DEI: Diversity in Evolutionary Inference for Quality-Diversity Search

John Donaghy [1] **Shikhar Rastogi** [1]

Abstract

We present DEI: Diversity in Evolutionary Inference, a distributed Quality-Diversity (QD) search framework that assigns heterogeneous large language models (LLMs) as mutation operators across peer nodes communicating with non-blocking collective operations. Unlike homogeneous parallel search, which replicates a single model's inductive biases across all workers, DEI treats each LLM's distinct creative prior as a complementary source of behavioral novelty. Extending the Digital Red Queen framework with DEI, nodes share local optimal solutions at the end of each round to seed the next round's population. This creates cross-model adversarial pressure that drives robustness beyond intra-model self-play. Evaluated on the Core War domain, a competitive programming benchmark in which Redcode warrior programs battle inside a simulated machine, a four-node heterogeneous ensemble (GPT-5.4-mini, Claude Sonnet 4.6, GPT-5.2, and Claude Haiku 4.5) achieves $+124\%$ higher merged-archive QD-Score (45.90 vs. 20.46) and $+28\%$ higher coverage (80.6% vs. 63.0% of cells) than a single-node baseline at equal total LLM-call budget. The heterogeneous ensemble also outperforms an equally-budgeted homogeneous ensemble on QD-Score, coverage, and held-out solution generality across all four model families. These results provide the first empirical evidence that *model diversity*, not merely parallelism, is the key driver of gain in distributed LLM-based QD search.

1. Introduction

Evolutionary algorithms search for high-performing solutions by maintaining a population of candidates and selecting the best from the population. The population is iteratively updated by generating new candidates either from scratch or by mutating existing solutions. Mutation is performed by the *mutation operator* which is a general procedure for taking an existing solution and perturbing it to yield a variant, whose fitness is then evaluated for inclusion in the population.

Large language models have emerged as surprisingly effective creative operators in evolutionary search. They generate novel program variants, mutate existing solutions with semantic awareness, and reason about fitness landscapes in ways that classic genetic operators cannot. Yet every LLM encodes a particular distribution over programs shaped by its training data, architecture, and alignment procedure. A model trained predominantly on Python tutorials will favor different control-flow patterns than one trained on competitive-programming corpora. A model fine-tuned for instruction following will hedge in ways that a code-completion model will not. These inductive biases should be harnessed deliberately to cover more of a behavioral space than any single model can reach alone.

Existing approaches to parallel LLM search treat scaling as a matter of *computation*, not *cognition*. FunSearch (Romera-Paredes et al., 2023) runs many independent LLM calls in parallel, but all calls go to the same model. Diversity emerges only from stochastic sampling, not from fundamentally different generative priors. The Digital Red Queen (DRQ) framework (Kumar et al., 2026) introduced adversarial evolutionary pressure into MAP-Elites (Mouret & Clune, 2015) Quality Diversity (QD) search by using round champions as opponents, but in its original formulation a single node runs a single model. Blind spots in that model's generative distribution become permanent gaps in the archive.

We argue for a qualitatively different regime: *parallel cognition*. When diverse LLMs simultaneously explore a shared behavioral space, each model's inductive bias causes it to preferentially populate distinct regions of the Quality-Diversity archive (Chatzilygeroudis et al., 2020). A growing body of work in reinforcement learning and multi-agent reasoning shows that explicitly promoting generative diversity, whether through diversity-aware RL objectives (Li et al., 2025; Hu et al., 2026) or multi-agent de-

[1]Gensyn. Correspondence to: John Donaghy <johnny@gensyn.ai>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

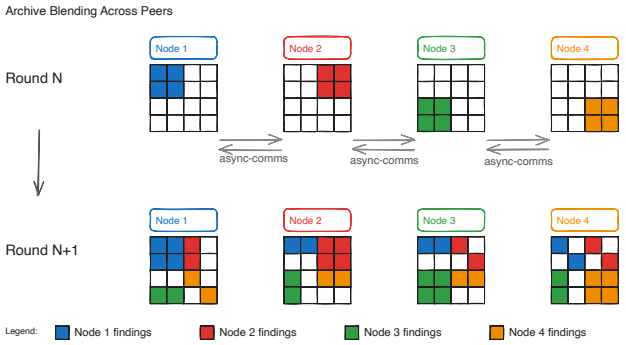

Figure 1. Heterogeneous ensembles as parallel cognition. In an early round, each node's inductive bias preferentially populates a distinct region of the shared Quality-Diversity archive (top). Asynchronous champion sharing then seeds peers with elites from niches they would rarely generate internally, so archives begin to blend across nodes in subsequent rounds (bottom).

bate across several LLMs (Liang et al., 2024), improves exploration and downstream task performance. We apply the same intuition to distributed Quality-Diversity search. Champions that one model considers elite may inhabit niches that another model never generates, providing ready-made seeds for cross-pollination. When these champions are shared as opponents via non-blocking communications, every node may progress at its own pace while competing against strategies it would never have generated internally, driving robustness that pure intra-model self-play cannot provide.

