we have to subtract the overlaps where two variables are 100. For example, if $a = 100$ and $b = 100$,
 2348 then $c = 100$, which is counted in all three cases. There are three such overlapping triples: $(100, 100, 100)$,
 2349 $(100, 100, 100)$, etc., but since it's the same triple, it's only counted once in each case. Wait, actually, when we
 2350 count the three cases, the triple $(100, 100, 100)$ is included in all three cases. So total overlaps are:
 2351
 2352 Each triple where two variables are 100 is counted in two cases. For example, $(100, 100, 100)$ is counted in all
 2353 three cases. Wait, no, if $a = 100, b = 100$, then $c = 100$, so it's counted once in each case. Wait, no, when $a =$
 2354 $100, b$ can be 100, which gives $c = 100$. Similarly for the other cases. So the triple $(100, 100, 100)$ is counted
 2355 three times in total (once in each case). Similarly, any triple where exactly two variables are 100 would be
 2356 counted twice. Wait, but if two variables are 100, say $a = 100$ and $b = 100$, then $c = 100$, which is the same as
 2357 all three being 100. So there are no triples where exactly two variables are 100 because that forces the third to
 2358 also be 100. Therefore, the only overlap is the triple $(100, 100, 100)$, which is counted three times.
 2359
 2360 Therefore, using inclusion-exclusion:
 2361
 2362 Total triples $= 3 \cdot 201 - 3 \cdot 1 + 1 \cdot 1 = 603 - 3 + 1 = 601$.
 2363 #####
 2364
 2365 ##### Verifying the counting with a small example #####
 2366 But let's check with a small example. Suppose we have variables summing to 3 instead of 300, and we want to
 2367 count triples where one variable is 1. Then:
 2368
 2369 Case 1: $a=1, b+c=2 \Rightarrow (1,0,2), (1,1,1), (1,2,0)$
 2370
 2371 Case 2: $b=1, a+c=2 \Rightarrow (0,1,2), (1,1,1), (2,1,0)$
 2372
 2373 Case 3: $c=1, a+b=2 \Rightarrow (0,2,1), (1,1,1), (2,0,1)$
 2374
 2375 Total listed triples: $(1,0,2), (1,1,1), (1,2,0), (0,1,2), (2,1,0), (0,2,1), (2,0,1)$. That's 7 triples. Using inclusion-
 exclusion:
 2376
 2377 $3 \cdot 3 - 3 \cdot 1 + 1 = 9 - 3 + 1 = 7$. Correct. So the formula works.
 2378 #####
 2379
 2380 ##### Extract the final answer #####
 2381 Thus, the final answer is 601.
 2382 #####

MLR Output (high-level) - GPQA (Diamond)

Problem:

Q: 3-(hydroxymethyl)-5-(prop-1-en-2-yl)cyclohexan-1-one is treated with sodium hydride, followed by benzyl bromide, forming product 1. 1 is treated with p-toluenesulfonyl hydrazide in the presence of a catalytic amount of HCl, forming product 2. 2 is treated at low temperature with n-butyllithium, followed by aqueous ammonium chloride, forming product 3. 3 is stirred with Pd/C under a hydrogen atmosphere, forming product 4. What is the structure of product 4?

Choices:

- A. ((3-isopropylcyclohexyl)methoxy)methylbenzene
- B. (3-isopropylcyclohexyl)methanol
- C. N'-(3-(hydroxymethyl)-5-isopropylcyclohexyl)-4-methylbenzenesulfonohydrazide
- D. 3-((benzyloxy)methyl)-1-butyl-5-isopropylcyclohexan-1-ol

Solution:

Understanding the problem and outlining the approach

Need to determine the structure of product 4 in the reaction sequence.

Approach: break down each step carefully to understand how product 4 is formed.

#####

Interpreting the starting material's structure

The starting material is 3-(hydroxymethyl)-5-(prop-1-en-2-yl)cyclohexan-1-one.

- The cyclohexanone ring has a hydroxymethyl group at position 3.

- At position 5, there is a prop-1-en-2-yl group, which is $\text{CH}_2=\text{C}(\text{CH}_3)-$ (a methyl group attached to a double bond).

- The structure is a cyclohexanone ring with these two substituents at positions 3 and 5.

#####

Analyzing the first reaction: NaH and benzyl bromide

NaH deprotonates the hydroxymethyl group to form an alkoxide, which then reacts with benzyl bromide to form a benzyl ether. Product 1 is 3-(benzyloxymethyl)-5-(prop-1-en-2-yl)cyclohexan-1-one, where the hydroxymethyl group is converted to a benzyloxymethyl group.

