

Population-Based RLVR: Breaking the Reweighting Ceiling via Independent Mode Discovery

Idea Sketch — Backburner

Rohan Sikand · Stanford University

March 2026

The Problem: Single-Seed Mode Collapse

RLVR training is stochastic. Different random seeds find different solution modes.

But we only ever train **one policy**. That policy converges to **one mode** — whichever the random seed happens to find first. All diversity methods (DAPO, STEER, AEPO, DIVER, DSDR) try to preserve diversity within a single run. They slow mode collapse but cannot prevent it.

Single-seed RLVR:

- Seed 1 finds quicksort
- Converges hard to quicksort
- Mergesort pruned by step 100
- Pass@1 up, Pass@k down

What if we ran K seeds?

- Seed 1 finds quicksort
- Seed 2 finds mergesort
- Seed 3 finds radix sort
- Union coverage $>$ any individual

The question: can we **combine** the modes found by independent runs into a single, broader policy?

The Reweighting Ceiling (Recap)

From Yue et al. (2025): RLVR does not discover new reasoning patterns. It reweights within the base model's existing distribution.

$$\text{Ceiling}(\pi_0) = \sup_{\pi \in \Delta(\pi_0)} \mathbb{E}_s[\text{Pass@1}_\pi(s)]$$

where $\Delta(\pi_0) = \{\pi : \text{supp}(\pi) \subseteq \text{supp}(\pi_0)\}$.

All single-seed methods hit this ceiling. Diversity methods slow the approach but don't raise it.

Key insight from Yue et al.: distillation *can* introduce new knowledge and expand the reasoning boundary. A policy trained on another policy's solutions gains capabilities the base didn't have.

The Proposal: Population-Based RLVR

Phase 1: Divergent Training

$$\pi_{\theta_0} \xrightarrow{\text{seed 1}} \pi_{\theta_1}^* \xrightarrow{\text{seed 2}} \pi_{\theta_2}^* \xrightarrow{\text{seed 3}} \pi_{\theta_3}^* \dots \xrightarrow{\text{seed K}} \pi_{\theta_K}^*$$

Phase 2: Solution Harvesting

$$\mathcal{D}_{\text{union}} = \bigcup_{k=1}^K \{(s, a) : a \sim \pi_{\theta_k}^*, R_{\text{env}}(s, a) = 1\}$$

Phase 3: Cross-Pollination

$$\pi_{\theta_0} \xrightarrow{\text{SFT on } \mathcal{D}_{\text{union}}} \pi_{\theta_{\text{merged}}} \xrightarrow{\text{continue RLVR}} \pi_{\theta_{\text{final}}}$$

Each phase uses known techniques. The novelty is the **pipeline** and the claim that the union of independently discovered modes yields a strictly broader policy than any single run.

Why Independent Seeds Find Different Modes

GRPO's advantage estimation is stochastic: which rollouts happen to be correct in the first few batches determines the initial gradient direction.

Empirical evidence (anecdotal but consistent):

- Different seeds on the same problem set converge to different Pass@1 scores
- The *set* of problems solved differs across seeds — not just accuracy
- Seed A solves problems 1, 3, 5 that seed B cannot, and vice versa

Theoretical intuition: RLVR is a non-convex optimization with many local optima in policy space. Different initializations of the stochastic process (seed, batch order, sampling noise) converge to different optima. Each optimum corresponds to a different “solution mode” — a cluster of related strategies the policy has sharpened on.

Phase 1: Divergent Training

Train K copies of the base model independently via RLVR:

Design choice	Options	Default
Number of policies K	4, 8, 16	4 (budget-limited)
Differentiation source	Random seed only + different problem subsets + different hyperparameters	Seed only
Training budget per policy	Full budget / K Full budget each (expensive)	Full budget / K
Base model	Same for all	Qwen3-4B

Key tradeoff: Iso-compute (same total budget, split K ways) vs. $K \times$ compute. Iso-compute is the fair comparison — each policy trains for fewer steps but the total GPU-hours match a single long run.

Phase 2: Solution Harvesting

After divergent training, collect each policy's correct solutions:

For each problem s_i in the training set:

1. Sample N rollouts from each policy $\pi_{\theta_k}^*$
2. Keep all correct rollouts: $\mathcal{D}_k = \{(s_i, a) : R_{\text{env}}(s_i, a) = 1\}$
3. Union: $\mathcal{D}_{\text{union}} = \bigcup_k \mathcal{D}_k$
4. Deduplicate by solution strategy (embedding clustering)

Key measurement at this stage:

- Coverage: how many problems are solved by the union that no single policy solves?
- Diversity: how many distinct solution clusters exist in $\mathcal{D}_{\text{union}}$ vs. any $\mathcal{D}_k$?

