FEAST: FAIR AND EFFICIENT ADAPTIVE SAMPLING FOR TASK-FREE ON-POLICY LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Language agents trained from self-generated experience must allocate a limited rollout budget across heterogeneous capabilities. A common strategy is to prioritize capabilities with strong current learning signals, but in on-policy learning these signals are endogenous: past allocation changes the policy and therefore the evidence available for future allocation. This can create a feedback loop in which under-trained capabilities receive increasingly fewer opportunities to improve. We formalize this effect through the Quota-Efficiency Law (QEL), which shows that allocation history can change learning efficiency even under matched capability-specific exposure. Motivated by this, we introduce FEAST, a task-free adaptive sampling framework that separates opportunity allocation from compute allocation. FEAST dynamically discovers capability clusters while preserving their learning histories, maintains continued sampling opportunities across clusters, and reserves a rollout budget for each selected cluster. Within each budget, it progressively concentrates computation on prompts with informative reward variation. Across GURU-18K and SMC, FEAST improves average performance over competitive adaptive sampling baselines under matched rollout budgets, while better preserving learning progress on weaker capabilities. Ablation studies further support the complementary roles of cross-cluster opportunity protection and within-cluster selective allocation. Code will be released for reproducibility.

1 INTRODUCTION

An agent that improves through reinforcement learning must decide where to spend its training experience. Reinforcement learning with verifiable rewards (RLVR) enables such self-evolution by allowing a language model to generate attempts, receive outcome feedback, and update its policy (Shao et al., 2024; Guo et al., 2025). As RLVR expands from mathematics to heterogeneous reasoning domains (Cheng et al., 2025; Ma et al., 2025; Akter et al., 2026), experience allocation becomes increasingly important: a fixed rollout budget must serve capabilities that differ in difficulty, current proficiency, and trainability. Moreover, predefined domain labels may not reflect the capabilities that matter to the current learner. The resulting challenge is to allocate experience where it is most needed without relying on a fixed task taxonomy.

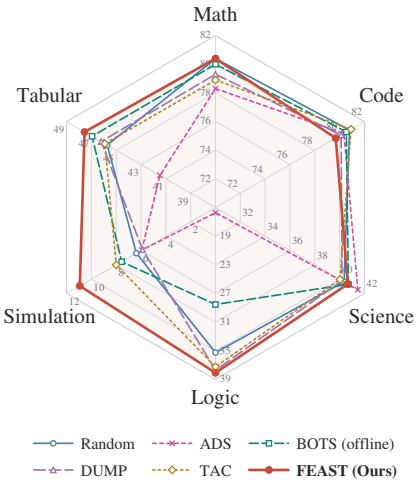


Figure 1: GURU-18K (Cheng et al., 2025) results with Qwen3-8B-Base after 100 updates.

Adaptive curricula address this challenge by estimating learning utility and prioritizing promising data. Learning progress, policy advantages, and cross-domain transfer all provide useful guidance (Graves et al., 2017; Wang et al., 2025a; Chen et al., 2025; Yang et al., 2026). In on-policy learning, however, utility is not an intrinsic property of an example or capability: allocation decisions change the policy, which in turn changes the outcomes available for future learning. Recent

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.

work has begun to recognize this endogenous nonstationarity (McKenzie et al., 2026). As a result, a weak capability may show little immediate learning signal simply because it has received too few opportunities to improve. Prioritizing only the strongest observed signals can then create a feedback loop in which under-allocation produces weak evidence, and weak evidence justifies further under-allocation.

We formalize this phenomenon through the Quota–Efficiency Law (QEL), which compares allocation paths at matched exposure to a target capability. Consider two schedules that train the same capability at different frequencies. We allow the sparser schedule to run longer until both schedules have generated the same number of samples for that capability. Updates on other capabilities move the policy before later target-capability samples are collected, affecting both its current performance and the quality of its subsequent self-generated experience. QEL therefore captures a path-dependent change in learning efficiency, rather than the trivial effect of receiving fewer samples at the same training step. This observation motivates *learning-opportunity fairness*: weak current evidence should not permanently exclude a capability from future training. This notion does not require equal quotas. Instead, it requires continued opportunities to revisit allocation decisions as the learner evolves. At the same time, protecting a capability does not imply spending computation uniformly on all of its prompts. Some prompts are already saturated, some remain beyond the learner's reach, and others exhibit informative variation in outcomes (Yu et al., 2025; Jiang et al., 2026). Effective allocation must therefore operate at two levels: deciding which capabilities receive learning opportunities and, within each, where additional computation is most useful.

We introduce **FEAST**, a **F**air and **E**fficient **A**daptive **S**ampling framework for **T**ask-free agent self-evolution. FEAST combines *fairness across capability clusters* with *efficiency within each cluster*. It discovers capability clusters from the learner's representations and learning dynamics while maintaining fine-grained learning histories as those clusters evolve. Across eligible clusters, adaptive priorities favor productive learning opportunities, while explicit protection ensures that low-scoring clusters continue to receive fresh evidence. Each selected cluster is assigned its own rollout budget. Within that budget, FEAST first probes prompts broadly and then progressively concentrates computation on those with informative outcome variation. Because prompts compete only within their cluster's reserved budget, high utility elsewhere cannot eliminate that cluster's opportunity to learn. Figure 1 previews FEAST's performance on GURU-18K. Our contributions are threefold:

- We introduce FEAST, a task-free framework that protects learning opportunities across evolving capability clusters and concentrates rollouts within each protected budget. Its hierarchy combines adaptive capability discovery, persistent learning histories, and selective computation.
- We characterize QEL at matched ability-specific exposure and analyze time-varying quota paths. This analysis motivates refreshing learning signals for under-allocated capabilities.
- We demonstrate improvements over state-of-the-art sampling baselines under matched training and rollout budgets, both in aggregate performance and learning gains on initially weaker capabilities. Together, the ablations and observed training dynamics support protecting learning opportunities and using rollouts selectively within each cluster.

2 Related Work

2.1 Multi-Domain Reinforcement Learning

Reinforcement learning with verifiable rewards increasingly spans heterogeneous reasoning domains. GURU provides multi-domain data and verifiers (Cheng et al., 2025), while General-Reasoner broadens answer verification with a generative verifier (Ma et al., 2025). Nemotron-CrossThink studies data construction and blending beyond mathematical reasoning (Akter et al., 2026). WildSci extends verifiable training to scientific questions derived from research literature (Liu et al., 2026). Together, these complementary efforts expand the experience available for RL and highlight the importance of its composition. Multi-domain data construction determines the available training pool, while an online allocation rule determines which capabilities receive prompts at each update. Under a fixed generation budget, these opportunities compete, and the resulting updates change the policy that produces future attempts and reward signals. Dataset labels also need not match the relevant allocation units: domains can exercise overlapping skills, while prompts within a domain can yield different learning signals for the current policy. FEAST addresses this allocation problem by discovering capability groups and revising their priorities from

Table 1: Comparison of data-sampling methods. Fairness and efficiency indicate whether these considerations are incorporated into the sampling design. Task-free means no supplied grouping labels. ClusterUCB uses target-task validation data. Extra generation includes refills and reference rollouts. *SPaCe uses extra generation only for model-estimated difficulty. †BOTS requires extra generation only when reference scores are unavailable.

Method	Task-free	Fairness	Efficiency	Extra generation
Skill-It (Chen et al., 2023)	–	✓	–	–
ClusterUCB (Wang et al., 2025b)	✓	✓	✓	–
SPaCe (Do et al., 2026)	✓	✓	✓	✓*
DAPO (dynamic sampling) (Yu et al., 2025)	✓	–	✓	✓
GRESO (Zheng et al., 2025)	✓	–	✓	✓
TAC (Yang et al., 2026)	–	✓	–	–
ADS (Xu et al., 2026)	✓	✓	✓	–
DUMP (Wang et al., 2025a)	–	✓	–	–
BOTS (Shen et al., 2026)	✓	–	✓	✓†
FEAST	✓	✓	✓	–

learning feedback within each capability. It can be applied to diverse, verifiable training data without assuming a fixed domain taxonomy.

2.2 Adaptive Data Sampling

Fair and balanced sampling. Group-level curricula address uneven learning by adapting data mixtures using learning feedback (Graves et al., 2017). Skill-It accounts for dependencies among skills (Chen et al., 2023). DUMP combines policy-advantage feedback with UCB exploration (Wang et al., 2025a), while TAC combines cross-domain transfer estimates with exploratory sampling (Yang et al., 2026). SEC updates category priorities through temporal-difference learning (Chen et al., 2025), while BMC models how training changes utility over a latent task geometry (McKenzie et al., 2026). DUMP's UCB bonus is larger for distributions with fewer samples (Wang et al., 2025a), while Fair-MAB requires each arm to receive a prespecified minimum fraction of pulls (Patil et al., 2021). DRATS focuses on the largest shortfall from a target return while enforcing a minimum task-sampling probability (Corrado et al., 2026). These approaches reflect different criteria for prioritizing training data. Advantage-based selection favors data distributions that currently provide strong training signals. Transfer-aware selection can prioritize a domain whose updates benefit other domains, even when its local learnability is modest.

Efficient sampling and rollout allocation. Efficiency also depends on which examples are selected and how much computation each receives. In supervised fine-tuning, ClusterUCB reduces gradient-based data-selection cost through clustering (Wang et al., 2025b). For RL, SPaCe uses semantic–difficulty clusters for curriculum sampling (Do et al., 2026), ADS combines semantic-cluster scheduling with difficulty-aware prompt selection (Xu et al., 2026), and ASTOR coordinates task budgets and prompt priorities (Chen et al., 2026). BOTS's interpolation estimator combines current-batch outcomes with reference-model scores to estimate difficulty for other prompts (Shen et al., 2026). Collecting missing scores incurs a one-time rollout cost that can be amortized across training runs. At the generation level, DAPO filters uniform-outcome groups and refills the batch (Yu et al., 2025), while ODF filters for intermediate-difficulty prompts (Bae et al., 2026). GRESO uses reward history to skip predicted uninformative prompts before generation (Zheng et al., 2025). VIGOR estimates reward variance from initial rollouts and progressively allocates a fixed total budget to prompts with higher variance (Jiang et al., 2026). Selection, filtering, and generation depth thus offer complementary ways to concentrate learning computation.

Table 1 summarizes these distinctions. TAC and DUMP schedule predefined groups (Yang et al., 2026; Wang et al., 2025a), while FEAST learns capability groups without task labels. ADS favors clusters with higher observed success (Xu et al., 2026); FEAST preserves opportunities even when

current utility is low. DAPO refills informative batches through extra generation (Yu et al., 2025), whereas FEAST redistributes a fixed rollout budget across selected prompts.

3 QEL: Why Allocation History Matters

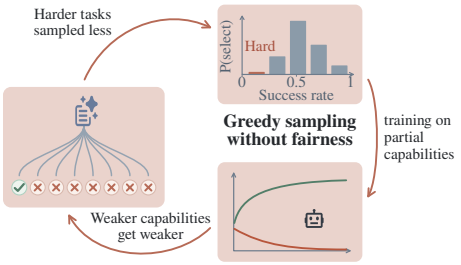


Figure 2: Greedy allocation can reinforce the neglect of weak capabilities.

For capability k, let $V_k(\theta)$ denote held-out performance and $q_{k,t}$ its update-t quota under a predictable, time-varying path $\mathcal{Q}$. For a target exposure $N > 0$, let T denote the first update at which the cumulative quota assigned to capability k reaches N. We define the matched-exposure learning efficiency as

$$\sum_{t<T} q_{k,t} = N, \quad \eta_k(N; \mathcal{Q}) = \frac{\mathbb{E}[V_k(\theta_T)] - V_k(\theta_0)}{N}. \tag{1}$$

This comparison fixes the total amount of capability-specific experience while allowing different allocation paths to reach the same exposure at different training steps.

History dependence. Equal exposure does not imply equal learning efficiency. Let $g_{j,t}$ denote the expected update direction from capability j at step t, before weighting by its quota. Under standard smoothness assumptions, the matched-exposure learning efficiency can be decomposed as

$$\eta_k(N; \mathcal{Q}) = \frac{1}{N}\mathbb{E}\sum_{t<T} \alpha_t \sum_j q_{j,t} \langle \nabla V_k(\theta_t), g_{j,t} \rangle + \mathcal{R}_N, \tag{2}$$

where $\mathcal{R}_N$ collects higher-order terms. The term $j = k$ captures direct learning from capability k, while $j \neq k$ captures cross-capability transfer. Because both the update directions and their effects are evaluated at the evolving policy θ_t, they depend on the preceding allocation history. Thus, two paths with the same capability-specific exposure can still produce different learning gains. Theorem 1 in Appendix A.1 gives the formal decomposition and bounds $\mathcal{R}_N$.

A suppressive regime. Cross-capability transfer can be beneficial, so sparse allocation does not always reduce learning efficiency. We therefore consider a sufficient regime in which it does. Let $h_{k,n}$ denote the amount of off-capability training between successive exposures to capability k. Consider two allocation paths, $\mathcal{Q}^+$ and $\mathcal{Q}^-$, with the same total exposure N, such that $h^+_{k,n} \leq h^-_{k,n}$ for all n. If additional off-capability training does not improve readiness for capability k, and the benefit of later k-specific experience is nondecreasing in that readiness, then

$$\eta_k(N; \mathcal{Q}^-) \leq \eta_k(N; \mathcal{Q}^+). \tag{3}$$

Thus, even with the same number of capability-specific samples, a path with more intervening training can achieve lower learning efficiency. Proposition 2 gives the formal conditions and strictness result. This ordering is sufficient rather than universal. Positive cross-capability transfer can reverse the comparison, and a lower average quota does not necessarily imply larger intervening gaps throughout training. More broadly, QEL shows that learning utility is history dependent: a weak signal observed under one allocation path may become stale as the learner evolves. The feedback loop in Figure 2 motivates regularly refreshing learning signals for under-allocated capabilities rather than using past estimates to exclude them from future experience.

4 FEAST: Fair Across Clusters, Efficient Within

Figure 3 illustrates the FEAST framework. We consider a policy π_θ, an unlabeled prompt pool $\mathcal{X}$, verifier feedback, and a fixed total budget of B generated trajectories per update.

Let $\mathcal{C}_t$ be the current capability clusters and $\mathcal{S}_t \subseteq \mathcal{C}_t$ the M clusters selected at update t. Each receives a disjoint budget $B_c = B/M$. Rollout counts $g_{x,t}$ for selected prompts $P_{c,t}$ satisfy

$$\sum_{x \in P_{c,t}} g_{x,t} = B_c, \qquad \sum_{c \in \mathcal{S}_t} B_c = B. \tag{4}$$

Thus a cluster's opportunity is decided before its prompts compete for additional rollouts.