#####

Analyzing the second reaction: p-toluenesulfonyl hydrazide and HCl

Product 1 reacts with p-toluenesulfonyl hydrazide (TsNHNH_2) and catalytic HCl to form a tosylhydrazone.

The ketone group (cyclohexanone) reacts with the hydrazide, losing water and forming a hydrazone. Product 2 is the tosylhydrazone of the ketone, with all other substituents (including prop-1-en-2-yl) unchanged. The structure is N'-(3-(benzyloxymethyl)-5-(prop-1-en-2-yl)cyclohexylidene)-4-methylbenzenesulfonohydrazide.

#####

Analyzing the third reaction: n-butyllithium and aqueous ammonium Product 2 is treated with n-

butyllithium and aqueous ammonium chloride (Shapiro reaction), converting the ketone-derived hydrazone into an alkene. The cyclohexanone ring becomes a cyclohexene ring with a double bond between C1 and C2.

Product 3 is a cyclohexene with a benzyloxymethyl group at C3 and an isopropenyl group at C5.

#####

Analyzing the fourth reaction: hydrogenation with Pd/C and H_2

- Hydrogenation with Pd/C and H_2 saturates all double bonds.

- The cyclohexene ring becomes a cyclohexane ring.

- The isopropenyl group ($\text{CH}_2=\text{C}(\text{CH}_3)-$) is reduced to an isopropyl group ($\text{CH}_2\text{CH}(\text{CH}_3)-$).

- Product 4: cyclohexane ring with a benzyloxymethyl group at C3 and an isopropyl group at C5.

- The ring is fully saturated; no other changes occur.

#####

Recapping the sequence and mapping substituent positions

Starting material: cyclohexan-1-one

Position 3: hydroxymethyl \rightarrow benzyloxymethyl (after NaH/benzyl bromide)

Position 1: ketone \rightarrow tosylhydrazone \rightarrow double bond (after Shapiro reaction)

Ring: cyclohexene (double bond between 1 and 2)

Position 5: prop-1-en-2-yl ($\text{CH}_2=\text{C}(\text{CH}_3)-$)

Final product: 3-(benzyloxymethyl)-5-isopropylcyclohexane

#####

Evaluating the effect of hydrogenolysis on the benzyl ether

The benzyloxymethyl group should remain after hydrogenation with Pd/C, as benzyl ethers are generally stable under these conditions. Therefore, product 4 should have a benzyloxymethyl group at position 3 and isopropyl at position 5 on cyclohexane. None of the options match this structure.

#####

Re-examining the possibility of benzyl ether cleavage under hydrogenation

- Ketone is at position 1 on the cyclohexane ring.
- Substituents at positions 3 (benzyloxymethyl) and 5 (isopropyl) are two positions apart.
- After Shapiro elimination, double bond forms between positions 1 and 2.
- Hydrogenation restores the single bond; substituents remain at positions 3 and 5.
- In product 4, cyclohexane ring has benzyloxymethyl at position 3 and isopropyl at position 5.
- This arrangement does not match any of the answer choices.

#####

Considering the effect of hydrogenolysis on the benzyl ether group

Step 1: alkylation to benzyloxymethyl—correct.

Step 2: tosylhydrazone formation at ketone—correct.

Step 3: Shapiro reaction forms alkene between positions 1 and 2—correct.

Step 4: hydrogenation saturates both the 1–2 alkene and isopropenyl group to isopropyl.

Final product: 3–(benzyloxymethyl)–5–isopropylcyclohexane.

None of the options exactly match this structure.

#####

Realizing benzyl ether cleavage under hydrogenation conditions

Starting material: 3–(hydroxymethyl)–5–(prop–1–en–2–yl)cyclohexan–1–one.

- Tosylhydrazone forms at position 1 (the ketone).
- Elimination forms an alkene between positions 1 and 2.
- Substituents at 3 (hydroxymethyl) and 5 (prop–1–en–2–yl) remain.
- Hydrogenation reduces the double bond (1,2) and the prop–1–en–2–yl group at 5.
- Final product: cyclohexane ring with substituents at 3 (hydroxymethyl) and 5 (isopropyl).
- Question: how to name this compound.

#####

Applying IUPAC nomenclature and lowest locant rule

After renumbering to give the substituents the lowest possible numbers, the hydroxymethyl group is at position 1 and the isopropyl at position 3. The correct IUPAC name is 1–(hydroxymethyl)–3–isopropylcyclohexane, also named as (3–isopropylcyclohexyl)methanol. Therefore, option B is correct.

#####

Final conclusion and answer selection

Numbering changes to give the lowest possible numbers, repositioning the substituents as 1 and 3 instead of 3 and 5.

Correct answer is B.

#####

Extract the final answer

Thus, the final answer is **B**.

