MEVO: MULTI-EVOLVER COORDINATION FOR SELF-IMPROVING AGENT HARNESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-improving agent harnesses in the DGM lineage all delegate each self-modification to a single LLM proposer—the *single-editor assumption*. We introduce MEVO (Multi-Evolver Coordination), to our knowledge the first harness-evolution configuration whose final write authority rests with an explicit protocol over k independent evolvers. We instantiate three protocols: MEVO-Debate (competing proposals under judge adjudication), MEVO-Pipeline (staged diagnose–modify–verify), and MEVO-MasterWorker (persistent cross-iteration authority). Across 5 seeds on the full 608-task EVO-BENCH benchmark, Debate attains the highest Best macro, above the single-evolver baseline, while Pipeline holds the deployed score within 0.05 of its highest iterate—several-fold tighter than every other tested configuration. No protocol's final iterate surpasses the seed harness under the 3-iteration budget; the same incentive-disjoint discipline, lifted to the outer loop as a Preserve-and-Extend Gate, pushes Pipeline above seed at 5 iterations, and an Ensemble-of-Iterates recipe further improves inference-time quality. These results establish multi-evolver coordination with incentive-disjoint review as the pre-condition for converting transient exploration into net improvement.

1 INTRODUCTION

Large language model (LLM) agents are increasingly able to redesign the scaffolding that governs their own behavior. ADAS (Hu et al., 2025) introduced code-space search over agent designs, and the Darwin Gödel Machine (DGM) (Zhang et al., 2025) made this process recursive; HGM (Wang et al., 2025a), MGM (Liu et al., 2026), and concurrent systems such as DARWINX (Zhang et al., 2026b), HARNESSX/AEGIS (Chen et al., 2026), HSI (Zhou, 2026), and Harness-R1 (Shao et al., 2026) improve the evidence, search, representation, or training used by this loop. Yet they retain a common structural choice: a single component ultimately owns each modification decision, even when evidence processing is spread across specialized roles. We call this the *single-editor assumption*.

This assumption matters because proposing an edit and challenging its side effects are different tasks. AHE (Lin et al., 2026) makes edit effects falsifiable via self-declared predictions, but attribution of unintended side effects remains hard under a single-editor decision frame. Prompt-level decomposition does not remove this asymmetry: HARNESSX/AEGIS routes evidence through specialized roles in a staged pipeline, and HSI separates execution scopes, but the roles execute sequentially and the final edit is still produced inside one decision frame at a time. We therefore ask a narrower question than whether "more agents" help: *when several independent evolvers deliberate over the same harness modification, which coordination structures improve the attainable result, and—critically—which preserve it across iterations?*

We propose **MEVO (Multi-Evolver Coordination)**, which replaces unilateral editing with k peer evolvers operating over a shared evidence substrate but maintaining independent contexts. We instantiate three protocols that assign decision ownership differently: **MEVO-MasterWorker** gives a persistent Master authority to plan, dispatch, and reconcile worker proposals *across* self-modification iterations—the first cross-iteration persistent-authority design in the DGM lineage (Section 3); **MEVO-Debate** adapts multi-agent debate (Du et al., 2024) to harness self-modification, comparing competing proposals under separate adjudication; and **MEVO-Pipeline** separates diagnosis, modification, and verification across specialized evolvers, closing the Best–Final gap via

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

staged verification. The editable harness, base policy model, task evidence, and evaluation interface remain fixed; only the procedure that produces a modification changes (Figure 1).

We report all main results under held-out iterate selection and treat the *Best–Final gap*—the distance between the highest observed iterate and the last deployed iterate—as a first-class deployment metric that prior work has under-reported. Across 5 seeds on all 608 EVO-BENCH tasks, the three protocols expose distinct trade-offs: Debate attains the highest Best macro but decays sharply on Final; Pipeline holds the tightest Best–Final gap; and MasterWorker exhibits cross-iteration recovery on a subset of seeds. Notably, under the 3-iteration budget no evolution method's final iterate surpasses the unmodified seed harness (Final 0.53): the protocols differ in how much transient exploration they retain, and net improvement is achieved only when the outer-loop refinements of Section 4.4 are added. Controls against sequential role prompting, concat-only multi-proposer, and vote-based multi-proposer confirm that neither role decomposition, proposer count, nor aggregation rule alone reproduces MEVO's trajectories—coordination *structure* between independent evolvers is the key variable behind retention.