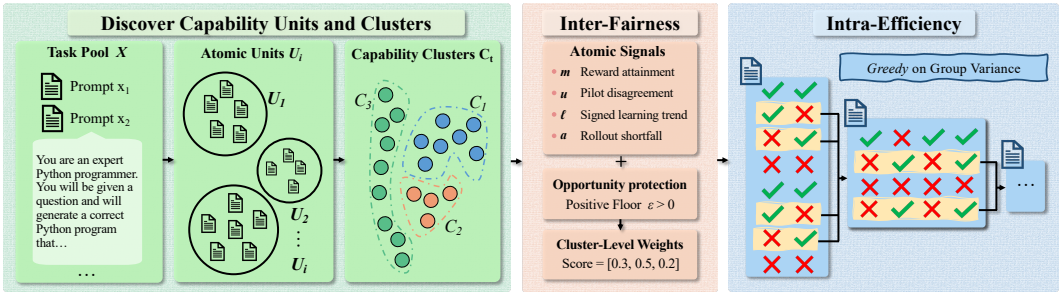


Figure 3: FEAST discovers capability clusters while retaining atomic-unit histories, protects selection opportunities and reserves cluster budgets, then progressively allocates rollouts within each budget. Pilot feedback and policy updates guide subsequent allocation and regrouping.

4.1 Discovering Capabilities While Retaining History

FEAST separates evolving capability groups from persistent learning histories. An initial fine-grained partition $\mathcal{U} = \{U_i\}_{i=1}^{J}$ defines atomic units that retain their observations when regrouped. Neither atoms nor clusters need align with ground-truth task boundaries.

Periodically, FEAST represents each atomic unit using the current policy and groups units by both semantic similarity and learning dynamics according to

$$\mathcal{C}_t = \mathrm{Group}\big(\{[z_i(\theta_t), \bar{\ell}_{i,t}]\}_{i=1}^{J}\big)\,, \tag{5}$$

where $z_i(\theta_t)$ is its policy-dependent representation and $\bar{\ell}_{i,t}$ is a smoothed signed learning trend. Semantically related units can receive different priorities when their learning trends diverge. Representation extraction, clustering, and state updates are specified in Appendix A.2.

4.2 Fair Learning Opportunities Across Clusters

For atomic unit i, let $m_{i,t}$ denote recent reward attainment, $u_{i,t}$ the mean squared difference between two pilot rewards, $\ell_{i,t}$ the exposure-normalized signed attainment trend, and $a_{i,t}$ the accumulated rollout shortfall from an equal per-update share, normalized and capped at 1. Its priority is

$$r_{i,t} = \underbrace{\epsilon + \lambda_a a_{i,t}}_{\text{opportunity protection}} + \underbrace{u_{i,t}\,\Phi(m_{i,t})\big(1 + \gamma|\ell_{i,t}|\big)}_{\text{learning-evidence surplus}}, \tag{6}$$

where $\epsilon > 0$, $\lambda_a, \gamma \geq 0$, and bounded Φ favors intermediate attainment with a modest preference for the less-mastered side. Absolute trends respond to improvement and regression. Attainment alone can conceal consistently successful and unsuccessful prompts with no within-prompt contrast, so FEAST measures within-prompt disagreement separately.

Cluster weights $w_{c,t}$ aggregate atomic priorities using counts of previously unsampled prompts and shrink toward ϵ as that capacity is exhausted (Appendix A.2). Cluster sampling therefore uses temperature-scaled weights $\widetilde{w}_{c,t} = w_{c,t}^{1/\mathcal{T}}$, with $\mathcal{T} > 0$.

The M clusters are drawn without replacement. Let $\mathcal{F}_t$ denote the history before update t and $\mathcal{R}_t$ the clusters remaining in the draw. The next cluster is sampled as

$$\Pr(c \text{ drawn next} \mid \mathcal{F}_t, \mathcal{R}_t) = \frac{\widetilde{w}_{c,t}}{\sum_{c' \in \mathcal{R}_t} \widetilde{w}_{c',t}}, \qquad c \in \mathcal{R}_t. \tag{7}$$

Learning evidence changes selection frequency while retaining eligibility. Selected budgets cannot be reassigned across clusters by prompt-level filtering, protecting each selected cluster's opportunity.

Uniformly positive, bounded weights give history-uniform positive selection probabilities for eligible current clusters, including after regrouping, and a geometric waiting-time bound (Proposition 4). Together with reserved budgets in equation 4, this protects access to experience and its use, but guarantees neither exposure to every member atom nor enough service to eliminate QEL effects. Observed service and learning dynamics assess its finite-horizon value.

4.3 Efficient Rollouts Within Each Opportunity

Within each selected cluster, FEAST first samples a broad prompt set $P_c^{(0)}$ and gives each selected prompt a small, equal number of pilot rollouts. During subsequent rounds, it then retains prompts with the largest cumulative verifier-score variance:

$$P_c^{(r+1)} = \mathrm{Top}_{\lceil \rho |P_c^{(r)}| \rceil}\{v_{x,r} : x \in P_c^{(r)}\}, \qquad \sum_r n_r |P_c^{(r)}| = B_c, \tag{8}$$

where $v_{x,r}$ is the empirical variance using all rollouts generated for prompt x through round r, $\rho \in (0, 1)$ is the retained fraction, and n_r is the number of additional rollouts per surviving prompt. The policy remains fixed throughout these rounds.

In reward-centered GRPO, identical sampled rewards yield zero task advantages; varying outcomes provide a relative signal (Shao et al., 2024; Yu et al., 2025), although regularization can remain active without task advantages (Wang et al., 2026). FEAST places variance-guided progressive allocation (Jiang et al., 2026) inside protected capability budgets: higher-variance prompts elsewhere cannot take a selected weak cluster's rollouts. Accordingly, every generated rollout, including both pilots and zero-signal groups, counts toward the total budget B.

FEAST estimates attainment and disagreement from common initial pilots, weighting prompts equally to avoid promotion bias. Additional rollouts refine training signals, and learning trends account for signal-bearing experience. Statistics remain attached to persistent units and guide regrouping and allocation together with the updated policy.

QEL shows that weak observed feedback can partly result from earlier under-allocation. A scheduler that uses this feedback to exclude a capability risks reinforcing the allocation pattern that produced it. FEAST preserves opportunities to reassess low-scoring clusters as the policy evolves. Within each selected cluster, prompts compete using current feedback, so preserving access to experience remains compatible with concentrating computation where useful learning signals are observed.

5 Experiments

In this section, we first test QEL at matched task exposure, then evaluate FEAST's overall performance, computational cost, and individual components.

5.1 QEL

5.1.1 Experimental Setup

We test the history dependence in Section 3 on Sokoban, Mastermind, and Countdown in TextArena (Guertler et al., 2025) with Qwen3-1.7B-Base. For each task, we compare uniform allocation (1:1:1) with a schedule that assigns it the smallest quota. Both schedules use the same task instances in the same order. Each update uses 32 prompt groups and eight rollouts per group. The two paths first reach $n = 256$ task-specific prompt groups at updates 24 and 104, and $n = 512$ at updates 48 and 208, respectively. As a result, the low-quota path trains longer to reach exactly the same task exposure, as required by the definition in Equation 1.

On Reasoning Gym (Stojanovski et al., 2025), we compare six fixed sampling ratios over Countdown, Spell backward, and Word sorting, aligning each task's held-out success by its cumulative prompt exposure. Each ratio trains for 400 updates. Evaluation uses a fixed panel of 256 held-out instances per task, shared across ratios, with one response per instance.

5.1.2 Results and Analysis

Allocation history changes the return from equal task exposure. At $n = 256$, low-quota paths in Figure 4 trail uniform allocation by 9.77, 18.36, and 1.95 percentage points for Sokoban, Mastermind, and Countdown, respectively. At $n = 512$, all three low-quota outcomes are zero, while uniform allocation improves each task over its initial policy. With common initialization and exposure denominators, these gaps also order empirical learning efficiencies, consistent with QEL's adverse direction and the need to protect opportunities.

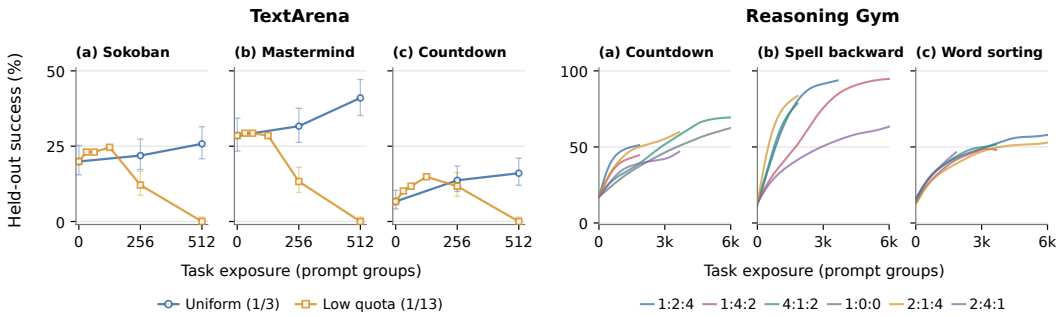


Figure 4: QEL at matched task exposure. TextArena panels (left) compare uniform allocation with low-quota S:M:C mixtures 1:4:8, 4:1:8, and 8:4:1, respectively, where S, M, and C denote Sokoban, Mastermind, and Countdown. Reasoning Gym panels (right) show six C:S:W sampling ratios for Countdown, Spell backward, and Word sorting. Curves are smoothed. Markers in the left panels show the observed scores with 95% Wilson intervals.

Sampling ratios change the average return per prompt. For Reasoning Gym, we measure held-out gain from each initialization per task-specific prompt (Equation 1). At $n = 1829$, the 1:2:4 and 1:4:2 mixtures yield 18.58 versus 14.95 percentage points of gain per 1,000 prompts for Countdown, and 38.87 versus 18.15 for Spell backward (Figure 4). The larger Spell backward gap shows task-dependent sensitivity to allocation history even after normalizing exposure, consistent with the own-ability alignment and signed cross-ability transfer terms in Theorem 1.

Additional task sets, longer-horizon comparisons, and experimental details appear in Appendix A.5.

5.2 FEAST

5.2.1 Experimental Setup

We evaluate FEAST on GURU-18K (Cheng et al., 2025) with Qwen3-1.7B-Base and Qwen3-8B-Base across mathematics, code, science, logic, simulation, and tabular reasoning. Baselines are uniform prompt sampling (Random), TAC (Yang et al., 2026), ADS (Xu et al., 2026), DUMP (Wang et al., 2025a), and BOTS (Shen et al., 2026). All train one policy across the shared pool for 100 optimizer updates and share evaluation panels within each scale. TAC switches domains between single-domain updates; the others allow mixed-domain batches. TAC and DUMP use domain identities, whereas FEAST requires none. Avg. equally weights nine benchmark scores across the six domains. Appendix A.3 gives per-seed results. We also evaluate Online Difficulty Filtering (ODF) (Bae et al., 2026) on GURU-18K at 1.7B as an additional filtering baseline.

We also evaluate FEAST on SMC (Science, Math, and Code), which combines WildSci (Liu et al., 2026), MATH (Hendrycks et al., 2021), and code from LeetCode, LiveCodeBench (Jain et al., 2024), PrimeIntellect (Mattern et al., 2025), and TACO (Li et al., 2023). Its 11,975 verifier-compatible training prompts span seven science categories, seven mathematics subjects, and one code category. Models trained separately on SMC are evaluated on 450 held-out prompts (30 per category, eight responses each), with terminal checkpoints at update 100 for 1.7B and 75 for 8B. Appendix A.3 gives benchmark results, data construction, and protocols.

Across both suites, GRPO comparisons and ablations use 1,024 online rollouts per update, including pilots and zero-variance groups, with matched update counts per dataset and scale. BOTS's offline generation is accounted for separately; Section 5.2.3 discusses refill-based comparisons.

5.2.2 Results

Utility-driven sampling can reinforce neglect. Under matched online rollout budgets at 1.7B, ADS, BOTS, DUMP, and TAC trail Random in Avg. (Table 2), consistent with Section 3. On Logic, ADS, BOTS, and TAC trail by 32.79, 29.66, and 12.19 points; ADS also trails by 13.03 points on Tabular. A similar imbalance also remains evident at 8B: ADS leads these schedulers

Table 2: GURU-18K results (%). Cells show mean ± sample SD across multiple training seeds (1.7B: seeds 42/43/44; 8B: seeds 42/43), each evaluated with avg@k.

Method	Math	Code	Science	Logic	Sim.	Tabular	Avg.
			Qwen3-1.7B-Base				
Base	51.40	49.29	13.66	1.46	2.88	11.18	22.67
Random	61.00±0.87	61.85±0.70	26.85±0.99	**34.24**±0.66	**4.25**±0.33	20.99±0.76	35.43±0.18
ADS	**58.27**±2.02	**60.01**±1.26	**28.22**±0.38	**1.45**±1.14	**4.25**±0.50	**7.96**±0.67	**28.48**±0.32
BOTS	62.80±1.83	**61.88**±0.53	27.52±0.82	4.58±1.70	4.92±0.44	24.16±1.38	33.27±0.30
DUMP	60.60±2.91	61.71±0.62	26.72±0.26	25.31±14.67	**5.38**±0.76	22.72±2.24	34.84±0.95
TAC	**64.13**±0.42	60.74±0.45	**25.46**±0.90	22.05±7.20	4.88±0.78	**25.78**±0.81	35.00±1.16
FEAST	60.40±2.25	61.81±0.49	27.00±1.38	31.67±1.97	4.79±0.36	25.57±1.60	**36.18**±0.13
DAPO	60.20±2.31	62.30±0.79	28.45±1.44	33.57±0.10	4.25±0.33	25.85±0.41	36.80±0.53
DAPO+FEAST	61.33±1.53	63.28±0.75	28.14±0.41	34.50±0.77	4.00±0.33	28.64±1.03	37.77±0.42
			Qwen3-8B-Base				
Base	73.00	77.08	34.03	2.77	8.50	32.02	41.17
Random	**80.40**±2.26	80.81±0.32	40.36±2.54	35.24±0.84	6.38±1.24	45.73±0.15	50.65±1.01
ADS	**78.30**±0.71	80.35±0.62	**41.47**±0.31	**15.72**±1.32	5.94±0.44	**41.48**±4.43	**47.40**±1.19
BOTS	80.00±1.70	80.55±0.68	40.71±0.73	28.53±0.59	7.56±2.21	46.95±0.26	50.28±0.05
DUMP	79.30±0.71	80.13±0.70	40.20±1.74	37.71±0.76	**5.88**±0.88	46.21±0.96	50.66±0.54
TAC	78.90±0.99	**80.93**±0.21	**40.03**±0.70	37.24±1.79	8.00±3.36	45.92±0.92	50.88±0.05
FEAST	**80.40**±0.28	**79.69**±0.10	40.67±1.26	**38.07**±0.66	**10.94**±3.27	**47.55**±0.14	**51.69**±0.08
DAPO	78.00±2.26	80.80±0.21	39.47±0.70	38.54±0.20	6.81±0.97	45.71±1.64	50.59±0.64
DAPO+FEAST	78.50±0.99	81.26±0.63	40.75±0.97	38.51±0.36	6.25±0.53	46.34±0.28	51.11±0.23

on Science, yet its Logic score remains 19.52 points below Random. At 1.7B, ADS leaves Logic near its initial score despite improving Science. This uneven progress is consistent with repeatedly favoring capabilities that already provide useful feedback. Periodic sampling of weak capabilities can refresh their learning signals as training on other domains changes the shared policy.

