Extended Abstract

Motivation How a model is taught—not only what it is taught—shapes how well it learns. Curriculum learning, ordering training by presenting tasks in a meaningful order, is a natural remedy for the sparse rewards that reinforcement learning with verifiable rewards (RLVR) faces on hard reasoning problems, and a recent Easy-to-Hard line reports large gains on 1.5–3B models. Yet the prevailing curricula are *preset*: the difficulty at each step is fixed in advance, blind to what the model has actually mastered. Human learning is not like this—we dwell just past the edge of our competence and return to skills that begin to slip. We ask whether an *adaptive*, knowledge-aware curriculum that mirrors this behaviour improves RLVR reasoning, and—since the question also bears on whether RLVR creates new ability or merely sharpens existing ability—we study both on Countdown, where difficulty is simply the operand count.

Method Our central contribution is a closed-loop scheduler that allocates training by where the model is *actually learning*. We frame difficulty selection as a non-stationary multi-armed bandit driven by a bidirectional learning-progress (LP) signal—the gap between a fast and a slow moving average of each level's success rate—so the curriculum chases the model's zone of proximal development and automatically revisits any level it begins to forget. Building on this, we introduce `lp_plus`, a difficulty-weighted exploration floor that guarantees the hardest learnable level a growing minimum share, curing the starvation a pure-progress bandit inflicts on a hard level whose signal is briefly flat. We compare both against a balanced baseline and the two preset Easy-to-Hard ramps (cosine and Gaussian).

Implementation We fine-tune the mandated Qwen2.5-0.5B base, warm-started with supervised fine-tuning, using RLOO against a rule-based verifier. To give an easy-to-hard curriculum a real foundation, we *synthesize* the two missing difficulty levels ourselves—2-op by exhaustive enumeration and 5-op by rejection sampling with forced exact divisors—building a balanced four-level dataset (we train on 2/3/4-op and hold out 5-op for out-of-distribution evaluation). All schedulers run under one identical regime so that only the schedule varies, and we evaluate with 16 samples per prompt after diagnosing and fixing an EOS-mismatch evaluation artifact.

Results The adaptive schedulers win. `lp_plus` is the single best scheduler in-distribution—best overall (pass@1 0.697) and best on the hardest level (4-op, 0.346)—and the LP family is the only curriculum to improve on balanced sampling, with an LP variant best at *every* difficulty level. The gains are real but modest, and our analysis turns that into a clear scientific picture: at this scale RLOO *sharpens* the base model's distribution rather than creating new ability (single-sample accuracy rises while coverage at $k{=}16$ does not), so the hardest level plateaus at a base-model ceiling shared by every schedule, alongside a collapse of policy entropy and output diversity. The same lens illuminates generalization: out-of-distribution performance is non-monotonic, peaking mid-training before continued sharpening collapses the output structure that harder, unseen problems require.

Discussion These results cohere into one message: a curriculum is an *allocation policy* over sharpening, and an adaptive, knowledge-aware allocation spends it best. Where the base model already holds latent capacity on hard levels—the 1.5–3B regime in which Easy-to-Hard reports its large gains—this is exactly the lever that converts coverage into reasoning; at 0.5B that capacity is largely absent, which is why every schedule converges toward a shared ceiling and the margins are small. Our `lp_plus` extracts the most of the headroom that does exist, and the same capacity argument predicts that its advantage should grow as the base model does.

Conclusion We contribute `lp_plus`, a learning-progress bandit with a difficulty-weighted floor that models the curriculum on how humans actually learn—training where progress is happening and protecting the hardest skills—and show it is the strongest scheduler in a controlled comparison. More broadly, we characterize *when* curriculum RL helps: it allocates a fixed budget of sharpening, helps most on the hardest learnable level, and is bounded only by the base model's capacity—a boundary our adaptive method is positioned to exploit as models scale.

When Does Curriculum Learning Help? A Knowledge-Aware Learning-Progress Curriculum for RLVR

Louis de Germay
Stanford University
louis2gc@stanford.edu

Arthur Gontier
Stanford University
agontier@stanford.edu

Abstract

How a model is taught, not only what it is taught, shapes how well it learns. We study curriculum learning for reinforcement learning with verifiable rewards (RLVR), fine-tuning a fixed Qwen2.5-0.5B base on Countdown, where difficulty is simply the operand count. Whereas the prevailing Easy-to-Hard curricula are *preset*—the difficulty at each step fixed in advance—our central contribution is a *knowledge-aware*, closed-loop scheduler that allocates training by where the model is actually learning: a non-stationary learning-progress bandit driven by the gap between a fast and a slow exponential moving average of per-level success, which chases the model's zone of proximal development and revisits levels it begins to forget. We extend it with `lp_plus`, a difficulty-weighted exploration floor that protects the hardest learnable level from starvation. Against a balanced baseline and the two parameter-free Easy-to-Hard ramps, our adaptive schedulers are the strongest in-distribution: `lp_plus` is best overall (pass@1 0.697) and best on the hardest level (4-op, 0.346), and the learning-progress family is the only curriculum to beat balanced. The gains are modest, and we explain why: at this scale RLOO *sharpens* the base model's distribution rather than creating new ability—pass@1 rises while pass@16 coverage does not, and the hardest level plateaus at a base-model ceiling, with policy entropy and output diversity collapsing. A curriculum thus reallocates a fixed budget of sharpening; an adaptive, knowledge-aware one spends it best, and the same capacity argument predicts its advantage should widen on larger, more capable models.