#####

MLR Output (low-level) - GPQA (Diamond)

Problem:

Q: 3–(hydroxymethyl)–5–(prop–1–en–2–yl)cyclohexan–1–one is treated with sodium hydride, followed by benzyl bromide, forming product 1. 1 is treated with p–toluenesulfonyl hydrazide in the presence of a catalytic amount of HCl, forming product 2. 2 is treated at low temperature with n–butyllithium, followed by aqueous ammonium chloride, forming product 3. 3 is stirred with Pd/C under a hydrogen atmosphere, forming product 4. What is the structure of product 4?

Choices:

- A. (((3–isopropylcyclohexyl)methoxy)methyl)benzene
- B. (3–isopropylcyclohexyl)methanol

C. N'-(3-(hydroxymethyl)-5-isopropylcyclohexyl)-4-methylbenzenesulfonohydrazide
 D. 3-((benzyloxy)methyl)-1-butyl-5-isopropylcyclohexan-1-ol

Solution:

Understanding the problem and outlining the approach

Okay, so I need to figure out the structure of product 4 in this reaction sequence. Let me start by breaking down each step carefully.

#####

Interpreting the starting material's structure

First, the starting material is 3-(hydroxymethyl)-5-(prop-1-en-2-yl)cyclohexan-1-one. Let's draw that out.

The cyclohexanone ring has a hydroxymethyl group at position 3 and a prop-1-en-2-yl group (which is an allyl group attached at the middle carbon, so it's actually a methyl group attached to a double bond, right? So like CH₂-C(CH₃)= maybe? Wait, prop-1-en-2-yl would be CH₂=C(CH₃)-, right? So the substituent is a methyl group attached to a double bond. So the cyclohexanone ring has that at position 5 and the hydroxymethyl at position 3.

#####

Analyzing the first reaction: NaH and benzyl bromide

Then, the first step is treatment with sodium hydride, followed by benzyl bromide. Sodium hydride is a strong base, so maybe it deprotonates something. The hydroxymethyl group has an -OH. Deprotonating that would give an alkoxide, which is a good nucleophile. Then benzyl bromide would act as an alkylating agent. So the benzyl group (Bn) would be attached to the oxygen, making a benzyl ether. So product 1 would be 3-(benzyloxymethyl)-5-(prop-1-en-2-yl)cyclohexan-1-one. That makes sense because the hydroxymethyl becomes benzyloxymethyl after alkylation.

#####

Analyzing the second reaction: p-toluenesulfonyl hydrazide and HCl

Next, product 1 is treated with p-toluenesulfonyl hydrazide (TsNHNH₂) with catalytic HCl. That's typically used to form a tosylhydrazone. The ketone group (cyclohexanone) would react with the hydrazide. The mechanism is that the ketone attacks the hydrazide, losing water and forming a hydrazone. So product 2 would be the tosylhydrazone of the ketone. So the cyclohexanone becomes a cyclohexylidene group linked to the tosylhydrazide. So structure would be N'-(3-(benzyloxymethyl)-5-(prop-1-en-2-yl)cyclohexylidene)-4-methylbenzenesulfonohydrazide. Wait, but maybe the prop-1-en-2-yl is still present? The reaction is only on the ketone, so the other substituents remain.

#####

Analyzing the third reaction: n-butyllithium and aqueous ammonium chloride (Shapiro reaction)

#####

Third step: product 2 is treated with n-butyllithium at low temperature, then aqueous ammonium chloride. This sounds like the Shapiro reaction. The Shapiro reaction uses tosylhydrazones treated with a strong base like n-BuLi to form alkenes via elimination. The mechanism involves deprotonation and elimination of the tosylhydrazide group, forming a double bond. So the cyclohexanone ring, which had the hydrazone, would become a cyclohexene ring? Wait, but the starting material already has an allyl group (prop-1-en-2-yl). Wait, maybe the Shapiro reaction here is converting the ketone-derived hydrazone into an alkene. But the original ketone is at position 1 (cyclohexan-1-one). So after Shapiro, that would form a double bond between C1 and C2? So the cyclohexanone becomes cyclohexene with the substituents. So product 3 would be 3-(benzyloxymethyl)-5-(prop-1-en-2-yl)cyclohexene, but the position of the double bond would be between C1 and C2? Wait, maybe I need to think about regiochemistry here. The Shapiro reaction typically gives less substituted alkenes because it's a concerted elimination. But in the cyclohexane ring, the original ketone is at position 1. So when eliminating, the hydrazone is at C1, and the adjacent hydrogen is at C2. So elimination would form a double bond between C1 and C2. But the ring already has substituents at C3 (benzyloxymethyl) and C5 (prop-1-en-2-yl). So after elimination, the ring becomes a cyclohexene with double bond between C1 and C2. So product 3 is 1,2-cyclohexenyl with substituents at C3 and C5. But the prop-1-en-2-yl is another substituent. Wait, but maybe the prop-1-en-2-yl is an isopropenyl group (CH₂=C(CH₃)-). So product 3 would have a cyclohexene ring (double bond between 1 and 2), a benzyloxymethyl group at 3, and an isopropenyl group at 5. Then, when treated with n-BuLi, but after that, aqueous ammonium chloride. Wait, the Shapiro reaction usually forms alkenes via deprotonation and elimination. But maybe in this case, after forming the hydrazone, treatment with n-BuLi leads to deprotonation and elimination of the tosylhydrazide, giving the alkene. So product 3 would be the cyclohexene derivative with those substituents.