Overall performance. As shown in Table 2, FEAST achieves the highest average scores among the compared GRPO schedulers, reaching 36.18 at 1.7B and 51.69 at 8B and exceeding Random by 0.75 and 1.04 points, respectively. At 1.7B, FEAST exceeds Random by 4.58 points on Tabular but trails it by 2.57 points on Logic. At 8B, FEAST matches Random on Math and achieves the highest scores among these schedulers on Logic, Simulation, and Tabular, exceeding Random by 2.83, 4.56, and 1.82 points, respectively. These three domains have lower initial scores than Math and Code, and their gains are consistent with FEAST preserving learning opportunities across a broader range of capabilities. FEAST nevertheless scores 1.12 points below Random on Code at 8B, indicating that its aggregate gains still involve trade-offs across domains. Across the reported training seeds, FEAST also has a lower sample SD of Avg. than Random at both scales, suggesting more stable performance.

Complementarity with DAPO. At 1.7B, DAPO+FEAST improves Avg. from 36.80 to 37.77, a gain of 0.97 points over DAPO, with improvements on Math, Code, Logic, and Tabular. At 8B, DAPO+FEAST achieves an Avg. score of 51.11 after 75 updates, compared with 50.59 for DAPO. This suggests that FEAST's allocation can complement DAPO's dynamic filtering. FEAST allocates the candidate rollouts, while DAPO selects which resulting groups contribute to the current policy update. In the combined method, feedback from candidate rollouts updates FEAST's learning histories even when their groups are excluded by filtering. These histories guide subsequent allocation before DAPO selects signal-bearing groups for optimization. Thus, FEAST can change the capability composition of the candidate pool while preserving DAPO's filtering criterion.

Results on SMC. FEAST exceeds Random by 3.33 points at 1.7B and 2.28 at 8B, outperforming ADS and DUMP at both scales (Table 3). Acc. is average success (%); TVA is mean gain over Random in the five least-improved categories. Negative TVA means that higher average performance need not improve every category. Among adaptive samplers, FEAST has the highest TVA at 1.7B and ties DUMP at 8B. Per-domain results are reported in Appendix Table 11.

Table 3: SMC results (avg@8).

	1.7B		8B	
Method	Acc.	TVA	Acc.	TVA
Random	61.42	0.00	78.42	0.00
ADS	58.86	−8.58	77.19	−5.42
DUMP	63.47	−3.67	78.75	−2.25
FEAST	64.75	−2.50	80.69	−2.25

5.2.3 Computational Cost

FEAST keeps the same per-update rollout budget as uniform sampling. Pilots and progressive rounds both draw from this budget, and all generated trajectories, including zero-variance groups, are counted. In the 1.7B GURU-18K run with seed 42, ten representation refreshes take 42.0 minutes in total, accounting for 7.1% of the 9.90-hour training run.

BOTS (Shen et al., 2026) additionally uses offline reference rollouts to construct its selection priors. DAPO (Yu et al., 2025) filters zero-variance groups, while the evaluated Online Difficulty Filtering (ODF) (Bae et al., 2026) retains groups of intermediate empirical difficulty. Both filtering procedures generate extra candidates to fill optimization batches, so rejected groups still incur generation cost. ODF consumes 737.5M tokens over 50 updates but achieves an average score of only 31.92%. Given its high generation cost and limited performance, we do not extend training to 100 updates. Appendix A.3 provides full computational accounting and ODF settings.

5.2.4 Ablation Study

Inter-Fairness and Intra-Efficiency. All ablations use Qwen3-1.7B-Base on GURU-18K with seed 42, 100 updates, and a fixed rollout budget. Global sampling retains progressive allocation (Table 4). Its lower score supports reserving capability-level budgets before prompts compete for additional rollouts. Fixing eight rollouts per prompt produces a larger decline, supporting the use of current reward variation to distribute computation within each protected budget. These losses therefore support combining capability-level protection with selective computation within each budget.

Table 4: FEAST ablations. Δ is relative to the full FEAST configuration.

Method	Avg.	Δ
FEAST	36.32	N/A
w/o Inter-Fairness	35.66	-0.66
w/o Intra-Efficiency	34.66	-1.67

Number of atomic units. Table 5 varies the number of persistent atomic units. Coarser units pool more prompts into each learning history, and regrouping can only move entire units. The lower score at $J = 32$ is consistent with this loss of resolution. With a fixed rollout budget, increasing J reduces the average exposure per unit. Learning trends may then rely on fewer observations, limiting the benefit of finer partitions. The default $J = 64$ achieves performance comparable to $J = 128$ with fewer learning histories to maintain and fewer atomic units to regroup.

Table 5: Atomic-unit count, with Δ relative to the default $J = 64$.

Atomic units J	Avg.	Δ
32	33.84	-2.48
64 (default)	36.32	N/A
128	36.39	$+0.07$

6 Conclusion

We presented FEAST, a task-free RL framework that combines fairness across capability clusters with efficiency within each cluster. Our QEL analysis shows that under suppressive conditions, allocation history can reduce learning gains even at matched capability-specific exposure. Experiments on GURU-18K and SMC with 1.7B and 8B models show improved average performance over other sampling methods under matched rollout budgets, with ablations supporting both allocation levels. These findings support learning-opportunity protection in on-policy self-evolution.

AI USE STATEMENT

We used generative AI tools to improve the language and readability of the manuscript and to assist with code implementation and debugging. The authors take full responsibility for the final text, code, and reported results, including all AI-assisted contributions.

REFERENCES

Syeda Nahida Akter, Shrimai Prabhumoye, Matvei Novikov, Seungju Han, Ying Lin, Evelina Bakhturina, Eric Nyberg, Yejin Choi, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-CrossThink: Scaling self-learning beyond math reasoning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 984–1002, 2026. doi: 10.18653/v1/2026.eacl-long.43. URL https://aclanthology.org/2026.eacl-long.43/.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. doi: 10.48550/arXiv.2108.07732. URL https://arxiv.org/abs/2108.07732.

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. Online difficulty filtering for reasoning oriented reinforcement learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 700–719, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021a. doi: 10.48550/arXiv.2107.03374. URL https://arxiv.org/abs/2107.03374.

Mayee F. Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! A data-driven skills framework for understanding and training language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 36000–36040, 2023. doi: 10.52202/075280-1562. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/70b8505ac79e3e131756f793cd80eb8d-Abstract-Conference.html.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for LLM reasoning. *arXiv preprint arXiv:2505.14970*, 2025. doi: 10.48550/arXiv.2505.14970. URL https://arxiv.org/abs/2505.14970.

Yujia Chen, Yang Ye, Xiao Chu, Yuchi Ma, and Cuiyun Gao. Schedule-and-calibrate: Utility-guided multi-task reinforcement learning for code LLMs. *arXiv preprint arXiv:2605.06111*, 2026. doi: 10.48550/arXiv.2605.06111. URL https://arxiv.org/abs/2605.06111.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, 2021b. doi: 10.18653/v1/2021.emnlp-main.300. URL https://aclanthology.org/2021.emnlp-main.300/.

Jorge (Zhoujun) Cheng, Shibo Hao, Tianyang Liu, Fan Zhou, Yutao Xie, Feng Yao, Yuexin Bian, Nilabjo Dey, Yonghao Zhuang, Yuheng Zha, Yi Gu, Kun Zhou, Yuqi Wang, Yuan Li, Richard Fan, Jianshu She, Chengqian Gao, Abulhair Saparov, Taylor W. Killian, Haonan Li, Mikhail Yurochkin, Eric P. Xing, Zhengzhong Liu, and Zhiting Hu. Revisiting reinforcement learning for LLM reasoning from a cross-domain perspective. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4479. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/c2f71567cd53464161cab3336e8fc865-Abstract-Datasets_and_Benchmarks_Track.html.

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019. doi: 10.48550/arXiv.1911.01547. URL https://arxiv.org/abs/1911.01547.

Nicholas E. Corrado, Wenyuan Huang, and Josiah P. Hanna. Distributionally robust multi-task reinforcement learning via adaptive task sampling. *arXiv preprint arXiv:2605.14350*, 2026. doi: 10.48550/arXiv.2605.14350. URL https://arxiv.org/abs/2605.14350.

Dai Do, Manh Nguyen, Svetha Venkatesh, and Hung Le. SPaCe: Unlocking sample-efficient large language models training with self-pace curriculum learning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3480–3507, 2026. doi: 10.18653/v1/2026.findings-acl.171. URL https://aclanthology.org/2026.findings-acl.171/.

Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1311–1320. PMLR, 2017. URL https://proceedings.mlr.press/v70/graves17a.html.

Leon Guertler, Bobby Cheng, Simon Yu, Bo Liu, Leshem Choshen, and Cheston Tan. TextArena. *arXiv preprint arXiv:2504.11442*, 2025. doi: 10.48550/arXiv.2504.11442. URL https://arxiv.org/abs/2504.11442.

Daya Guo et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

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 J. Vanschoren and S. Yeung (eds.), *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.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024. doi: 10.48550/arXiv.2403.07974. URL https://arxiv.org/abs/2403.07974.

Heyang Jiang, Henry Liu, and Baharan Mirzasoleiman. Learning as reasoning unfolds: Progressive rollout allocation for efficient reinforcement learning. *arXiv preprint arXiv:2607.22002*, 2026. doi: 10.48550/arXiv.2607.22002. URL https://arxiv.org/abs/2607.22002.

Junlong Li, Daya Guo, Dejian Yang, Runxin Xu, Yu Wu, and Junxian He. CodeI/O: Condensing reasoning patterns via code input-output prediction. *arXiv preprint arXiv:2502.07316*, 2025. doi: 10.48550/arXiv.2502.07316. URL https://arxiv.org/abs/2502.07316.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. TACO: Topics in algorithmic COde generation dataset. *arXiv preprint arXiv:2312.14852*, 2023. doi: 10.48550/arXiv.2312.14852. URL https://arxiv.org/abs/2312.14852.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. doi: 10.48550/arXiv.2305.20050. URL https://arxiv.org/abs/2305.20050.

Tengxiao Liu, Deepak Nathani, Zekun Li, Kevin Yang, and William Yang Wang. WildSci: Advancing scientific reasoning from in-the-wild literature. *arXiv preprint arXiv:2601.05567*, 2026. doi: 10.48550/arXiv.2601.05567. URL https://arxiv.org/abs/2601.05567.

M-A-P Team, Xinrun Du, Yifan Yao, et al. SuperGPQA: Scaling LLM evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*, 2025. doi: 10.48550/arXiv.2502.14739. URL https://arxiv.org/abs/2502.14739.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. General-reasoner: Advancing LLM reasoning across all domains. In *Advances in Neural Information Processing Systems*, volume 38, pp. 56596–56618, 2025. doi: 10.52202/085713-1893. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/51c52cbb4b3faa891ca0621d13efe1be-Abstract-Conference.html.

Justus Mattern, Sami Jaghouar, Manveer Basra, Jannik Straube, Matthew Di Ferrante, Felix Gabriel, Jack Min Ong, Vincent Weisser, and Johannes Hagemann. SYNTHETIC-1: Two million collaboratively generated reasoning traces from DeepSeek-R1, 2025. URL https://www.primeintellect.ai/blog/synthetic-1-release.

Darrien McKenzie, Nicklas Hansen, and Xiaolong Wang. Manifold bandits: Bayesian curriculum learning over the latent geometry of large language models. *arXiv preprint arXiv:2606.19750*, 2026. doi: 10.48550/arXiv.2606.19750. URL https://arxiv.org/abs/2606.19750.

NVIDIA. OpenScienceReasoning-2. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/nvidia/OpenScienceReasoning-2.

Vishakha Patil, Ganesh Ghalme, Vineet Nair, and Y. Narahari. Achieving fairness in the stochastic multi-armed bandit problem. *Journal of Machine Learning Research*, 22(174):1–31, 2021. URL https://jmlr.org/papers/v22/20-704.html.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments. In *Conference on Robot Learning*, pp. 835–853. PMLR, 2020.

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. doi: 10.48550/arXiv.2311.12022. URL https://arxiv.org/abs/2311.12022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

Qianli Shen, Daoyuan Chen, Yilun Huang, Zhenqing Ling, Yaliang Li, Bolin Ding, and Jingren Zhou. BOTS: A unified framework for bayesian online task selection in LLM reinforcement finetuning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=rQXBw8fn3Z.

Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. Reasoning Gym: Reasoning environments for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2505.24760*, 2025. doi: 10.48550/arXiv.2505.24760. URL https://arxiv.org/abs/2505.24760.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. DUMP: Automated distribution-level curriculum learning for RL-based LLM post-training. *arXiv preprint arXiv:2504.09710*, 2025a. doi: 10.48550/arXiv.2504.09710. URL https://arxiv.org/abs/2504.09710.

Zige Wang, Qi Zhu, Fei Mi, Minghui Xu, Ruochun Jin, and Wenjing Yang. ClusterUCB: Efficient gradient-based data selection for targeted fine-tuning of LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 18867–18880, 2025b. doi: 10.18653/v1/2025.findings-emnlp.1026. URL https://aclanthology.org/2025.findings-emnlp.1026/.

Zihan Wang, Chi Gui, Xing Jin, Qineng Wang, Licheng Liu, Kangrui Wang, Shiqi Chen, Linjie Li, Zhengyuan Yang, Pingyue Zhang, Yiping Lu, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN-2: Reasoning collapse in agentic RL. *arXiv preprint arXiv:2604.06268*, 2026. doi: 10.48550/arXiv.2604.06268. URL https://arxiv.org/abs/2604.06268.

Runzhe Wu, Ankur Samanta, Ayush Jain, Scott Fujimoto, Jeongyeol Kwon, Ben Kretzu, Youliang Yu, Kaveh Hassani, Boris Vidolov, and Yonathan Efroni. Imbalanced gradients in RL post-training of multi-task LLMs. *arXiv preprint arXiv:2510.19178*, 2025. doi: 10.48550/arXiv.2510.19178. URL https://arxiv.org/abs/2510.19178.

