PROBEPO: SPARSE PROBES, REDISTRIBUTED CREDIT FOR CRITIC-FREE LLM REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from verifiable rewards assigns one terminal reward to an entire reasoning trajectory, obscuring how individual reasoning steps contributed to the outcome. GRPO broadcasts a group-normalized advantage to every response token, while finer-grained approaches often rely on dense auxiliary rollouts or branching schemes that can also alter the data used for policy updates. We cast the missing middle as *budgeted measurement*: with a limited rollout budget, which partial solutions should be evaluated, and how should their value estimates refine credit? ProbePO selects partial-solution states with a reward-free uncertainty score and estimates their expected verifier rewards under a capped continuation budget. ProbePO redistributes, rather than replaces, GRPO's trajectory-level advantage across tokens while preserving each trajectory's mean token-level advantage; probe continuations serve only as measurements. Across three DeepSeekMath and Qwen-family models spanning 1.5B–7B parameters, ProbePO consistently outperforms GRPO and achieves the highest accuracy among the four evaluated methods in 16 of 18 model–benchmark comparisons. Against dense-probing VinePPO on DeepSeekMath-7B, ProbePO achieves higher accuracy on all six benchmarks with 76% less end-to-end training time.

1 INTRODUCTION

Reinforcement learning from verifiable rewards (RLVR) has become a central approach for improving the mathematical and coding reasoning of large language models (Shao et al., 2024; DeepSeek-AI et al., 2025). Its supervision is nevertheless coarse: a verifier evaluates the completed response, leaving the contribution of intermediate reasoning implicit. Group Relative Policy Optimization (GRPO) (Shao et al., 2024) assigns each completion a group-normalized advantage and broadcasts it to every response token. This is simple and effective, but ignores the different roles tokens play within a trajectory. Under a positive trajectory-level advantage, an early detour in the reasoning trajectory and the later step that repairs it are reinforced equally; under a negative advantage, sound progress is penalized together with the mistake that invalidates it.

A response to a difficult prompt typically unfolds over multiple reasoning steps, creating natural intermediate points at which progress can be assessed. Intermediate value measurements at these reasoning boundaries offer a direct way to refine this coarse allocation. At such a boundary, we hold the generated prefix fixed, sample several possible suffixes from the current policy, and score the resulting completed responses with the verifier. We call each sampled suffix (from the boundary through a final answer) a *probe continuation*; averaging their rewards estimates the partial solution's expected terminal verifier reward under the current policy. Differences between value estimates at successive boundaries indicate whether expected terminal verifier reward rose or fell across the intervening segment. Evidence suggests that this expectation changes little across many reasoning steps, while a smaller number change it sharply (Kazemnejad et al., 2025). Because every probe continuation requires additional generation, measuring every eligible boundary can be costly. A fixed continuation budget therefore creates a measurement-allocation problem:

Which partial solutions are worth measuring, and how should those measurements redistribute credit along the original trajectory?

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

We present ProbePO (*probe policy optimization*), a budgeted measurement layer for GRPO (Figure 1). A lightweight, reward-free selector ranks reasoning-step boundaries using uncertainty signals. A capped auxiliary-rollout budget is then spent on a small number of highly ranked boundaries. At each selected boundary, the method samples completions directly from the rollout policy and uses verifier outcomes to estimate the chance that the partial solution will succeed. Continuations sampled from a pivot estimate its expected verifier reward and serve exclusively as measurements, allowing ProbePO to refine credit. To improve the precision of each sparse estimate, ProbePO regularizes it toward the group-level baseline already supplied by GRPO. Changes between successive measured states become segment-wise adjustments. Rather than replacing GRPO's trajectory-level advantage, ProbePO adds these adjustments as a zero-average redistribution across the response. This preserves each trajectory's mean token-level advantage at its GRPO value while allowing different reasoning segments to receive different emphasis (Proposition 1).

Contributions.

1. **Budgeted credit measurement.** We formulate intermediate credit assignment as allocating a finite continuation budget across partial-solution states, separating where to measure from how sparse measurements are converted into credit.
2. **GRPO-anchored redistribution.** We introduce ProbePO, combining reward-free pivot selection, regularized continuation-based value estimates, and a centered segment-wise correction. The correction exactly preserves each trajectory's mean token-level advantage at its GRPO value and vanishes when no pivot is selected.
3. **Accuracy and efficiency.** Across three models and six benchmarks, ProbePO consistently outperforms GRPO. Controlled studies support both structured placement and shrinkage. At matched training time, ProbePO remains more accurate than GRPO; against dense-probing VinePPO, it achieves higher accuracy on all six benchmarks with 76% less wall-clock training time.

2 PRELIMINARIES

Setup and GRPO. A policy π_θ generates a continuation $y = (y_1, \ldots, y_T)$ for a prompt q; a verifier returns an undiscounted terminal reward $R \in [0, 1]$. Let $s_t := (q, y_{1:t})$ denote the partial-solution state after t generated tokens, with $s_0 = q$. Training optimizes a clipped PPO surrogate (Schulman et al., 2017) over per-token advantages $A_{i,t}$. We use *advantage* for the action-value contrast that weights a policy-gradient update, *credit* for its allocation across tokens, and *residual* for the zero-mean correction introduced in Section 3.2; a process reward is an intermediate reward signal rather than a synonym for advantage. For each prompt, GRPO (Shao et al., 2024) samples a group of G continuations with rewards $r_1, \ldots, r_G$ and assigns continuation i the group-normalized advantage

$$A_i = \frac{r_i - \mu_g}{\sigma_g}, \qquad \mu_g = \tfrac{1}{G}\sum_j r_j, \quad \sigma_g = \mathrm{std}(\{r_j\}) + \varepsilon, \tag{1}$$

broadcast to every token: $A_{i,t} = A_i$ for all t. The group mean μ_g provides GRPO's prompt-level endpoint anchor; the standardized return contrast does not identify where within the continuation that contrast arose.

Token-level credit as value increments. For a fixed policy, let $(\mathcal{F}_t)_{t=0}^T$ be the filtration generated by the prompt and successive tokens, so $\mathcal{F}_t$ contains the information in $s_t = (q, y_{1:t})$. Define $V_t^\pi := \mathbb{E}_\pi[R \mid \mathcal{F}_t]$. By the tower property, $\{V_t^\pi\}_{t=0}^T$ is a Doob martingale, with $V_0^\pi = \mathbb{E}_\pi[R \mid q]$ and $V_T^\pi = R$ when the completed response determines the verifier score. Because appending token y_t deterministically maps s_{t-1} to s_t and reward is terminal, $Q^\pi(s_{t-1}, y_t) = V_t^\pi$. The token-level action advantage, denoted $A^\pi(s_{t-1}, y_t)$, is therefore

$$A^\pi(s_{t-1}, y_t) = V_t^\pi - V_{t-1}^\pi, \qquad \sum_{t=1}^{T} A^\pi(s_{t-1}, y_t) = R - V_0^\pi. \tag{2}$$

GRPO does not estimate these tokenwise increments. Instead, it assigns the trajectory-level advantage $A_i = (r_i - \mu_g)/\sigma_g$ to every response token: $A_{i,t} = A_i$. VinePPO takes the dense alternative,