#####

Analyzing the fourth reaction: hydrogenation with Pd/C and H₂

Then, product 3 is stirred with Pd/C under hydrogen. Hydrogenation. So any double bonds would be hydrogenated. The cyclohexene ring's double bond (from the Shapiro reaction) would be hydrogenated to a single bond. Also, the prop-1-en-2-yl group (isopropenyl, $\text{CH}_2=\text{C}(\text{CH}_3)-$, so that's a vinyl group with a methyl substituent on the double bond) would be hydrogenated, making it a prop-2-yl group (isopropyl group). Wait, prop-1-en-2-yl is $\text{CH}_2=\text{C}(\text{CH}_3)-$, so hydrogenation would make $\text{CH}_2\text{CH}(\text{CH}_3)$. So the isopropenyl becomes isopropyl. Also, the cyclohexene from the Shapiro reaction (double bond between 1 and 2) would become cyclohexane again. So product 4 would have a cyclohexane ring with substituents: at C3, benzyloxymethyl; at C5, isopropyl. But wait, after hydrogenation, the ring is fully saturated. Also, the hydroxymethyl group at 3 is now a benzyloxymethyl. But after hydrogenation, is there any other change?

#####

Recapping the sequence and mapping substituent positions

Wait, let's recap: Starting material: cyclohexan-1-one (so position 1 is the ketone). Substituents: at 3, hydroxymethyl; at 5, prop-1-en-2-yl ($\text{CH}_2=\text{C}(\text{CH}_3)-$). Step 1: deprotonate hydroxymethyl's OH with NaH, then benzyl bromide gives benzyloxymethyl at position 3. Step 2: form tosylhydrazone at the ketone (position 1). So now, the hydrazone is at position 1. Step 3: Shapiro reaction ($n\text{-BuLi}$) eliminates the tosylhydrazide, forming a double bond between position 1 and 2. So the ring becomes cyclohexene (positions 1 and 2 double bond). So now, the substituents are at 3 (benzyloxymethyl), 5 ($\text{CH}_2=\text{C}(\text{CH}_3)-$), and the ring has a double bond between 1 and 2. Step 4: hydrogenation with Pd/C and H_2 . The double bonds in the molecule (both the cyclohexene ring's 1,2-double bond and the prop-1-en-2-yl group's double bond) would be hydrogenated. So the cyclohexene becomes cyclohexane (single bond between 1 and 2), and the prop-1-en-2-yl becomes isopropyl ($\text{CH}(\text{CH}_3)_2$). So the substituent at position 5 becomes isopropyl. So product 4 would be 3-(benzyloxymethyl)-5-isopropylcyclohexane. Wait, but cyclohexane is a six-membered ring. The substituents are at positions 3 and 5. But then, after hydrogenation, the ring is fully saturated. Also, the hydroxymethyl was at 3, which became benzyloxymethyl. But after hydrogenation, is there any other change?

#####

Evaluating the effect of hydrogenolysis on the benzyl ether

Wait, but in the final hydrogenation step, maybe the benzyloxymethyl group is hydrogenated? But benzyl ethers are generally stable under hydrogenation unless there's a catalyst that cleaves ethers. But Pd/C usually doesn't cleave ethers. So the benzyloxymethyl should remain. So product 4 should have a benzyloxymethyl group at position 3 and isopropyl at position 5 on cyclohexane. But none of the options have that.

#####

Re-examining the possibility of benzyl ether cleavage under hydrogenation

Wait, maybe I messed up the numbering. Let me try to visualize the cyclohexane ring. Let's imagine the ketone at position 1. Then positions 2, 3, 4, 5, 6 go around the ring. So if the ketone is at position 1, then substituents at 3 and 5 would be two positions apart. After elimination (Shapiro), the double bond is between 1 and 2. Then hydrogenation would bring back the single bond. So the substituents at 3 and 5 are in the same positions. So in product 4, the cyclohexane ring has a benzyloxymethyl group at position 3 and an isopropyl group at position 5. But the answer choices don't have that.