Zicheng Xu, Ruixuan Zhang, Yu-Neng Chuang, Xiuyi Lou, Hoang Anh Duy Le, Oren Gal, Alexander S. Szalay, Zhaozhuo Xu, Guanchu Wang, and Vladimir Braverman. Learning at the right pace: Adaptive data scheduling improves LLM reinforcement learning. *arXiv preprint arXiv:2606.22305*, 2026. doi: 10.48550/arXiv.2606.22305. URL https://arxiv.org/abs/2606.22305.

Yongjin Yang, Jiarui Liu, Yinghui He, Lechen Zhang, Bernhard Schölkopf, and Zhijing Jin. Transferability for general reasoning: An automated curriculum for multi-domain RLVR. *arXiv preprint arXiv:2606.25178*, 2026. doi: 10.48550/arXiv.2606.25178. URL https://arxiv.org/abs/2606.25178.

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, pp. 113222–113244, 2025. doi: 10.52202/085713-3775. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Yilun Zhao, Yunxiang Li, Chenying Li, and Rui Zhang. MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6588–6600, 2022. doi: 10.18653/v1/2022.acl-long.454. URL https://aclanthology.org/2022.acl-long.454/.

Haizhong Zheng, Yang Zhou, Brian R. 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, pp. 124321–124346, 2025. doi: 10.52202/085713-4143. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/b42fb82fc3ffb88b872c4714f093875d-Abstract-Conference.html.

A Appendix

A.1 Theory for the Quota–Efficiency Law

This section presents the assumptions and proofs for the QEL analysis in Sec. 3.

A.1.1 Stochastic setup

Let $(\mathcal{F}_t)_{t\geq 0}$ be the filtration recording the training history before update t, and fix the initial policy parameters θ_0. The parameters θ_t, quota $\mathbf{q}_t \in \Delta^{K-1}$, and step size α_t are $\mathcal{F}_t$-measurable, with $0 \leq \alpha_t \leq \alpha_{\max} < \infty$ almost surely. For capability j, let $\widehat{g}_{j,t}$ be the square-integrable update direction estimated from current-policy data, measurable with respect to $\mathcal{F}_{t+1}$. Its conditional mean is the expected update direction used in Sec. 3:

$$g_{j,t} := \mathbb{E}[\widehat{g}_{j,t} \mid \mathcal{F}_t]. \tag{9}$$

We use the first-order update model

$$\theta_{t+1} = \theta_t + \alpha_t \widehat{g}_t, \qquad \widehat{g}_t = \sum_{j=1}^{K} q_{j,t}\widehat{g}_{j,t}. \tag{10}$$

For a scheduling rule $\mathcal{Q}$, define

$$N_k(T) = \sum_{t=0}^{T-1} q_{k,t}, \qquad \tau_k^{\mathcal{Q}}(N) = \inf\{T : N_k(T) \geq N\}. \tag{11}$$

We compare allocation paths at the same cumulative exposure $N > 0$. For each path, set $T = \tau_k^{\mathcal{Q}}(N)$ and assume $N_k(T) = N$ almost surely. The measurability of the quotas makes T a stopping time.

For a fixed capability k, assume that:

1. V_k has an L-Lipschitz gradient;
2. $\mathbb{E}[\|\widehat{g}_t\|^2 \mid \mathcal{F}_t] \leq G^2$;
3. T is almost surely bounded by a deterministic $T_{\max}$.

The bounded step sizes and second-moment bound give $\mathbb{E}\|\theta_t\|^2 < \infty$ for $t \leq T_{\max}$. The Lipschitz gradient then gives integrability of the stopped first-order terms and value differences.

A.1.2 History-dependent learning efficiency

Write $A_{k,t} = \langle \nabla V_k(\theta_t), g_{k,t}\rangle$ for the direct learning contribution and $C_{j\to k,t} = \langle \nabla V_k(\theta_t), g_{j,t}\rangle$ for transfer from capability j to k. Positive $C_{j\to k,t}$ means beneficial transfer; negative values mean interference. These quantities are evaluated at θ_t using the conditional update directions $g_{j,t}$.

Theorem 1 (Quota-path decomposition)**.** *Under the setup and assumptions of Sec. A.1.1, the learning efficiency in equation 1 satisfies*

$$\eta_k(N;\mathcal{Q}) = \frac{1}{N}\mathbb{E}\left[\sum_{t<T}\alpha_t\left(q_{k,t}A_{k,t} + \sum_{j\neq k} q_{j,t}C_{j\to k,t}\right)\right] + \mathcal{R}_N, \tag{12}$$

$$|\mathcal{R}_N| \leq \frac{LG^2}{2N}\mathbb{E}\left[\sum_{t<T}\alpha_t^2\right]. \tag{13}$$

Proof of Theorem 1. The L-Lipschitz gradient condition gives, at every update,

$$V_k(\theta_{t+1}) - V_k(\theta_t) = \alpha_t\langle \nabla V_k(\theta_t), \widehat{g}_t\rangle + r_t, \qquad |r_t| \leq \frac{L\alpha_t^2}{2}\|\widehat{g}_t\|^2. \tag{14}$$

Summing to $T \le T_{\max}$ and taking expectations gives

$$\begin{aligned}&\mathbb{E}[V_k(\theta_T) - V_k(\theta_0)] \\ &\quad = \sum_{t=0}^{T_{\max}-1} \mathbb{E}\big[\mathbf{1}\{t < T\}\big(V_k(\theta_{t+1}) - V_k(\theta_t)\big)\big]\,. \end{aligned} \tag{15}$$

The variables $\mathbf{1}\{t < T\}$, α_t, and θ_t are $\mathcal{F}_t$-measurable. Taking conditional expectations of the first-order term in equation 14 therefore yields

$$\begin{aligned}&\mathbb{E}[\mathbf{1}\{t < T\}\alpha_t\langle \nabla V_k(\theta_t), \widehat{g}_t\rangle] \\ &\quad = \mathbb{E}\left[\mathbf{1}\{t < T\}\alpha_t \sum_{j=1}^{K} q_{j,t}\langle \nabla V_k(\theta_t), g_{j,t}\rangle\right]. \end{aligned} \tag{16}$$

Separating $j = k$ and dividing by N gives equation 12, with $\mathcal{R}_N = N^{-1}\mathbb{E}[\sum_{t<T} r_t]$. The conditional second-moment bound gives

$$|\mathcal{R}_N| \le \frac{L}{2N}\mathbb{E}\left[\sum_{t<T}\alpha_t^2\|\widehat{g}_t\|^2\right] \tag{17}$$

$$\le \frac{LG^2}{2N}\mathbb{E}\left[\sum_{t<T}\alpha_t^2\right], \tag{18}$$

which is equation 13. □

A.1.3 A SUPPRESSIVE REGIME

Theorem 1 covers positive and negative cross-capability transfer. The readiness model below assumes that readiness is nonincreasing in the amount of off-capability training.

Training between equal-exposure milestones. Fix two deterministic quota paths. To measure exposure between update boundaries, extend each path to $q_k(r) = q_{k,\lfloor r\rfloor}$ and define

$$E_k(t) = \int_0^t q_k(r)\,dr, \qquad \tau_k^{\mathcal{Q}}(u) = \inf\{t : E_k(t) \ge u\}. \tag{19}$$

At exactly attained update boundaries, this agrees with equation 11. For an integer $m \ge 1$, set $\delta = N/m$. The cumulative amount of training on other capabilities, and its amount in interval n, are

$$H_k^{\mathcal{Q}}(u) = \int_0^{\tau_k^{\mathcal{Q}}(u)} (1 - q_k(r))\,dr = \tau_k^{\mathcal{Q}}(u) - u, \qquad h_{k,n}^{\mathcal{Q}} = H_k^{\mathcal{Q}}(n\delta) - H_k^{\mathcal{Q}}((n-1)\delta). \tag{20}$$

Set $\tau_k^{\mathcal{Q}}(0) = H_k^{\mathcal{Q}}(0) = 0$. Each interval contains δ exposure to capability k, and $h_{k,n}^{\mathcal{Q}}$ counts all off-capability training within that interval, including interleaved updates. Assume the two paths satisfy

$$h_{k,n}^{\mathcal{Q}^+} \le h_{k,n}^{\mathcal{Q}^-} \quad \text{for every } n = 1, \dots, m. \tag{21}$$

For a constant quota $q_k > 0$, $h_{k,n} = \delta(1 - q_k)/q_k$ in every interval.

Readiness assumptions. Let the scalar $s_{k,n}$ measure readiness to learn capability k at the start of interval n. Assume the interval follows

$$\widetilde{s}_{k,n} = \mathsf{D}_{k,h_{k,n}}(s_{k,n}), \qquad s_{k,n+1} = \mathsf{U}_k(\widetilde{s}_{k,n}). \tag{22}$$

The map $\mathsf{D}_{k,h}$ describes the effect of training on other capabilities, and U_k describes the effect of δ exposure to capability k. We assume these maps are shared by both paths and that $s_{k,n}$, $h_{k,n}$, and δ determine each interval's state update.

Assume $\mathsf{D}_{k,h}(s)$ is nondecreasing in s and nonincreasing in h, and that U_k, rollout quality r_k, and held-out score v_k are nondecreasing in readiness. At the common initial state and each path's exactly attained final checkpoint, assume

$$V_k(\theta_0) = v_k(s_{k,1}), \qquad \mathbb{E}[V_k(\theta_{\tau_k^{\mathcal{Q}}(N)})] = v_k(s_{k,m+1}^{\mathcal{Q}}). \tag{23}$$

Proposition 2 (QEL in a suppressive regime)**.** *Suppose the two paths start from the same policy and readiness state and attain the same final exposure N. Under equation 21, the readiness assumptions, and equation 23, for every $n = 1, \dots, m$,*

$$\widetilde{s}^{\mathcal{Q}^+}_{k,n} \geq \widetilde{s}^{\mathcal{Q}^-}_{k,n}, \qquad r_k(\widetilde{s}^{\mathcal{Q}^+}_{k,n}) \geq r_k(\widetilde{s}^{\mathcal{Q}^-}_{k,n}), \tag{24}$$

and the final score and efficiency obey

$$v_k(s^{\mathcal{Q}^+}_{k,m+1}) \geq v_k(s^{\mathcal{Q}^-}_{k,m+1}), \qquad \eta_k(N; \mathcal{Q}^+) \geq \eta_k(N; \mathcal{Q}^-). \tag{25}$$

Proof. Assume inductively that $s^+_{k,n} \geq s^-_{k,n}$. The intervalwise inequality equation 21 and monotonicity of D give

$$\mathsf{D}_{k,h^+_{k,n}}(s^+_{k,n}) \geq \mathsf{D}_{k,h^+_{k,n}}(s^-_{k,n}) \geq \mathsf{D}_{k,h^-_{k,n}}(s^-_{k,n}). \tag{26}$$

Monotonicity of U_k therefore gives $s^+_{k,n+1} \geq s^-_{k,n+1}$. Induction from the common initial state proves the state order, and monotonicity of r_k proves equation 24. Equation 23 and the common initial score give the final-score order. Subtracting that common score and dividing by the common exposure N proves equation 25. □

Strict inequalities. Assume the hypotheses of Proposition 2. Suppose some interval n_0 satisfies

$$\mathsf{D}_{k,h^+_{k,n_0}}(s^-_{k,n_0}) > \mathsf{D}_{k,h^-_{k,n_0}}(s^-_{k,n_0}).$$

Assume additionally that U_k and v_k are strictly increasing, and that $\mathsf{D}_{k,h^+_{k,n}}$ is strictly increasing in its state argument for every $n > n_0$. Equation 26 gives $\widetilde{s}^+_{k,n_0} > \widetilde{s}^-_{k,n_0}$. The same induction propagates this strict order to $s^+_{k,m+1} > s^-_{k,m+1}$, and equation 23 gives strict inequalities in equation 25.

A concrete suppressive model. For $s \in [0,1]$, take $\mathsf{D}_{k,h}(s) = e^{-\lambda h}s$, $\mathsf{U}_k(s) = s + \kappa s(1-s)$, and $r_k(s) = v_k(s) = s$, with $\lambda > 0$ and $0 < \kappa < 1$. These maps preserve $[0,1]$. Their derivatives satisfy $\partial_s \mathsf{D} = e^{-\lambda h} > 0$, $\partial_h \mathsf{D} = -\lambda e^{-\lambda h}s < 0$ for $s > 0$, and $\mathsf{U}'_k(s) = 1 + \kappa - 2\kappa s \geq 1 - \kappa > 0$. Under equation 21, positive initial readiness and $h^+_{k,n_0} < h^-_{k,n_0}$ in one interval satisfy the strictness conditions above.

Exposure times and interval loads. Let $N = 2$, with $q^+_k(t) = 1$ on $[0,1)$ and $q^+_k(t) = 1/3$ on $[1,4]$, and let $q^-_k(t) = 1/2$ on $[0,4]$. Then

$$\tau^+_k(u) = \begin{cases} u, & u \in [0,1], \\ 3u - 2, & u \in [1,2], \end{cases} \qquad \tau^-_k(u) = 2u. \tag{27}$$

These paths satisfy $\tau^+_k(u) \leq \tau^-_k(u)$ for all $u \in [0,2]$. With $\delta = 1$, their interval loads are $(h^+_{k,1}, h^+_{k,2}) = (0,2)$ and $(h^-_{k,1}, h^-_{k,2}) = (1,1)$. Thus $h^+_{k,2} = 2 > 1 = h^-_{k,2}$, which violates equation 21.

A.1.4 A SUFFICIENT EFFECTIVE-QUOTA THRESHOLD

Corollary 3 (Sufficient effective-quota threshold for nonprogress)**.** *Under the hypotheses of Theorem 1, assume a constant step size $\alpha_t = \alpha > 0$ and constants $a, b > 0$ such that, for every $t < T$, almost surely,*

$$A_{k,t} \leq a, \qquad \sum_{j \neq k} q_{j,t} C_{j \to k,t} \leq -b(1 - q_{k,t}). \tag{28}$$

Define the effective quota as $q^{\mathrm{eff}}_{k,N} = N/\mathbb{E}[T]$, the cumulative exposure divided by the expected number of updates. If $b > LG^2\alpha/2$, then

$$q^{\mathrm{eff}}_{k,N} \leq q^\star_{k,\alpha} := \frac{b - LG^2\alpha/2}{a + b} \quad \Longrightarrow \quad \eta_k(N; \mathcal{Q}) \leq 0. \tag{29}$$

In the small-step limit, $q^\star_{k,\alpha} \to b/(a+b)$.