Realizing this vision requires solving a practical systems problem: heterogeneous LLM nodes operate at wildly different latencies. A local Qwen-35B instance running on Apple Silicon with MLX may take ten seconds per call; a frontier model may respond in under two. A synchronization barrier would throttle the entire ensemble to the slowest participant. We therefore develop fully asynchronous collective communications, allowing each node advance on independently, regardless of latencies.

The main contributions of the paper can be listed as:

- **DEI:** A distributed QD search framework that explicitly exploits *heterogeneous* LLM ensembles as mutation operators, assigning each model a distinct node in the search graph.
- **Async gossip integration:** A fully asynchronous champion-sharing protocol that enables heterogeneous hardware participation without synchronization barriers achieving higher throughput than a synchronous approach.
- **Diversity aware improvements on generality, archive coverage and QD-Score:** By holding total compute budget fixed across conditions, we show the gain from heterogeneous nodes over homogeneous nodes cannot be attributed to extra computation, isolating diversity as the causal factor.

2. Background

We use the procedure developed by Kumar et al. (2026) as a single node baseline for extension to evolution with a diverse ensemble of mutation operators.

2.1. Core War and the MARS Simulator

Core War is a competitive programming game in which two or more *warrior* programs, written in the Redcode assembly language, battle inside a virtual circular memory called the Memory Array Redcode Simulator (MARS). Each warrior attempts to crash all opponents by overwriting their instruction pointers or halting their processes, while surviving as long as possible itself. The memory array wraps around, all addresses are relative and modular, and multiple processes per warrior are permitted, making the space of viable strategies remarkably rich: imps that replicate forward, dwarfs that bomb the core with `DAT` instructions, scanners that locate and attack enemy code, and fortress warriors that protect a compact core with a paper layer.

In our experiments, the MARS configuration duplicates the hyperparameters used by the single node implementation in the original DRQ paper Kumar et al. (2026). See Appendix A for the details. The fitness of a warrior w_i against a set of opponents $\mathcal{O}$ is:

$$f(w_i, \mathcal{O}) = \sum_{\tau \in \mathcal{T}} \frac{N}{\mathcal{T}} \frac{A_\tau^i}{\sum_{o \in \mathcal{O}} A_\tau^o} \tag{1}$$

where $\mathcal{T}$ is the number of simulation timesteps, and A_τ^i indicates if warrior i is alive at timestep τ, and N is the number of warriors in the battle.

2.2. MAP-Elites and Quality-Diversity Search

Quality-Diversity (QD) search (Chatzilygeroudis et al., 2020) seeks not a single optimal solution but a large, diverse collection of high-performing solutions spanning a user-defined behavioral space. MAP-Elites (Mouret & Clune, 2015) is the canonical QD algorithm: it maintains an *archive* $\mathcal{A}$ of cells indexed by a discretized behavioral characteristic (BC) space $\mathcal{B}$. Each cell stores at most one elite, the highest-fitness solution observed at that BC coordinate. At each iteration, an existing elite is selected, mutated, evaluated, and placed in the archive if it either fills an empty cell or improves over the current occupant.

Keeping with Kumar et al. (2026), we use a two-dimensional BC space defined by:

- **Time-Space Product (TSP):** the product of the warrior's code length and its average lifespan across battles, capturing the trade-off between spatial footprint and temporal persistence.
- **Memory Coverage (MC):** the fraction of the core that the warrior touches (reads, writes, or executes) during a battle, reflecting how aggressively it explores the address space.

Differing from the fitness used for archive placement (Equation (1)) which varies with iteration, we evaluate solution quality using a static held out cohort of human authored warriors. Here we take a solution's Generality as indicative of its quality:

$$\text{generality}(w_r, \mathcal{H}) = \frac{|\{h \in \mathcal{H} : \texttt{win_tie}(w_r, h)\}|}{|\mathcal{H}|} \quad (2)$$

where r is the round number, $\mathcal{H}$ is a fixed held-out set of human-authored warriors, and $\texttt{win_tie}()$ is 1 if warrior w_r wins/ties against warrior h and 0 if it loses.

2.3. Digital Red Queen (DRQ)

The Digital Red Queen framework (Kumar et al., 2026) integrates an LLM as the primary mutation operator inside MAP-Elites, replacing hand-crafted genetic operators with natural-language-conditioned program generation. At each iteration, an elite is randomly sampled from the archive, are serialized into a prompt and mutated by the LLM, or a new warrior is generated from scratch by the LLM. The adversarial Red Queen pressure is introduced by maintaining a pool of *round champions*, the highest-fitness warrior from each completed round, and using them as the opponent set in Equation (1). As the archive improves, the opponent pool grows stronger, continuously raising the selection pressure in a co-evolutionary arms race.

2.4. Asynchronous Champion Sharing

In a distributed evolutionary system with heterogeneous LLMs, nodes operate at different latencies, a local open-weight model may take an order of magnitude longer per call than a frontier model. A synchronization barrier would throttle the entire ensemble to the slowest participant. We therefore adopt a fully asynchronous all-gather communication pattern: each node publishes its round champion to its peers whenever ready, and peers consume incoming champions at their own pace. This design achieves reliable champion propagation without synchronization barriers, and is detailed in Appendix C.