#####

Considering the effect of hydrogenolysis on the benzyl ether group

Wait, but let me double-check the steps again. Maybe I made a mistake. Step 1: alkylation of hydroxymethyl to benzyloxymethyl. Correct. Step 2: formation of tosylhydrazone at the ketone. Correct. Step 3: Shapiro reaction with $n\text{-BuLi}$. The Shapiro reaction involves deprotonation adjacent to the hydrazone, leading to elimination of the tosylhydrazide and formation of an alkene. So the ketone (position 1) becomes a double bond between positions 1 and 2. Step 4: hydrogenation of the alkene (positions 1-2) and the isopropenyl group (position 5) to isopropyl. So product 4 would have a cyclohexane ring with substituents: at position 3, benzyloxymethyl; at position 5, isopropyl. So the structure is 3-(benzyloxymethyl)-5-isopropylcyclohexane. But looking at the options, none of them exactly match.

#####

Realizing benzyl ether cleavage under hydrogenation conditions

Wait, maybe I messed up the positions during the Shapiro reaction. Let's think again. The starting material is 3-(hydroxymethyl)-5-(prop-1-en-2-yl)cyclohexan-1-one. When we form the tosylhydrazone at position 1 (the ketone), then eliminate to form the alkene between positions 1 and 2. Now, the ring has a double bond between 1 and 2. The substituents at 3 and 5 are still in their original positions. Then, hydrogenation reduces the double bond between 1 and 2 and the prop-1-en-2-yl group at 5. So after hydrogenation, the ring is cyclohexane with substituents at 3 (hydroxymethyl) and 5 (isopropyl). But how is this named?

#####

Applying IUPAC nomenclature and lowest locant rule

Alternatively, maybe the numbering changes when the ketone is removed. Originally, the ketone is at position 1. After converting it to CH₂ via hydrogenation, maybe the numbering starts from the hydroxymethyl group as position 1. In IUPAC nomenclature, the numbering is done to give the substituents the lowest possible numbers. So if the hydroxymethyl is at position 1 and the isopropyl at position 3, that would be lower than if they were at 3 and 5. So the new numbering would be: 1: hydroxymethyl (original 3) 2: CH₂ (original 4) 3: isopropyl (original 5) 4: CH₂ (original 6) 5: CH₂ (original 1) 6: CH₂ (original 2) Wait, this might not be accurate. Let me think of the cyclohexane as a chair. If the original positions are 1 (top), 2, 3, 4 (bottom), 5, 6. After removing the ketone (position 1 becomes CH₂), and having substituents at 3 and 5. If we renumber to give the lowest numbers, we need to choose a direction where the substituents are at lower positions. If we start numbering from the hydroxymethyl (original 3) as position 1, then the isopropyl is at original 5, which would be position 3 in the new numbering (since it's two positions away in the ring). This gives substituents at positions 1 and 3, which is lower than 3 and 5. Therefore, the correct IUPAC name would be 1-(hydroxymethyl)-3-isopropylcyclohexane, which is named as (3-isopropylcyclohexyl)methanol. Because the methanol group (hydroxymethyl) is considered as the substituent at position 1, and the isopropyl at position 3. Thus, option B is correct.

#####

Final conclusion and answer selection

Ah, so the key here is that the numbering changes to give the lowest possible numbers, which would reposition the substituents as 1 and 3 instead of 3 and 5. Therefore, the correct answer is B.

#####

Extract the final answer

Thus, the final answer is B.

#####

MLR Output (high-level) - BoardGameQA (Hard)

Problem:

A few players are playing a boardgame. The current state of the game is as follows. The dachshund has a football with a radius of 15 inches. And the rules of the game are as follows. Rule1: If the dachshund has a football that fits in a 36.6 x 33.1 x 40.7 inches box, then the dachshund brings an oil tank for the worm. Rule2: If at least one animal brings an oil tank for the worm, then the stork does not dance with the butterfly. Based on the game state and the rules and preferences, does the stork dance with the butterfly?

Solution:

Recap the problem and rules

The dachshund has a football with a radius of 15 inches.

The football fits in a 36.6 x 33.1 x 40.7 inches box.

#####

Identify the key condition to check

Key condition: If the dachshund's football fits in the box, Rule 1 applies (dachshund brings oil tank), triggering Rule 2 (stork can't dance).

If the football doesn't fit, Rule 1 doesn't apply (no oil tank), so Rule 2 might not be triggered, allowing the stork to dance.

#####

Analyze the football's dimensions and assumptions

The football has a radius of 15 inches.

Assuming it is a sphere, the diameter is 30 inches.

The problem uses ``radius'', so the football is treated as a sphere.