Proof. Exact attainment gives $\sum_{t<T} q_{k,t} = N$ and $\sum_{t<T}(1 - q_{k,t}) = T - N$ almost surely. Applying Theorem 1 and its remainder bound gives

$$\eta_k(N; \mathcal{Q}) \le \alpha a - \alpha b \frac{\mathbb{E}[T] - N}{N} + \frac{L\alpha^2 G^2}{2} \frac{\mathbb{E}[T]}{N} \tag{30}$$

$$= \alpha \left[a - b \frac{1 - q^{\text{eff}}_{k,N}}{q^{\text{eff}}_{k,N}} \right] + \frac{L\alpha^2 G^2}{2 q^{\text{eff}}_{k,N}}, \tag{31}$$

where $q^{\text{eff}}_{k,N} = N/\mathbb{E}[T]$. Multiplying the right-hand side by the positive quantity $q^{\text{eff}}_{k,N}/\alpha$ shows that it is nonpositive whenever

$$(a + b) q^{\text{eff}}_{k,N} - b + \frac{L\alpha G^2}{2} \le 0, \tag{32}$$

which is equation 29. □

For a constant quota, $q^{\text{eff}}_{k,N} = q_k$. The small-step limit holds with a, b, L, G fixed.

A.1.5 Learning signals under binary-reward GRPO

For a fixed prompt and policy, suppose the rewards of $G_r \ge 2$ rollouts are independent Bernoulli trials with success probability p. Call a group informative if it contains both a success and a failure. Its probability is

$$\Pr(\text{informative} \mid p) = 1 - \Pr(\text{all success}) - \Pr(\text{all failure}) = 1 - p^{G_r} - (1-p)^{G_r} = \iota_{G_r}(p). \tag{33}$$

Differentiating gives

$$\iota'_{G_r}(p) = G_r\big((1-p)^{G_r - 1} - p^{G_r - 1}\big). \tag{34}$$

Hence ι_{G_r} is increasing on $[0, 1/2]$ and decreasing on $[1/2, 1]$. As $p \to 0$, $\iota_{G_r}(p) = G_r p + O(p^2)$.

Groups with constant rewards have zero reward-centered GRPO advantages and zero task-gradient contribution under the zero-advantage convention for zero reward variance (Yu et al., 2025). For a prompt x in capability k, write $p_x(\theta)$ for success probability and $a_{k,x}(\theta)$ for the expected inner product of the task-gradient estimator with $\nabla V_k(\theta)$. Let $\mu_{k,x}(\theta)$ be this expected alignment conditional on an informative group. For $0 < p_x(\theta) < 1$, conditioning on whether the group is informative gives

$$a_{k,x}(\theta) = \iota_{G_r}(p_x(\theta)) \mu_{k,x}(\theta). \tag{35}$$

At $p_x(\theta) \in \{0, 1\}$, $a_{k,x}(\theta) = 0$. For two policy states, write $p^{\pm}_x = p_x(\theta^{\pm})$ and $\mu^{\pm}_x = \mu_{k,x}(\theta^{\pm})$. Assume $0 < p^-_x \le p^+_x \le 1/2$ and $0 \le \mu^-_x \le \mu^+_x$. Then

$$a_{k,x}(\theta^+) - a_{k,x}(\theta^-) = \big[\iota_{G_r}(p^+_x) - \iota_{G_r}(p^-_x)\big]\mu^-_x + \iota_{G_r}(p^+_x)\big(\mu^+_x - \mu^-_x\big) \ge 0. \tag{36}$$

For $p^-_x = 0$ and $0 < p^+_x < 1$, the condition $\mu^+_x \ge 0$ gives $a_{k,x}(\theta^+) \ge a_{k,x}(\theta^-) = 0$; $p^+_x \in \{0, 1\}$ gives equality. For $0 < p^-_x < p^+_x \le 1/2$ and $0 < \mu^-_x \le \mu^+_x$, the first term in equation 36 is positive and the second is nonnegative, so the inequality is strict. Averaging the pointwise inequality over the same prompt distribution gives the corresponding inequality for expected alignment.

Under the independent binary-reward assumptions above, $\iota_2(p) = 2p(1-p)$ is the expectation of FEAST's two-rollout disagreement indicator.

A.1.6 Greedy starvation

Consider a deterministic greedy scheduler with utility estimate $z_{k,t}$. Assume $z_{k,t} \le 0$ implies $q_{k,t} = 0$, and $q_{k,t} = 0$ implies $z_{k,t+1} \le z_{k,t}$. Fix an update t_0 with $z_{k,t_0} \le 0$. The two assumptions give $q_{k,t_0} = 0$ and $z_{k,t_0+1} \le 0$. Induction yields $z_{k,t} \le 0$ and $q_{k,t} = 0$ for every $t \ge t_0$.

These results complement analyses of gradient imbalance in multi-task RL (Wu et al., 2025), adaptive sampling for worst-task return gaps (Corrado et al., 2026), and weak task-gradient signals from low reward variance (Wang et al., 2026). QEL focuses on how allocation history changes learning efficiency at equal capability-specific exposure.

A.2 FEAST Details and Fair-Service Analysis

Algorithm 1 summarizes FEAST's training loop. The following subsections specify its grouping, allocation, and feedback rules and prove the group-opportunity bound.

Algorithm 1 FEAST training

Input: Policy π_{θ_0}, prompt pool $\mathcal{X}$, verifier, update horizon T.
Budget: B rollouts per update, shared equally among M selected clusters.

```
Initialize persistent atomic units, their histories, and capability clusters.
for t = 0, ..., T − 1 do
    Periodically regroup units using current-policy representations and learning trends.
    Compute positive cluster weights from atomic histories (Eqs. 39–41).
    Sample M clusters without replacement; reserve B_c = B/M rollouts for each.
    D_t ← ∅; hold π_{θ_t} fixed during generation.
    for each selected cluster c do
        Sample prompts using atomic priorities and remaining prompt capacity.
        Generate and score an equal number of pilot rollouts per prompt.
        Spend the remaining rollouts in rounds, retaining prompts by cumulative reward variance.
        Add all B_c rollouts to D_t, grouped by prompt.
    end for
    θ_{t+1} ← PolicyUpdate(θ_t, signal-bearing prompt groups in D_t).
    Refresh pilot-based attainment and productivity, learning trends, and rollout shortfalls.
end for
return π_{θ_T}
```

A.2.1 Persistent units and policy-adaptive groups

We embed the unlabeled training prompts with the base policy and partition them into $J = 64$ atomic units. Each prompt is rendered with the model's chat template, including the assistant-generation prefix, and truncated to the training input cap (2,048 tokens for GURU-18K). We use the final transformer hidden states after the model's final normalization and before the language-model head. The prompt vector is the attention-mask-weighted mean over all non-padding input tokens, including template tokens; no response tokens or ℓ_2 normalization are used. Pooling accumulates in float32. The initial partition fits Euclidean K-means to these raw prompt vectors, using K-means++ initialization, Lloyd iterations, ten restarts, at most 300 iterations, tolerance 10^{-4}. Each prompt keeps its nearest-centroid atomic assignment throughout training; the initial partition is not recomputed after input-length filtering.

Every ten optimizer updates, the current policy re-embeds the retained training pool using the same hidden layer and pooling rule. We take an equal-weight mean of prompt vectors within each atomic unit. The retained pool consists of prompts eligible for training after length filtering and need not contain every prompt used to construct the initial partition. Let $e_{i,t}$ be the resulting centroid. We center the 64 centroids, project them onto the leading eight right-singular vectors, and divide each projected coordinate by its empirical standard deviation, obtaining $z_{i,t} \in \mathbb{R}^8$. The regrouping feature is $[z_{i,t}, \sqrt{\lambda_y}\,\bar{\ell}_{i,t}]$, where $\bar{\ell}_{i,t}$ is a slowly smoothed signed progress signal and $\lambda_y = 1$.

For each component count from 8 through 16, we fit a diagonal-covariance Gaussian mixture and compute its BIC. We then repair assignments, using log posterior responsibilities to avoid probability underflow, to require at least two atomic units per group. Among feasible repaired assignments, we select the component count with the smallest BIC of its original fitted mixture. Regrouping changes only the map $i \mapsto c_t(i)$, while all attainment, productivity, and exposure statistics remain indexed by atomic unit.

The states $m, u, \ell, \bar{\ell}$ are initialized to zero and remain unchanged for atoms that are not sampled. Pilot feedback uses the common two-rollout first round. For each served prompt x, let $R_{x,j} \in [0,1]$ be the raw verifier reward on pilot j, preserving partial credit. The attainment observation is the prompt-equal mean $\widehat{m} = |P|^{-1} \sum_{x \in P} (R_{x,1} + R_{x,2})/2$, and the productive-signal observation is $\widehat{u} = |P|^{-1} \sum_{x \in P} (R_{x,1} - R_{x,2})^2$. With binary rewards the latter equals the disagreement indicator, while for partial rewards it retains the magnitude of the difference. The first observation sets m and

u to $\widehat{m}$ and $\widehat{u}$; subsequent observations use $m_{i,t+1} = 0.9m_{i,t} + 0.1\widehat{m}$ and $u_{i,t+1} = 0.9u_{i,t} + 0.1\widehat{u}$. Promotion and optimizer-signal detection also use the original verifier scores.

Let $n_{i,t}^{\mathrm{sig}}$ be the number of generated trajectories in the served batch belonging to prompt groups with nonzero verifier-score variance. The instantaneous signed trend and the slower regrouping trend are

$$\ell_{i,t+1} = \mathrm{clip}\left(\frac{m_{i,t+1} - m_{i,t}}{n_{i,t}^{\mathrm{sig}}/8}, -1, 1\right), \qquad \bar{\ell}_{i,t+1} = 0.7\bar{\ell}_{i,t} + 0.3\ell_{i,t+1}, \tag{37}$$

with $\ell_{i,t+1} = 0$ on the first observation or when $n_{i,t}^{\mathrm{sig}} = 0$. FEAST tracks each atom's accumulated rollout shortfall relative to an equal per-update share. With $J = 64$ atoms, equal-share increment $b = B/J = 16$, and all-round generated trajectory count $g_{i,t}$, it updates

$$d_{i,0} = 0, \qquad d_{i,t+1} = \max\{0, d_{i,t} + b - g_{i,t}\}, \qquad a_{i,t} = \min\{1, d_{i,t}/(10b)\}. \tag{38}$$

Shortfalls are updated for all atoms, counting zero-variance trajectories as exposure.

A.2.2 Cluster priorities and prompt quotas

The atomic priority is

$$r_{i,t} = \epsilon + u_{i,t}\,\phi(m_{i,t})\,\psi(m_{i,t})\big(1 + \gamma|\ell_{i,t}|\big) + \lambda_a a_{i,t}, \tag{39}$$

where

$$\phi(m) = \exp\left[-\frac{1}{2}\left(\frac{m - 1/2}{\sigma}\right)^2\right], \qquad \psi(m) = 1 + \frac{\eta}{1 + \exp((m - 1/2)/\tau)}. \tag{40}$$

The factor Φ in equation 6 is $\Phi(m) = \phi(m)\psi(m)$. The resolved values are $\epsilon = 0.05$, $\sigma = 0.22$, $\eta = 0.20$, $\tau = 0.05$, $\gamma = 0.50$, and $\lambda_a = 0.05$. Because $u, m, a \in [0, 1]$ and $|\ell| \le 1$, the current instantiation obeys $0.05 \le r_{i,t} \le 1.90$.

Let $n_{i,t}^{\mathrm{unseen}}$ count prompts in atom i that have never been sampled during training, let $P = 32$ be the prompt budget of a selected group, and let $U_{c,t} = \sum_{i:c_t(i)=c} n_{i,t}^{\mathrm{unseen}}$. For $U_{c,t} > 0$, define the capacity-weighted atomic mean $\bar{r}_{c,t} = U_{c,t}^{-1} \sum_{i:c_t(i)=c} n_{i,t}^{\mathrm{unseen}} r_{i,t}$. If $U_{c,t} = 0$, use the unweighted atomic mean. As lifetime-unseen capacity is exhausted, the group score shrinks toward the exploration floor according to

$$R_{c,t} = \epsilon + \min\left\{1, \frac{U_{c,t}}{P}\right\}(\bar{r}_{c,t} - \epsilon), \qquad \widetilde{w}_{c,t} = R_{c,t}^{1/\mathcal{T}}. \tag{41}$$

Here $R_{c,t} = w_{c,t}$ in the main-text notation. FEAST samples four groups without replacement using the next-draw probabilities in equation 7. Within a selected group, each of its 32 prompt slots is assigned sequentially to an atomic unit with probability proportional to its priority times remaining lifetime-unseen capacity. After this capacity is exhausted, the fallback uses each atomic unit's total eligible prompt capacity minus the slots already assigned to it in the current batch. Prompt identity selection is handled separately through stable per-atom without-replacement cycles, with repeats admitted after cycle exhaustion.

A.2.3 Fair-service bound

Proposition 4 (Non-absorbing group opportunity)**.** *Consider an indefinitely continued FEAST process. Suppose that at every update the current group containing a persistent atomic unit i is eligible, there are at most $C_{\max}$ eligible groups, and the temperature-scaled sampling weights satisfy $0 < \underline{w} \le \widetilde{w}_{c,t} \le \overline{w} < \infty$ almost surely. FEAST samples at least one group without replacement. Let W_i^{grp} count future updates through and including the first selection of the then-current group containing i, with $W_i^{\mathrm{grp}} = \infty$ if no such selection occurs. Conditional on any training history, with $p_{\min} = \underline{w}/(C_{\max}\overline{w})$ and integer $h \ge 0$,*

$$\Pr(W_i^{\mathrm{grp}} > h \mid \mathcal{F}_t) \le (1 - p_{\min})^h, \qquad \mathbb{E}[W_i^{\mathrm{grp}} \mid \mathcal{F}_t] \le p_{\min}^{-1}. \tag{42}$$

Proof. At each update, the first draw selects the current group containing i with probability $\widetilde{w}_{c_t(i),t}/\sum_c \widetilde{w}_{c,t} \geq \underline{w}/(C_{\max}\overline{w}) = p_{\min}$. Its inclusion probability is therefore at least $p_{\min}$. Conditioning successively on each update's history gives $\Pr(W_i^{\mathrm{grp}} > h \mid \mathcal{F}_t) \leq (1-p_{\min})^h$. The tail-sum formula then yields $\mathbb{E}[W_i^{\mathrm{grp}} \mid \mathcal{F}_t] = \sum_{h\geq 0} \Pr(W_i^{\mathrm{grp}} > h \mid \mathcal{F}_t) \leq \sum_{h\geq 0}(1-p_{\min})^h = p_{\min}^{-1}$. □

For the score above, one may take $\underline{w} = 0.05^{1/\mathcal{T}}$ and $\overline{w} = 1.90^{1/\mathcal{T}}$. With $\mathcal{T} = 1$ and $C_{\max} = 16$, the first-draw bound gives $p_{\min} = 1/608$.