1 Introduction

Reinforcement learning with verifiable rewards (RLVR) has become the standard recipe for turning a pretrained language model into a competent reasoner: a programmatic verifier supplies a scalar reward, and a policy-gradient method such as PPO, GRPO, or RLOO pushes the model toward higher-reward generations. A central difficulty is that hard reasoning problems return reward only when solved, so early in training the model receives little or no signal on precisely the problems it most needs to learn—the sparse-reward problem. Curriculum learning—presenting tasks in a meaningful order to improve performance and learn faster (Bengio et al., 2009)—is a natural remedy: mastering easy instances first builds foundational skills and a denser reward signal, and narrows the distribution gap between what the base model can already do and the target task before the model is exposed to its hardest variants. This intuition drives a recent Easy-to-Hard (E2H) line that schedules training problems from easy to hard and reports large gains for RLVR reasoning (Parashar et al., 2026). These schedules are nonetheless preset: the difficulty sampled at step t is a fixed function of t, independent of what the model has so far mastered. This project pursues two goals. First, we design a curriculum that adapts to the model's current ability—allocating difficulty by where the policy is measurably learning rather than by a preset schedule. Second, we ask whether, and under

what conditions, such a difficulty curriculum helps at all—particularly when the base model is small and capacity-limited, a regime the E2H results (obtained on 1.5–3B models) leave open.

Interpreting curriculum results requires a position on a second, still-unresolved question: what RLVR does in the first place. It is widely assumed that RL lets a model acquire genuinely new reasoning ability, but recent work casts doubt. Yue et al. (2025) find that, measured by pass@k at large k, RLVR raises single-sample accuracy without expanding the base model's reasoning boundary—the base catches up once enough samples are drawn—and Cui et al. (2025) show that policy entropy collapses during RLVR, curbing exploration and saturating gains. These results point toward *sharpening* (re-ranking solutions the base model can already produce) rather than *emergence* (new capability), though the question remains open. The distinction bears directly on curricula: if RL mainly sharpens, a curriculum cannot add capability—it can only decide where a fixed amount of sharpening is spent. We therefore treat sharpening versus emergence not as a question we attempt to settle, but as an interpretive lens for understanding when, and why, a schedule can help.

We study these questions on Countdown, a compact arithmetic-reasoning task with a clean, monotone notion of difficulty. Given a target number and a list of numbers, the model must emit an equation (wrapped in `<answer>...</answer>` tags) that combines the numbers to reach the target; a deterministic verifier scores each generation 0.0 (no valid answer), 0.1 (well-formatted but wrong), or 1.0 (correct). Difficulty is simply the operand count, which lets us define discrete difficulty *levels* and treat "which level to sample at step t" as the central curriculum decision while holding everything else fixed. We fix the base model to Qwen2.5-0.5B, warm-start it with supervised fine-tuning (SFT), and run online RL with RLOO against the verifier reward.

We compare five difficulty schedulers, each producing a probability vector over levels at every step: a balanced baseline, the two E2H ramps of Parashar et al. (2026) (cosine and Gaussian), and a learning-progress (LP) bandit that treats levels as arms of a non-stationary multi-armed bandit and allocates budget by a learning-progress signal (the absolute difference of a fast and a slow exponential moving average of per-level success rate). Unlike these preset ramps, the LP bandit is *closed-loop*—it steers budget toward the levels where the model is actually improving and circles back to any that begin to slip, much as a learner lingers just past their current reach and returns to a fading skill. The LP bandit is adapted from the automatic-curriculum literature. Our methodological contribution is `lp_plus`, which replaces the bandit's *uniform* exploration floor with a *difficulty-weighted* one. The uniform floor is the wrong inductive bias for a small model: when the hardest level is barely solvable, both EMAs of its success rate sit near zero, its learning-progress signal vanishes, and it receives only the tiny uniform share—so the level most in need of exposure gets the least. `lp_plus` guarantees the hardest learnable level a protected, growing minimum budget, deliberately inverting the classical rule of *dropping* zero-progress tasks on the wager that protecting hard-level structure aids generalization.

We make three contributions. **(1)** `lp_plus`, our *ability-aware* curriculum scheduler: a learning-progress bandit whose difficulty-weighted exploration floor guarantees the hardest learnable level a protected, growing share, fixing the hard-level starvation to which uniform-floor bandits are structurally prone. **(2)** A clean, controlled comparison of open-loop and closed-loop difficulty schedulers on a fixed small baseholding the warm-started policy, RLOO algorithm, data, and budget constant, under a corrected evaluation protocol—in which our adaptive learning-progress schedulers are the strongest, an LP variant best at every difficulty level. **(3)** A mechanistic account—through pass@k, policy entropy, and output diversity—of *why* difficulty scheduling reallocates sharpening rather than creating capability at this scale.

2 Related Work

Curriculum learning and Easy-to-Hard RL for reasoning. Curriculum learning—ordering training from easy to hard (Bengio et al., 2009)—is a natural remedy for the sparse rewards of RLVR on hard problems. The most direct precursor to our work is the Easy-to-Hard (E2H) line of Parashar et al. (2026), who schedule problem difficulty probabilistically with the two parameter-free ramps we replicate—cosine (E2H-C) and moving-center Gaussian (E2H-G)—and report large gains on Countdown and other reasoning tasks (e.g. medium-difficulty accuracy $28.8 \rightarrow 70.4$) for models in the 1.5–3B range, arguing that fading out easy tasks is essential to avoid forgetting and overfitting. Other curricula for LLM post-training switch from easy to hard after a fixed number of steps or

filter problems online by on-policy difficulty. The evidence is mixed, however: in a controlled study across model families and schedules, Mordig et al. (2026) report "no robust advantage in difficulty-based sequencing over standard random sampling" under both SFT and RL on synthetic reasoning. We sit between these camps—faithfully reimplementing E2H's schedulers but on the 0.5B base its results leave untested—and read the contrast through model capacity.

Learning progress and automatic curricula as bandits. A second line selects tasks to maximize *learning progress*, originating in the intrinsic-motivation literature (Oudeyer et al., 2007; Baranes and Oudeyer, 2013) and cast as a non-stationary multi-armed bandit over tasks by Graves et al. (2017). Teacher–Student Curriculum Learning (Matiisen et al., 2019) combines an ε-exploration floor with sampling proportional to the absolute slope of reward, giving anti-forgetting behavior; ALP-GMM (Portelas et al., 2020) operationalizes "absolute learning progress" for continuous task spaces; and Kanitscheider et al. (2021) introduce the bidirectional learning-progress estimator—the absolute difference of a fast and a slow EMA of success rate—that our LP bandit adopts. This formalizes Vygotsky's Zone of Proximal Development. Closest to us, Self-Evolving Curriculum (Chen et al., 2025) frames difficulty selection for LLM RL as a non-stationary bandit with absolute advantage as the learning-gain proxy, and improves out-of-distribution generalization— though notably its gains shrink toward the random baseline on a stronger base model, an effect its authors attribute to that model already covering the harder problems. Our LP bandit is a faithful instantiation of this lineage; our contribution, `lp_plus`, is a single change—a difficulty-weighted exploration floor that protects the hardest learnable level—which inverts the classical prescription to abandon zero-progress arms.