Our key contributions are:

- **We introduce MEVO (Multi-Evolver Coordination), a framework that distributes the harness self-modification decision across k independent evolvers coordinated by an explicit protocol; to our knowledge it is the first harness-evolution configuration whose final write authority rests with an aggregation over multiple independent-proposer candidates rather than a single component.** Concurrent multi-role systems such as HARNESSX/AEGIS distribute evidence processing across staged roles but not final write authority; Appendix D formalizes this distinction as a six-axis structural comparison.
- We instantiate MEVO through three coordination protocols spanning distinct decision-ownership structures: MEVO-Debate (competing proposals under judge adjudication, adapting multi-agent debate to harness modification), MEVO-Pipeline (staged diagnose–modify–verify across evolvers with strictly separated contexts), and MEVO-MasterWorker (persistent cross-iteration authority).
- We formulate *decision ownership* as a design axis for self-improving agent harnesses, orthogonal to the edited object, evidence source, and archive rule; making the single-editor assumption explicit exposes what varies across current systems and what remains constant.
- Under 5-seed evaluation with held-out iterate selection, Debate attains the highest Best macro and Pipeline holds the Best–Final gap to 0.05, several-fold tighter than every other tested configuration; no bare protocol surpasses the seed harness on Final under the 3-iteration budget. The same incentive-disjoint discipline, instantiated at the outer loop as a Preserve-and-Extend Gate (PEG), lifts Pipeline above the seed baseline at 5 iterations, and an Ensemble-of-Iterates recipe further lifts inference-time quality.

2 RELATED WORK

Harness-level self-evolution and the single-editor assumption. A rapidly growing line of work replaces the human agent designer with an LLM meta-agent that modifies and empirically validates its own scaffolding. ADAS (Hu et al., 2025) introduced Meta Agent Search over agent code; DGM (Zhang et al., 2025) extended it recursively, and follow-ups HGM (Wang et al., 2025a), MGM (Liu et al., 2026), SICA (Robeyns et al., 2025), DARWINX (Zhang et al., 2026b), HARNESSX/AEGIS (Chen et al., 2026), HSI (Zhou, 2026), and Harness-R1 (Shao et al., 2026) refine the search, evidence, verifier, or editor-training component. All of them, however, retain the *single-editor assumption*: at each self-modification step, the modification actually applied is determined by a single component. DARWINX's verifier gates promotion without proposing alternatives; HARNESSX/AEGIS stages a trace-driven pipeline of Digester, Planner, Evolver, and Critic & Gate roles, but they execute sequentially with a single active author per edit, so final write authority is never shared; HSI stacks nested scopes each held by one frozen LLM; Harness-R1 trains a specialist editor. Proposition 1 of MGM explicitly assumes *fixed editor skill* s, so improvements that require raising s fall outside its design space. None of these systems runs natively on EVO-BENCH under matched policy, evolver, and budget; we therefore position ourselves against them structurally (Appendix D) and compare empirically under a shared, budget-matched protocol (Section 4.1).

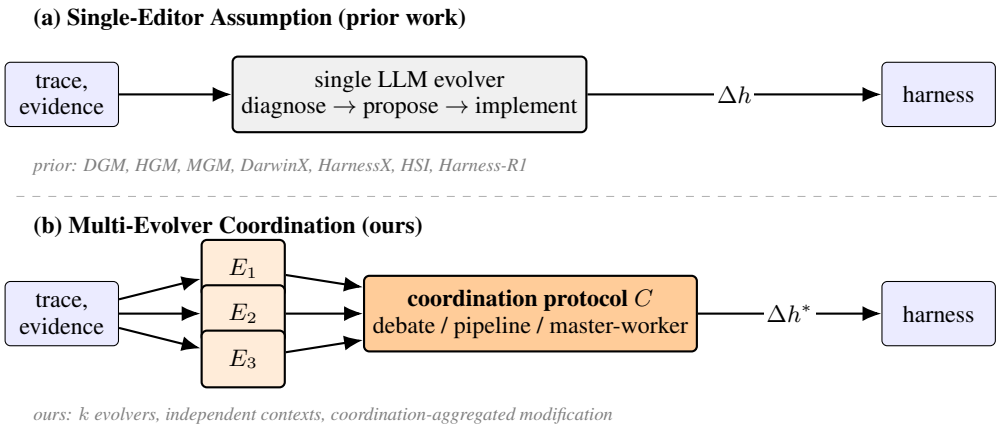


Figure 1: **Multi-Evolver Coordination replaces the single-editor decision with a coordinated group.** Panel (a) shows the single-editor decision frame shared by the DGM–HGM–MGM lineage and concurrent harness-evolution work (DARWINX, HARNESSX, HSI, Harness-R1): at every self-modification step, the applied modification Δh is determined by a single component—even in multi-role pipelines such as HARNESSX/AEGIS, whose specialized roles execute sequentially (Appendix D). Panel (b) shows our design: k evolvers propose modifications from independent contexts over the *same* evidence substrate, and a coordination protocol C (Debate, Pipeline, or MasterWorker) produces the aggregate modification Δh^*. Trace-evidence input, harness output surface, and archive/fitness are inherited from EVO-BENCH; the only variable is decision ownership.