A.2.4 Budget-preserving progressive allocation

For every selected group, the current schedule starts from 32 prompts with two rollouts each, retains the top half by cumulative verifier-score variance, adds four rollouts to each survivor, retains the top half again, and adds sixteen rollouts to each final survivor. Thus one group uses $32 \times 2 + 16 \times 4 + 8 \times 16 = 256$ trajectories. Four selected groups use exactly 1,024 trajectories, equal to assigning eight rollouts uniformly to 128 prompts. The final prompt-group sizes are 64 groups of size 2, 32 of size 6, and 32 of size 22. Stable tie-breaking is used when variance scores agree. The policy is held fixed across the within-update rounds, and all generated trajectories are associated with their original prompt before the reward-centered policy update. Prompt groups with zero verifier-score variance contribute no reward-centered optimizer signal, although an implementation may retain the minimum set of such groups solely for data-parallel alignment. All generated trajectories still count against the rollout budget.

A.2.5 Hyperparameter design rationale

Table 13 reports the default settings, which were fixed before the ablation studies. We summarize their theoretical and structural rationale below.

Atomic resolution. The atomic-unit count J controls the granularity of persistent statistics. Larger J permits finer tracking but requires more histories and centroids to maintain and regroup. With $J = 64$ and 8–16 dynamic groups, each group contains 4–8 atomic units on average, separating the resolution of persistent tracking from that of group-level allocation.

Exposure compensation. The positive priority floor ϵ supplies the lower bound used in Appendix A.2.3. The term $\lambda_a a_{i,t}$ adds a bounded correction for accumulated rollout shortfall. Since $a_{i,t} \in [0, 1]$, setting $\lambda_a = \epsilon = 0.05$ bounds this correction by the priority floor while allowing priority to increase with shortfall.

Pilot disagreement and the attainment frontier. For two conditionally independent rollouts of a fixed prompt under the same policy,

$$\mathbb{E}[(R_1 - R_2)^2 \mid x] = 2\,\mathrm{Var}(R \mid x). \tag{43}$$

With Bernoulli reward of success probability p, this becomes $2p(1-p)$, maximized at $p = 1/2$. This supports using pilot disagreement as a variance signal and motivates the moderate-attainment frontier, also used as a target in BOTS (Shen et al., 2026).

Absolute progress. Using $|\ell|$ allows both improvement and regression to increase priority. ALP-GMM (Portelas et al., 2020) similarly uses absolute learning progress to respond to improving and deteriorating competence.

Progressive rollout allocation. VIGOR (Jiang et al., 2026) provides a precedent for two initial rollouts and repeated top-half, variance-guided promotion. Appendix A.2.4 gives the budget accounting for FEAST's three-round schedule.

A.3 Experimental Protocol and Additional Results

GURU-18K construction. We construct a fixed pool with 3,000 examples from each of mathematics, code, science, logic, simulation, and tabular reasoning, following the per-domain sam-

pling rule of GURU (Cheng et al., 2025). Five domains come from the GURU-92K release at revision `0f7f6e8fc2cd36d46a9629310c9a4f9d8c1ffa3d`. The science domain is replaced by a pinned NVIDIA OpenScienceReasoning-2 (NVIDIA, 2025) subset (revision `174b02c9cdf231f220765b2a1d5ece4550921894`). Input-length filtering retains 17,067 of the 18,000 prompts. The same retained pool is used for each compared scheduler, and evaluation domain labels are not supplied to FEAST.

Training and comparison scope. The GRPO comparisons use Qwen3-1.7B-Base and Qwen3-8B-Base, training seeds 42/43/44 at 1.7B and seeds 42/43 at 8B, and the checkpoint after 100 optimizer updates at both scales. DAPO and DAPO+FEAST use 100 updates at 1.7B and 75 at 8B, while ODF uses 50 updates at 1.7B. Each GRPO update samples 128 prompts and generates exactly 1,024 online trajectories, including pilots and zero-variance groups. Random, TAC, ADS, DUMP, and BOTS use eight rollouts per prompt, while FEAST and its variant without Inter-Fairness redistribute the same budget through progressive rounds. We use GRPO with learning rate 5×10^{-7}, weight decay 0.1, PPO mini-batches of eight prompt groups, clipping bounds 0.2/0.2, and disabled reward KL and actor KL loss. Training prompt and response limits are both 2,048 tokens, with temperature 1.0 and top-p 1.0. Training and evaluation share the domain-specific verifier semantics. Code is checked with the pinned Reasoning360 `coder1` path and a local SandboxFusion-compatible service, with a 30-second program timeout. Science uses strict boxed multiple-choice scoring.

TAC and DUMP receive the six domain identities. TAC selects one domain for each update and samples all 128 prompts from that domain, alternating domains over training. The other schedulers allow batches to mix domains without requiring per-update coverage of all six domains. ADS uses 62 semantic clusters obtained by merging small clusters after a 64-cluster initialization. FEAST uses the persistent atomic units and current-policy regrouping described in Appendix A.2, with raw pilot rewards, squared pilot-reward differences, rollout shortfalls, log-responsibility GMM repair, absolute progress in its priority, and progressive increments $[2, 4, 16]$. The Inter-Fairness ablation removes the capability structure and samples 128 prompts globally without replacement before applying the progressive allocation. The Intra-Efficiency ablation retains FEAST's structure and allocation across clusters but fixes every response group at eight.

DAPO+FEAST combines FEAST's grouping and atomic priorities with eight rollouts per prompt, refilling until 128 signal-bearing prompt groups are retained. Selected clusters and atomic quotas remain fixed across refill batches. Attainment and productivity use the first two raw rewards of every candidate prompt, including rejected groups. The trend denominator counts trajectories retained for optimization. The shortfall increment equals the total generated trajectories per update divided by J, including refill, while its normalization remains $10 \times 1024/J$.

BOTS uses the official interpolation-based Bayesian selector with target success 0.5, forgetting coefficient 0.1, and implicit-evidence weight 0.1. Its prior features pair Qwen3-1.7B (the post-trained model) with Qwen2.5-7B-Instruct. We generate 16 reference responses for all 18,000 prompts with Qwen3-1.7B and for the 3,000 substituted science prompts with Qwen2.5-7B-Instruct, using a 4,096-token response cap. The other 15,000 prompts reuse published 7B pass-rate features. Newly generated reference evaluations therefore comprise 336,000 responses. The completed feature artifact is shared across the two policy scales.

Evaluation panels and metrics. Evaluation panels are taken from GURU (Cheng et al., 2025). The nine primary benchmarks use input cap 4,096 and evaluation seed 42. MATH-500 (Lightman et al., 2023), HumanEval (Chen et al., 2021a), MBPP (Austin et al., 2021), GPQA-Diamond (Rein et al., 2023), and SuperGPQA (M-A-P Team et al., 2025) use temperature 0.7 and top-p 0.8, while the other four benchmarks use temperature 1.0 and top-p 0.7. Scores average all sampled responses per question. Table 6 specifies the panel sizes and response caps for the primary and supplementary evaluations.

Zebra (Cheng et al., 2025) uses official cell-level partial credit. FinQA (Chen et al., 2021b) retains eight duplicate question–answer pairs and their original weights, giving 1,093 distinct identities among 1,101 records. CodeIO (Li et al., 2025) retains the official NaN reference at record 103, which receives zero.

Benchmark selection and supplementary evaluations. The main table reports MATH-500 for mathematics and the code benchmarks HumanEval and MBPP. GPQA-Diamond and SuperGPQA cover science, Zebra covers logic, CodeIO covers simulation, and FinQA and MultiHierTT (Zhao et al., 2022) cover tabular reasoning. These nine benchmarks span all six training domains and have complete evaluations for the reported checkpoints, allowing each method and seed to use the same aggregate.

The supplementary math panels use input cap 4,096, response cap 8,192, temperature 0.7, top-p 0.8, and evaluation seed 42. ARC400 (Chollet, 2019) is the 400-question ARC-AGI-1 panel, scored by exact-grid accuracy. Its input cap is 24,576 to retain all prompts within the 32,768-token context. Ordering100 uses input cap 4,096 and strict ordering accuracy. Both use temperature 1.0, top-p 0.7, and evaluation seed 42. For Ordering100, we report both the official score and a case-normalized score. The official scorer lowercases predictions but not reference answers. The case-normalized score applies the same normalization to the references while reusing the original responses.

Table 6: GURU-18K evaluation configuration.

Benchmark	Questions	k	Cap
MATH-500	500	1	4,096
HumanEval	164	4	4,096
MBPP	500	1	4,096
GPQA-Diamond	198	4	4,096
SuperGPQA	1,000	1	4,096
Zebra	200	4	8,192
CodeIO	200	4	8,192
FinQA	1,101	1	8,192
MultiHierTT	336	4	8,192
AMC	83	4	8,192

k denotes responses per question, and cap is the maximum number of response tokens.

Aggregation and uncertainty. Math, Logic, and Simulation report MATH-500, Zebra, and CodeIO, respectively. Code averages HumanEval and MBPP, Science averages GPQA-Diamond and SuperGPQA, and Tabular averages FinQA and MultiHierTT. Avg. equally weights the nine benchmarks, equivalent to domain weights 1:2:2:1:1:2. All aggregates use unrounded scores. Green and red mark the highest and lowest displayed means among the GRPO schedulers, including ties.

Table 2 reports mean $\pm$ sample standard deviation across training seeds 42/43/44 for 1.7B methods with three completed training seeds and across seeds 42/43 for all trained 8B methods, using denominator $n - 1$. Domain and overall scores are computed separately for each seed before calculating their means and SDs. Tables 7 and 8 list each checkpoint's nine primary benchmark scores and AMC for 1.7B and 8B, respectively. In the detailed tables, MATH, H.Eval, GPQA, S.GPQA, and M.Hier abbreviate MATH-500, HumanEval, GPQA-Diamond, SuperGPQA, and MultiHierTT, respectively. Base rows evaluate the original, untrained checkpoints at update zero and omit error bars. At 1.7B, ODF uses training seed 42.

Ablations use seed 42 at 1.7B with a matched full-method reference.

Table 7: Individual benchmark scores (%) for Qwen3-1.7B-Base, grouped by training seed. Trained checkpoints are at update 100, except ODF at update 50. Base is the untrained checkpoint at update 0. AMC uses an 8,192-token response cap and is excluded from the main-table Avg.

Method	MATH	AMC	H.Eval	MBPP	GPQA	S.GPQA	Zebra	CodeIO	FinQA	M.Hier
Base	51.40	27.41	46.19	52.40	15.03	12.30	1.46	2.88	8.90	13.47
Training seed 42										
Random	62.00	27.41	69.51	55.80	30.43	21.80	34.90	4.00	16.44	23.81
TAC	64.60	28.31	67.53	53.40	29.67	21.70	23.97	5.50	23.07	27.60
ADS	60.60	30.12	68.90	54.00	33.08	24.20	0.38	3.75	3.54	11.09
DUMP	57.80	28.61	68.29	54.00	32.58	21.00	34.63	5.00	19.16	27.23
BOTS	61.20	33.73	68.14	54.40	30.56	22.70	3.60	5.38	23.98	27.23
FEAST	61.80	30.42	68.90	53.80	31.69	21.40	29.50	5.00	26.98	27.83
DAPO	58.00	28.61	69.51	55.60	30.18	23.70	33.60	4.38	23.16	27.60
DAPO+FEAST	63.00	33.43	70.12	57.60	32.07	23.60	35.32	4.25	26.16	32.07

Continued on the next page.

Table 7 (continued). Qwen3-1.7B-Base.

Method	MATH	AMC	H.Eval	MBPP	GPQA	S.GPQA	Zebra	CodeIO	FinQA	M.Hier
ODF	58.40	29.52	68.29	56.20	32.58	21.20	1.13	3.75	17.89	27.83
w/o Inter-Fairness	57.80	29.82	66.92	53.80	29.29	24.50	33.74	5.25	22.62	27.01
w/o Intra-Efficiency	61.00	26.20	67.23	57.40	30.43	24.40	19.98	5.12	20.98	25.37
Training seed 43										
Random	60.60	29.22	67.99	55.00	32.95	23.00	34.26	4.12	18.17	24.48
TAC	63.80	29.82	67.53	55.00	30.05	22.40	28.10	4.00	25.07	28.35
ADS	57.00	24.10	65.85	52.40	31.44	24.40	1.34	4.25	5.54	11.76
DUMP	63.60	29.52	68.75	56.00	30.05	22.80	8.41	4.88	21.62	27.75
BOTS	62.40	27.11	70.12	54.20	33.59	21.80	3.61	4.88	21.16	26.86
FEAST	57.80	27.11	68.29	55.20	33.59	23.50	32.15	5.00	23.52	26.26
DAPO	62.60	31.33	68.45	54.40	36.99	22.60	33.46	3.88	23.07	29.24
DAPO+FEAST	60.00	31.33	70.27	54.60	34.72	22.50	34.40	3.62	26.70	31.99
Training seed 44										
Random	60.40	28.92	67.99	54.80	31.31	21.60	33.58	4.62	17.08	25.97
TAC	64.00	30.42	66.01	55.00	27.65	21.30	14.09	5.12	23.34	27.23
ADS	57.20	27.11	65.70	53.20	33.08	23.10	2.64	4.75	4.45	11.38
DUMP	60.40	30.42	68.60	54.60	30.18	23.70	32.90	6.25	16.62	23.96
BOTS	64.80	28.01	68.60	55.80	33.46	23.00	6.55	4.50	20.71	25.00
FEAST	61.60	31.63	68.45	56.20	28.91	22.90	33.36	4.38	22.43	26.41
DAPO	60.00	30.12	69.66	56.20	34.72	22.50	33.64	4.50	23.61	28.42
DAPO+FEAST	61.00	31.33	71.49	55.60	31.94	24.00	33.79	4.12	25.16	29.76

Table 8: Individual benchmark scores (%) for Qwen3-8B-Base, grouped by training seed. GRPO schedulers use update 100; DAPO and DAPO+FEAST use update 75. Base is the untrained checkpoint at update 0. AMC uses an 8,192-token response cap and is excluded from the main-table Avg.