What RLVR does: sharpening, entropy, and diversity. Interpreting curriculum effects requires a position on what RLVR does to a policy. Yue et al. (2025) show via pass@k that RLVR raises single-sample accuracy but does not expand—and can shrink—the base model's reasoning boundary: base models catch up or overtake at large k, and perplexity analysis places RL-sampled paths within the base model's distribution, so RLVR sharpens rather than creates, bounded by the base. Cui et al. (2025) document the mechanism on the entropy side: policy entropy collapses monotonically and is traded for reward along a predictable curve, implying a performance ceiling once entropy is exhausted—which conditionally supports a base-model bound. We adopt this sharpening-versus-emergence view as an interpretive lens for difficulty scheduling: if RL mainly reallocates a fixed pool of sharpening, a curriculum decides where that sharpening lands rather than adding capability.

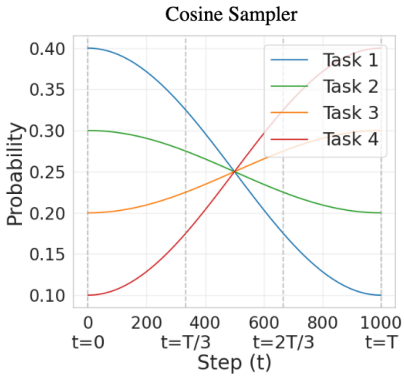
3 Method

Task formalization. A Countdown instance is a pair $(\mathcal{N}, \tau)$ of a multiset of source numbers $\mathcal{N} = \{n_1, \ldots, n_d\}$ and a target $\tau \in \mathbb{N}$. The policy must emit an arithmetic expression e, wrapped in `<answer>...</answer>` tags, that uses each source number exactly once with the operators $\{+, -, \times, \div\}$. A deterministic verifier $r : e \mapsto \{0.0, 0.1, 1.0\}$ scores a generation: $r = 0.0$ if no well-formed answer is found, $r = 0.1$ if the tagged expression parses but is wrong or uses the wrong numbers, and $r = 1.0$ if the expression is valid, uses exactly $\mathcal{N}$, and evaluates to τ. We define the *difficulty level* of an instance as its operand count $d \in \{2, 3, 4, 5\}$, which is the single axis our curriculum schedules over.

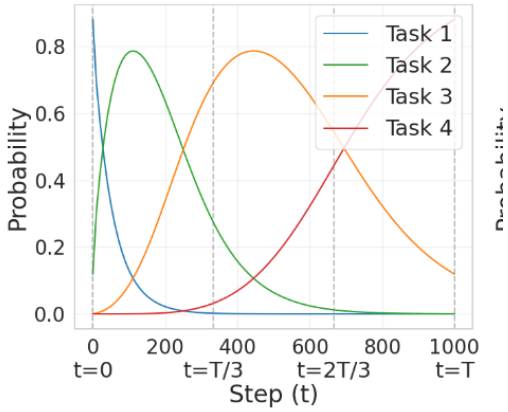
Pipeline and RL objective. The base model is fixed to Qwen2.5-0.5B. We warm-start with supervised fine-tuning (SFT) on correct chain-of-thought solutions, and then run online RL with RLOO (Reinforce Leave-One-Out). For each prompt we draw a group of G samples $\{y_1, \ldots, y_G\}$ from the current policy π_θ and apply the policy-gradient estimator

$$\nabla_\theta \mathcal{J}(\theta) \;=\; \mathbb{E}\left[\sum_{i=1}^{G} \big(r(y_i) - b_{-i}\big)\, \nabla_\theta \log \pi_\theta(y_i \mid x)\right], \qquad b_{-i} \;=\; \frac{1}{G-1}\sum_{j \neq i} r(y_j), \tag{1}$$

where the leave-one-out mean b_{-i} of the other samples in the group serves as a low-variance baseline. A token-level KL penalty to the SFT reference and an entropy bonus are added to the loss with small coefficients (both 10^{-3}).



(a) Cosine sampler (E2H-C).



(b) Gaussian sampler (E2H-G).

Figure 1: Per-level sampling probability across training for the two open-loop E2H schedules, reproduced from Parashar et al. (2026, Figs. 3–4). **(a)** The cosine ramp shifts mass smoothly and monotonically from the easiest to the hardest level. **(b)** The Gaussian sampler slides a localised band of difficulty rightward over training (shown for three (β, σ) settings; we use $\beta = \sigma = 0.5$).

Difficulty schedulers. The only quantity a scheduler controls is $S(t, k)$, the probability of sampling difficulty level k (with $k = 0$ the easiest) at training step t; each batch is then drawn from the per-level data pools according to $S(t, \cdot)$. We implement five schedulers:

- **`random` (balanced).** $S(t, k) = 1/K$, uniform over all levels at every step. This is our baseline.
- **`cosine` (E2H-C).** A parameter-free Easy-to-Hard ramp (Parashar et al., 2026): it maintains a smooth distribution over all levels whose mass shifts from easy to hard. The per-level weights interpolate between an easy-favouring and a hard-favouring linear profile under a cosine schedule,

  $$S_{\cos}(t, k) \;\propto\; \alpha_t\,(K - 1 - k) \;+\; (1 - \alpha_t)\,k, \qquad \alpha_t = \tfrac{1}{2}\big(1 + \cos(\pi t/T)\big),$$

  where $k = 0$ is the easiest level and the weights are renormalised to a distribution at each step. At $t = 0$, $\alpha_t = 1$, so the weight decreases linearly from the easiest level to the hardest (which receives zero mass); at $t = T$, $\alpha_t = 0$ reverses the profile; intermediate steps cosine-interpolate between the two. The result is a smooth, monotone transfer of probability mass from easy to hard with no tunable hyperparameters (Fig. 1a).
- **`gaussian` (E2H-G).** An Easy-to-Hard ramp that, instead of interpolating monotonically, samples a soft *band* of adjacent difficulties whose centre slides from easy to hard (Parashar et al., 2026). Assigning each level the unit-spaced mean $\mu_k = k$, the per-level weight is

  $$S_{\mathcal{N}}(t, k) \;\propto\; \exp\!\Big(-\tfrac{(x_t - \mu_k)^2}{2\sigma^2}\Big), \qquad x_t = (t/T)^{\beta}\,(K - 1),$$

  renormalised each step. The band centre x_t moves rightward across the levels over training: β controls how fast it moves ($\beta < 1$ spends fewer steps on the easy levels and dwells longer on the hard ones, limiting easy-task overfitting), and σ controls the width of the band (smaller σ concentrates sampling on a single level, approaching sequential CL). We use the paper's balanced defaults $\beta = \sigma = 0.5$ (Fig. 1b).
- **`lp` / `lp_plus` (learning-progress bandit).** A non-stationary multi-armed bandit over levels, detailed below.

Learning-progress bandit. Unlike the preset ramps above, our scheduler is knowledge-aware: it watches the policy's per-level success and steers training to where the model is actually learning. For each level k present in a batch we observe its current success rate $s_k(t)$ (fraction of samples with $r = 1.0$) and maintain a fast and a slow exponential moving average,

$$m_k^{\text{fast}}(t) = (1 - \alpha_{\text{f}})\, m_k^{\text{fast}}(t-1) + \alpha_{\text{f}}\, s_k(t), \qquad m_k^{\text{slow}}(t) = (1 - \alpha_{\text{s}})\, m_k^{\text{slow}}(t-1) + \alpha_{\text{s}}\, s_k(t), \quad (2)$$

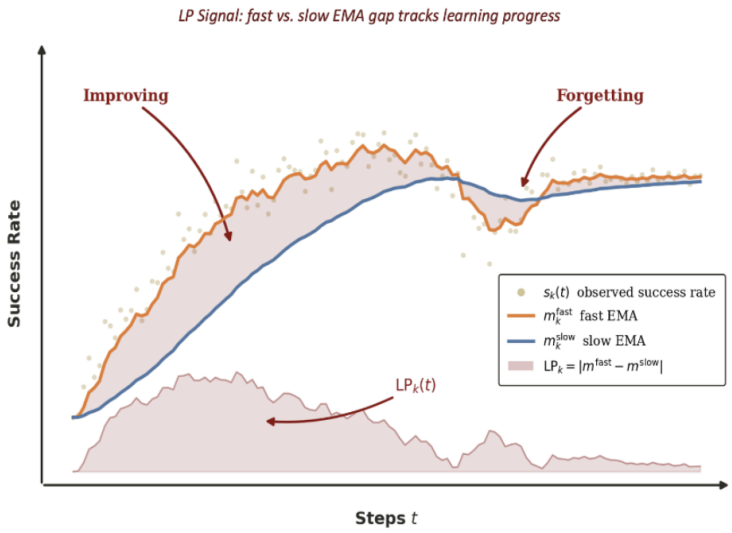


Figure 2: The learning-progress signal. For each level we track the observed success rate $s_k(t)$ with a fast and a slow EMA; their gap $\mathrm{LP}_k = |m_k^{\mathrm{fast}} - m_k^{\mathrm{slow}}|$ (shaded) is large both when the level is *improving* (fast above slow) and when it is being *forgotten* (fast dipping below slow), and collapses to zero once the level saturates. The absolute value is what gives the bandit its anti-forgetting behaviour: a mastered level whose success rate decays re-acquires a large LP and is pulled back into the sampling mix.

with $\alpha_{\mathrm{f}} = 0.3$ ($\sim$3-step horizon) and $\alpha_{\mathrm{s}} = 0.05$ ($\sim$20-step horizon). The bidirectional LP signal is the absolute difference $\mathrm{LP}_k(t) = \left|m_k^{\mathrm{fast}}(t) - m_k^{\mathrm{slow}}(t)\right|$; the absolute value provides anti-forgetting, because a mastered level whose success rate *decays* produces a large LP and is pulled back into the sampling mix (Figure 2). We convert LP into a tempered distribution $q_k(t) \propto \mathrm{LP}_k(t)^{\tau}$ (default $\tau = 1$) and mix it with an exploration floor:

$$S(t,k) \;=\; \mathrm{floor}_k \;+\; (1-\varepsilon)\, q_k(t), \qquad \varepsilon = 0.1. \tag{3}$$

The two bandit variants differ *only* in the floor. The standard **vanilla** `lp` uses a *uniform* floor $\mathrm{floor}_k = \varepsilon/K$. Our `lp_plus` (contribution) uses a *difficulty-weighted* floor

$$\mathrm{floor}_k \;=\; \varepsilon\, \frac{k+1}{\sum_j (j+1)}, \tag{4}$$

which guarantees the hardest level a minimum share growing linearly with its index.

Why a difficulty-weighted floor. Figure 3 shows the failure mode `lp_plus` is designed to fix. Under sharpening, the easy levels saturate quickly: their LP spikes early and then collapses to near zero by $\sim$step 100. The hardest in-distribution level (4-op), however, *never produces a significant LP signal*—both its fast and slow EMAs sit near zero because the model rarely solves it—so a pure LP-proportional policy allocates it almost nothing, and the uniform floor gives it only ε/K. This is precisely backwards: the one level most in need of exposure is starved. The difficulty-weighted floor forces a protected, non-trivial budget onto the hard level regardless of its (vanishing) LP.