#####

Compare the football's size to the box

Diameter of football: 30 inches.

Box dimensions: 36.6 x 33.1 x 40.7 inches.

30 < 36.6 (length), 30 < 33.1 (width), 30 < 40.7 (height).

The football fits in the box in all dimensions.

#####

Apply Rule 1 based on the fit

Diameter (30 inches) is less than all box dimensions, so the football fits.

By Rule 1, the dachshund brings an oil tank for the worm.

#####

Apply Rule 2 and conclude

Applying Rule 2: since at least one animal (the dachshund) brings an oil tank, the stork does not dance with the butterfly.

#####

Extract the final answer

Thus, the final answer is disproved.

#####

MLR Output (low-level) - BoardGameQA (Hard)

Problem:

A few players are playing a boardgame. The current state of the game is as follows. The dachshund has a football with a radius of 15 inches. And the rules of the game are as follows. Rule1: If the dachshund has a football that fits in a 36.6 x 33.1 x 40.7 inches box, then the dachshund brings an oil tank for the worm. Rule2: If at least one animal brings an oil tank for the worm, then the stork does not dance with the butterfly. Based on the game state and the rules and preferences, does the stork dance with the butterfly?

Solution:

Recap the problem and rules

Okay, let's try to figure out whether the stork dances with the butterfly based on the given rules and the current game state.

First, let's recap the information provided. The dachshund has a football with a radius of 15 inches. The rules are:

If the dachshund's football fits in a 36.6 x 33.1 x 40.7 inches box, then the dachshund brings an oil tank for the worm.

If at least one animal brings an oil tank for the worm, then the stork does not dance with the butterfly.

The question is whether the stork dances with the butterfly. To determine that, we need to check if any animal (in this case, possibly the dachshund) brings an oil tank for the worm. If they do, then by Rule 2, the stork won't dance. If not, then maybe the stork does dance.

#####

Identify the key condition to check

So, the key here is whether the dachshund's football fits into the specified box. If it does, then Rule 1 applies, and the dachshund brings the oil tank, which triggers Rule 2, preventing the stork from dancing. If the football doesn't fit, then Rule 1 doesn't apply, so the dachshund doesn't bring the oil tank, and Rule 2 might not be triggered, allowing the stork to dance.

#####

Analyze the football's dimensions and assumptions

Now, let's focus on the football's dimensions. The football has a radius of 15 inches. Wait, a football is typically an elongated shape, like a prolate spheroid, but maybe in this game, it's considered a sphere? The problem mentions a radius, which is a term used for spheres. If it's a sphere with radius 15 inches, then the diameter would be 30 inches (since diameter is twice the radius). However, if it's an American football shape, the dimensions would be different. But since the problem specifies a radius, I think we can assume it's a sphere here.

#####

Compare the football's size to the box

So, diameter of the football is 30 inches. Now, the box has dimensions 36.6 x 33.1 x 40.7 inches. To fit the sphere inside the box, the sphere must fit in all three dimensions. That is, the diameter of the sphere must be less than or equal to each of the box's dimensions.

Checking each dimension:

Length: 36.6 inches. The diameter is 30, which is less than 36.6. So okay.

Width: 33.1 inches. 30 is less than 33.1. Okay.

Height: 40.7 inches. 30 is less than 40.7. Okay.

2700	#####
2701	
2702	##### Apply Rule 1 based on the fit #####
2703	Since the diameter (30 inches) is less than all the box's dimensions, the football should fit inside the box.
2704	Therefore, according to Rule 1, the dachshund brings an oil tank for the worm.
2705	#####
2706	##### Apply Rule 2 and conclude #####
2707	Then, applying Rule 2: since at least one animal (the dachshund) brings an oil tank, the stork does not dance
2708	with the butterfly.
2709	
2710	Therefore, the answer is that the stork does not dance with the butterfly.
2711	#####
2712	##### Extract the final answer #####
2713	Thus, the final answer is disproved .
2714	#####
2715	
2716	
2717	Long-horizon Reasoning Test Prompt Template
2718	You are given multiple problems:
2719	
2720	{Problem 1}
2721	
2722	{Problem 2}
2723	
2724	...
2725	{Problem N}
2726	
2727	Solve these problems in order. Please reason step by step for each problem, and at the end provide all final
2728	answers in a single JSON list**, where each answer is wrapped in <code>\boxed{}</code> . If you cannot solve a problem
2729	, write <code>\boxed{N/A}</code> for that entry.
2730	Output format:
2731	<code>\\json</code>
2732	<code>[\"\\boxed{answer1}\", \"\\boxed{answer2}\", ..., \"\\boxed{answerN}\"]</code>
2733	<code>\\</code>
2734	
2735	MLR (high-level only) Output
2736	
2737	Problem:
2738	Define
2739	$p = \sum_{k=1}^{\infty} \frac{1}{k^2}$ and $q = \sum_{k=1}^{\infty} \frac{1}{k^3}$. Find a way to write
2740	$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$ in terms of p and q .
2741	
2742	Solution:
2743	##### Understanding the problem #####
2744	The goal is to express
2745	$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$
2746	$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$
2747	in terms of p and q . The current stage involves understanding the problem and identifying the key
2748	components to work with.
2749	#####
2750	##### Recall the definitions of p and q #####
2751	$p = \sum_{k=1}^{\infty} \frac{1}{k^2}$ and $q = \sum_{k=1}^{\infty} \frac{1}{k^3}$ are defined as
2752	the sums of the reciprocals of squares and cubes, respectively.
2753	#####