Method	MATH	AMC	H.Eval	MBPP	GPQA	S.GPQA	Zebra	CodeIO	FinQA	M.Hier
Base	73.00	42.47	84.76	69.40	38.76	29.30	2.77	8.50	28.70	35.34
Training seed 42										
Random	78.80	56.93	88.87	73.20	42.93	34.20	34.65	5.50	41.69	49.55
TAC	78.20	51.81	87.96	73.60	44.57	34.50	38.50	10.38	39.87	50.67
ADS	78.80	54.22	85.82	74.00	48.48	34.90	16.65	6.25	41.24	47.99
DUMP	78.80	52.11	87.04	74.20	43.94	34.00	38.25	5.25	41.14	49.93
BOTS	81.20	54.52	86.74	73.40	46.84	35.60	28.11	6.00	41.51	52.75
FEAST	80.60	54.82	86.13	73.40	47.73	35.40	38.53	8.62	41.51	53.79
DAPO	76.40	53.61	86.89	75.00	42.55	35.40	38.40	7.50	38.87	50.22
DAPO+FEAST	79.20	55.12	89.02	72.60	46.09	36.80	38.77	5.88	40.78	52.31
Training seed 43										
Random	82.00	50.30	87.96	73.20	47.73	36.60	35.83	7.25	41.14	50.52
TAC	79.60	54.22	87.96	74.20	43.56	37.50	35.97	5.62	41.87	51.26
ADS	77.80	52.41	88.57	73.00	47.10	35.40	14.78	5.62	34.15	42.56
DUMP	79.80	52.11	85.06	74.20	47.35	35.50	37.17	6.50	42.60	51.19
BOTS	78.80	47.59	88.26	73.80	46.09	34.30	28.95	9.12	42.78	50.74
FEAST	80.20	51.51	86.43	72.80	43.56	36.00	37.60	13.25	41.42	53.50
DAPO	79.60	55.12	88.11	73.20	43.43	36.50	38.69	6.12	41.14	52.60
DAPO+FEAST	77.80	52.71	88.41	75.00	45.83	34.30	38.26	6.62	40.87	51.41

ODF results. We evaluate the balanced filtering variant of ODF (Bae et al., 2026) on GURU-18K with Qwen3-1.7B-Base and training seed 42, using the same optimizer settings and evaluation panels as DAPO. Each candidate prompt receives eight responses, and we retain the complete group when its empirical success rate satisfies $0.3 < \hat{p} < 0.7$, equivalent to three through five correct responses. We use DAPO's synchronous refill backend to collect 128 accepted groups per update, with a maximum of 64 candidate batches. We evaluate the update-50 checkpoint and count all generated responses, including rejected and excess groups, toward the reported 841,728 rollouts and 737.5M tokens (Table 9).

Table 9: ODF performance (%) and generation cost. Results use 1.7B with seed 42 at update 50.

Metric	1.7B
Math	58.40
Code	62.25
Science	26.89
Logic	1.13
Simulation	3.75
Tabular	22.86
Avg.	31.92
Rollouts (k)	841.7
Tokens (M)	737.5

Online token accounting. Table 10 reports computation for training seed 42 at the reported checkpoints. Online totals include non-padding prompt and response tokens for every generated trajectory, including pilots, progressive rounds, zero-variance groups, and refill attempts. Prompt tokens are counted once per rollout, without prefix-cache discounts. At 1.7B, the totals are 91.3M (Random), 107.6M (TAC), 74.5M (ADS), 94.3M (DUMP), 92.3M (BOTS), 101.4M (FEAST) and 357.8M (DAPO). DAPO uses clipping 0.2/0.28 with clip-c 10, no KL term, and at most 64 refill batches per update. It retains 128 groups of eight responses per update, generating 425,984 trajectories in total. At 8B, DAPO+FEAST generates 206,848 trajectories and consumes 201.0M tokens over 75 updates. Totals follow the retained training history.

The shared BOTS preparation artifact contains 336,000 newly generated reference responses, totaling 1,149.8M measured tokens, and is used at both policy scales.

Table 10: Online computation on GURU-18K (training seed 42). GRPO schedulers use 100 updates at both scales; DAPO and DAPO+FEAST use 100 at 1.7B and 75 at 8B. BOTS additionally uses a shared offline preparation artifact: 336.0k rollouts and 1,149.8M tokens.

	Qwen3-1.7B-Base		Qwen3-8B-Base	
Method	Rollouts (k)	Tokens (M)	Rollouts (k)	Tokens (M)
Base	0	0	0	0
Random	102.4	91.3	102.4	110.9
ADS	102.4	74.5	102.4	101.1
BOTS	102.4	92.3	102.4	108.8
DUMP	102.4	94.3	102.4	115.2
TAC	102.4	107.6	102.4	107.3
FEAST	102.4	101.4	102.4	99.8
DAPO	426.0	357.8	205.8	180.1
DAPO+FEAST	399.4	373.5	206.8	201.0

SMC protocol. The SMC training pool contains 11,975 prompts across 15 categories. Evaluation uses 30 questions per category, eight responses per question, temperature 0.7, top-p 0.8, and a 14,336-token response cap. All methods use training seed 42, with final checkpoints at update 100 for 1.7B and update 75 for 8B. Each compared GRPO scheduler generates 1,024 online rollouts per update, including pilots and zero-variance groups, giving 102,400 rollouts at 1.7B and 76,800 at 8B.

Table 11: SMC results by domain (avg@8). Avg. averages all 15 categories. Endpoints are update 100 at 1.7B and update 75 at 8B. Green and red mark the highest and lowest displayed scores within each scale, including ties.

	Qwen3-1.7B-Base				Qwen3-8B-Base			
Method	Math	Code	Science	Avg.	Math	Code	Science	Avg.
Random	57.44	**10.00**	72.74	61.42	72.50	32.50	90.89	78.42
ADS	**57.50**	10.42	**67.14**	**58.86**	**71.96**	**27.50**	**89.52**	**77.19**
DUMP	**55.71**	**17.08**	77.86	63.47	72.20	33.75	91.73	78.75
FEAST	56.67	**17.08**	**79.64**	**64.75**	**74.29**	**43.75**	**92.38**	**80.69**

For category k, let $\delta_{m,k} = r_{m,k} - r_{\mathrm{Random},k}$ denote the success-rate difference at the same model scale and checkpoint. Sorting these 15 differences in ascending order, Tail Value-Added is

$$\mathrm{TVA}_{1/3}(m) = \frac{100}{5} \sum_{i=1}^{5} \delta_{m,(i)}. \tag{44}$$

Thus the tail consists of the five least-improved categories.

Self-Evolving Curriculum (SEC) (Chen et al., 2025) updates category scores using temporal-difference learning with mean absolute advantage as feedback, then samples categories from a softmax distribution over these scores. Figure 5 shows SEC-Greedy, a diagnostic variant on SMC that replaces this stochastic selection with deterministic selection of the four highest-scoring categories after a 15-step balanced warmup. Its left panel reports held-out success, and its right panel counts class dispatches after warmup. Code receives no further dispatches, and Algebra and Prealgebra receive three each. These allocation patterns illustrate how greedy selection can deprive weak capabilities of fresh feedback, motivating FEAST's explicit protection of learning opportunities.

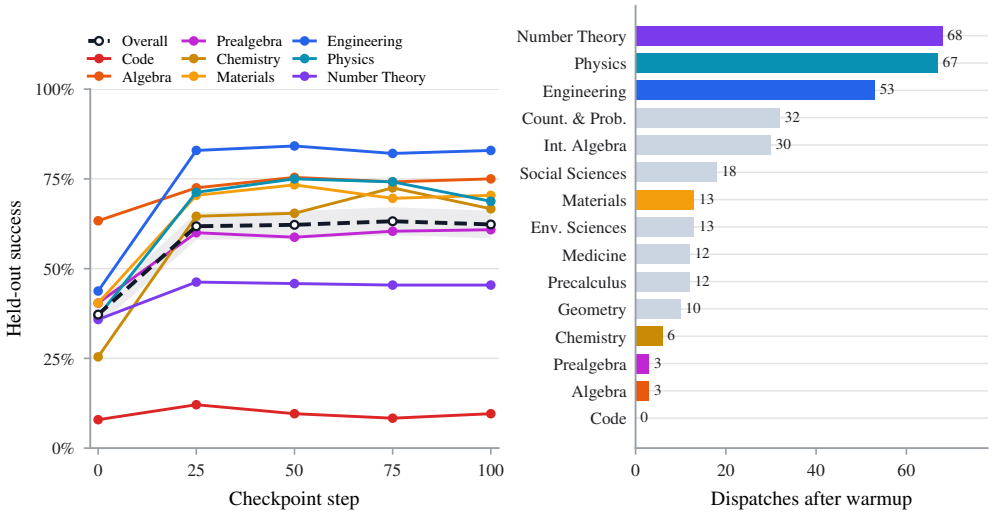


Figure 5: SEC-Greedy, a deterministic Top-4 variant of Self-Evolving Curriculum (SEC), on SMC (Qwen3-1.7B). The left panel shows held-out success for the overall panel and selected clusters. The right panel shows class allocation after the 15-step balanced warmup. Code is never selected after warmup, while Algebra and Prealgebra receive only three dispatches each. Colored bars match the left-panel clusters.

A.4 Domain-Level Allocation and Training Rewards

Figure 6 tracks the distribution of online rollouts across the six GURU-18K domains at both model scales. Counts include all generated trajectories before filtering, including FEAST's progressive rounds and DAPO's refill attempts. Each point pools counts within a non-overlapping five-update window and is plotted at the window midpoint.

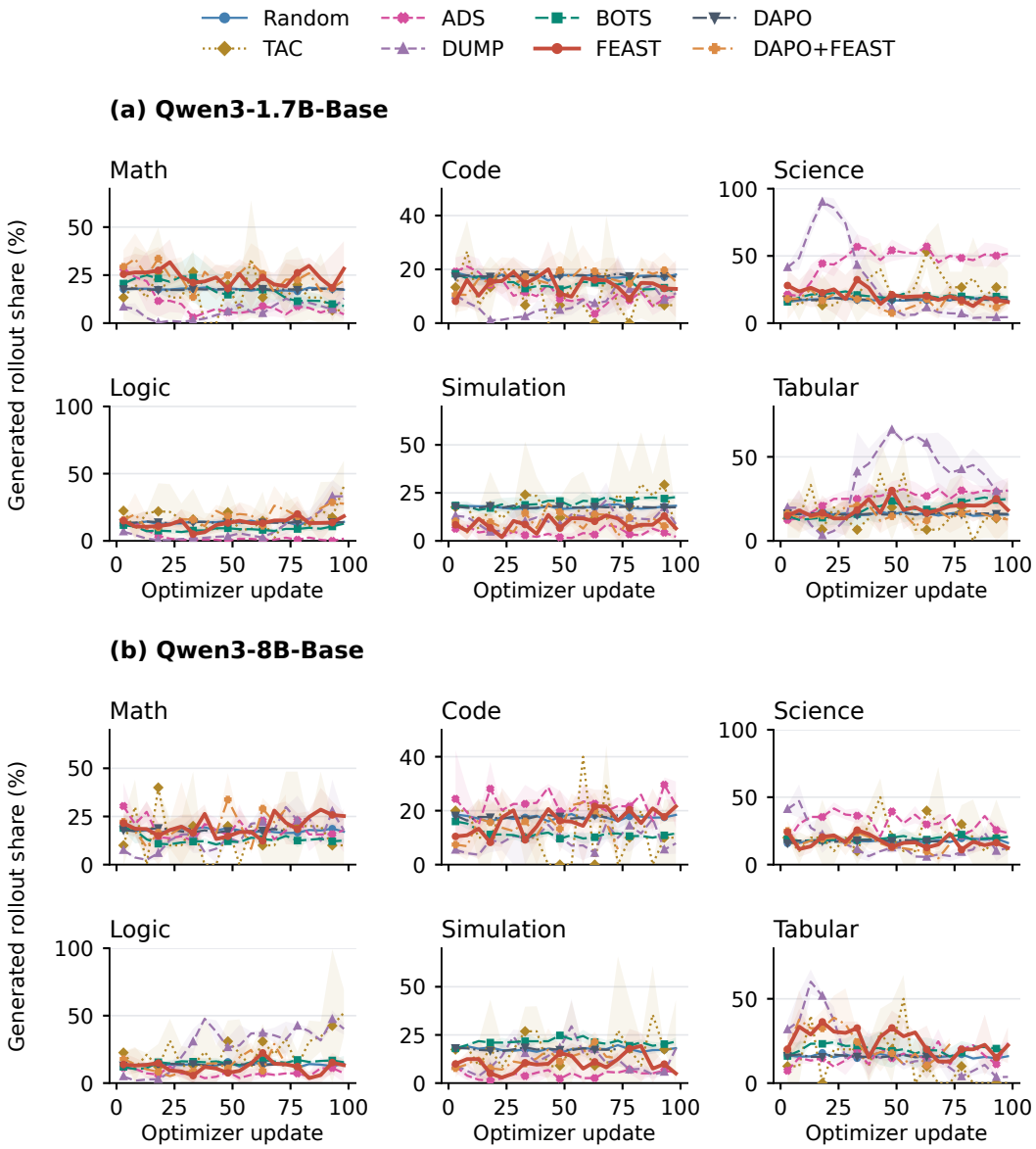


Figure 6: Domain-level rollout allocation on GURU-18K. **(a)** Qwen3-1.7B-Base, seeds 42/43/44. **(b)** Qwen3-8B-Base, seeds 42/43. Lines show seed means and shaded regions show sample SD. At 8B, DAPO and DAPO+FEAST end at update 75; all other runs end at update 100.

Figure 7 tracks the mean raw verifier reward over generated training trajectories within each domain, retaining partial credit. These rewards depend on the prompts selected by each method and therefore describe training dynamics rather than performance on a common held-out panel.