4 Experimental Setup

Data and difficulty axis. Constructing the difficulty axis is itself part of our contribution. The standard Countdown data spans only 3- and 4-operand problems—too narrow to define an Easy-to-Hard curriculum, with no genuinely easy entry point and no harder level to generalize to. We synthesize the two missing endpoints ourselves and assemble a balanced four-level set (`countdown_curriculum_2to5`): 2-op (trivial, 6,773), 3-op (easy, 240,558), 4-op (medium, 249,656), and 5-op (hard, 20,000), all in the source data's exact prompt format and scored by the

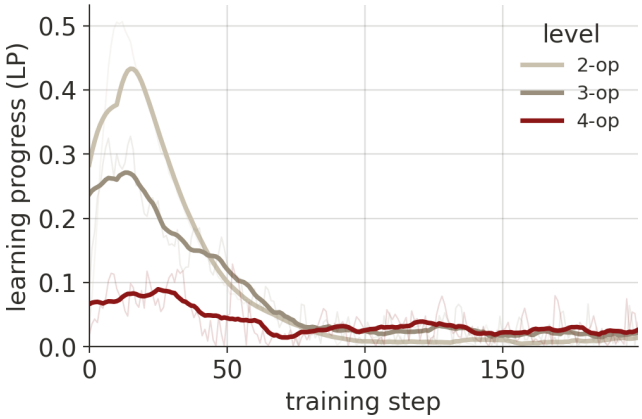


Figure 3: Per-level learning-progress signal $\mathrm{LP}_k = |m^{\mathrm{fast}} - m^{\mathrm{slow}}|$ during the `lp_plus` run (T=200). Easy levels (2-op, 3-op) spike early and collapse by ∼step 100; the hardest in-distribution level (4-op) never produces an LP signal, so a pure LP bandit would starve it. This motivates the difficulty-weighted floor.

same verifier. We generate the **2-op** level by *exhaustive enumeration*—every operand pair and operator with an exact, in-range result, deduplicated—which covers the entire valid two-number problem space. We generate the **5-op** level by *rejection sampling*: an expression is built left-to-right, and whenever the operator is division we force the next operand to be an exact divisor of the running result, so no step can break exactness and every generated instance is 100% verifier-conformant (without this trick, non-exact divisions reject nearly all candidates and division all but vanishes from the data). The 2-op endpoint is what makes an easy-to-hard schedule possible at all, while the held-out 5-op endpoint provides a clean out-of-distribution probe. Training uses only the three easiest levels (2/3/4-op, K=3), with 5-op reserved entirely for out-of-distribution evaluation; we evaluate on a balanced held-out test set of 50 instances per level.

Model, RL, and decoding. All runs initialize from the same supervised-fine-tuned (SFT) checkpoint of the Qwen2.5-0.5B base. The SFT stage trains on the cognitive-behaviors Countdown corpus (`Asap7772/cog_behav_all_strategies`), with the loss applied only to completion tokens, for 3 epochs at learning rate 5×10^{-5} (batch size 32 with 4 gradient-accumulation steps, weight decay 0.01, warmup ratio 0.05); we select this checkpoint as the best by pass@1 from a learning-rate/epoch sweep. From this warm start, every RLOO run is trained on Modal (H100 GPUs), with rollouts sampled by vLLM. Shared RLOO hyperparameters are learning rate 10^{-5}, group size $G = 8$, KL coefficient 10^{-3}, and entropy coefficient 10^{-3}, for T=200 steps at batch size 64; across runs only `--schedule` varies. Training decodes at temperature 1.0 (top-p 1.0, top-k -1); evaluation decodes at temperature 0.6, top-p 0.95, top-k 20, drawing 16 samples per prompt. We report **pass@1** (single-sample correctness, our primary metric) and **pass@16** (coverage, the fraction of prompts solved by at least one of the 16 samples).

Evaluation artifact and fix. We diagnosed and corrected a subtle evaluation artifact. The SFT policy learns to end its turn on the Qwen chat token `<|im_end|>`, whereas the base model's EOS is `<|endoftext|>`, which is what vLLM watches; as a result no generation terminates, decoding runs to the token budget, and the verifier scores the *last* `<answer>` of a rambling output—inflating measured capability. All evaluations in this paper use `stop=["</answer>"]`, which aligns standalone evaluation with the in-training metric and correctly measures true first-answer capability.

Training and evaluation protocol. Every scheduler is trained under one common regime: K=3 over 2/3/4-op, T=200 steps, batch 64, all initialized from the same SFT checkpoint and run to completion (200/200). Out-of-distribution generalization is measured on a separate set of 500 held-out 5-op prompts.

Table 1: In-distribution per-level pass@1. `lp_plus` is best overall and on 4-op; the two adaptive LP schedulers are the only curricula that beat balanced on the aggregate.

Run	2-op	3-op	4-op	≤4 agg.
SFT	0.159	0.383	0.139	0.227
`random` (balanced)	0.996	0.714	0.319	0.676
`cosine` (E2H-C)	0.965	0.751	0.307	0.675
`gaussian` (E2H-G)	0.943	0.760	0.294	0.665
`lp` (LP-bandit)	**1.000**	**0.764**	0.290	0.685
`lp_plus` (ours)	0.993	0.751	**0.346**	**0.697**

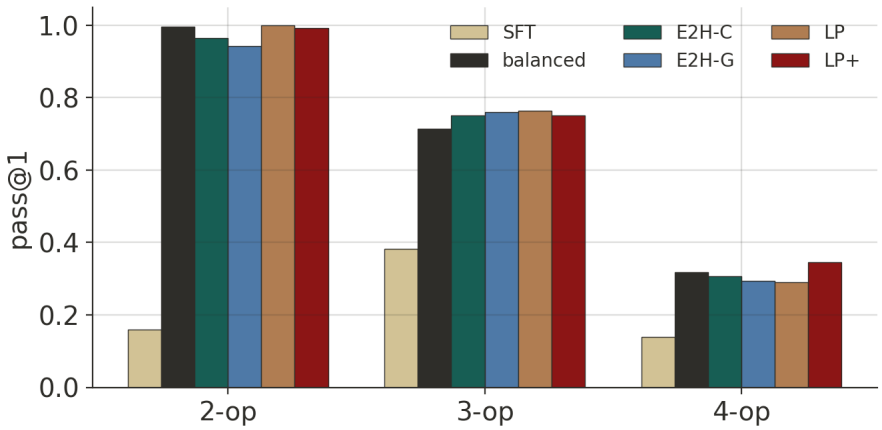


Figure 4: Final pass@1 per level (T=200). A large SFT→RL jump, then a close race across schedulers on 2/3-op; on the hardest in-distribution level (4-op) `lp_plus` leads the field, the payoff of its difficulty-weighted floor.