Table 1. Experimental conditions. The budget per node is inversely proportional to N, keeping total LLM calls fixed.

Condition	N	Diversity	Purpose
Solo [DRQ]	1	n/a	Single-node baseline (each LLM)
Homogeneous Ensemble	4	Homogeneous	Same LLM on all nodes
Diverse Ensemble	4	Heterogeneous	Mixed LLMs

3. Method

3.1. System Architecture

Each participating node in DEI instantiates two cooperating subsystems: (i) an *asynchronous messaging layer* that handles champion sharing with peers, and (ii) a local *DRQ optimizer* that couples MAP-Elites with an LLM mutation engine. The DRQ code is the open source implementation supplied by Kumar et al. (2026), and was used unmodified for the *solo* runs with the communication layer included for the ensemble runs.

The communication layer handles asynchronous champion propagation between nodes. The DRQ optimizer runs locally and independently on each node, querying its assigned LLM for warrior generation and mutation.

3.2. Per-Node DRQ Loop

Within each round a node performs T LLM calls. With probability 0.1 the LLM *generates* a new warrior from scratch; with probability 0.9 it *mutates* an existing warrior sampled uniformly from the current archive. The fitness of a candidate warrior w under a round r's opponent pool $\mathcal{O}_r$ is given by Equation (1). The pool $\mathcal{O}_r$ comprises the round's initial seed warriors plus champion warriors retained from the previous K rounds, and the warriors collected from peer nodes.

At the end of each round the *champion* is selected:

$$\hat{w}_r = \arg\max_{w \in \mathcal{A}_r} f(w, \mathcal{O}_r). \quad (3)$$

The champion is broadcast to the node's peers. The node then drains its receive buffer to collect champions published by peer nodes during the same round. Received champions are (a) appended to the local opponent pool, providing cross-model adversarial pressure, and (b) seeded into the local archive if they occupy previously empty cells. The complete procedure is formalized in Algorithm 1.

3.3. Experimental Conditions

To disentangle the effect of additional compute from the effect of model diversity, we design three conditions, all holding the total LLM call budget constant. Table 1 summarizes the design.

The heterogeneous condition (Diverse Ensemble) dis-

Algorithm 1 DEI (node i)

Require: LLM_i, initial opponents $\mathcal{O}_0$, rounds R, iters T
$\mathcal{A}_i \leftarrow \{\}$; $\mathcal{O}_i \leftarrow \mathcal{O}_0$
for $r = 1$ **to** R **do**
  for $t = 1$ **to** T **do**
    if rand() < 0.1 **then**
      $w \leftarrow \text{LLM}_i.\textsc{Generate}(\text{new_prompt})$
    else
      $p \leftarrow \textsc{Sample}(\mathcal{A}_i)$
      $w \leftarrow \text{LLM}_i.\textsc{Mutate}(\text{mutate_prompt}, p)$
    end if
    $\mathbf{bc} \leftarrow \textsc{ComputeBC}(w)$ $\{(TSP, MC)$ from simulation$\}$
    $f \leftarrow \textsc{Evaluate}(w, \mathcal{O}_i)$ {Eq. (1)}
    $\mathcal{A}_i \leftarrow \textsc{MapElitesUpdate}(\mathcal{A}_i, \mathbf{bc}, w, f)$
  end for
  $\hat{w}_r \leftarrow \arg\max_{w \in \mathcal{A}_i} f(w, \mathcal{O}_i)$
  $\textsc{Publish}(\hat{w}_r)$
  $\mathcal{R} \leftarrow \textsc{Drain}()$
  $\mathcal{O}_i \leftarrow \mathcal{O}_i \cup \mathcal{R}$
  $\mathcal{A}_i \leftarrow \textsc{Seed}(\mathcal{A}_i, \mathcal{R})$
end for
return $\arg\max_{w \in \mathcal{A}_i} f(w, \mathcal{O}_i)$

tributes nodes across four models: GPT-5.4-mini, Claude Sonnet 4.6, GPT-5.2, and Claude Haiku 4.5. The homogeneous condition runs all four nodes with the same model, giving four Homogeneous Ensemble variants. The solo condition runs the DRQ algorithm on a single node in isolation.

3.4. Evaluation Metrics

We measure performance across four dimensions:

Generality: Fraction of a held-out set of human authored warriors the final champion defeats or ties Equation (2).

Archive coverage: Fraction of MAP-Elites grid cells occupied:

$$\text{coverage}(\mathcal{A}) = \frac{|\{b \in \mathcal{B} : \mathcal{A}(b) \neq \emptyset\}|}{|\mathcal{B}|},$$

where $\mathcal{B}$ is the full set of discretized behavioral cells and $\mathcal{A}(b)$ denotes the elite stored at cell b. For distributed conditions (Diverse Ensemble, Homogeneous Ensemble) we report the coverage of the *cross-node merged archive*, the union of the four nodes' final-round archives, keeping the best warrior per cell. This equalises total LLM-call budget against Solo ($4 \times 62 \approx 1 \times 250$ iters per round), so coverage differences reflect ensemble effects rather than per-node budget asymmetry.