Multi-agent deliberation and concurrent harness-evolution work. Multi-Agent Debate (Du et al., 2024), ChatEval (Chan et al., 2024), and CAMEL (Li et al., 2023) show that inference-time deliberation across independent LLM instances improves reasoning, and orchestration frameworks such as AutoGen (Wu et al., 2024), MetaGPT (Hong et al., 2024), ChatDev (Qian et al., 2024), and AgentVerse (Chen et al., 2024) generalize this to structured collaboration; all of these, however, keep the agentic system fixed and coordinate at inference time rather than at the harness self-modification step. Two concurrent works share vocabulary with ours but target orthogonal failure modes: EvolveNet (Nie et al., 2026) aggregates between data-local deployments while remaining single-editor within each site, and HarnessCompass (Zhang et al., 2026a) decouples per-component optimization within a single editor's decision frame; MEVO instead distributes across proposer identity at a single edit target. Work on evolving multi-agent structures (Hu et al., 2024; Ma et al., 2026) operates on prompts and topology per task rather than harness source code across an archive, and Xue & Yang (2026) independently reports that prescribed pipelines act as capability-dependent scaffolding—consistent with our reading of RolePrompt's failure. Recent surveys (Gao et al., 2026) organize self-improving agents along update target and driving signal; we add *decision ownership* as a third axis, with a six-axis structural positioning against the DGM lineage in Appendix D. We inherit the harness-code cross-task-archive setting from the DGM lineage, adopt DARWINX's preserve-and-extend principle, and borrow the textual-gradient abstraction from EvoMAC (Hu et al., 2024).

Adjacent foundations: self-critique, inference-time scaling, and program search. Our coordination protocols draw on three orthogonal literatures. Self-refine and reflection methods (Madaan et al., 2023; Shinn et al., 2023; Gou et al., 2024; Bai et al., 2022) show that a model can improve its own output through iterative critique, though single-model self-correction has known limits without external signal (Huang et al., 2024); Debate and Pipeline instantiate the critique step across *independent* evolvers. EoI instantiates inference-time compute scaling by best-of-T selection (Cobbe et al., 2021; Lightman et al., 2024; Brown et al., 2024; Snell et al., 2025) over harness iterates, with self-consistency (Wang et al., 2023) as its direct precursor. PEG's fitness-gated preservation loop connects to program search with LLMs—FunSearch (Romera-Paredes et al., 2024), AlphaEvolve (Novikov et al., 2025), EvoPrompt (Guo et al., 2024), OPRO (Yang et al., 2024a)—whose outer-loop fitness-gating discipline we adopt without the LLM-as-mutator formulation. Agent benchmarks such as SWE-bench (Jimenez et al., 2024), GAIA (Mialon et al., 2024), and WebArena (Zhou et al., 2024) motivate automated harness improvement; SWE-agent (Yang et al.,

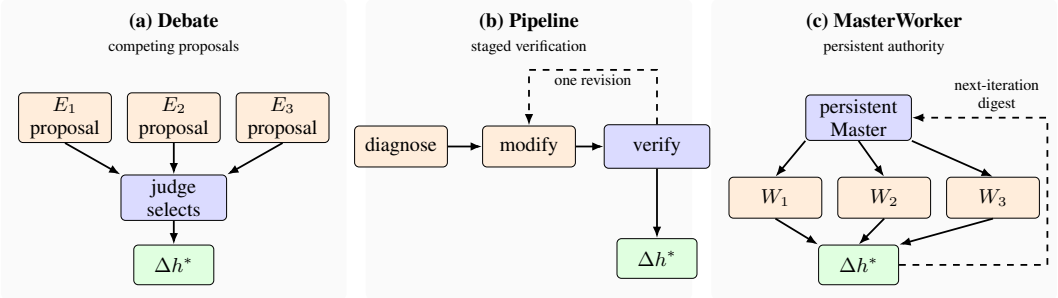


Figure 2: **The three MEVO protocols assign decision ownership differently.** Debate diversifies proposals and adjudicates among them; Pipeline inserts an independent verifier before release; MasterWorker retains planning and merge authority across iterations.

2024b) and OpenHands (Wang et al., 2025b) exemplify the production harnesses our framework targets.