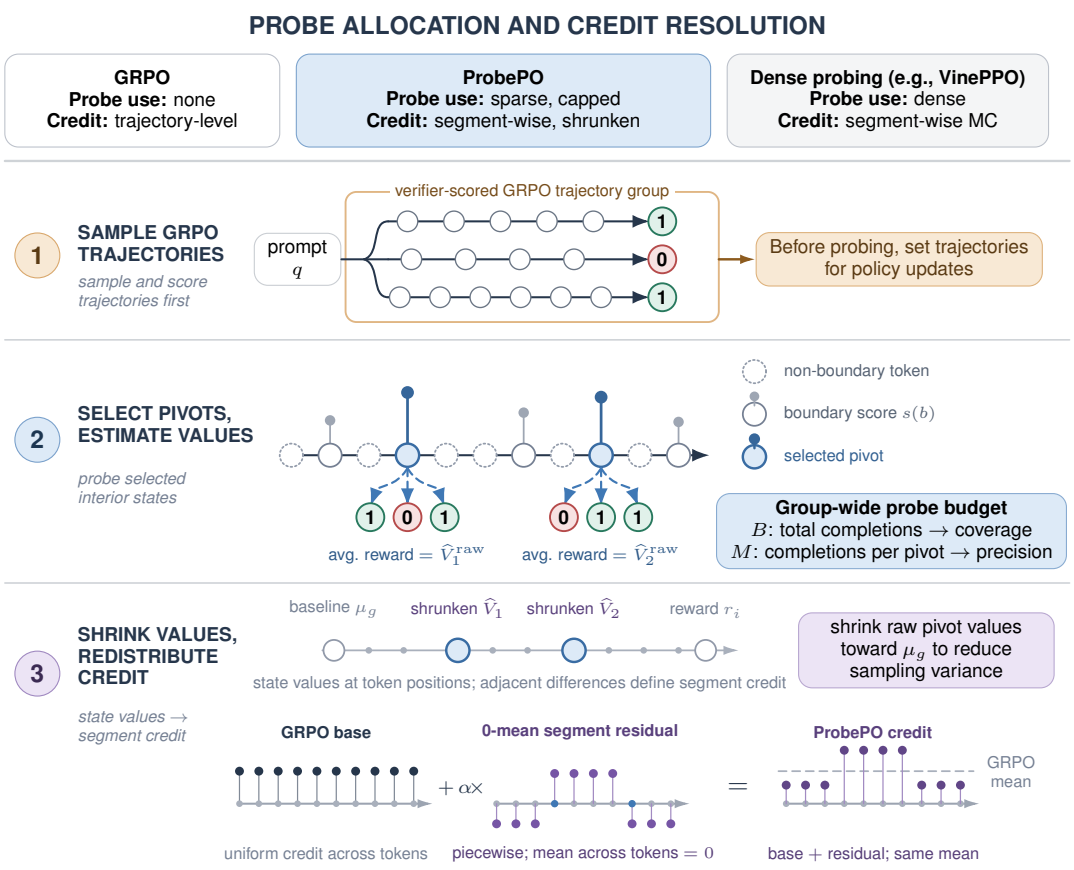

Figure 1: **ProbePO improves cost-quality tradeoff by assigning within-trajectory credit with a sparse-probe budget.** GRPO incurs no auxiliary-probe overhead but provides only trajectory-level credit. Dense per-boundary probing, exemplified by VinePPO, provides fine resolution at high rollout cost, while finite-sample Monte Carlo differences remain noise-sensitive. ProbePO instead caps the auxiliary-probe budget, concentrates it on selected pivots, and shrinks their value estimates before deriving segment-wise credit. The pipeline first fixes the policy-update trajectories, then ranks and probes candidate boundaries under the group-wide budget, and finally adds a token-mean-zero segment residual to the GRPO base advantage.

estimating values at every eligible reasoning-step boundary and differencing adjacent estimates to construct step-level credit (Kazemnejad et al., 2025).

What should a placement signal track? For a state s and next-token action $a \sim \pi$, let $Q^\pi(s,a) = \mathbb{E}_\pi[R \mid s,a]$. The law of total variance gives

$$\mathrm{Var}(R \mid s) = \underbrace{\mathbb{E}_a[\mathrm{Var}(R \mid s,a)]}_{\text{continuation variance}} + \underbrace{\mathrm{Var}_a(Q^\pi(s,a))}_{\text{action-fork variance}}. \tag{3}$$

The action-fork term measures how strongly the immediate choice can change expected terminal reward and, under the deterministic token transition above, equals the conditional second moment of the token action advantage. By contrast, total conditional variance controls the sampling variance of an M-continuation value estimate, $\mathrm{Var}(\hat{V}_{\text{raw}}(s) \mid s) = \mathrm{Var}(R \mid s)/M$. Offline, we approximate action-fork variance by sampling several next-token actions and estimating each action's mean return from repeated continuations. We evaluate a selector using its rank correlation with this proxy and, under a one-boundary budget, the share of a trajectory's total proxy variance captured by its top-ranked boundary.

3 PROBEPO

ProbePO has an acquisition layer that selects boundaries and allocates the rollout budget in fixed-M chunks (Section 3.1), and an estimation layer that converts the resulting sparse value estimates into token-level advantages (Section 3.2). A selected boundary is a *pivot*, and a verifier-scored suffix generated from it is a *probe continuation*. Probe continuations serve only as measurements; the actor batch remains fixed. Figure 1 illustrates the pipeline, and Algorithm 1 states one training iteration.

3.1 THE REWARD-FREE SELECTOR

Candidate boundaries. Within each trajectory we detect candidate boundaries deterministically at reasoning-step delimiters (e.g., paragraph breaks, enumerated step markers), discard any terminal boundary $t_b = T$, and merge boundaries closer than a minimum token gap.

Why reward-free? The ideal placement statistic would rank boundaries by how strongly the next action can change eventual verifier return, captured by the action-fork variance $\mathrm{Var}_{a\sim\pi_\theta}(Q^{\pi_\theta}(s_b, a))$. Estimating that quantity online would itself require verifier-scored continuations at every candidate, spending the measurement budget before any pivot is selected. We therefore rank candidate boundaries using the reward-free forward-window maximum entropy score $H_{\max}$, computed from backbone logits without auxiliary rollouts or reward queries, and evaluate its empirical agreement with action-fork variance in Section 5.6.

Score. For a backbone state after token t, define its full-vocabulary next-token entropy H_t; each candidate boundary b at position t_b receives

$$\begin{aligned} H_t &= -\sum_{v\in\mathcal{V}} \pi_\theta(v \mid q, y_{1:t}) \log \pi_\theta(v \mid q, y_{1:t}), \\ H_{\max}(b) &= \max_{\substack{0\le\ell<L \\ t_b+\ell<T}} H_{t_b+\ell}, \\ s(b) &= H_{\max}(b)\, w_{\mathrm{pos}}(t_b/T). \end{aligned} \tag{4}$$

Here $L = 8$ in every reported arm, and the maximum is evaluated on the realized backbone tokens. The optional factor $w_{\mathrm{pos}}(u)$ can encode a preference over normalized boundary position $u = t_b/T$; all reported experiments use $w_{\mathrm{pos}}(u) \equiv 1$, so this factor does not affect the ranking. The forward maximum reduces sensitivity to exact boundary alignment by allowing a nearby high-entropy state to represent the boundary. Because it uses logits produced during backbone generation, scoring requires no additional rollouts. The top fraction of candidates per trajectory forms the shortlist for group-pooled allocation.

Group-pooled budget. We define budget B as the number of probe continuations across all G trajectories. We rank shortlisted boundaries jointly and select $\lfloor B/M \rfloor$ pivots. Each pivot is evaluated with M independent continuations. Pooling is deliberate: a trajectory rich in high-ranked boundaries may receive several pivots while a flat one receives none.

Probing. At each selected pivot τ in trajectory i we freeze the prefix $y_{i,1:\tau}$, sample M probe continuations to completion from the same current policy π_θ that generated the backbone batch, score each with the same verifier that produces training rewards, and set $\hat{V}_{\mathrm{raw}}(\tau) = \frac{1}{M}\sum_m \tilde{r}_m$. The probe mean estimates the current policy's value $V^{\pi_\theta}(s_\tau)$ under the training verifier, with standard error at most $0.5/\sqrt{M}$ for conditionally independent rewards in $[0, 1]$, provided sampling matches π_θ, continuations terminate, and the verifier score is recoverable from the generated suffix (Appendix C). Probe continuations are used only to estimate values and are excluded from the policy-update batch. These raw means and their error scale are the inputs the estimation layer consumes.

3.2 GRPO-ANCHORED CREDIT REDISTRIBUTION

Consider trajectory i from group g, with length T_i, reward r_i, group statistics μ_g, σ_g, and $K \ge 0$ selected pivots at positions $0 < \tau_1 < \cdots < \tau_K < T_i$. Let $A_{\mathrm{base},i}$ denote the trajectory-level base

advantage assigned by the configured GRPO rule; under Eq. (1), $A_{\text{base},i} = (r_i - \mu_g)/\sigma_g$. We add endpoint anchors $\tau_0 = 0$ and $\tau_{K+1} = T_i$, with values $\hat{V}(\tau_0) = \mu_g$ and $\hat{V}(\tau_{K+1}) = r_i$.

Shrinkage. Each interior value averages M probe returns. We regularize this estimate toward the group baseline using linear pseudo-count shrinkage:

$$\hat{V}(\tau_k) = \frac{M\,\hat{V}_{\text{raw}}(\tau_k) + n_0\,\mu_g}{M + n_0}, \qquad k = 1, \ldots, K. \tag{5}$$

Here $n_0 \geq 0$ controls the shrinkage strength, and $M/(M+n_0)$ is the weight on the raw estimate. For binary verifier rewards, Eq. (5) is the posterior mean under a standard Beta–Bernoulli conjugate model (Gelman et al., 2013, Ch. 2); Appendix B gives this interpretation and the estimator's conditional bias–variance decomposition.

Segment-wise redistribution. The pivots partition the response into segments $I_k = \{t : \tau_{k-1} < t \leq \tau_k\}$ with lengths $\ell_k = \tau_k - \tau_{k-1}$. For each segment, define the value increment

$$d_k = \hat{V}(\tau_k) - \hat{V}(\tau_{k-1}), \qquad k = 1, \ldots, K+1.$$

The endpoint anchors imply $\sum_{k=1}^{K+1} d_k = r_i - \mu_g$. When $K = 1$, the two increments measure the change from the group baseline to the pivot and from the pivot to the observed return, respectively.

Let $k(t)$ denote the segment containing token t. We broadcast each segment increment to its tokens as the raw token-level advantage $A^{\text{raw}}_{i,t} = d_{k(t)}/\sigma_g$, then subtract its per-trajectory token mean:

$$\tilde{A}_{i,t} = A^{\text{raw}}_{i,t} - \frac{1}{T_i}\sum_{s=1}^{T_i} A^{\text{raw}}_{i,s}, \qquad \sum_{t=1}^{T_i} \tilde{A}_{i,t} = 0. \tag{6}$$

The final token-level advantage is

$$\boxed{A_{i,t} = A_{\text{base},i} + \alpha\,\tilde{A}_{i,t}}, \tag{7}$$

where α controls the magnitude of the segment-wise redistribution. The GRPO base advantage determines the token mean of $A_{i,t}$, while the centered residual redistributes credit among the trajectory's segments.