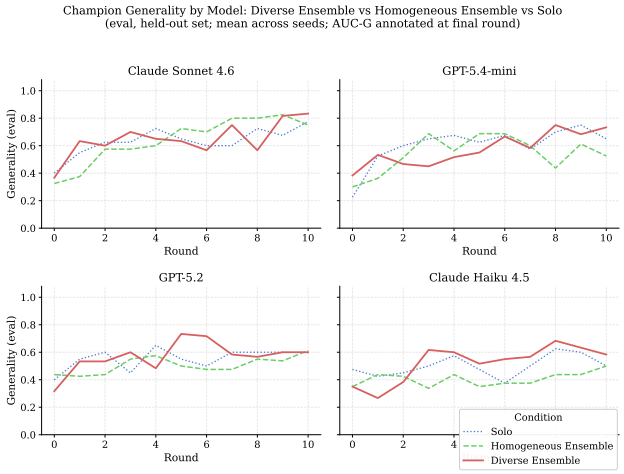

Figure 2. Champion generality (eval, held-out) over rounds per model. Each panel shows Diverse Ensemble, Homogeneous Ensemble, and Solo for one LLM. Diverse Ensemble achieves the highest generality for all four models, consistent with the diversity hypothesis.

QD-Score: Sum of fitness values across all occupied archive cells:

$$\text{QD-Score}(\mathcal{A}) = \sum_{b \in \mathcal{B}} f(\mathcal{A}(b), \mathcal{O}) \cdot \mathbf{1}[\mathcal{A}(b) \neq \emptyset],$$

where $f(\mathcal{A}(b), \mathcal{O})$ is the fitness of the elite at cell b against opponent set $\mathcal{O}$ (Equation (1)). QD-Score rewards both breadth (coverage) and quality (per-cell fitness), making it sensitive to improvements on either axis.

Niche novelty η: For each received champion w, $\eta = \mathbb{E}[\mathbf{1}[\mathbf{bc}(w) \notin \mathcal{A}_i^{(r-1)}]]$, the fraction of received champions landing in previously empty cells of the receiving node's archive.

4. Results

4.1. Individual Node

Results in Table 2 confirm the primary hypothesis across all four model families: Diverse collaboration achieves higher generality than homogeneous collaboration, which in turn outperforms or is comparable with solo across most conditions, under equal total LLM call budget.

Figure 2 shows the mean champion generality over rounds, broken down by model.

Niche novelty, while undefined in the solo case, meaningfully improves when moving from homogeneous to diverse ensembles. This supports the hypothesis that the breadth and quality of the archive is being increased due to collaborative peers, rather than originating from the node's LLM directly.

Table 2. Main results across all conditions. Mean ± std shown; bold = best within each model group. Horizontal rules separate model families.

Condition	Peak Gen.	Niche Nov. η
Diverse Ensemble (Claude Sonnet 4.6)	$\mathbf{0.850 \pm 0.087}$	$\mathbf{0.483 \pm 0.120}$
Homogeneous Ensemble (Claude Sonnet 4.6)	0.825 ± 0.106	0.348 ± 0.039
Solo [DRQ] (Claude Sonnet 4.6)	0.775 ± 0.035	—
Diverse Ensemble (GPT-5.4-mini)	$\mathbf{0.767 \pm 0.076}$	$\mathbf{0.422 \pm 0.072}$
Homogeneous Ensemble (GPT-5.4-mini)	0.725 ± 0.029	0.119 ± 0.013
Solo [DRQ] (GPT-5.4-mini)	0.750 ± 0.000	—
Diverse Ensemble (GPT-5.2)	$\mathbf{0.767 \pm 0.076}$	$\mathbf{0.454 \pm 0.070}$
Homogeneous Ensemble (GPT-5.2)	0.700 ± 0.000	0.091 ± 0.010
Solo [DRQ] (GPT-5.2)	0.650	—
Diverse Ensemble (Claude Haiku 4.5)	$\mathbf{0.700 \pm 0.050}$	$\mathbf{0.443 \pm 0.132}$
Homogeneous Ensemble (Claude Haiku 4.5)	0.538 ± 0.063	0.139 ± 0.040
Solo [DRQ] (Claude Haiku 4.5)	0.650 ± 0.141	—

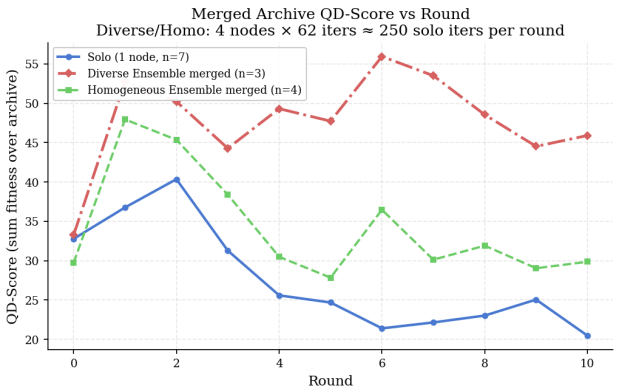

Figure 3. Merged archive QD-Score vs. round at equal total compute (4 nodes × 62 iters ≈ 250 solo iters per round). Each curve is the mean across all available runs. Both ensembles outperform Solo from round 1 onward; the homogeneous ensemble sustains the highest QD-Score through the final round, while the diverse ensemble leads on archive coverage (Table 3).