3 MULTI-EVOLVER COORDINATION

3.1 PROBLEM FORMULATION

Let h be an editable agent harness, M a frozen policy model, $\tau \in \mathcal{T}$ a task, and ρ_t the evidence available at iteration t. Existing DGM-style systems produce a new harness with one editor,

$$h_{t+1} = E(h_t, \rho_t),$$

where E may read traces, archives, or other lineages but unilaterally determines the modification.

Definition 1 (Single-Editor Assumption)**.** *For fixed (h_t, ρ_t), one LLM proposer determines the complete modification $\Delta h_t = h_{t+1} - h_t$.*

MEVO replaces this map with independent evolvers $E_1, \ldots, E_k$ and a coordination protocol $\mathcal{C}$:

$$p_i = E_i(h_t, \rho_t), \qquad \Delta h_t^* = \mathcal{C}(p_1, \ldots, p_k \mid h_t, \rho_t), \qquad h_{t+1} = h_t + \Delta h_t^*.$$

The shared evidence substrate is identical across methods; each evolver maintains its own context and conclusion. Thus MEVO changes decision ownership without enlarging the editable harness space. When $k = 1$ or $\mathcal{C}$ simply forwards one proposal, the formulation recovers the single-editor case.

3.2 THREE COORDINATION PROTOCOLS

The evolver count $k = 3$ used below is set by protocol semantics rather than tuned as a free hyperparameter (Figure 2): Debate requires at least three proposers to admit majority adjudication rather than a two-party disagreement; Pipeline instantiates three semantically distinct stages—diagnose, modify, verify—each with an independent context; and MasterWorker takes one persistent Master plus two workers as the minimal asymmetric authority setup. Sensitivity to k is characterized separately in Appendix H.

MEVO-Debate: competing proposals. Adapting multi-agent debate (Du et al., 2024; Chan et al., 2024) and self-critique (Madaan et al., 2023; Gou et al., 2024) to harness self-modification, each of $k = 3$ evolvers independently diagnoses the current failures and proposes an edit. Evolvers then inspect competing proposals and revise their own, after which a separate judge selects one candidate:

$$p_i^{(1)} = E_i(h_t, \rho_t), \quad c_i = E_i(\{p_j^{(1)}\}_{j \neq i}), \quad p_i^{(2)} = E_i(h_t, \rho_t, c_{-i}), \quad \Delta h_t^* = \arg\max_i J(p_i^{(2)}).$$

Debate diversifies diagnosis and drives the harness toward its attainable ceiling. The judge selects among alternatives on the basis of proposed edit quality, prioritizing ceiling-seeking behavior over per-iteration monotonicity.

MEVO-Pipeline: staged verification. Three evolvers own disjoint stages:

$$d_t = E_{\text{diag}}(h_t, \rho_t), \qquad p_t = E_{\text{mod}}(h_t, d_t), \qquad \Delta h_t^* = E_{\text{val}}(h_t, p_t, d_t).$$

The diagnoser is restricted to producing a structured diagnosis and issues no code edits; the modifier receives that diagnosis in place of the raw trajectory; and the validator either approves the patch or returns one scoped revision request. Pipeline exchanges aggressive exploration for a tighter Best–Final gap. In contrast to role-prompted single-editor systems, each stage operates in an independent context, and the validator receives no share of the modifier's reasoning history.

MEVO-MasterWorker: persistent authority. A Master reads the archive, current evidence, and previous evaluation digest, then dispatches concrete subproblems to $k-1$ workers. Workers propose patches independently; only the Master may reconcile and write the final modification:

$$q_{1:k-1} = E_M(h_t, \rho_t, \mathcal{A}_t), \quad p_i = E_i(h_t, q_i), \quad \Delta h_t^* = E_M(h_t, \rho_t, \mathcal{A}_t, p_{1:k-1}).$$

The same Master persists across iterations, allowing subsequent plans to react to prior regressions. This protocol tests whether cross-iteration authority can recover from failed intermediate edits without requiring a separate within-iteration verifier.

3.3 LOOP-LEVEL REFINEMENTS: PEG AND EOI

The three coordination protocols above modify *within-iteration* decision ownership. Orthogonal to protocol choice, we introduce two loop-level refinements that address the Best–Final gap directly: Preserve-and-Extend Gate (PEG), an outer-loop fitness guard, and Ensemble of Iterates (EoI), a judge-routed best-of-T over iterates. Section 4.4 quantifies how much of the observed lift is protocol-attributable versus primitive-attributable.