Proposition 1 (Token-mean preservation and recovery at zero pivots)**.** *With $A_{\text{base},i}$ set to the GRPO advantage in Eq.* (1)*, for every trajectory, pivot set, collection of interior estimates, and value of α,*

$$\frac{1}{T_i}\sum_{t=1}^{T_i} A_{i,t} = \frac{r_i - \mu_g}{\sigma_g}.$$

If $K = 0$, then $A_{i,t} = (r_i - \mu_g)/\sigma_g$ for every response token.

The centered residual preserves the trajectory's GRPO token-mean advantage while redistributing its token-level policy-gradient weights. Appendix B gives the proof and the matched conditions under which $K = 0$ recovers the GRPO update.

Algorithm 1 ProbePO training step

Input: prompts and current policy π_θ
Parameters: group size G, probe budget B, continuations per pivot M, shrinkage n_0, and residual weight α
For each prompt: sample G backbone completions from π_θ; verify $\rightarrow$ rewards r_i
for each group g **do**
    Compute (μ_g, σ_g) and the configured base advantages $A_{\text{base},i}$
    for each trajectory i in group g **do**
        Detect candidate boundaries, score them by Eq. (4), and retain the top fraction
    end for
    Rank shortlisted candidates and select $\lfloor B/M \rfloor$ pivots.
    for each selected pivot (i, τ) **do**
        Generate M fresh probe continuations from π_θ; verify and average rewards $\rightarrow \hat{V}_{\text{raw}}(\tau)$
    end for
    for each trajectory i in group g **do**
        Set the endpoint anchors and shrink interior values by Eq. (5)
        Compute $\tilde{A}_{i,t}$ and $A_{i,t}$ by Eqs. (6)–(7)
    end for
end for
Update θ with the standard GRPO objective, using $A_{i,t}$ in place of GRPO's broadcast trajectory-level advantage

4 Related work

Trajectory-level credit assignment. GRPO (Shao et al., 2024) and GSPO (Zheng et al., 2025a) broadcast a group-relative advantage across each completion. DAPO (Yu et al., 2025) removes zero-variance groups after rollout, whereas GRESO (Zheng et al., 2025b) skips predicted zero-variance prompts beforehand; Dr. GRPO (Liu et al., 2025) removes length normalization and group standardization. Recent variants rescale or transform trajectory advantages (λ-GRPO (Wang et al., 2025); f-GRPO (Haldar et al., 2026)), iteratively refine trajectories (iGRPO (Hatamizadeh et al., 2026)), or prioritize replay using trajectory rewards (Yoo et al., 2026). A trajectory-level signal cannot distinguish helpful from harmful regions of the same response; ProbePO resolves this while retaining GRPO's trajectory-level anchor.

Token- and step-level credit assignment. Process reward models (PRMs) learn dense step feedback from labeled or automatically constructed process data (Wang et al., 2024; Setlur et al., 2025). RED differences a sequence reward model's scores on consecutive prefixes to obtain token rewards, whereas T-REG regularizes implicit token rewards toward targets produced by contrastive prompts (Li et al., 2025; Zhou et al., 2025). VinePPO instead estimates policy-matched values by probing every reasoning-step boundary (Kazemnejad et al., 2025). Together, these methods expose a supervision–cost trade-off: learned or self-generated supervision adds label construction, model inference, or prompted-scoring costs and may not match current-policy verifier success, while dense policy-matched probing incurs high rollout cost. ProbePO targets the middle ground with verifier-grounded credit under a capped sparse-measurement budget and no separate reward model.

Segment-level credit assignment. Segment methods assign shared credit within contiguous spans. SPO partitions trajectories at fixed intervals or low-probability cutpoints, estimates boundary values from policy continuations, and differences adjacent MC means (Guo et al., 2025). Other approaches derive segment scores from Shapley or intervention-based attribution, generative PRMs, or gold-answer probabilities (Cao et al., 2025; Xie et al., 2025; Yang et al., 2026; Wang et al., 2026; Li et al., 2026b). SPO provides current-policy segment values, but its rollout cost grows with the number of probed boundaries. ProbePO instead concentrates a fixed group-wide budget on high-ranked boundaries, regularizes the resulting estimates, and centers the segment correction so local refinements preserve the trajectory's overall GRPO credit.

Tree-based credit assignment. Online tree methods obtain intermediate evidence by branching trajectories and commonly reuse descendants as policy-update samples (Guo et al., 2025; Yang et al.,

2025b; Li et al., 2026a; Zhao et al., 2026). TEMPO constructs a prefix tree post hoc from an ordinary GRPO batch (Tran et al., 2025), while OmegaPRM allocates trees offline to construct PRM labels (Luo et al., 2024). Online reuse amortizes branching cost, but branch placement and fanout affect both value evidence and which samples enter the actor update, making credit gains difficult to separate from exploration and data reuse. ProbePO also branches, but uses a capped sparse continuation budget only for measurement while leaving the backbone actor batch unchanged (Appendix E).

5 EXPERIMENTS

We ask four questions: Does ProbePO improve reasoning accuracy across model families? Do structured placement and shrinkage contribute to its performance? How do its accuracy and training cost compare with GRPO and dense probing? Does the reward-free selector identify informative boundaries?

5.1 EXPERIMENTAL SETUP

Backbones and baselines. We evaluate three complementary backbones spanning 1.5B–7B parameters: the math-specialized DeepSeekMath-7B-SFT (Shao et al., 2024), the compact reasoning-distilled R1-Distill-Qwen-1.5B (DeepSeek-AI et al., 2025), and Qwen3-4B-Base (Yang et al., 2025a). GRPO provides the trajectory-level reference. SPO-tree estimates segment credit from branching rollouts (Guo et al., 2025), while TreePO uses tree-structured exploration and credit assignment (Li et al., 2026a). For each model, GRPO, SPO-tree, TreePO, and ProbePO use the same training problems, evaluation protocol, and terminal horizon, enabling direct within-model comparability. We evaluate VinePPO separately as the dense policy-matched baseline, which estimates values at every eligible reasoning-step boundary (Kazemnejad et al., 2025). Tables 7–9 in Appendix G give the headline ProbePO recipes and evaluation protocol.

Training data. Following established RL-on-math protocols for each model family, DeepSeekMath and R1 use VinePPO's PRM800K/MATH repartition (Lightman et al., 2024; Kazemnejad et al., 2025), while Qwen3 uses the original MATH training split adopted by ROSE (Zhao et al., 2026). All four algorithms for a given model train on the same problem identities. Appendix G specifies the split construction, filtering, and prompt templates.

Metrics and checkpoint rule. MATH-500, OlympiadBench, CollegeMath, AMC, and Minerva-Math use 16 sampled responses per problem; AIME24 uses 32. Mean@k averages correctness over k sampled responses per problem and then across problems, and **Avg-6** is the unweighted mean of all six reported dataset-level point estimates. Table 1 reports each method's terminal checkpoint at the shared within-model horizon: step 1,000 for DeepSeekMath and R1, and step 27 for Qwen3.

5.2 MAIN RESULTS

Findings. Across all three models, ProbePO achieves the highest Avg-6, improving over GRPO by $+1.72$, $+1.88$, and $+2.85$ points on DeepSeekMath, R1-Distill-Qwen-1.5B, and Qwen3, respectively. It outperforms GRPO in all 18 model–benchmark comparisons and ranks first among the four methods in 16 of 18, showing that the gains persist across model families and benchmarks rather than being concentrated in a single setting.

Within-response credit. Figure 2 illustrates Eq. (7) at actual token positions for two selected held-out responses. In panel (a), an estimated-value dip over the graph-search detour corresponds to a negative token advantage, followed by the largest positive advantage over the final permutation argument. In panel (b), the late reversal from $QA = 10$ to 5 receives the strongest penalty; the preceding derivation is penalized less than under GRPO but remains negative. This is the intended anchored design: ProbePO redistributes rather than replaces trajectory-level credit, preserving its GRPO token mean by construction (Proposition 1).

Table 1: **Terminal accuracy** (%). Avg-6 averages six benchmarks; within-model best in bold.