4.2. Merged Archive

The generality and niche novelty metrics above reflect individual node performance. A complementary view asks: when the archives from all four collaborative nodes are pooled into a single merged archive (keeping the best warrior per behavioral cell), how does the combined population compare to a solo node at equal total compute?

Figure 3 shows merged QD-Score vs. round, averaged across all available runs. At each round, the merged diverse archive is built from 4 nodes × 62 iterations = 248 total LLM calls, matching the solo budget of 250 calls per round.

Table 3 summarises the final-round merged archive. The homogeneous ensemble does not outperform the solo case in coverage. The diverse ensemble outperforms the solo baseline and the homogeneous ensemble on coverage and QD-Score, achieving $+124\%$ higher QD-Score and $+28\%$ more coverage than a solo node. The diverse ensemble yields the broadest coverage of the cell space (80%), while the homogeneous ensemble concentrates higher per-cell fitness which helps it to outperform the solo case on QD-Score at the final round.

Table 3. Cross-node merged archive at the final round vs. the single-node Solo archive, at equal total LLM-call budget.

Condition	Coverage	QD-Score
Solo	63.0%	20.46
Homo merged	59.0%	29.85
Diverse merged	80.6%	45.90

5. Discussion

The central finding, that diverse ensembles outperform homogeneous ensembles, which in turn outperform solo, holds across two of the three hypothesized complementary measurement axes: individual-node generality on the held-out set, (Table 2), and the merged population archive Coverage/QD-Score (Figure 3). The merged archive comparison is particularly compelling because it is compute-equalised by construction: four collaborative nodes running 62 iterations each contribute the same total LLM calls as one solo node running 250, yet the diverse merged archive achieves higher Coverage and QD-Score by the final round. These results reinforce the intuition from recent diversity-aware reinforcement learning (Li et al., 2025; Hu et al., 2026) and multi-agent reasoning work (Liang et al.,

2024): explicitly sourcing generative diversity, here by assigning distinct LLM families to distinct nodes, yields measurable gains over single-distribution baselines at matched compute.

The fully asynchronous design means that adding a slow node (e.g., local MLX inference on a laptop) improves coverage without penalizing faster nodes.

We study Core War as a self-contained benchmark with a clear fitness function and well-defined behavioral characteristics. The generality of our findings to domains with less structured BC spaces or more expensive fitness evaluations remains to be established. While there is no theoretical reason to believe these findings would not hold on other domains, our emprical results are strictly within this domain, it requires further testing to verify the applicability elsewhere.

Promising directions for further exploration include: adaptive topology (dynamically connecting nodes whose archives are most complementary), heterogeneous BC axes per node (each model tracks different behavioral dimensions), and extension to multi-agent collaborative tasks beyond Core War.

6. Related Work

LLMs as evolutionary operators: The use of LLMs as mutation and crossover operators has grown rapidly since the seminal Evolution through Large Models (Lehman et al., 2022) framework, which showed that an LLM pretrained on code can serve as a context-aware mutation operator in an evolutionary algorithm. EvoPrompting (Chen et al., 2023) applies LLM-driven code-level neural architecture search, and OPRO (Yang et al., 2023) frames optimization broadly as in-context learning. FunSearch (Romera-Paredes et al., 2023) scales this to a distributed setting with a large homogeneous pool of LLM evaluators searching for combinatorial mathematical functions. Wu et al. (2024) provide a comprehensive survey and roadmap of evolutionary computation in the LLM era. AlphaEvolve (Novikov et al., 2025) uses an ensemble of LLMs within an evolutionary loop, but restricts diversity to model *size* (pairing a smaller and a larger model from the same family) to "balance computational throughput with the quality of generated solutions." In contrast, we diversify across model *families* with the explicit goal of increasing sample diversity. We are the first to study such *heterogeneous* model ensembles in distributed evolutionary search.