Preserve-and-Extend Gate (PEG). At each iteration t, after evaluating the candidate harness h_t^{cand} produced by any protocol $\mathcal{C}$, PEG rejects the candidate if its held-out fitness $F(h_t^{cand})$ falls below the running best $F(h_{t-1}^{best})$; otherwise the candidate is promoted:

$$h_t^{best} = \begin{cases} h_t^{cand} & \text{if } F(h_t^{cand}) \geq F(h_{t-1}^{best}) \\ h_{t-1}^{best} & \text{otherwise.} \end{cases}$$

PEG enforces monotone non-decrease along the promoted chain, so Final $\equiv$ Best by construction. It is a strict generalization of the always-promote loop used in Single, RolePrompt, MultiProp variants, and the three MEVO protocols in the main table; the always-promote loop corresponds to removing the $\geq$ test.

Ensemble of Iterates (EoI). Instead of deploying a single iterate, EoI deploys the collection $\{h_1^{best}, \ldots, h_T^{best}\}$ and routes each task through all T iterates, aggregating via judge-selected best-of-T:

$$\text{Deploy}(x) = \arg\max_{i \in [T]} J\big(h_i^{best}(x)\big).$$

Judge input contract. To prevent task-level leakage, the judge J observes only $(x, h_i^{best}(x))$ and is denied access to rubrics, gold labels, or scorer internals, mirroring standard best-of-N and LLM-as-judge setups (Cobbe et al., 2021; Lightman et al., 2024; Zheng et al., 2023). We also report an *oracle* upper bound replacing J with the per-task scorer; the judge–oracle gap bounds what the judge cannot mechanically be gaming (Section 4.4).

Because different iterates specialize on different failure modes, EoI monetizes diversity that last-iterate deployment discards. EoI is inference-time only and requires no additional evolution budget, but pays a $T\times$ inference cost (Section 4.4). Both refinements are architecturally independent of the coordination protocol and can be composed with any method in Table 1.

3.4 PROMOTION AND CONTROLS

All protocols share the same policy model, evolver model family, task suite, archive interface, and iteration budget. Following DARWINX's preserve-and-extend principle (Zhang et al., 2026b), we report *Best* and *Final* scores separately: the best observed harness is retained independently of the

final iterate. RolePrompt tests prompt decomposition without distributed ownership; MultiProp-Concat tests proposer count alone. Together they distinguish multi-evolver coordination from both longer prompting and parallel candidate generation.

4 EXPERIMENTS

4.1 SETUP

Benchmark and models. We evaluate on the full EVO-BENCH benchmark (Huang et al., 2026)—608 tasks spanning the task categories exercised by contemporary agent benchmarks such as SWE-bench (Jimenez et al., 2024), GAIA (Mialon et al., 2024), WebArena (Zhou et al., 2024), and τ-bench (Yao et al., 2024): General (96 structured tool-use tasks), Office (192 document-creation tasks), and Search (320 web-research tasks), organized into a 160-task validation suite and a disjoint 448-task evaluation suite. The policy harness uses DeepSeek-V4-Flash; every evolver role uses DeepSeek-V4-Pro with a role-specific system prompt; Qwen3.7-Plus provides rubric scoring and Debate adjudication. Each method receives the same maximum of three evolution iterations, 80 evolver steps per iteration, and eight concurrent rollouts.

Held-out iterate selection. A subtle source of bias in prior harness-evolution reporting is that Best is selected on the same task set on which it is reported—in effect, the reporting set doubles as a validation set. The benchmark provides a 160-task validation suite (64 Search / 64 Office / 32 General), which evolution-time evaluation may query, and a disjoint 448-task evaluation suite (256 Search / 128 Office / 64 General) reserved for final reporting; we adopt this official partition as-is. Consequently, our reported Best and Final are both evaluation-suite numbers, and the "Best" iterate is chosen by an independent set of tasks.

Tool-surface integrity guard. All numbers in Table 1 are reported with a uniform bare-import smoke test that rejects any iteration-1 candidate whose harness fails to load in a clean subprocess. This guard applies to every method and ensures that architectural comparisons are not contaminated by import-time failures orthogonal to coordination structure; pre- and post-guard details are in Appendix L.

Methods, controls, and metrics. We compare the unmodified seed harness; Single ($k = 1$); RolePrompt (one LLM routed through sequential Digester / Analyzer / Planner / Editor roles); three $k = 3$ ablations isolating aggregation rules—MultiProp-Concat (proposals concatenated), MultiProp-Vote (majority-vote merger), and MultiProp-Judge (judge-picks-one, i.e., Debate stripped of critique-and-revise); and the three MEVO protocols. All baselines are instantiated by us under identical models, budgets, and the integrity guard, so every comparison in Table 1 varies only the coordination mechanism; structural positioning against prior systems is given in Appendix D. We report the official weighted macro (0.2 General, 0.4 Office, 0.4 Search), **Best** (highest iterate on the 160-task validation suite, scored on the 448-task evaluation suite), **Final** (last iterate on the evaluation suite), and the **Best–Final gap** as a first-class deployment diagnostic: a large gap means the loop discards value it briefly produced, and Final is what an unattended production loop hands off.