Model	Method	MATH 500	Olympiad Bench	College Math	AMC	Minerva Math	AIME24 (@32)	Avg-6
DeepSeekMath-7B-SFT	GRPO	42.55	12.67	22.57	14.38	11.79	1.77	17.62
	SPO-tree	37.58	9.93	20.11	12.65	10.96	0.83	15.34
	TreePO	43.86	11.73	22.46	11.22	9.38	0.83	16.58
	ProbePO	**45.81**	**14.47**	**22.77**	**17.09**	**12.96**	**2.92**	**19.34**
R1-Distill-Qwen-1.5B	GRPO	80.10	46.08	55.45	54.89	29.69	12.71	46.49
	SPO-tree	78.93	43.33	54.88	52.41	31.09	**15.42**	46.01
	TreePO	80.36	46.94	54.73	53.92	**31.18**	14.69	46.97
	ProbePO	**82.34**	**48.44**	**57.31**	**58.13**	30.28	13.75	**48.37**
Qwen3-4B-Base	GRPO	76.00	38.50	31.59	47.14	24.26	12.08	38.26
	SPO-tree	75.93	37.86	31.44	47.82	23.85	13.70	38.43
	TreePO	77.48	37.89	30.29	49.02	24.06	15.52	39.04
	ProbePO	**79.40**	**41.13**	**31.99**	**50.53**	**26.10**	**17.50**	**41.11**

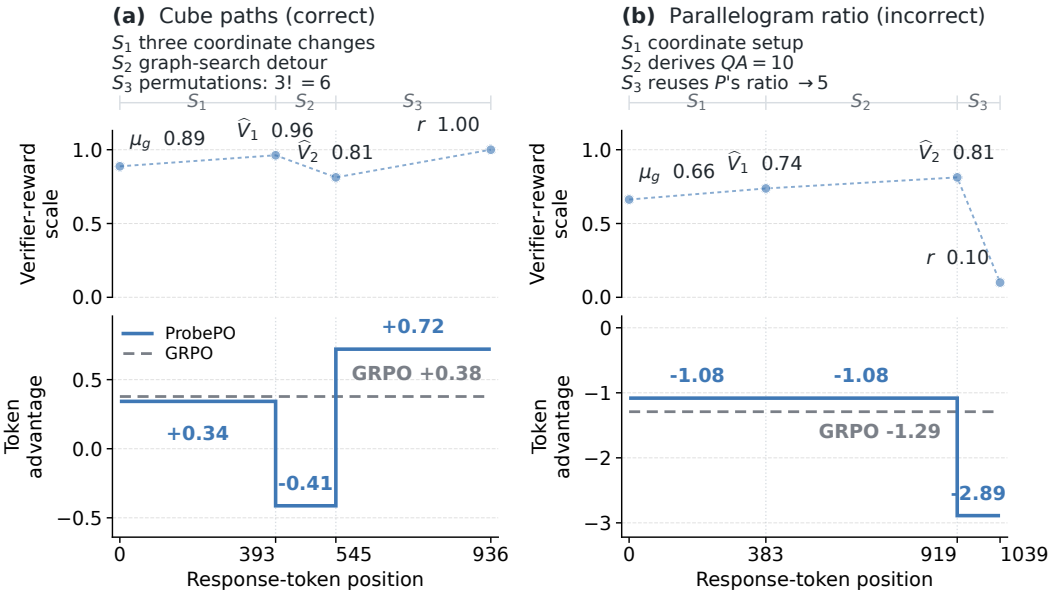

Figure 2: **Selected within-response credit redistributions.** Two illustrative held-out MATH responses from a replay of the trained R1-Distill-Qwen-1.5B checkpoint. Upper markers show the group anchor, deployed shrunk pivot estimates ($M = 8, n_0 = 4$), and terminal reward (connecting lines guide the eye only); lower panels compare ProbePO and uniform-GRPO token advantages over the actual token spans. Each profile preserves GRPO's token-length-weighted mean. Here $r = 0.10$ denotes a boxed incorrect answer. Appendix D.5 gives the full contexts and pooled-$M = 32$ sensitivity check.

5.3 Controlled ablations: placement and shrinkage

Controlled design. We ablate the structured placement module and shrinkage in a controlled Qwen3-4B study. All arms share the training and evaluation setup and retain every generated group; probe-based arms use the same budget ($B = 40$, $M = 8$).

Ablation arms. Random placement samples eligible interior token positions uniformly instead of using the structured selector, while keeping the probe budget and credit estimator fixed. No shrinkage keeps the structured selector and probe budget fixed but uses the unregularized continuation mean.

Results. At the matched checkpoint, replacing structured placement with random placement lowers MATH-500 by 2.29 points and Avg-6 by 1.30; removing shrinkage lowers them by 3.17 and 1.24 points, respectively. Removing shrinkage also increases the 99th-percentile token-advantage magnitude by 19.6%, consistent with its intended regularization role. These complementary losses support the central design: sparse measurements are most effective when allocated to informative states and regularized before credit redistribution. Appendix D.2 gives the controlled-study protocol.

5.4 Accuracy at matched training time

Using the same learning rate, KL coefficient, and eight-B200 allocation, we compare ProbePO and GRPO at matched cumulative training time, including backbone and auxiliary generation, optimization, evaluation, and checkpointing. Figure 3 traces MATH-500 accuracy over cumulative training time.

Variant	MATH-500	Avg-6
ProbePO	**79.35**	**41.36**
Random placement	77.06	40.06
No shrinkage	76.18	40.12

Table 2: **Qwen3-4B ablations** (%).

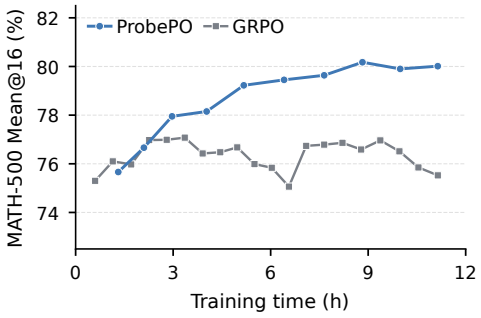

Figure 3: **Qwen3-4B MATH-500 accuracy at matched training time.**

At 11.14 training hours, ProbePO outperforms GRPO by 4.49 points on MATH-500 and 1.95 points on Avg-6. Sparse, structured measurements therefore improve accuracy per unit of end-to-end training time. Endpoint and timing details appear in Appendix D.3.

5.5 Efficiency against dense probing

VinePPO is the closest dense policy-matched comparator: it estimates values at every eligible reasoning boundary (Kazemnejad et al., 2025). Under the shared DeepSeekMath-7B-SFT/MATH setup, ProbePO exceeds VinePPO on all six benchmarks (19.34 versus 17.31 Avg-6) while using 76% less training time; VinePPO uses 10.7× as many auxiliary continuations per update. Appendix D.1 reports the per-benchmark results and training-cost details.

5.6 Selector evidence

Offline selector diagnostic. On frozen DeepSeekMath-7B-SFT trajectories, the offline forward-entropy diagnostic correlates positively with the action-fork-variance proxy on MATH-precalc ($\rho = 0.271$) and GSM8K (0.268). With one selected boundary, it captures 18.0 and 10.9 percentage points more proxy variance than random selection, respectively. Appendix F gives the full protocol and uncertainty scope.

6 Conclusion

ProbePO allocates sparse measurements to reasoning boundaries and redistributes GRPO credit while preserving each trajectory's mean token-level advantage. Across three models and six benchmarks, it consistently outperforms GRPO, while controlled ablations support structured placement and shrinkage. At matched training time, ProbePO improves Avg-6 over GRPO by 1.95 points; against VinePPO, it improves Avg-6 by 2.03 points with 76% less wall-clock training time. Together, these results show that budgeted, measurement-only probing enables efficient fine-grained credit without dense process supervision.

AI USE STATEMENT

We used OpenAI Codex and Anthropic Claude Code to assist with polishing the manuscript's writing and with writing, debugging, and refining code. The authors reviewed all AI-assisted text and code before inclusion and take full responsibility for the final manuscript, code, results, and conclusions.

ETHICS STATEMENT

This work uses publicly available models and benchmarks and involves no human subjects or private personal data. It studies a general credit-assignment method for language-model reinforcement learning and does not introduce a deployment-facing application, release a new high-risk dataset, or release a trained model checkpoint. The models, datasets, frameworks, evaluators, baselines, and adapted visual elements used in this work are cited and used under their stated terms.

REPRODUCIBILITY STATEMENT

Algorithm 1 specifies the method, Section 5.1 describes the datasets and evaluation design, and Appendix G provides the prompting, acquisition, optimization, decoding, checkpoint, and compute details needed to reproduce the reported setup. Appendices A and B provide the assumptions and derivations for the theoretical results, and Appendix F documents the offline selector diagnostic and its bootstrap intervals.

ACKNOWLEDGMENTS

Acknowledgments omitted for double-blind review.

REFERENCES

Meng Cao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. SCAR: Shapley credit assignment for more efficient RLHF. *arXiv preprint arXiv:2505.20417*, 2025. URL https://arxiv.org/abs/2505.20417.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen

Zhang. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, Boca Raton, FL, 3 edition, 2013. ISBN 9780429113079. doi: 10.1201/b16018. URL https://doi.org/10.1201/b16018.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in RL for large language models. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 114399–114431. Curran Associates, Inc., 2025. doi: 10.52202/085713-3815. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a6536243037d1e32c20de85137d478da-Abstract-Conference.html.

Rajdeep Haldar, Lantao Mei, Guang Lin, Yue Xing, and Qifan Song. f-GRPO and beyond: Divergence-based reinforcement learning algorithms for general LLM alignment. *arXiv preprint arXiv:2602.05946*, 2026. URL https://arxiv.org/abs/2602.05946.

Ali Hatamizadeh, Shrimai Prabhumoye, Igor Gitman, Ximing Lu, Seungju Han, Wei Ping, Yejin Choi, and Jan Kautz. iGRPO: Self-feedback-driven LLM reasoning. *arXiv preprint arXiv:2602.09000*, 2026. URL https://arxiv.org/abs/2602.09000.

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. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29557–29590. PMLR, 2025. URL https://proceedings.mlr.press/v267/kazemnejad25a.html.

Jiahui Li, Lin Li, Tai-Wei Chang, Kun Kuang, Long Chen, Jun Zhou, and Cheng Yang. RED: Unleashing token-level rewards from holistic feedback via reward redistribution. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4993–5022, Suzhou, China, nov 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.252. URL https://aclanthology.org/2025.emnlp-main.252/.