Quality-Diversity algorithms: MAP-Elites (Mouret & Clune, 2015) introduced the foundational archive-plus-BC structure that all QD methods build on. Chatzilygeroudis et al. (2020) provide a thorough taxonomy. Differentiable QD (Fontaine & Nikolaidis, 2021) extends the framework to gradient-based methods; CVT-MAP-Elites (Vassiliades et al., 2016) scales the archive to high-dimensional BC spaces via centroidal Voronoi tessellations. Cully & Demiris (2018) introduce unsupervised behavioral descriptors that allow an archive to adapt its BC axes without hand-engineering, a direction orthogonal to, but complementary with, our multi-model approach. Quality-Diversity through AI Feedback (Bradley et al., 2023) replaces handcrafted fitness functions with LLM judges, a complementary direction to our work, which retains the simulation-based fitness but diversifies the *generator*. Flageat et al. (2022) provide a systematic benchmarking study of QD algorithms on neuroevolution tasks, establishing baselines we draw on for experimental protocol. Lehman & Stanley (2011) established that novelty alone (without explicit quality pressure) can drive exploration; our adversarial gossip mechanism adds quality pressure from cross-model opponents.

Distributed and island-model EAs: The island model (Cantú-Paz, 2001) is the classical blueprint for distributing evolutionary search: subpopulations evolve in parallel and exchange migrants periodically. Castillo et al. (2011) extends this to REST-based web services. Our asynchronous gossip substrate is in the spirit of the island model but differs in three key ways: (i) migration uses publish-subscribe rather than point-to-point topology, (ii) compute budgets across nodes are heterogeneous by design, and (iii) the *identity* of the LLM rather than random drift is the primary source of diversity. Core War has been explored with island-model co-evolution previously (Corno et al., 2003), but without LLM operators or QD objectives.

Adversarial co-evolution and Red Queen dynamics: The Red Queen hypothesis (Van Valen, 1973) describes evolutionary arms races in which species must continually adapt simply to maintain relative fitness against co-evolving competitors. Hillis (1990) brought this dynamic into computational search by co-evolving sorting networks against adversarial test cases, demonstrating that competitive co-evolution can escape local optima where direct optimisation stalls. Rosin & Belew (1997) formalised the methodology for competitive co-evolution in evolutionary algorithms, introducing hall-of-fame archives and shared sampling to stabilise the arms race. Kumar et al. (2026) extended this lineage to LLM-driven program evolution with the Digital Red Queen framework, using Core War as a testbed and instantiating Red Queen pressure inside MAP-Elites by retaining each round's champion as an opponent in subsequent rounds. We extend this further to the distributed setting: cross-node gossip delivers champions from heterogeneous LLMs as opponents, creating inter-model Red Queen pressure that cannot arise in single-

model self-play.

Code generation with LLMs: Our system relies on the ability of diverse LLMs, Codex-family models (Chen et al., 2021), instruction-tuned models like Claude and Gemini, and open-weight models such as Qwen, to generate syntactically valid Redcode warriors. AlphaCode (Li et al., 2022) demonstrated competition-level program synthesis; our work repurposes diverse code-generation capabilities as QD search operators.

Diversity aware RL and multi-agent reasoning: There is growing evidence in reinforcement learning that diversity can play a strong role in reasoning and multi-agent systems. DARLING (Li et al., 2025) jointly optimises a quality reward with a learned semantic-diversity signal during online RL post-training, and reports gains in both quality and novelty over quality-only RL on instruction-following and creative-writing benchmarks, alongside improved pass@k on competition math. DIVER (Hu et al., 2026) adds a global, sequence-level diversity intrinsic reward via potential-based reward shaping inside the RL-with-verifiable-rewards paradigm, and beats competing RLVR baselines on in- and out-of-domain reasoning at both pass@1 and pass@k. At inference time, multi-agent debate (Liang et al., 2024) elicits divergent thinking by having several LLM agents argue under a judge, addressing the degeneration of thought failure mode of single-agent self-reflection. These papers compliment this work and and inspire our exploration of diversity applied to distributed Quality-Diversity search.

7. Conclusion

We introduced DEI, a distributed Quality-Diversity search framework that exploits the distinct inductive biases of heterogeneous LLMs as the primary source of behavioral diversity. By connecting nodes via an asynchronous distributed communication layer, we enable cross-model adversarial pressure and niche novelty without synchronization barriers. A coverage argument shows that heterogeneous BC distributions yield strictly higher expected archive coverage than any single distribution at equal compute, and our experiments confirm this empirically: at equal total LLM call budget, the merged diverse archive achieves $+124\%$ higher QD-Score and $+28\%$ more coverage than a solo node. We argue that *model diversity*, not just parallelism, should be a first-class design principle in distributed LLM-based search.

Broader Impact

This work empirically advances distributed evolutionary search using heterogeneous LLM ensembles. The asynchronous, gossip-based architecture enables researchers without dedicated GPU clusters to contribute to collaborative search by running local open-weight models alongside cloud-hosted counterparts. Democratizing participation in LLM-driven program synthesis could accelerate scientific discovery in domains ranging from combinatorial optimization to automated theorem proving. The Quality-Diversity framing also produces richer, more interpretable result sets than single-objective optimization, supporting human-in-the-loop workflows where practitioners inspect the full archive rather than a single champion.

References

Arceliar and Alexander, N. Yggdrasil: An experiment in scalable routing as an encrypted IPv6 overlay network. https://github.com/Arceliar/yggdrasil-go/blob/master/doc/Whitepaper.md, 2019. Project whitepaper.