Identify the desired expression #####
The desired expression is:

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4}$$
This is the sum of the reciprocals of the fourth powers of the sums of (j) and (k) , from $(j = 1)$ to $(j = \infty)$ and from $(k = 1)$ to $(k = \infty)$.

Consider symmetry and possible simplifications #####
The expression $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4}$ involves summing the reciprocals of the fourth powers of the sums of (j) and (k) . The symmetry between (j) and (k) suggests potential simplifications or transformations. The current stage is focusing on identifying the key components and considering symmetry to simplify the expression.

Attempt to express the sum in terms of (p) and (q) #####
The expression $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4}$ involves summing the reciprocals of the fourth powers of the sums of (j) and (k) . The current stage is attempting to express this sum in terms of (p) and (q) , which are defined as $\sum_{k=1}^{\infty} \frac{1}{k^2}$ and $\sum_{k=1}^{\infty} \frac{1}{k^3}$, respectively. The goal is to find a relationship between these sums and the given expression.

Recall known results for symmetric sums #####
The expression $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4}$ involves summing the reciprocals of the fourth powers of the sums of (j) and (k) . The goal is to express this sum in terms of (p) and (q) , where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$. The current stage is exploring known results for symmetric sums and potential simplifications.

Attempt to simplify using known results #####
The expression $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4}$ involves summing the reciprocals of the fourth powers of the sums of (j) and (k) . The current stage is attempting to simplify this expression using known results for symmetric sums, focusing on possible relationships with (p) and (q) .

Consider using known results for symmetric sums #####
The expression $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4}$ is being considered in terms of (p) and (q) , where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$. The goal is to find a relationship between (p) and (q) and the given sum.

Recall the identity for the sum of reciprocals of squares #####
The identity for the sum of reciprocals of squares is:

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = p$$
where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$.

Recall the identity for the sum of reciprocals of cubes #####
The identity for the sum of reciprocals of cubes is:

$$\sum_{k=1}^{\infty} \frac{1}{k^3} = q$$
where $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$.

Recall the identity for the sum of reciprocals of fourth powers #####
The identity for the sum of reciprocals of fourth powers is:

$$\sum_{k=1}^{\infty} \frac{1}{k^4} = \frac{p^2}{2} + \frac{1}{4}q$$

where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$.
#####

Apply the identity to the given sum #####
The given sum $(\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4})$ can be expressed in terms of (p) and (q) using the identity:

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4} = \frac{p}{2} + \frac{1}{4}q$$
#####

Verify the result with a specific example #####
The given sum $(\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4})$ can be expressed in terms of (p) and (q) using the identity:

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4} = \frac{p}{2} + \frac{1}{4}q$$
where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$.
#####

Alternative approach using generating functions #####
The sum $(\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4})$ can be expressed in terms of (p) and (q) as:

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4} = \frac{p}{2} + \frac{1}{4}q$$
where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$.
#####

Review the result for potential simplifications #####
The sum $(\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4})$ can be expressed in terms of (p) and (q) as:

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4} = \frac{p}{2} + \frac{1}{4}q$$
where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$.
#####

Finalize the answer #####
The sum $(\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4})$ can be expressed in terms of (p) and (q) as:

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^4} = \frac{p}{2} + \frac{1}{4}q$$
where $(p = \sum_{k=1}^{\infty} \frac{1}{k^2})$ and $(q = \sum_{k=1}^{\infty} \frac{1}{k^3})$.
#####

Thus, the final answer is $\frac{1}{2}p + \frac{1}{4}q$.
#####

Plan-and-Solve Prompt

Q: Let x, y and z be positive real numbers that satisfy the following system of equations:

$$\log_2 \left(\frac{x}{yz} \right) = \frac{1}{2}$$

$$\log_2 \left(\frac{y}{xz} \right) = \frac{1}{3}$$

$$\log_2 \left(\frac{z}{xy} \right) = \frac{1}{4}$$

Then the value of $|\log_2(x^4 y^3 z^2)|$ is $\frac{m}{n}$ where m and n are relatively prime positive integers. Find $m + n$.