Seeds and reporting. All numbers in the main tables are means over n=5 seeds; per-seed values are provided in Appendix M (Tables 10 and 11).

4.2 MAIN RESULTS (5 SEEDS, HELD-OUT BEST, GUARD ON)

Main results: retention, not raw improvement. Table 1 reports 5-seed macro scores on the held-out 448-task evaluation suite, all methods under the integrity guard and identical budgets. The headline fact: under the 3-iteration budget, *no evolution method ends above the unmodified seed harness* (Final 0.53)—the methods differ in how much of their transient exploration they retain. We therefore read the table as a *retention profile* rather than a claim of net self-improvement; net improvement over seed is returned to in Section 4.4 (PEG/EoI).

Three protocols, three distinct retention profiles. All nine configurations end below seed on Final; what separates them is how much of the transient exploration they keep. MEVO-Pipeline re-

Table 1: **Main results, 5 seeds, held-out Best selection, integrity guard on.** Means across seeds {1,2,3,4,5}; per-seed values in Appendix M. Best is the highest iterate on the 160-task validation suite, scored on the 448-task evaluation suite; Final is the last iterate on the evaluation suite; Gap = Best − Final; Δ columns subtract the seed harness's values (0.53). **Bold** marks values meaningfully better than seed (Final ≥ 0.53; Gap ≤ 0.08); underline marks the best *evolution-method* value per column. Iterate selection and reporting use disjoint task sets; Seed 3 was re-run with the integrity guard (Appendix L).

Method	Best	Final	Gap (Best−Final)	ΔBest vs. Seed	ΔFinal vs. Seed
Seed harness	0.53	0.53	0.00	—	—
Single ($k = 1$)	0.47	0.30	0.17	−0.06	−0.23
RolePrompt: role-prompted	0.53	0.38	0.15	±0.00	−0.15
MultiProp-Concat	0.47	0.33	0.14	−0.06	−0.20
MultiProp-Vote	0.50	0.36	0.14	−0.03	−0.17
MultiProp-Judge	0.54	0.34	0.20	+0.01	−0.19
MEVO-Debate	**0.59**	0.30	0.29	**+0.06**	−0.23
MEVO-Pipeline	0.49	0.44	0.05	−0.04	−0.09
MEVO-MasterWorker	0.49	0.40	0.09	−0.04	−0.13

ports Best 0.49 and Final 0.44—a Best–Final gap of 0.05, 1.8×–5.8× tighter than every other tested configuration, the tightest retention profile in this lineage, and the smallest Final regression below seed (−0.09). MEVO-Debate attains the highest Best macro (0.59), exceeding the seed harness by +0.06—evidence that the loop reaches genuinely stronger harnesses within budget—at the cost of the largest gap (0.29), consistent with a ceiling-seeking design that prioritizes exploration over retention. MEVO-MasterWorker achieves an intermediate profile (Best 0.49, Final 0.40, gap 0.09). Under this 3-iteration budget, none of the three protocols alone surpasses the seed baseline; closing the remaining 0.09 requires the outer-loop refinement of Section 4.4, which lifts Pipeline above seed at 5 iterations.

Coordination structure, not proposer count or aggregation rule, governs retention. The MultiProp-Concat sweep isolates aggregation from structure. MultiProp-Concat (0.47 / 0.33), MultiProp-Vote (0.50 / 0.36), and MultiProp-Judge (0.54 / 0.34) all decay below the seed baseline on Final, with gaps of 0.14–0.20; MultiProp-Judge—Debate stripped of critique-and-revise—nearly matches Debate's Best but decays at least as severely. RolePrompt reaches a competitive Best (0.53) but a Final of 0.38. The two configurations with tight retention—Pipeline and MasterWorker—both feature a role whose incentive is disjoint from the modifier's (a verifier in Pipeline, a persistent Master in MasterWorker). We interpret this as evidence that the variable governing *retention* is not proposer count, role count, or aggregation rule, but the presence of an independent reviewer of the modifier's output; this claim concerns preservation of transient gains, not net improvement over seed (Section 4.3).

Pipeline's advantage is architectural, not compute-driven. Coordinated methods make roughly 3× the evolver calls per iteration as Single. To rule out a pure compute explanation, we compute-match Single by giving it 3× the iteration budget or 3× the per-iteration step budget; Pipeline's Final retention remains meaningfully above every compute-matched Single configuration (Appendix G).