Yizhi Li, Qingshui Gu, Zhoufutu Wen, Ziniu Li, Ruibin Yuan, Tianshun Xing, Shuyue Guo, Tuney Zheng, Xin Zhou, Xingwei Qu, Wangchunshu Zhou, Zheng Zhang, Wei Shen, Wei Xue, Qian Liu, Chenghua Lin, Jian Yang, Ge Zhang, and Wenhao Huang. TreePO: Enhancing policy efficacy and inference efficiency with tree modeling. In *Forty-Third International Conference on Machine Learning*, 2026a. URL https://icml.cc/virtual/2026/poster/61677.

Yugu Li, Zehong Cao, Jianglin Qiao, and Siyi Hu. Ssvpo: Effective step-level credit assignment for rl training of language models. In *The Fourteenth International Conference on Learning Representations*, 2026b.

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 *International Conference on Learning Representations*, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-Like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision, 2024. URL https://arxiv.org/abs/2406.06592.

Jerzy Neyman. On the two different aspects of the representative method: The method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4): 558–625, 1934. doi: 10.2307/2342192. URL https://doi.org/10.2307/2342192.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *International Conference on Learning Representations*, volume 2025, pp. 60808–60838, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/98711dea460bdefe0e651ca23ec98ba2-Abstract-Conference.html.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Hieu Tran, Zonghai Yao, and Hong Yu. Exploiting tree structure for credit assignment in RL training of LLMs. *arXiv preprint arXiv:2509.18314*, 2025. URL https://arxiv.org/abs/2509.18314.

Jingyi Wang, Lei Zhu, Tengjin Weng, Song-Li Wu, Haochen Tan, Jierun Chen, Chaofan Tao, Haoli Bai, Lu Hou, Lifeng Shang, and Xiao-Ping Zhang. GRPO-VPS: Enhancing group relative policy optimization with verifiable process supervision for effective reasoning. *arXiv preprint arXiv:2604.20659*, 2026. URL https://arxiv.org/abs/2604.20659.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, aug 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Yining Wang, Jinman Zhao, Chuangxin Zhao, Shuhao Guan, Gerald Penn, and Shinan Liu. λ-GRPO: Unifying the GRPO frameworks with learnable token preferences. *arXiv preprint arXiv:2510.06870*, 2025. URL https://arxiv.org/abs/2510.06870.

Guofu Xie, Yunsheng Shi, Hongtao Tian, Ting Yao, and Xiao Zhang. CAPO: Towards enhancing LLM reasoning through generative credit assignment. *arXiv preprint arXiv:2508.02298*, 2025. URL https://arxiv.org/abs/2508.02298.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Matthew Yang, Hao Bai, Ian Wu, Gene Yang, Amrith Setlur, and Aviral Kumar. InT: Self-proposed interventions enable credit assignment in LLM reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 85054–85091, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/89062e4d480c0c3a88d36c20c5694459-Abstract-Conference.html.

Zhicheng Yang, Zhijiang Guo, Yinya Huang, Xiaodan Liang, Yiwei Wang, and Jing Tang. TreeRPO: Tree relative policy optimization, 2025b. URL https://arxiv.org/abs/2506.05183.

Gyeongtae Yoo, Sanghyeok Park, Soohyuk Jang, Ik-hwan Kim, and Sungroh Yoon. Rollout-level advantage-prioritized experience replay for GRPO. *arXiv preprint arXiv:2606.04560*, 2026. URL https://arxiv.org/abs/2606.04560.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 113222–113244. Curran Associates, Inc., 2025. doi: 10.52202/085713-3775. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Ziqi Zhao, Zhaochun Ren, Jiahong Zou, Liu Yang, Zhiwei Xu, Xuri Ge, Zhumin Chen, Xinyu Ma, Daiting Shi, Shuaiqiang Wang, Dawei Yin, and Xin Xin. Reinforced efficient reasoning via semantically diverse exploration. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 47994–48007, San Diego, California, United States, jul 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.2216. URL https://aclanthology.org/2026.acl-long.2216/.

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

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for LLM reasoning via selective rollouts. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 124321–124346. Curran Associates, Inc., 2025b. doi: 10.52202/085713-4143. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/b42fb82fc3ffb88b872c4714f093875d-Abstract-Conference.html.

Wenxuan Zhou, Shujian Zhang, Lingxiao Zhao, and Tao Meng. T-REG: Preference optimization with token-level reward regularization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27876–27889, Vienna, Austria, jul 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.1353. URL https://aclanthology.org/2025.acl-long.1353/.

A A CONDITIONAL PRECISION-ALLOCATION BENCHMARK

ProbePO does not optimize the number of continuations per selected pivot: it uses $n_b = M$ throughout. This appendix records a benchmark for a different, conditional question—if the pivot set were already fixed and per-pivot sample counts could vary, how would a first-order approximation allocate a total budget B to reduce MC-induced gradient perturbation? The benchmark is not used by the algorithm and does not derive the $H_{\max}$ placement score.

Setup. Condition on a fixed accepted backbone trajectory and a fixed pivot set $\mathcal{P}$. To isolate MC precision, let the raw, unshrunk mean at $b \in \mathcal{P}$ use n_b continuations, with error $\epsilon_b = \hat{V}_{\mathrm{raw}}(s_b) - V^{\pi_\theta}(s_b)$, $\mathbb{E}[\epsilon_b \mid s_b] = 0$, and $\mathrm{Var}(\epsilon_b \mid s_b) = \mathrm{Var}(R \mid s_b)/n_b$. For this appendix only, let $A^{(0)}_{i,t}$ denote Eq. (7) with the raw pivot means inserted directly (equivalently, $n_0 = 0$), and define $J_{tb} := \partial A^{(0)}_{i,t} / \partial \hat{V}_{\mathrm{raw}}(s_b)$. Writing $\psi_t = \nabla_\theta \log \pi_\theta(y_t \mid s_{t-1})$, the probe-induced gradient perturbation is $\delta g = \sum_t \big(\sum_{b\in\mathcal{P}} J_{tb}\epsilon_b\big)\psi_t$. For this unclipped gradient proxy and conditionally independent probe sets,

$$\mathbb{E}_\epsilon \|\delta g\|^2 \;=\; \sum_{b\in\mathcal{P}} \frac{\mathrm{Var}(R \mid s_b)}{n_b}\, C_b, \qquad C_b = \Big\| \sum_t J_{tb}\psi_t \Big\|^2 .$$

Assuming one continuation per budget unit and relaxing n_b to positive reals, minimizing this proxy under $\sum_{b\in\mathcal{P}} n_b = B$ gives the Neyman-style allocation (Neyman, 1934)

$$n^\star_b = B\, \frac{\sqrt{C_b \,\mathrm{Var}(R \mid s_b)}}{\sum_{b'\in\mathcal{P}} \sqrt{C_{b'}\,\mathrm{Var}(R \mid s_{b'})}} \;\propto\; \sqrt{C_b \,\mathrm{Var}(R \mid s_b)}. \tag{8}$$

Scope and approximations. First, Eq. (8) allocates precision among an already selected pivot set; it neither selects pivots nor supplies an analytic justification for $H_{\max}$. Second, before centering, a boundary estimate enters only the two adjacent segments; after Eq. (6), it also changes the trajectory-wide centering offset. For an interior boundary $b = \tau_j$, let $I^-_b = I_j$ denote the segment ending at b and $I^+_b = I_{j+1}$ the segment beginning after b. Under the unshrunk map, the corresponding final advantage sensitivity is

$$J_{tb} = \frac{\alpha}{\sigma_g}\left[\mathbf{1}\{t \in I^-_b\} - \mathbf{1}\{t \in I^+_b\} - \frac{|I^-_b| - |I^+_b|}{T_i}\right].$$

Thus C_b must use this full centered Jacobian; adjacent-segment length alone is not generally the correct footprint. Third, the benchmark ignores policy-ratio clipping and optimizer dynamics, ignores cross-pivot covariance, treats n_b as continuous, and assigns equal cost to continuations despite variation in generated suffix length. Fourth, it omits the contraction in Eq. (5), which introduces bias and changes mean-squared error. Finally, Eq. (8) depends on total continuation variance $\mathrm{Var}(R \mid s_b)$, whereas placement targets action-fork variance $\mathrm{Var}_{a\sim\pi_\theta}(Q^{\pi_\theta}(s_b, a))$. For all of these reasons, we retain Eq. (8) only as a precision benchmark; the implemented method ranks with $H_{\max}$ and assigns the same M continuations to every selected pivot.

B ESTIMATOR DETAILS AND TOKEN-MEAN PRESERVATION

Proof of Proposition 1. Equation (6) gives $\sum_t \tilde{A}_{i,t} = 0$; hence Eq. (7) yields $T_i^{-1}\sum_t A_{i,t} = A_{\mathrm{base},i} = (r_i - \mu_g)/\sigma_g$, proving the first claim. If $K = 0$, $A^{\mathrm{raw}}_{i,t} = d_1/\sigma_g$ is constant across the response, so its centered residual vanishes and $A_{i,t} = A_{\mathrm{base},i} = (r_i - \mu_g)/\sigma_g$. □

Pseudo-count interpretation and bias–variance decomposition. For binary rewards, let $S_k = M\hat{V}_{\mathrm{raw}}(\tau_k)$ and $v_k = V^{\pi_\theta}(s_{\tau_k})$. When $n_0 > 0$ and $0 < \mu_g < 1$, the working prior $v_k \sim \mathrm{Beta}(n_0\mu_g, n_0(1-\mu_g))$ yields Eq. (5) as the posterior mean after observing S_k successes in M trials. Because the prior center μ_g is estimated from the accepted backbone group, this is a plug-in empirical-Bayes interpretation rather than a fully specified generative model.