A: Let's first understand the problem, extract relevant variables and their corresponding numerals, and make a complete plan. Then, let's carry out the plan, calculate intermediate variables (pay attention to correct numerical calculation and commonsense), solve the problem step by step, and show the answer.

Output:

Given:

$$\log_2 \left(\frac{x}{yz} \right) = \frac{1}{2}$$

$$\log_2 \left(\frac{y}{xz} \right) = \frac{1}{3}$$

$$\log_2 \left(\frac{z}{xy} \right) = \frac{1}{4}$$

The value of $|\log_2(x^4 y^3 z^2)|$ is $\frac{m}{n}$ where m and n are relatively prime positive integers.

Plan:

1. Convert each logarithmic equation to its exponential form.
2. Use properties of exponents and logarithms to simplify the expressions.
3. Combine like terms and solve for the values of x , y , and z .
4. Substitute these values into the expression $|\log_2(x^4 y^3 z^2)|$.
5. Simplify this expression to find $|\log_2(x^4 y^3 z^2)|$.
6. Express it as a fraction $\frac{m}{n}$ with m and n being relatively prime.
7. Find $m + n$.

Calculation:

Step 1: Convert each logarithmic equation to its exponential form

From the given equations, we have:

$$x = 2^{1/2} yz$$

$$y = 2^{1/3} xz$$

$$z = 2^{1/4} xy$$

Step 2: Use properties of exponents and logarithms to simplify the expressions

We can rewrite each equation using the property of exponents:

$$x = \sqrt{2} yz$$

$$y = \sqrt[3]{2} xz$$

$$z = \sqrt[4]{2} xy$$

Step 3: Solve for the values of x , y , and z

Multiply all three equations together:

$$(\sqrt{2} yz)(\sqrt[3]{2} xz)(\sqrt[4]{2} xy) = (\sqrt{2})(\sqrt[3]{2})(\sqrt[4]{2})xyz^2$$

This simplifies to:

$$x^2 y^2 z^2 = \sqrt{2}(\sqrt[3]{2})(\sqrt[4]{2})$$

Since $(\sqrt{2})(\sqrt[3]{2})(\sqrt[4]{2}) = 2^{(1/2+1/3+1/4)} = 2^{(6+4+3)/12} = 2^{13/12}$, we get:

$$x^2 y^2 z^2 = 2^{13/12}$$

Taking the square root on both sides:

$$xyz = 2^{13/24}$$

Now, substitute back to find individual values:

$$x = \sqrt{2} yz$$

$$y = \sqrt[3]{2} xz$$

$$z = \sqrt[4]{2} xy$$

Step 4: Substitute these values into the expression $|\log_2(x^4 y^3 z^2)|$

Substitute $x = \sqrt{2} yz$ into $x^4 y^3 z^2$:

$$x^4 y^3 z^2 = (\sqrt{2} yz)^4 (yz)^3 (z)^2 = 2^2 (y^4 z^4) y^3 z^2 = 4(y^4 z^4) y^3 z^2 = 4(y^7 z^6)$$

So,

2916 $|x^4 y^3 z^2| = 4(y^7 z^6)$
 2917
 2918 Using the earlier result $xyz = 2^{13/24}$, we know:
 2919 $y^7 z^6 = \left(\frac{2^{13/24}}{x}\right)^7 \cdot \left(\frac{2^{13/24}}{y}\right)^6 = \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6$
 2920
 2921 Simplifying further:
 2922 $= \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6 = \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6 = \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6$
 2923
 2924 Finally:
 2925 $|x^4 y^3 z^2| = 4(y^7 z^6) = 4 \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6 = 4 \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6 = 4 \left(\frac{2^{13/24}}{\sqrt{2}yz}\right)^7 \cdot \left(\frac{2^{13/24}}{\sqrt[3]{2}xz}\right)^6$
 2926
 2927 Therefore:
 2928 $|\log_2(x^4 y^3 z^2)| = \log_2(4) = 2$
 2929
 2930 Thus:
 2931 $|\log_2(x^4 y^3 z^2)| = \frac{2}{1}$
 2932
 2933 ### Answer:
 2934 $m = 2, n = 1$
 2935 $m + n = 2 + 1 = 3$
 2936
 2937 The final answer is 3.
 2938
 2939
 2940
 2941
 2942
 2943
 2944
 2945
 2946
 2947
 2948
 2949
 2950
 2951
 2952
 2953
 2954
 2955
 2956
 2957
 2958
 2959
 2960
 2961
 2962
 2963
 2964
 2965
 2966
 2967
 2968
 2969