5 Results

5.1 Finding 1: LP+ is the strongest scheduler in-distribution

Table 1 and Figure 4 report final in-distribution pass@1. `lp_plus` is the best scheduler overall (0.697 on the ≤4-op aggregate) and, decisively, the best on the hardest in-distribution level—0.346 pass@1 on 4-op, against ≤ 0.32 for every other schedule, with the tied-best 0.520 pass@16 coverage there. The gain is exactly where the method is designed to act: the difficulty-weighted floor protects the hardest learnable level and converts that protection into the best hard-level accuracy. More broadly, the *adaptive* learning-progress schedulers are the only curricula to improve on balanced sampling at all (`lp` 0.685 and `lp_plus` 0.697 both exceed balanced 0.676), whereas the fixed Easy-to-Hard ramps do not (`cosine` 0.675, `gaussian` 0.665); and across the LP family an LP variant is best at *every* level (`lp` on 2/3-op, `lp_plus` on 4-op and overall). The improvements over balanced are consistent but modest in magnitude, with all RLOO runs falling in a narrow 0.665–0.697 band—because, as we show next, the schedulers share a common ceiling set by the base model rather than by the schedule.

5.2 Finding 2: the ceiling is the base model's, and RLOO sharpens up to it

Why do all schedulers land in the same narrow band, and why is LP+'s lead on 4-op real but bounded? Figure 5 resolves the in-distribution gains with the pass@k lens. On every level the RLOO policies start far above the SFT model at k=1, but the gap closes as k grows and the SFT model *catches up and overtakes* by k=16 on 3-op and 4-op. RLOO therefore raises single-sample accuracy by re-ranking solutions the base model can already produce, not by adding new ones: at the prompt level the SFT model already covers $24/24$ of the 3-op test prompts in pass@16, so net emergence there is zero. The 4-op plateau is thus a property of the *base model*—its pass@16 coverage ($\approx$ 0.50) is the ceiling—and giving 4-op more budget, as LP+'s floor does, pushes pass@1

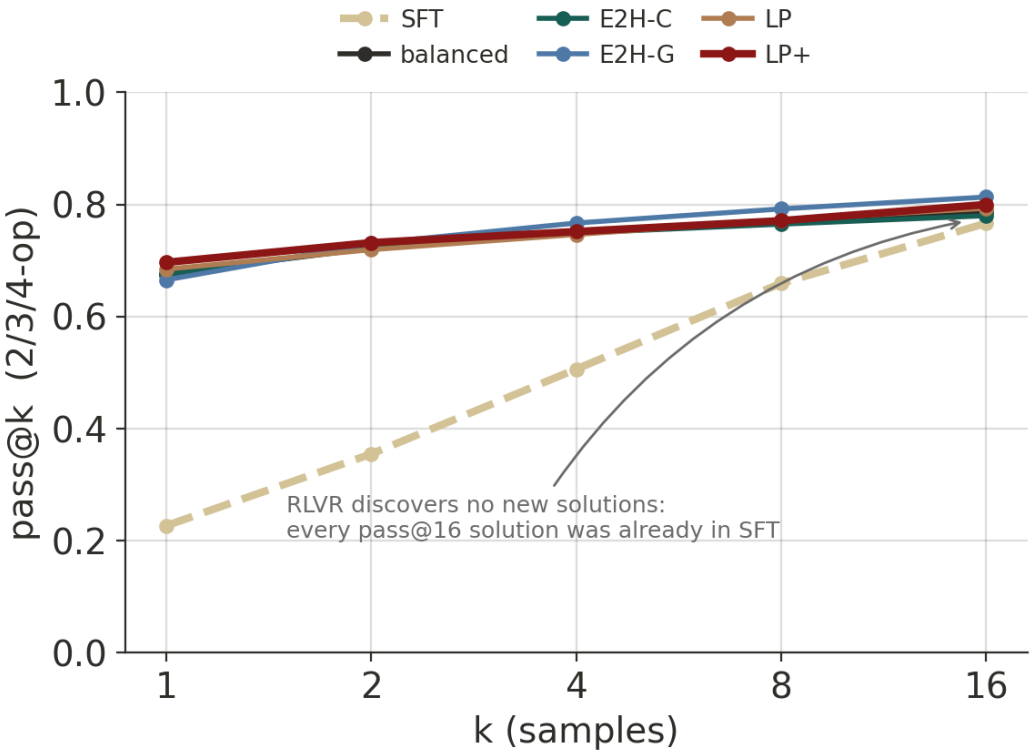


Figure 5: pass@k, SFT vs. RLOO. In-distribution (2/3/4-op), RLOO wins at k=1 but the SFT model catches up and overtakes by k=16: the gains are sharpening, not new coverage. Only on 5-op (OOD) do the RLOO policies remain above the SFT model at all k.

toward that ceiling more effectively than any other schedule, which is precisely why LP+ wins 4-op yet no schedule clears the band. This is the sharpening signature of Yue et al. Yue et al. (2025), and it reframes the modest spread as a feature of the regime, not of our method: a curriculum redistributes where sharpening lands, and LP+ redistributes it best, but no schedule can lift a coverage ceiling the base model sets.

5.3 Entropy and output-diversity collapse

The sharpening is visible directly in the policy's distributions (Figure 6). Token entropy falls monotonically from $\sim$0.55 to 0.20–0.28 for all schedulers, with the bandits sharpest; meanwhile the KL penalty is effectively inactive (the logged $\beta \cdot \mathrm{KL}$ is $\sim 10^{-4}$, roughly $150\times$ smaller than the loss magnitude), so nothing restrains the collapse. Output diversity tells a similar story: the SFT model produces $\sim$10–12 distinct `<answer>` equations per 16 samples uniformly across levels, whereas `lp_plus` collapses to $\sim$1–2 distinct solutions on the mastered 2-op level but stays diffuse ($\sim$6–7) on the unmastered 4-op level. Sharpening is thus *local to learnability*: RL amplifies a single canonical derivation only where it has actually learned the level—consistent with LP+'s gains concentrating where the model is genuinely competent.