More generally, set $\lambda = M/(M + n_0)$ and $\varsigma_k^2 = \mathrm{Var}(R \mid s_{\tau_k})$. Conditional on the selected state and the realized target μ_g, independent on-policy probes give

$$\mathbb{E}[\hat{V}(\tau_k)] - v_k = (1-\lambda)(\mu_g - v_k), \qquad \mathrm{Var}(\hat{V}(\tau_k)) = \lambda^2 \frac{\varsigma_k^2}{M},$$

and hence

$$\mathrm{MSE}(\hat{V}(\tau_k)) = \lambda^2 \frac{\varsigma_k^2}{M} + (1-\lambda)^2 (v_k - \mu_g)^2.$$

Thus n_0 makes explicit the tradeoff between Monte Carlo variance and mismatch between the pivot value and the group-level target.

Exact-update conditions. On the same accepted backbone trajectories, $K = 0$ or $\alpha = 0$ reproduces the configured GRPO advantage tensor when group statistics, response masks, and standardization are identical. It also reproduces the optimization update when the policy parameters, objective (including clipping and KL terms), and optimizer state and settings are matched.

Token-mean advantages For the unclipped score-function estimator at the rollout policy, the residual adds

$$\Delta g_i = \frac{\alpha}{T_i} \sum_{t=1}^{T_i} \tilde{A}_{i,t} \nabla_\theta \log \pi_\theta(y_{i,t} \mid q_i, y_{i,<t}).$$

Although $\sum_t \tilde{A}_{i,t} = 0$, this score-weighted sum is generally nonzero because the token score vectors differ. Proposition 1 therefore preserves each trajectory's token mean advantage, not the update produced by a nonzero residual.

Centering and segment length. Because the centering subtracts the token-mean, a residual's effect on a segment scales with the complement of that segment's length share; long segments retain proportionally less of their raw delta than short ones.

Shrinkage is not a fallback. Suppose $K \geq 1$ and every interior estimate is contracted fully to μ_g. Writing $p = (T_i - \tau_K)/T_i$ for the terminal segment's token fraction and $b_i = (r_i - \mu_g)/\sigma_g$, the raw token-level advantage is 0 before the terminal segment and b_i on it. After centering, the residual is therefore $-pb_i$ before that segment and $(1-p)b_i$ on it, which is generally nonzero. Removing all pivots ($K = 0$) or setting the residual weight to zero ($\alpha = 0$) guarantees the configured GRPO advantage tensor independently of the probe values; contraction alone does not.

C CUTOFF-RESCUE PROBE POLICY

The R1-Distill-Qwen-1.5B regime (2K context) uses a cutoff-rescue policy for probes that would otherwise exhaust the context window without a gradeable answer. From each selected prefix, a probe first continues normally under π_θ. If it terminates before the operational cutoff, its continuation is kept and scored unchanged. If it remains unfinished at the cutoff, the continuation is trimmed there and scored directly when the trimmed text is already gradeable. Otherwise, an open reasoning block is closed when necessary, an answer template is appended, and a short reserved decode produces the answer submitted to the verifier. Thus natural and rescued samples are outcomes of one modified continuation policy $\pi_{\mathrm{rescue},k}$, and their mean estimates $V^{\pi_{\mathrm{rescue},k}}(s)$ rather than the unmodified $V^{\pi_\theta}(s)$. In the recipe-matched replay, 98.0% of probe continuations finish naturally; the remaining 2.0% use cutoff handling.

D EXTENDED EXPERIMENTAL RESULTS

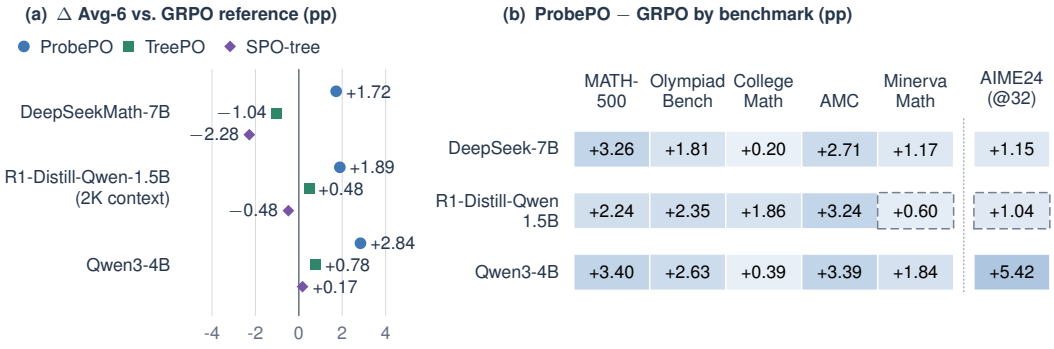

Figure 4: **Terminal-checkpoint gains.** *(a)* Avg-6 difference from each setting's GRPO reference. *(b)* ProbePO minus GRPO by benchmark. AIME24 uses Mean@32; dashed cells mark tasks on which another method scores higher.

D.1 FULL DENSE-PROBING COMPARISON

Table 3: **Quality and cost against dense probing.** Panel (a) reports terminal benchmark accuracy and panel (b) end-to-end training cost under the shared DeepSeekMath setting.

(a) Per-benchmark quality

Method	MATH 500	Olympiad Bench	College Math	AMC	Minerva Math	AIME24 (@32)	Avg-6
ProbePO	**45.81**	**14.47**	**22.77**	**17.09**	**12.96**	**2.92**	**19.34**
VinePPO	43.69	11.65	22.11	15.81	10.27	0.31	17.31

(b) End-to-end training cost

Method	Auxiliary continuations per update	H200-hours	Relative time
ProbePO	$\leq$ 2,560	**231**	**1.00**×
VinePPO	27,265 mean	944	4.08×

ProbePO's auxiliary-continuation count is a configured upper bound, whereas VinePPO's is the observed per-update mean. H200-hours are rounded; relative time uses the measured end-to-end durations.

The original VinePPO paper reports 46.0% for its MATH-500 entry. Our evaluation instead uses the strict verl/Hendrycks grader rather than VinePPO's Minerva-style grader and runs on eight H200 rather than H100 GPUs. Both methods start from the same DeepSeekMath-7B-SFT initialization and MATH source split, use 64 prompt groups with $G = 8$ backbone responses, cap responses at 1,024 tokens, and train for 1,000 outer updates on one exclusive node. VinePPO retains its released nine-continuation-per-boundary recipe.

D.2 CONTROLLED QWEN3 COMPONENT STUDY

The study in Section 5.3 uses Qwen3-4B-Base, the common ROSE MATH data and objective, the same frozen evaluator, and one 8×B200 node. Every arm retains all generated prompt groups, and the probe-based arms share the same normalization, coefficient clipping, and $B = 40$, $M = 8$, and $\alpha = 1$ settings.

The structured reference uses $H_{\max}$ with $L = 8$, a 60-token separation, a two-pivot core cap per trajectory, and post-answer exclusion. Random placement replaces this complete placement module

with uniform sampling over interior pre-EOS token positions while preserving the probe budget and credit estimator. It therefore evaluates structured placement as a module rather than entropy ranking alone. The no-shrinkage arm changes only n_0 from 4 to 0, removing the group-baseline contribution to the value estimate. Table 2 reports the matched component results. At each update, we compute the 99th percentile of $|A_t|$ over response tokens included in the policy loss and average it across the common component-study training window. Removing shrinkage raises this statistic from 2.672 to 3.195 (19.6%), consistent with its regularization role.

D.3 Accuracy at matched training time

Table 4: **Equal-training-time Qwen3-4B comparison** (%). Both methods use approximately 11.14 logged trainer hours on the same 8×B200 hardware.

Method	Training time (hours)	MATH-500	Avg-6
ProbePO	11.14	**80.01**	**41.03**
GRPO	11.14	75.52	39.08
Δ	—	+4.49	+1.95

Cumulative logged trainer time includes backbone and auxiliary generation, optimization, periodic evaluation, and checkpointing.

D.4 Headline Qwen3-4B training dynamics

Figure 5 complements the terminal comparison with performance over the shared training horizon. On MATH-500, ProbePO maintains the strongest trajectory throughout the later evaluations. At the final monitored checkpoint, it also leads on both AIME benchmarks and remains competitive on AMC. Together, these curves show that the headline MATH-500 result reflects sustained training behavior rather than an isolated checkpoint.