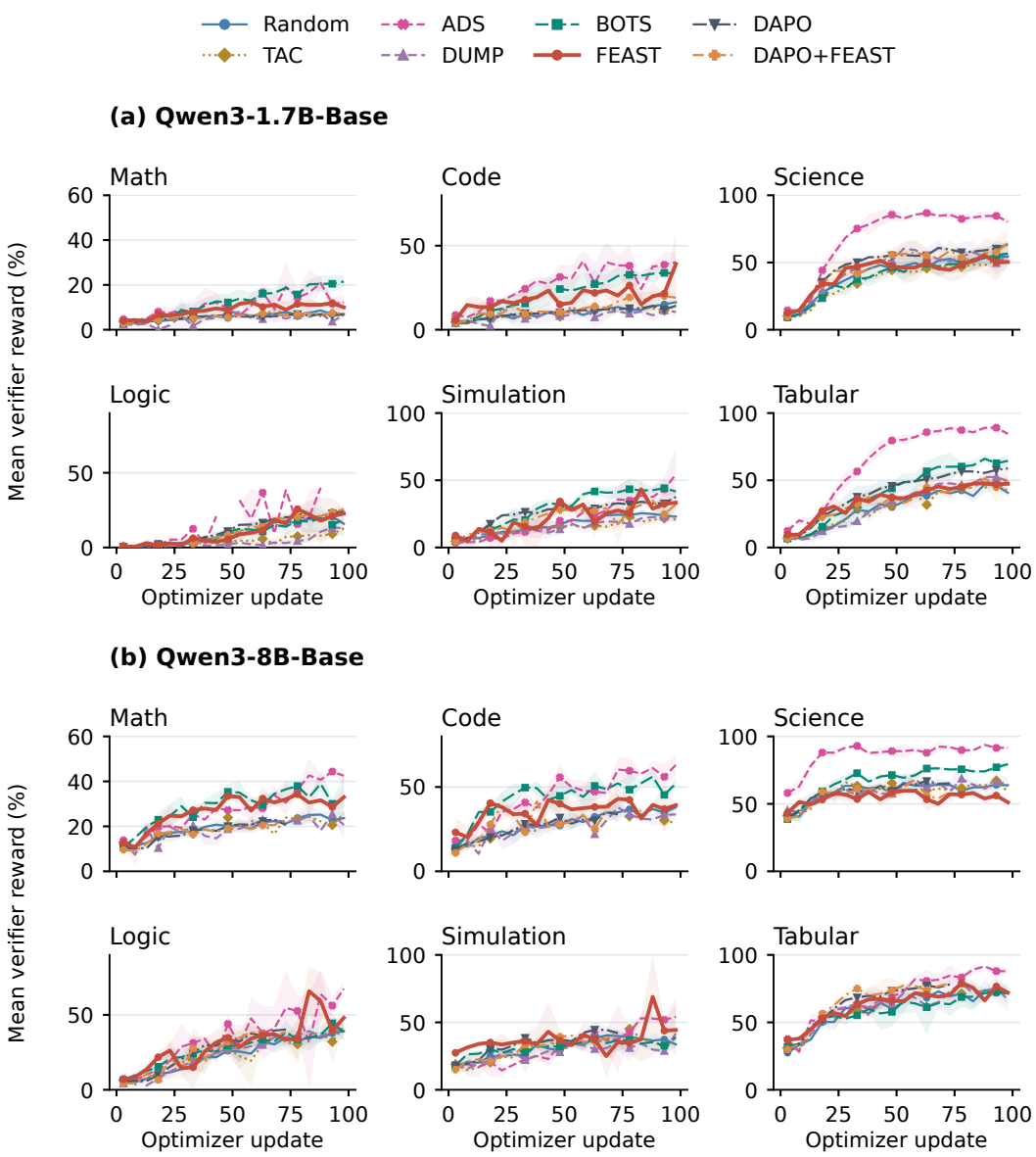


Figure 7: Domain-level training rewards on GURU-18K at **(a)** 1.7B and **(b)** 8B, with the seeds, windows, and training horizons of Figure 6. Lines and shaded regions show the mean and sample SD across seeds with domain samples. Windows with no samples from a domain have no reward estimate.

A.5 Additional QEL Evidence

Main-figure protocol. For the TextArena comparisons in Figure 4, both schedules use the same task instances in the same order for each task. We compute 95% Wilson intervals over 256 held-out instances per task. The Reasoning Gym panels in that figure use six completed 400-update runs, with zero-quota task–mixture pairs omitted. Their display smoothing applies a weighted three-point average $(0.2, 0.6, 0.2)$ followed by a Gaussian filter with bandwidth 350 prompt groups. Odd reflection preserves the observed endpoints; the numerical comparisons use the unsmoothed observations.

The additional Reasoning Gym (Stojanovski et al., 2025) comparisons use 256 fixed held-out instances per task. The 1.7B experiments use one response per instance; the 4B experiment averages

eight responses (avg@8). Each experiment retains its own panel, shared across ratios. Markers show observed checkpoints, connected without smoothing; numerical comparisons use exactly attained exposures.

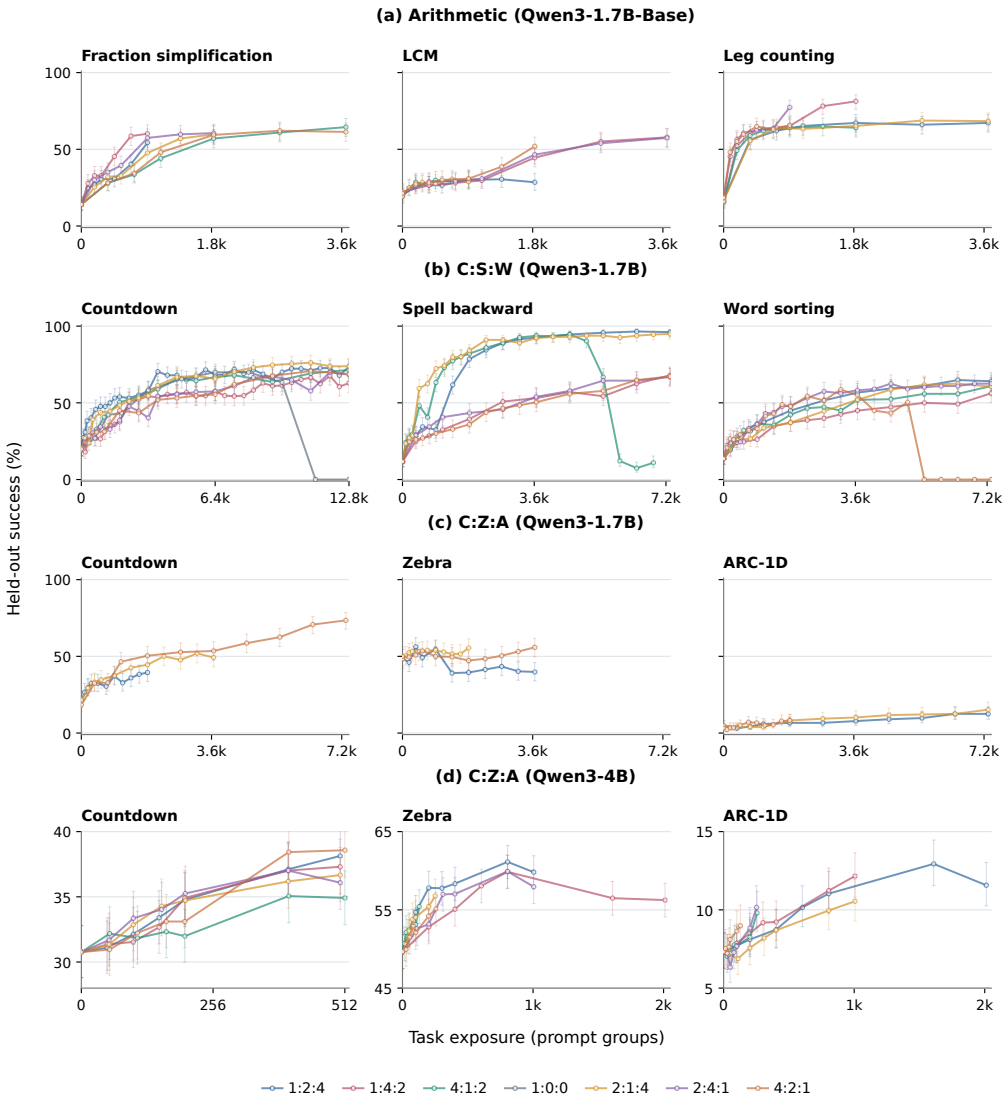


Figure 8: Additional QEL results at matched exposure; ratios follow each row's task order. Markers show unsmoothed observations: avg@1 in (a)–(c), avg@8 in (d). Error bars show pointwise 95% response-level Wilson intervals on fixed evaluation panels.

Task composition matters at fixed target-task quota. Figure 8(a) compares all six permutations of 1:2:4 over fraction simplification, least common multiple (LCM), and leg counting. These runs use Qwen3-1.7B-Base for 200 updates, with 32 prompt groups and eight rollouts per group, learning rate 10^{-5}, LoRA rank and scaling 32, and entropy coefficient 0.001. Evaluation uses temperature 0.6, top-p 0.95, and top-k 20.

At update 200, fraction simplification reaches 54.30% under 1:2:4 and 60.16% under 1:4:2. Both paths allocate it quota $1/7$ and exactly 914 prompt groups. Leg counting reaches 81.25% under 1:4:2 and 64.06% under 4:1:2, both with quota $2/7$ and exactly 1,828 groups. Within each of these two pairs, the ordered target-task prompts are identical. Changing the companion-task allocation changes

the policy path and the return from the same target-task sequence, consistent with the signed transfer and path-dependent alignment terms in Theorem 1.

Allocation-history gaps persist during extended training. Figure 8(b) uses a separate collection of Qwen3-1.7B runs on Countdown, Spell backward, and Word sorting (C:S:W). Training starts afresh with the first 400 updates' data preserved from the shorter protocol. The 1:2:4 and 1:4:2 runs reach 2,800 updates, the other mixed ratios reach 1,600, and the Countdown-only control reaches 400. The batch, rollout, learning-rate, LoRA, entropy, and evaluation settings match the preceding experiment. After a task/difficulty pool is exhausted, its fixed prompt permutation cycles. The longer-horizon runs use a separate frozen evaluation panel. The 4:2:1 run uses a different host from the other six runs.

At 6,400 Spell backward groups and update 1,400, 2:1:4 yields 93.75%, whereas 4:1:2 yields 7.42%. Both assign Spell backward quota $1/7$; the latter path had reached 94.53% at update 1,000 before declining. Conversely, at 12,800 Countdown groups, the single-task control scores 0.00% and 1:2:4 scores 73.44%.

QEL comparisons on an additional task set. Figure 8(c) compares three allocation ratios on Countdown, Zebra, and ARC-1D (C:Z:A) over 400 training updates.

QEL comparisons with a 4B model. Figure 8(d) adds six completed Qwen3-4B runs on C:Z:A, covering all permutations of 1:2:4. Training uses 32 prompt groups per update, eight rollouts per group, learning rate 10^{-5}, LoRA rank and scaling 32, and entropy coefficient 0.001. Runs stop after 28, 55, or 110 updates, when Countdown exposure first reaches at least 500 groups; the attained counts are 503 or 512. Evaluation averages eight responses per instance at temperature 0.6, top-p 0.95, and top-k 20. At exactly 512 Countdown groups and update 28, 4:2:1 reaches 38.57% and 4:1:2 reaches 34.91%, both with Countdown quota $4/7$. This is a descriptive allocation-path difference at 4B.

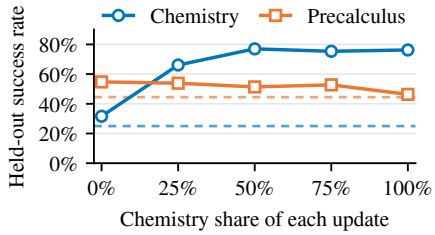


Figure 9: Held-out performance after 25 pure-random updates across fixed allocations. Dashed lines indicate the shared step-0 baselines.

Allocation diagnostics. Figure 9 varies the Chemistry–Precalculus allocation and evaluates held-out performance after 25 pure-random updates. Dashed lines mark the common initial policy.

A.6 Detailed Experimental Hyperparameters

Table 12 lists the TextArena QEL settings. QEL matches task exposure, whereas FEAST matches optimizer updates and rollout count.

Table 12: TextArena QEL hyperparameters. S:M:C denotes Sokoban:Mastermind:Countdown.

Hyperparameter	Value
Model	
Base policy	Qwen3-1.7B-Base
Adaptation	LoRA on all linear layers; rank 32; $\alpha = 32$
Distributed strategy	FSDP; one GPU per training arm
Environment	
Framework	TextArena
Sokoban	Room 6 × 6; one box; maximum 10 turns
Mastermind	Code length 3; symbols 1–4; no duplicates; maximum 8 turns
Countdown	Six numbers; target absent initially; shortest solution has two operations; maximum 5 turns

Continued on the next page.

Table 12 (continued).

Hyperparameter	Value
Policy reward	Strict terminal success: success $= 1$, every failure mode $= 0$
Quota schedule	
S:M:C ratios	1:1:1, 1:4:8, 4:1:8, 8:4:1, 4:8:1, 8:1:4
Prompt groups per update	32
Rollouts per prompt group	8 (256 trajectories per update)
Exposure milestones	$n \in \{256, 512, 1024, 2048\}$ selected prompt groups per task
Terminal updates	192 for 1:1:1; 832 for each skewed-ratio arm
Schedule/data seeds	1234 / 20260819
Optimization	
Objective	GRPO; one PPO epoch
Learning rate / schedule	1×10^{-5} / constant
Adam β / weight decay	$(0.9, 0.999)$ / 0.01
PPO mini-batch size	8 prompt groups
PPO clipping	lower 0.20; upper 0.28
Entropy coefficient	0.001
KL reward / actor KL loss	disabled / disabled
Actor token cap per GPU	10,240
Rollout	
Backend	vLLM server-based asynchronous rollout
Sampling	temperature 1.0; top-p 0.95; top-k 20
Per-turn response / thinking budget	256 / 192 tokens
Full trajectory / model context limits	8,192 / 12,288 tokens
Maximum concurrent sequences / batched tokens	256 / 32,768
Evaluation	
Frozen panel	256 deterministic instances per task; one trajectory per instance
Sampling	temperature 0.6; top-p 0.95; top-k 20
Response / thinking / trajectory limits	256 / 192 / 8,192 tokens
Metric	Strict terminal success (%); compared at matched task exposure n

Table 13: FEAST hyperparameters. Appendix A.2.5 summarizes their theoretical and structural rationale.

Hyperparameter	Value
Capability structure	
Persistent atomic units J	64; base-policy semantic partition; no task labels
Semantic refresh / regroup interval	Current policy / every 10 optimizer updates
Regrouping feature	8-dimensional semantic representation plus smoothed signed progress
Dynamic groups	8–16 diagonal-covariance mixture components; assignment repair to at least 2 atomic units per group
Persistent state	Raw attainment, squared pilot differences, learning trend, and rollout shortfall remain attached to atomic units
Inter-Fairness	
Selected groups M / prompts per group	4 / 32
Sampling	Sequential weighted sampling without replacement; temperature 1.0
Productive signal / attainment	First-round squared raw-reward difference and raw-reward mean; prompt-equal EMA coefficient 0.10
Priority floor / rollout shortfall	Floor $\epsilon = 0.05$; shortfall weight $\lambda_a = 0.05$; shortfall normalized by 10 equal-share rollout increments and capped at 1
Adaptive surplus	Frontier width $\sigma = 0.22$; under-mastered bonus $\eta = 0.20$; transition scale $\tau = 0.05$; absolute-progress coefficient $\gamma = 0.50$
Atomic prompt quota	Lifetime-unseen capacity weighted by atomic priority; eligible-capacity fallback after exhaustion
Intra-Efficiency	
Candidate / selected prompts per group	At most 64 candidates; 32 selected; stable without replacement within each atomic-unit cycle, with repeats only after exhaustion
Progressive rollout rounds	$[2, 4, 16]$ with top-half promotion by cumulative verifier-score variance
Final prompt-group sizes per update	64 groups of 2; 32 groups of 6; 32 groups of 22
Feedback for cluster selection	Attainment and productivity: prompt-equal first-round outcomes; progress: normalized by signal-bearing trajectories from all rounds
Optimizer filtering	Prompt groups with zero verifier-score variance do not contribute a reward-centered GRPO update
Total rollout budget	1,024 per update