5.4 Finding 3: out-of-distribution generalization over-sharpens away

On the held-out 5-op level a correct answer must be a five-number equation, so a policy that stops emitting this form cannot solve 5-op at all (Table 2). Tracking `lp_plus` over training (Figure 7) shows the effect is non-monotonic: generalization is strongest mid-training (pass@16 0.258 at step 160, well above the SFT model's 0.092), then OOD pass@1, pass@16, and the five-number-equation rate all decline monotonically to below the SFT level by step 200. Continued RL sharpens output structure onto the in-distribution maximum of ≤ 4 numbers, progressively unlearning the five-number form a 5-op solution requires. The low convergence value is therefore an artifact of

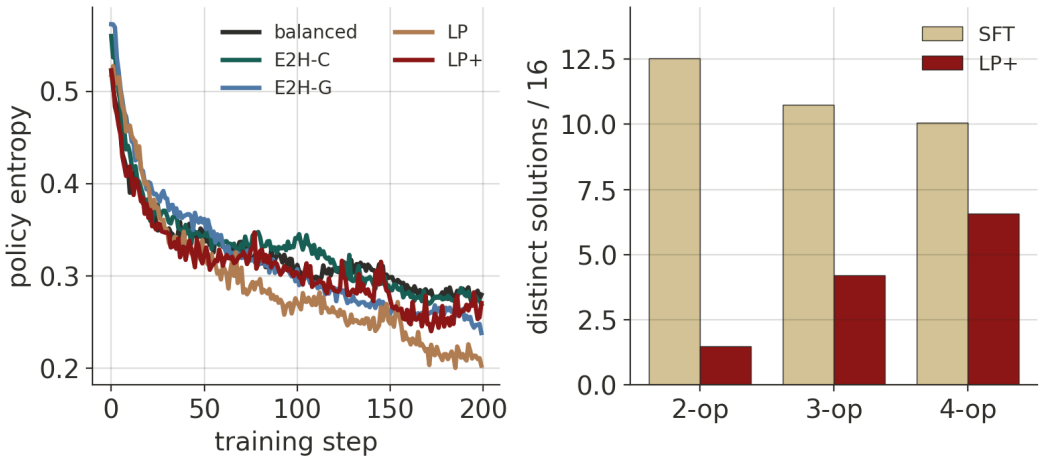


Figure 6: Left: policy (token) entropy collapses for all schedulers from ∼0.55 to 0.20–0.28. Right: distinct `<answer>` equations per 16 samples; `lp_plus` collapses to ∼1 on mastered 2-op but remains diverse on unmastered 4-op—sharpening is local to where the model has learned.

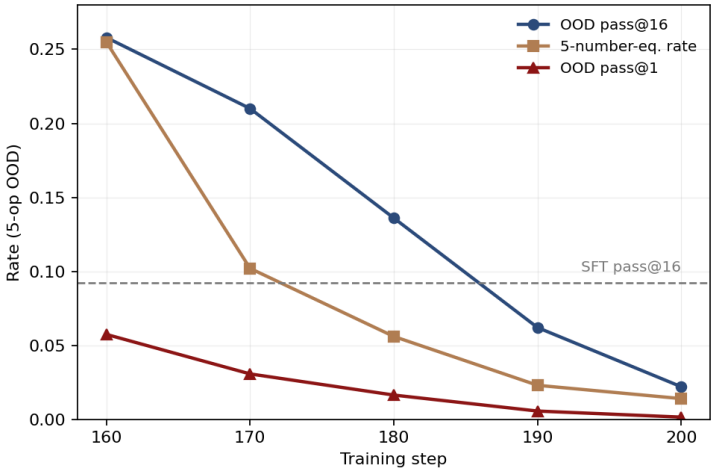


Figure 7: Out-of-distribution (5-op) performance of `lp_plus` over training. OOD pass@1, pass@16, and the five-number-equation emission rate all decline monotonically from step 160 to 200 and in lockstep; pass@16 starts well above the SFT level and crosses below it by ∼step 190. The convergence (step-200) OOD number is the endpoint of this over-sharpening collapse, not an intrinsic property of the schedule.

over-training rather than the schedule, and OOD here is governed by checkpoint selection (we characterize this for `lp_plus`; the other runs are measured at convergence only).

6 Discussion

Within the regime, the learning-progress bandits win. Even where the schedule cannot move the ceiling, it can still order the field—and within our controlled comparison the adaptive learning-progress bandits come out on top. The two LP schedulers are the only curricula to improve on balanced sampling overall (0.685 and 0.697 vs. 0.676), where the fixed Easy-to-Hard ramps do not, and across the LP family an LP variant is best at *every* difficulty level. Our `lp_plus` is the single strongest scheduler: best overall and best on the hardest learnable level. The margins are modest and single-seed, as the capacity ceiling dictates, but the direction is consistent—when a difficulty curriculum helps at this scale, an adaptive, learning-progress-driven one helps most.

Table 2: Out-of-distribution 5-op generalization at convergence (500 prompts, step $\sim$200). Emitting five-number equations is necessary to solve 5-op; `lp_plus` collapses to a 1.4% emission rate. The trajectory of `lp_plus` toward these convergence values is shown in Figure 7.