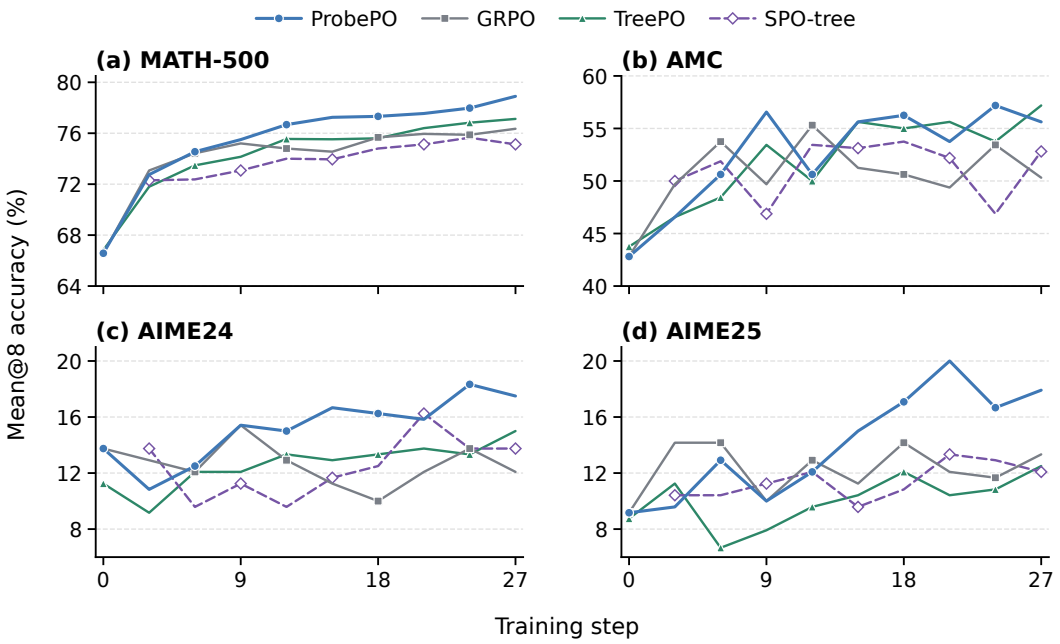

Figure 5: **Qwen3-4B training dynamics.** Monitoring accuracy over the shared headline horizon; points denote evaluations and lines connect successive checkpoints.

D.5 TRAINED-CHECKPOINT CREDIT REDISTRIBUTION

Figure 2 makes the redistribution in Eq. (7) concrete using two responses from a replay of the trained R1-Distill-Qwen-1.5B ProbePO checkpoint. The replay uses the reported $G = 8$, $M = 8$ generation, selector, budget, shrinkage, and cutoff-rescue settings on a fixed cohort of 64 held-out MATH validation prompts. The R1 training grader assigns 1 to a correct boxed answer, 0.1 to a boxed but incorrect answer, and 0 when no valid boxed answer can be extracted; each interior value therefore estimates expected verifier reward rather than literal success probability. This replay visualizes the resulting advantage tensor on concrete responses; Table 1 provides the efficacy evidence and Figure 6 evaluates selector placement. We select two complementary responses from the fixed cohort to illustrate distinct redistribution patterns. Each has two semantic pivots and three segments of at least 64 tokens. The plotted values use the deployed $M = 8$ estimator, and Table 5 reports the pooled-$M = 32$ sensitivity check.

Table 5: **Sensitivity of the displayed token-advantage profiles.** The pooled-$M = 32$ calculation preserves the $(+, -, +)$ pattern in panel (a) and the terminal-most-negative ordering in panel (b).

Profile	Segment 1	Segment 2	Segment 3
Panel (a), deployed $M = 8$	$+0.34$	-0.41	$+0.72$
Panel (a), pooled $M = 32$	$+0.41$	-0.10	$+0.53$
Panel (b), deployed $M = 8$	-1.08	-1.08	-2.89
Panel (b), pooled $M = 32$	-1.37	-0.80	-3.24

The $M = 32$ calculation is a descriptive top-up sensitivity check, not an independent replication or an alternative training configuration.

Pivot-aligned question and response context. The questions below omit only their raw diagram code; the cube diagram is verbalized explicitly. The sampled-response excerpts are verbatim apart from whitespace and mathematical typesetting, ellipses mark omitted spans, and [P1]/[P2] are editorial markers placed at the selector's boundaries. The selector produced these pivots before the examples were chosen for display; the unabridged responses and probe records are retained in the companion artifact.

Example A: cube paths (correct response). Question (diagram verbalized). How many different three-step paths along the edges of a cube are there that take you from vertex A to vertex B? (A step is from a vertex to an adjacent vertex sharing an edge.) The diagram labels $A = (0, 0, 1)$ and $B = (1, 1, 0)$, opposite cube vertices. **Ground truth:** 6.

Sampled response excerpts.

> **Segment 1** ($+0.34$)**.** "...from A $(0, 0, 1)$ to B $(1, 1, 0)$, the coordinates need to change from $(0, 0, 1)$ to $(1, 1, 0)$. So, the changes needed are $+1$ in x, $+1$ in y, and -1 in z. But each step can only change one coordinate at a time. So, we need to find the number of sequences of three moves that result in changing x from 0 to 1, y from 0 to 1, and z from 1 to 0."
>
> **[P1: selected paragraph boundary, token 393]**
>
> **Segment 2** (-0.41)**.** "This is similar to a path counting problem in a grid, but in three dimensions. Hmm, maybe using recursion or combinatorics. Alternatively, we can model this as a graph problem where each vertex is a cube vertex, and edges connect adjacent vertices. ...From A $(0, 0, 1)$, the possible first moves are to $(1, 0, 1)$, $(0, 1, 1)$, or $(0, 0, 0)$. But $(0, 0, 0)$ is not B, so we need to see which moves lead closer to B. Wait, but actually, we need to reach B in exactly three steps. So"
>
> **[P2: selected "So" boundary, token 545]**
>
> **Segment 3** ($+0.72$)**.** ", maybe we can use multinomial coefficients or consider the number of ways to arrange the moves. ...We need one move in x, one move in y, and one move in z, but in some order. ...The number of ways to arrange the moves in x, y, and z ...is $3! = 6$. ...The number of different three-step paths from vertex A to vertex B on a cube is $\boxed{6}$."

The first pivot follows the correct reduction to three coordinate changes; the intervening segment detours into a less directed graph search, and the second pivot precedes the return to the permutation argument. Thus the selected boundaries make the temporary estimated-value dip and recovery in Figure 2(a) directly interpretable.

Example B: parallelogram ratio reversal (incorrect response). Question (diagram code omitted). Let $ABCD$ be a parallelogram. We have that M is the midpoint of AB and N is the midpoint of BC. The segments DM and DN intersect AC at P and Q, respectively. If $AC = 15$, what is QA? **Ground truth:** 10.

Sampled response excerpts.

> **Segment 1** (-1.08)**.** "I think using coordinate geometry might help here. ... Let me place point A at $(0, 0)$. Since it's a parallelogram, let me assume point B is at $(2b, 0)$, point D at $(0, 2d)$, and then point C would be at $(2b, 2d)$. ... $M = (b, 0)$. ... $N = (2b, d)$."
>
> **[P1: selected paragraph boundary, token 383]**
>
> **Segment 2** (-1.08)**.** "To find P, the intersection of DM and AC, we set the equations equal ... $s = 1/3$. ... Similarly, now find Q, which is the intersection of DN and AC. ... $2d - ds = 2ds \Rightarrow 2d = 3ds \Rightarrow s = 2/3$. So, $s = 2/3$, so Q is at $(4b/3, 4d/3)$. ... But the answer seems to be $QA = 10$, but I need to confirm."
>
> **[P2: selected paragraph boundary, token 919]**
>
> **Segment 3** (-2.89)**.** "Alternatively, using coordinates, the length of AC is 15. ... From earlier, $s = 1/3$ for Q, so $AQ = 1/3$ of $AC = 5$, and $QC = 10$. Wait, but the question is about QA, which is 5. ... The length of QA is $\boxed{5}$."

The first pivot separates coordinate setup from the intersection calculations. The middle segment reaches the correct Q ratio $s = 2/3$ and states $QA = 10$; after the second pivot, the response reuses the earlier $s = 1/3$ ratio associated with the other intersection and changes its answer to 5. This makes the terminal-most-negative allocation in Figure 2(b) consistent with the displayed reasoning transition.

E COMPARISON WITH RELATED CREDIT-ASSIGNMENT METHODS

Table 6: **Mechanism comparison.** Cells summarize the cited configurations. ✓ denotes support, × absence, and ± partial or implementation-qualified support; § denotes the surrogate-probe family (Li et al., 2026b; Wang et al., 2026). "Exact GRPO" denotes a $K = 0$ or $\alpha = 0$ tensor setting with fixed accepted trajectories and matched preprocessing.

(a) Measurement and value estimation

Method	Budgeted spend	Reward-free interior pilot	Estimates $V^{\pi_\theta}(s)$
GRPO (Shao et al., 2024)	(no spend)	—	—
VinePPO (Kazemnejad et al., 2025)	×	— (uniform)	✓
SPO-tree (Guo et al., 2025)	×	×	±
TreeRPO (Yang et al., 2025b)	×	×	±
TreePO (Li et al., 2026a)	±	×	±
ROSE (Zhao et al., 2026)	±	✓	×
Surrogate probes§	±	±	×
PRM-label MC (Luo et al., 2024)	✓	× (offline)	✓
ProbePO	✓	✓	✓

(b) Credit and update behavior

Method	Token-mean-preserving injection	Exact GRPO tensor setting	Actor multiset unchanged
GRPO (Shao et al., 2024)	(base)	(base)	✓
VinePPO (Kazemnejad et al., 2025)	×	×	✓
SPO-tree (Guo et al., 2025)	×	×	×
TreeRPO (Yang et al., 2025b)	×	×	×
TreePO (Li et al., 2026a)	×	×	×
ROSE (Zhao et al., 2026)	×	×	×
Surrogate probes§	×	×	✓
PRM-label MC (Luo et al., 2024)	(offline)	(offline)	(offline)
ProbePO	✓	✓	✓

Table 6 highlights ProbePO's combination of budgeted policy-matched measurement, token-mean-preserving credit, exact GRPO recovery, and an unchanged actor batch.