Bradley, H., Dai, A., Teufel, H., Zhang, J., Oostermeijer, K., Bellagente, M., Clune, J., Stanley, K., Schott, G., and Lehman, J. Quality-diversity through AI feedback, 2023. URL https://arxiv.org/abs/2310.13032.

Cantú-Paz, E. *Efficient and Accurate Parallel Genetic Algorithms*. Genetic Algorithms and Evolutionary Computation. Springer US, 2001. ISBN 9781461543695. doi: 10.1007/978-1-4615-4369-5. URL http://dx.doi.org/10.1007/978-1-4615-4369-5.

Castillo, P. A., Arenas, M. G., Mora, A. M., Laredo, J. L. J., Romero, G., Rivas, V. M., and Merelo, J. J. Distributed evolutionary computation using REST, 2011. URL https://arxiv.org/abs/1105.4971.

Chatzilygeroudis, K., Cully, A., Vassiliades, V., and Mouret, J.-B. Quality-Diversity optimization: a novel branch of stochastic optimization, 2020. URL https://arxiv.org/abs/2012.04322.

Chen, A., Dohan, D. M., and So, D. R. EvoPrompting: Language models for code-level neural architecture search, 2023. URL https://arxiv.org/abs/2302.14838.

Chen, M., Tworek, J., Jun, H., Yuan, Q., de Oliveira Pinto, H. P., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam,

J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., and Zaremba, W. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Corno, F., Sanchez, E., and Squillero, G. Exploiting coevolution and a modified island model to climb the Core War hill. In *The 2003 Congress on Evolutionary Computation (CEC '03)*, volume 3, pp. 2217–2221. IEEE, 2003. doi: 10.1109/CEC.2003.1299947. URL http://dx.doi.org/10.1109/CEC.2003.1299947.

Cully, A. and Demiris, Y. Hierarchical behavioral repertoires with unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '18)*, pp. 69–76. ACM, July 2018. doi: 10.1145/3205455.3205571. URL http://dx.doi.org/10.1145/3205455.3205571.

Flageat, M., Lim, B., Grillotti, L., Allard, M., Smith, S., and Cully, A. Benchmarking quality-diversity algorithms on neuroevolution for reinforcement learning, 2022. URL https://arxiv.org/abs/2211.02193.

Fontaine, M. C. and Nikolaidis, S. Differentiable quality diversity, 2021. URL https://arxiv.org/abs/2106.03894.

Gensyn AI. AXL: A p2p network for decentralized agentic and AI/ML applications. https://github.com/gensyn-ai/axl, 2026. Open-source software.

Hillis, W. D. Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena*, 42(1–3):228–234, 1990. doi: 10.1016/0167-2789(90)90076-2.

Hu, Z., Zhang, S., Li, Y., Yan, J., Hu, X., Cui, L., Qu, X., Chen, C., Cheng, Y., and Wang, Z. Diversity-incentivized exploration for versatile reasoning. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2509.26209.

Kumar, A., Bahlous-Boldi, R., Sharma, P., Isola, P., Risi, S., Tang, Y., and Ha, D. Digital red queen: Adversarial program evolution in core war with llms, 2026. URL https://arxiv.org/abs/2601.03335.

Lehman, J. and Stanley, K. O. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, June 2011. ISSN 1530-9304. doi: 10.1162/EVCO_a_00025.

Lehman, J., Gordon, J., Jain, S., Ndousse, K., Yeh, C., and Stanley, K. O. Evolution through large models, 2022. URL https://arxiv.org/abs/2206.08896.

Li, T., Zhang, Y., Yu, P., Saha, S., Khashabi, D., Weston, J., Lanchantin, J., and Wang, T. Jointly reinforcing diversity and quality in language model generations, 2025. URL https://arxiv.org/abs/2509.02534.

Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Lago, A. D., Hubert, T., Choy, P., de Masson d'Autume, C., Babuschkin, I., Chen, X., Huang, P.-S., Welbl, J., Gowal, S., Cherepanov, A., Molloy, J., Mankowitz, D. J., Robson, E. S., Kohli, P., de Freitas, N., Kavukcuoglu, K., and Vinyals, O. Competition-level code generation with AlphaCode, 2022. URL https://arxiv.org/abs/2203.07814.

Liang, T., He, Z., Jiao, W., Wang, X., Wang, Y., Wang, R., Yang, Y., Shi, S., and Tu, Z. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17889–17904, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.992. URL https://aclanthology.org/2024.emnlp-main.992/.

Mouret, J.-B. and Clune, J. Illuminating search spaces by mapping elites, 2015. URL https://arxiv.org/abs/1504.04909.