Phase 3: Cross-Pollination

The critical step. How to merge K policies' knowledge into one:

Option A: SFT on $\mathcal{D}_{\text{union}}$, then continue RLVR

- SFT the *base model* (not any trained policy) on the diverse solution set
- This expands the base model's support to include all discovered modes
- Continue RLVR from this SFT checkpoint to sharpen
- Cleanest separation: SFT for discovery, RL for exploitation

Option B: Forward-KL distillation from mixture

- Define mixture policy: $\pi_{\text{mix}}(\cdot|s) = \frac{1}{K} \sum_k \pi_{\theta_k}^*(\cdot|s)$
- Distill student via forward-KL: $\min_{\theta} D_{\text{KL}}(\pi_{\text{mix}} \parallel \pi_{\theta})$
- Forward-KL is mass-covering — student must cover all modes (cf. DPH-RL)

Why Forward-KL Matters for Merging

The choice of distillation divergence determines whether modes survive:

Reverse KL (standard RLVR):

- Mode-seeking
- Student covers one peak well
- Other modes dropped
- → Merging collapses back to one mode

Forward KL (DPH-RL style):

- Mass-covering
- Student must cover all teacher mass
- All modes preserved
- → Merging retains diversity

DPH-RL (2025) showed that forward-KL divergence preserves diversity in single-policy RLVR. We apply the same principle to multi-policy merging: distill from the mixture π_{mix} using forward-KL to ensure the student inherits all K modes, not just the dominant one.

Connection to EGSR

EGSR provides natural synergies with population-based training:

1. Dense evaluation of cross-pollinated solutions

When policy A 's solutions are SFT'd into policy B , EGSR's teacher can evaluate them with failure-mode granularity — not just did it pass ' ' but how does this new strategy compare to the existing ones."

2. Efficient per-policy training

Each of the K policies trains with EGSR's dense reward, reaching its local optimum faster. If EGSR reduces the group size needed (EGSR $G=4 \approx$ GRPO $G=16$), the per-policy budget goes further.

3. Judgment diversity across policies

Prior Work Landscape

Paper	Approach	Gap from ours
PBT (Jaderberg 2017)	Population-based training	Classical RL, not LLMs
Yue et al. (2025)	RLVR = reweighting	Diagnosis only, no population
DPH-RL (2025)	Forward-KL for diversity	Single policy, no population
DIVER / DSDR / AEPO	Diversity rewards	Single policy, within-run
POPE (CMU 2026)	Oracle-guided exploration	External oracle, single policy
Ensembles of distilled	Ensemble for generalization	Classical RL, not LLM RLVR
BroRL (NVIDIA)	Scale rollout breadth	Single policy, no merging

The gap: No one has done independent multi-seed RLVR training → solution harvesting → diversity-preserving cross-pollination for LLM reasoning. The individual components exist; the pipeline doesn't.

Experimental Plan

Phase 0: Mode divergence validation (1 week)

- Train $K=4$ RLVR runs (Qwen3-4B, different seeds) for 200 steps each
- Measure per-run coverage, pairwise overlap, union coverage
- **Go/no-go:** if union coverage meaningfully exceeds any individual ($>5\%$ more problems solved), proceed

Phase 1: Cross-pollination (3 weeks)

- Option A: SFT base on $\mathcal{D}_{\text{union}}$, continue RLVR
- Option B: Forward-KL distillation from π_{mix}
- Compare both against: single run (same total compute), single run ($K \times$ compute), single run with diversity bonus

Phase 2: Analysis (2 weeks)

The Iso-Compute Question

The fairest comparison is **iso-compute**: same total GPU-hours.

Method	Total compute	Prediction
Single RLVR, 800 steps	C	High Pass@1, low diversity
$K=4$ RLVR, 200 steps each	C	Lower per-policy Pass@1
+ cross-pollination	$C + \epsilon$	Higher coverage?
Single RLVR + diversity bonus	C	Moderate both

The key hypothesis: K short runs + merge $>$ 1 long run, even at the same compute budget.

Why this might hold: 200 steps is enough to identify a mode (early training). The last 600 steps of a long run are pure sharpening — diminishing returns. Spending those 600 steps finding 3 other modes, then merging, could yield more total coverage.

What Could Go Wrong

Risk	Severity	Mitigation
Seeds find the same mode	High	Measure in Phase 0; diversify problem subsets per seed
SFT merging forgets modes	High	Forward-KL distillation; replay buffer from all seeds
Per-policy undertrained (iso)	Medium	Find minimum viable steps per policy (ablate)
$K \times$ compute not justified	Medium	Iso-compute comparison; show coverage $>$ long run
Merged policy can't execute all modes reliably	Medium	Continue RLVR post-merge to sharpen

Kill criterion: Phase 0 shows $\$ < \$3\%$ union coverage gain over best individual. Seeds aren't diverging enough to justify the pipeline.

Theoretical Angle

Coverage amplification: Let C_k be the set of problems solved by policy k . If policies are trained independently:

$$\mathbb{E}\left[\left|\bigcup_k C_k\right|\right] \geq |\mathbb{E}[C_1]|$$

with equality only if all policies find identical solutions. Under diversity (different seeds), the union grows sublinearly but strictly faster than a single run.

Connection to the reweighting ceiling: Cross-pollination via SFT expands $\text{supp}(\pi_\theta)$. If policy 2's solution was not in policy 1's support, SFT on it literally raises policy 1's reweighting ceiling. Subsequent RLVR then optimizes within this expanded support.

Formal claim:

Honest Assessment

Pros:

- Conceptually clean: independent exploration + unified exploitation
- No architectural changes, no external oracle
- Directly addresses the reweighting ceiling with a clear mechanism
- Individual components (RLVR, SFT, distillation) are all proven
- **Least crowded** of the five exploration ideas

Cons:

- $K \times$ compute (even iso-compute means shorter per-run training)
- Merging without mode collapse is an unsolved problem in general
- “Population-based training + distillation” is conceptually simple — may feel