F ORACLE STUDY DETAILS

The oracle uses up to five candidate boundaries per trajectory, six head actions with 32 continuations per action, and a 16-continuation rejection-sampled tail stratum when head-action coverage is below 0.9. The MATH-precalc and GSM8K artifacts contain 286 and 212 candidates, of which 239 and 104 have finite action-fork variance estimates. The offline oracle tests whether forward entropy tracks action-fork variance on archived trajectories. It computes entropy from the top 10 returned token log-probabilities and uses an adaptive forward window; the deployed selector uses full-vocabulary entropy with fixed $L = 8$. The controlled random-placement ablation in Section 5.3 evaluates the complete deployed placement module. The archived backbone trajectories use temperature 0.35 and top-p 0.9, first-action weights use temperature 1 and top-p 1, and forced suffixes use temperature 0.35 and top-p 0.95. For Figure 6A, we resample prompts with replacement, retain every finite boundary belonging to each sampled prompt (including multiplicities), and recompute the pooled tied-rank Spearman correlation. There is one oracle trajectory per prompt, so prompt and trajectory clusters coincide; all 100,000 replicates are finite. For panel B, we retain trajectories with at least two finite candidates and positive total oracle variance, average uniformly over ties in the selector maximum, and compare captured variance with the exact conditional random-selection expectation $1/m_i$ for a trajectory with m_i candidates. The interval resamples the paired selector-minus-random differences

by trajectory. The reported intervals quantify prompt- or trajectory-level variation conditional on the oracle estimates.

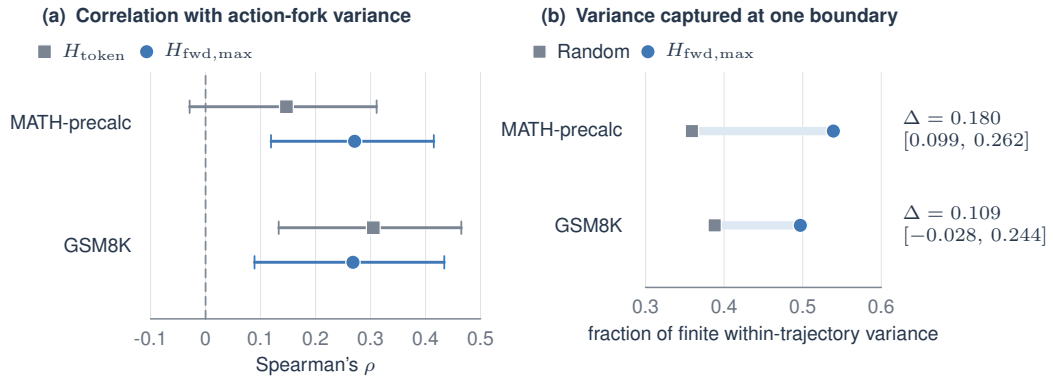

Figure 6: **Offline selector diagnostics.** *(a)* Rank correlation between reward-free signals and action-fork variance. *(b)* Oracle variance captured by the top-ranked versus a uniformly selected boundary. Error bars and brackets show the bootstrap intervals described in the text.

G Recipes, hyperparameters, and reproducibility

Training data. The original MATH benchmark contains 12,500 competition problems, split into 7,500 training and 5,000 test problems (Hendrycks et al., 2021). The DeepSeekMath-7B-SFT and R1-Distill-Qwen-1.5B regimes follow VinePPO's OpenAI/PRM800K repartition (Lightman et al., 2024; Kazemnejad et al., 2025): the original 7,500 training problems and 4,500 test problems form a 12,000-problem development pool, while the remaining 500 uniformly sampled test problems form MATH-500. VinePPO reserves another 500 development problems for validation, leaving 11,500 source training problems. DeepSeekMath uses VinePPO's prompt wrapper; R1 renders the same identities with the SPO/R1 template. Qwen3-4B-Base follows ROSE (Zhao et al., 2026), using the original 7,500-problem MATH training split from `DigitalLearningGmbH/MATH-lighteval`. ROSE's 2,048-token prompt filter removes none under our non-thinking Qwen3 rendering.

Prompt and answer templates. We retain the prompt convention paired with each model family rather than imposing one cross-family wrapper. DeepSeekMath receives VinePPO's baked user string, `[MATH_TASK] Problem:` followed by the problem and `Solution:`; a pass-through chat template adds no system instruction, and correctness is read from the generated final answer by the verl/Hendrycks grader. R1 receives the literal SPO/R1 wrapper: it asks the model to solve efficiently and clearly, think step by step, and end with `Therefore, the final answer is: \boxed{ANSWER}. I hope it is correct`; the rendered assistant prefix opens `<think>`, and the GRESO/math-verify path grades the final boxed answer. Qwen3 retains ROSE's system message requesting step-by-step reasoning and a final answer in `\boxed{}`, followed by the raw problem as the user message. Following ROSE's *no-CoT* (non-thinking) configuration (Zhao et al., 2026), we render this conversation with Qwen3's native chat template and `enable_thinking=false`; this suppresses the model-specific `<think>... </think>` block even though the retained ROSE system message still requests a worked solution. The frozen ROSE evaluator extracts the last boxed expression and applies normalized exact match.

Table 7: **Shared ProbePO acquisition and estimator settings.** These values apply to the three headline ProbePO arms in Table 1.

Parameter	Value
Backbone group size G	8 responses per prompt
Group-wide probe budget B	40 continuation rollouts
Continuations per selected pivot M	8
Shrinkage pseudo-count n_0	4
Residual weight α	1
Stage-1 score	$H_{\max}$ with forward window $L = 8$
Per-trajectory shortlist	$\max\{1, \mathrm{round}(0.30J)\}$ of J candidates
Positional weight w_{pos}	uniform (1)

Table 8: **Optimization and batching for the headline ProbePO runs.** Batch and mini-batch are prompt counts.

Parameter	DeepSeekMath 7B	R1-Distill-Qwen 1.5B (2K context)	Qwen3 4B
Learning rate	5×10^{-7}	1×10^{-6}	1×10^{-6}
Schedule	constant; no warmup	3% linear warmup, then constant	constant; no warmup
Batch / mini	64/8	64/16	512/8
PPO epochs	1	1	1
PPO clip ratio	0.2	0.2	0.2
KL regularization	loss-side 10^{-3} `low_var_kl`	loss-side 10^{-3} `low_var_kl`	loss-side 10^{-3} `low_var_kl`
Reward-side KL	off	off	off
Loss aggregation	`token-mean`	`token-mean`	`seq-mean-token-sum-norm`

Table 9: **Decoding, context, and training horizons for the headline ProbePO runs.** Probe continuations inherit the backbone decoding parameters.

Parameter	DeepSeekMath 7B	R1-Distill-Qwen 1.5B (2K context)	Qwen3 4B
Train/probe temperature, top-p	0.8, 0.95	1.0, 1.0	1.0, 1.0
Maximum prompt tokens	512	512	2,048
Configured response-token limit	1,024	4,096	4,096
Rollout context limit	4,096	2,048 (context binding)	6,144
Training horizon (steps)	1,000	1,000	27

The launcher field `budget_per_trajectory=5` is denominated in continuation rollouts rather than selected pivots; with $G = 8$, it gives the group-wide $B = 40$ budget in Table 7, sufficient for at most five $M = 8$ pivots per prompt group.

Compute resources. Each headline experiment used one node with eight NVIDIA H200 GPUs; every controlled Qwen3 arm used one node with eight NVIDIA B200 GPUs.

Evaluation protocol. DeepSeekMath uses the verl grader with a 4,096-token evaluation context, R1-Distill-Qwen-1.5B uses math-verify with a 2,048-token context, and Qwen3 uses the frozen ROSE last-boxed exact-match evaluator with a 6,144-token total context and a 4,096-token generation cap. The GRPO reference is matched to each headline ProbePO arm on context, optimizer microbatch, rollout temperature, KL coefficient, and learning rate. The controlled Qwen3 arms use the same

frozen six-dataset evaluation protocol. The retained manifest maps every table row to its run and evaluator aggregate.

Collection. Across the controlled Qwen3 arms, we hold prompt batches, advantage normalization, coefficient clipping, and group handling fixed.

Checkpoint protocol. All six scores in a row of Table 1 come from the same checkpoint; AMC or MinervaMath add-on evaluations are joined only at that same checkpoint. This removes benchmark-specific test-set checkpoint selection, and all rows within a setting share the same training horizon: step 1,000 for DeepSeekMath and R1, and step 27 for Qwen3. The component arms are compared at their first common checkpoint. The equal-time comparison uses checkpoints with nearly identical cumulative logged training time of approximately 11.14 hours.