Run	pass@1	pass@16	5-number-eq. rate
SFT	0.0088	0.092	10.5%
`random` (balanced)	0.0204	0.150	14.2%
`cosine` (E2H-C)	0.0445	0.224	14.2%
`gaussian` (E2H-G)	0.0196	0.156	14.0%
`lp` (LP-bandit)	0.0140	0.094	7.8%
`lp_plus` (ours)	0.0015	0.022	1.4%

When does a curriculum help? A capacity account. Our central in-distribution result—no scheduler beats balanced by much—appears to contradict the large gains of E2H Parashar et al. (2026) and SEC Chen et al. (2025), yet matches the controlled negative result of Mordig et al. Mordig et al. (2026). We reconcile them through model capacity. Under the pass@k lens of Yue et al. Yue et al. (2025), RLVR re-ranks solutions the base model can already produce rather than creating new ones, so a curriculum can only *redistribute* where that sharpening is spent. Redistribution pays off only when the base retains latent coverage of the hard levels that balanced sampling under-elicits—present on the 1.5–3B models E2H and SEC use, but absent on the mandated 0.5B base, whose 4-op pass@16 ceiling (≈ 0.50) every scheduler shares. Tellingly, SEC reports that its own advantage shrinks toward balanced on stronger bases that already cover the hard problems: curriculum gains seem to require capacity that is latent but not yet realized, a band our 0.5B base sits below. Our study is thus best read as the small-model boundary case that delimits when E2H/SEC-style gains are even possible.

What `lp_plus` buys, and what it costs. The difficulty-weighted floor does what it was designed to do: it rescues the hardest in-distribution level from the learning-progress starvation a uniform floor suffers (Figure 3) and yields the best 4-op accuracy of any scheduler (Table 1). Its cost appears only out-of-distribution, and only under over-training. Because the floor sharpens the 4-op level hardest, it drives the policy most firmly onto $\leq$ 4-number equations and, carried to convergence, progressively unlearns the five-number form a 5-op solution requires (Figure 7). Crucially, the erosion is a *training-time* effect—`lp_plus` generalizes strongly mid-training—so out-of-distribution performance is governed by checkpoint selection, not by an intrinsic flaw in the floor. The broader lesson is that in-distribution reward is the wrong objective for preserving out-of-distribution structure: separating "train the hard level" from "preserve generalizing structure" would require a signal defined on output structure (e.g. the five-number-equation rate) rather than on reward.

Limitations. Our claims rest on single-seed runs at a single, mandated 0.5B model size with $k \leq 16$, so we foreground qualitative patterns over third-decimal orderings. The 5-op OOD numbers are small in absolute terms—the level may lie largely outside the base model's support—and we characterize the OOD trajectory for `lp_plus` only. Most directly, we did not run the larger-model control that our capacity account most sharply predicts.

Future work. Two directions follow. First, repeat the identical schedulers on a 1.5–3B base: our account predicts that E2H/SEC-style gains reappear exactly where latent hard-level capacity exists, and that `lp_plus`'s out-of-distribution erosion weakens. Second, replace the floor's in-distribution objective with a structure-aware signal that rewards the output structure generalization requires, decoupling "train the hard level" from "generalize beyond it."

7 Conclusion

We compared five difficulty schedulers for RLVR on Countdown under one controlled regime on a fixed 0.5B base, and introduced `lp_plus`, a learning-progress bandit whose difficulty-weighted floor protects the hardest learnable level. The floor works—`lp_plus` is the best scheduler in-distribution and on the hardest level—but no scheduler beats balanced by much, because RL here sharpens the base model's existing distribution rather than extending it. On a capacity-limited base, curriculum

RL is therefore an allocation policy over a fixed budget of sharpening: it decides *where* that sharpening is spent and helps most on the hardest learnable level, but cannot exceed the base model's coverage—and, pushed to convergence, can erode the output structure that generalization requires.

8 Team Contributions

- **Arthur Gontier:** generated the synthetic 2-op and 5-op Countdown data, co-designed the LP/LP+ scheduler framework, and led the entropy and output-diversity analysis.
- **Louis de Germay:** designed and implemented the difficulty schedulers, and led the out-of-distribution (5-op) and pass@k analysis.

Changes from Proposal Relative to our proposal, we (i) augmented the standard 3–4-operand dataset with synthetic 2-op and 5-op levels, creating a balanced difficulty axis with a held-out out-of-distribution level; (ii) reframed our adaptive scheduler from its initial self-evolving design into the learning-progress bandit (`lp`) and our difficulty-weighted variant (`lp_plus`); and (iii) devoted much of our effort to explaining the capacity ceiling and the sharpening-versus-emergence picture—an unanticipated but central finding.

References

Adrien Baranes and Pierre-Yves Oudeyer. 2013. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems* 61, 1 (2013), 49–73.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*. 41–48.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. 2025. Self-Evolving Curriculum for LLM Reasoning. *arXiv preprint arXiv:2505.14970* (2025).

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. 2025. The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models. *arXiv preprint arXiv:2505.22617* (2025). arXiv:2505.22617 [cs.LG]

Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*. 1311–1320.

Ingmar Kanitscheider, Joost Huizinga, David Farhi, William Hebgen Guss, Brandon Houghton, Raul Sampedro, Peter Zhokhov, Bowen Baker, Adrien Ecoffet, Jie Tang, Oleg Klimov, and Jeff Clune. 2021. Multi-task curriculum learning in a complex, visual, hard-exploration domain: Minecraft. *arXiv preprint arXiv:2106.14876* (2021).

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. 2019. Teacher–student curriculum learning. *IEEE Transactions on Neural Networks and Learning Systems* 31, 9 (2019), 3732–3740.

Maximilian Mordig, Andreas Opedal, Weiyang Liu, and Bernhard Schölkopf. 2026. Rethinking Easy-to-Hard: Limits of Curriculum Learning in Post-Training for Deductive Reasoning. *arXiv preprint arXiv:2603.27226* (2026).

Pierre-Yves Oudeyer, Frederic Kaplan, and Verena V. Hafner. 2007. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation* 11, 2 (2007), 265–286.

Shubham Parashar et al. 2026. Curriculum Reinforcement Learning from Easy to Hard Tasks Improves LLM Reasoning. In *International Conference on Learning Representations (ICLR)*. arXiv:2506.06632.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. 2020. Teacher algorithms for curriculum learning of deep reinforcement learning in continuously parameterized environments. In *Proceedings of the Conference on Robot Learning (CoRL)*. 835–853.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. 2025. Does Reinforcement Learning Really Incentivize Reasoning Capacity in LLMs Beyond the Base Model? *arXiv preprint arXiv:2504.13837* (2025).