Novikov, A., V, N., Eisenberger, M., Dupont, E., Huang, P.-S., Wagner, A. Z., Shirobokov, S., Kozlovskii, B., Ruiz, F. J. R., Mehrabian, A., Kumar, M. P., See, A., Chaudhuri, S., Holland, G., Davies, A., Nowozin, S., Kohli, P., and Balog, M. Alphaevolve: A coding agent for scientific and algorithmic discovery, 2025. URL https://arxiv.org/abs/2506.13131.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J. R., Ellenberg, J. S., Wang, P., Fawzi, O., Kohli, P., and Fawzi, A. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, December 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06924-6. URL http://dx.doi.org/10.1038/s41586-023-06924-6.

Rosin, C. D. and Belew, R. K. New methods for competitive coevolution. *Evolutionary Computation*, 5(1):1–29, 1997. doi: 10.1162/evco.1997.5.1.1.

Van Valen, L. A new evolutionary law. *Evolutionary Theory*, 1:1–30, 1973.

Vassiliades, V., Chatzilygeroudis, K., and Mouret, J.-B. Using centroidal Voronoi tessellations to scale up the multi-dimensional archive of phenotypic elites algorithm, 2016. URL https://arxiv.org/abs/1610.05729.

Vyzovitis, D., Napora, Y., McCormick, D., Dias, D., and Psaras, Y. Gossipsub: Attack-resilient message propagation in the filecoin and ETH2.0 networks, 2020. URL https://arxiv.org/abs/2007.02754.

Wu, X., hao Wu, S., Wu, J., Feng, L., and Tan, K. C. Evolutionary computation in the era of large language model: Survey and roadmap, 2024. URL https://arxiv.org/abs/2401.10034.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers, 2023. URL https://arxiv.org/abs/2309.03409.

A. MARS Configuration Details

All simulations use the following MARS configuration, held constant across all experimental conditions:

- Core size: 8,000 instructions
- Maximum cycles per battle: 80,000
- Rounds per pair: 20
- Initial warrior placement: random, minimum separation enforced
- Process limit per warrior: unlimited (standard MARS)

B. LLM Prompt Templates

Two prompt templates govern LLM calls:

New warrior prompt (used for 10% of calls): instructs the LLM to generate a novel Redcode warrior from scratch, given a description of Core War rules and the current archive state.

Mutation prompt (used for 90% of calls): provides an existing warrior as context along with its fitness score and BC coordinates, and asks the LLM to produce an improved variant.

Full prompt text is included in the code release and in the DRQ repository: https://github.com/SakanaAI/drq/.

C. Network Implementation Details

C.1. Yggdrasil Overlay

Yggdrasil (Arceliar & Alexander, 2019) assigns each node a stable IPv6 address derived from its public key and performs NAT traversal via a distributed spanning-tree routing scheme. This allows nodes behind firewalls or consumer routers to participate without manual port forwarding.

C.2. AXL: Application Interface to the Network Layer

The bridge between the DRQ application and the Yggdrasil transport is provided by AXL (Gensyn AI, 2026), an open-source P2P network for decentralized agentic and AI/ML applications, developed by Gensyn AI. AXL is a Go binary that embeds the Yggdrasil core and exposes a local HTTP API for application interface. The DRQ process communicates exclusively with this local interface and never opens sockets to remote peers directly.

Architecture. AXL consists of four layers: (i) an *HTTP API* that accepts application requests on localhost; (ii) an inbound *message multiplexer* that dispatches arriving TCP messages to the appropriate handler; (iii) a *userspace TCP/IP stack* (gVisor) that requires no TUN device or root privileges; and (iv) the *Yggdrasil core*, which manages the node's ed25519 keypair, derives a stable `200::/7` IPv6 address from the public key, and peers over TLS/TCP.

API. The three endpoints used by DEI are:

- `POST /send` — fire-and-forget delivery of a raw byte payload to a peer identified by its 64-character hex-encoded ed25519 public key (header `X-Destination-Peer-Id`).
- `GET /recv` — polling dequeue of inbound messages; returns `204` when the queue is empty, or `200` with the raw payload and an `X-From-Peer-Id` header identifying the sender.
- `GET /topology` — returns this node's IPv6 address, public key, and current peer list, used at startup to obtain the node identity shared with collaborating nodes.

C.3. GossipSub Protocol

We implement Gossipsub on top of AXL over Yggdrasil as the asynchronous communication layer for our ensemble DRQ experiments. GossipSub (Vyzovitis et al., 2020) maintains a *mesh* of D peers per topic to which it eagerly pushes full message payloads. We use $D = 3$, giving a propagation diameter of $O(\log N)$ hops for N nodes. Beyond the mesh, the node lazily announces message availability via `IHAVE` control messages; recipients lacking the

announced message may request it with `IWANT`. A periodic heartbeat (default 1 s) triggers graft and prune operations to repair the mesh after churn.

D. LLM and Agent Usage

In the spirit of disclosure, we used an LLM-based coding and writing agent in the preparation of this paper. Specifically, the agent was used to: (i) co-author and revise prose in the manuscript, including drafting, restructuring, and tightening sections under the authors' direction; (ii) debug the experimental codebase; and (iii) write the plotting code that generates the tables and figures presented in the paper. All scientific claims, experimental design choices, and final wording were reviewed and approved by the authors, who take responsibility for the content.