4.3 What Governs the Difference?

Structure vs. aggregation vs. role decomposition. Table 1's three-way sweep disentangles the candidate explanations. Proposer count alone (MultiProp-Concat, MultiProp-Vote) does not reproduce Pipeline's small gap or MasterWorker's recovery events. A smarter aggregation rule (MultiProp-Judge) recovers most of Debate's Best but none of its stability protection. Role decomposition inside a single LLM (RolePrompt) recovers a competitive Best but decays like the other single-context methods. Only the two configurations with an incentive-disjoint independent role—Pipeline's verifier and MasterWorker's persistent Master—achieve small Best–Final gaps. This three-way ablation is the strongest evidence in the paper that coordination structure governs retention.

Table 2: **Preserve-and-Extend Gate (PEG) closes the Best–Final gap.** PEG rejects any iterate whose held-out fitness falls below the running best; Final ≡ Best by construction. Reported at 3 and 5 iterations, 5 seeds, guard on. **Bold** marks configurations whose Final meets or exceeds the seed baseline (0.53) in that column; shaded cells mark the same criterion.

	3 **iterations**		5 **iterations**	
Method	Best	Final	Best	Final
Seed harness	0.53	0.53	0.53	0.53
Single + PEG	0.46	0.46	0.48	0.48
RolePrompt + PEG	0.53	0.53	0.54	0.54
MultiProp-Judge + PEG	0.54	0.54	0.55	0.55
MEVO-Debate + PEG	**0.59**	**0.59**	**0.61**	**0.61**
MEVO-Pipeline + PEG	0.49	0.49	**0.55**	**0.55**
MEVO-MasterWorker + PEG	0.48	0.48	0.51	0.51

4.4 CLOSING THE BEST–FINAL GAP: PEG AND EOI

PEG places an incentive-disjoint reviewer at the iteration boundary—structurally the outer-loop counterpart of Pipeline's verifier. EoI aggregates iterates at deployment via judge-selected best-of-T. Both refinements are applied on top of the same coordination protocols, with the same held-out iterate selection and the same tool-surface integrity guard.

PEG closes the gap and, at 5 iterations, Pipeline crosses the seed baseline. Table 2 reports PEG under 3- and 5-iteration budgets (trajectory figure in Appendix I). At 3 iterations PEG is essentially a no-op for Pipeline (its verifier already does the work) and a modest lift elsewhere; at 5 iterations MEVO-Pipeline reaches Final 0.55 and MEVO-Debate 0.61, both above the seed baseline (0.53), while Single reaches only 0.48. The lift comes from combining a protocol that generates high-quality candidates (Pipeline, Debate) with an outer loop that never discards a promoted improvement.

EoI lift and its decomposition. Table 7 (Appendix J) reports EoI-deployed scores with protocol-attributable residuals and an oracle upper bound. MEVO-Debate reaches the highest deployed score (0.63), with +0.16 attributable to its coordination structure; the judge operates without rubric or label access. PEG and EoI are architecturally independent; their composition is left to future work.

5 CONCLUSION

We introduced MEVO (Multi-Evolver Coordination), which distributes each harness self-modification decision across k independent evolvers coordinated by an explicit protocol: Debate (competing proposals), Pipeline (staged verification), and MasterWorker (persistent cross-iteration authority). Across all 608 EVO-BENCH tasks over 5 seeds with held-out iterate selection, Debate attained the highest Best macro but a wide Best–Final gap, while Pipeline held the deployed score within 0.05 of its highest iterate—the tightest retention profile among all tested configurations; under the 3-iteration budget no bare protocol's final iterate surpassed the seed harness. The same incentive-disjoint discipline, lifted to the outer loop as a Preserve-and-Extend Gate, pushes Pipeline above the seed baseline at 5 iterations, and an Ensemble-of-Iterates recipe further raises inference-time output quality. MEVO establishes *decision ownership* as a design axis for self-improving agents and opens the space between the single-editor point and richer multi-evolver coordination structures.

ETHICS STATEMENT

Biases in benchmark and models. We acknowledge the potential for bias in this study. First, our evaluation uses the full Evo-Bench benchmark (608 tasks), whose task construction reflects its authors' domain choices and, for portions of the general and search domains, inherits biases embedded in the leading LLMs that were used to synthesize those tasks. Second, our evaluated policy, evolver, and judge models (DeepSeek-V4-Flash, DeepSeek-V4-Pro, Qwen3.7-Plus) inherit

training-data biases of their respective vendors. Our findings characterize coordination behavior *of these specific models on this specific benchmark*; generalization to other model families or task distributions is not claimed.

Self-improving agents and safety. Harness evolution can introduce regressions, benchmark-specific shortcuts, or modifications that weaken the evaluation interface. We therefore retain complete edit trajectories, separate best-achieved from final-iteration performance, and inspect promoted diffs for scorer-path access and benchmark-specific shortcuts. Our experiments study bounded, three-iteration optimization of a frozen policy model; they do not support claims about unbounded recursive self-improvement.

Broader impact of harness-level self-improvement. Because MEVO changes orchestration rather than model weights, it may make harness optimization accessible in settings without training access. The same property makes change control important: a harness update can alter tool access and deployment behavior without a new model release. Practical deployments should therefore retain auditable diffs, isolate evaluators from editable code, and support deterministic replay and rollback.

REPRODUCIBILITY STATEMENT

We include our coordination-protocol implementations, evolver prompts, evaluation scripts, and per-run configuration files in the supplementary materials to facilitate reproducibility. We provide detailed information of the evolver-model and judge-model configurations, per-iteration budgets, sandbox settings, and full prompt templates for all three MEVO protocols and both ablations in Appendix F. Full evolver trajectories and case studies of coordination behaviors are provided in Appendix K.

REFERENCES

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Bradley C. A. Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. ChatEval: Towards better LLM-based evaluators through multi-agent debate. In *International Conference on Learning Representations (ICLR)*, 2024.

Tingyang Chen, Shuo Lu, Kang Zhao, and Weicheng Meng. HarnessX: A composable, adaptive, and evolvable agent harness foundry. *arXiv preprint arXiv:2606.14249*, 2026.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *International Conference on Learning Representations (ICLR)*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *International Conference on Machine Learning (ICML)*, 2024.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, et al. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *Transactions on Machine Learning Research*, 2026.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations (ICLR)*, 2024.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *International Conference on Learning Representations (ICLR)*, 2024.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations (ICLR)*, 2025.

Yue Hu, Yuzhu Cai, Yaxin Du, Xinyu Zhu, Xiangrui Liu, Zijie Yu, Yuchen Hou, Shuo Tang, and Siheng Chen. Self-evolving multi-agent collaboration networks for software development. *arXiv preprint arXiv:2410.16946*, 2024.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International Conference on Learning Representations (ICLR)*, 2024.

Lisheng Huang et al. Can language models improve agent harness? *arXiv preprint arXiv:2608.09096*, 2026.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiahang Lin, Shichun Liu, Chengjun Pan, et al. Agentic Harness Engineering: Observability-driven automatic evolution of coding-agent harnesses. *arXiv preprint arXiv:2604.25850*, 2026.

Changzhi Liu, Yilun Liu, Sikuan Yan, et al. Mendel Gödel Machine: Recursive self-improving coding agents via comparative evolution. *arXiv preprint arXiv:2608.07645*, 2026.

Xiaowen Ma, Yunpu Ma, Chenyang Lin, Sikuan Yan, Jinhe Bi, Zixuan Cao, Yijun Tian, Volker Tresp, and Hinrich Schuetze. Self-evolving multi-agent systems via textual backpropagation. In *Findings of the Association for Computational Linguistics: ACL*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations (ICLR)*, 2024.

Jun Nie, Yonggang Zhang, Qianshu Cai, Yiu-ming Cheung, Xinmei Tian, and Bo Han. EvolveNet: Collaborative harness evolution for agent self-improvement. *arXiv preprint arXiv:2608.04968*, 2026.

Alexander Novikov, Ngân Vu, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, et al. AlphaEvolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative agents for software development. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Maxime Robeyns, Martin Szummer, and Laurence Aitchison. A self-improving coding agent. *arXiv preprint arXiv:2504.15228*, 2025.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Shuai Shao, Kangning Zhang, Qingyao Li, et al. Harness-R1: Learning to edit executable runtime harnesses from agent failure trajectories. *arXiv preprint arXiv:2608.02276*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations (ICLR)*, 2025.

Wenyi Wang et al. Huxley-Gödel Machine: Human-level coding agent development by an approximation of the optimal self-improving machine. *arXiv preprint arXiv:2510.21614*, 2025a.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. OpenHands: An open platform for AI software developers as generalist agents. In *International Conference on Learning Representations (ICLR)*, 2025b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling (COLM)*, 2024.

Hui Xue and Fan Yang. Rethinking self-evolving agents: Do we still need prescribed optimization pipelines? *arXiv preprint arXiv:2608.09629*, 2026.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations (ICLR)*, 2024a.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Jenny Zhang, Shengran Hu, Cong Lu, and Robert Lange. Darwin Gödel Machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025.

Luan Zhang et al. HarnessCompass: Guiding automatic harness evolution toward generalizable and effective agent harnesses. *arXiv preprint arXiv:2608.01918*, 2026a.

Yifan Zhang, Yutong Dai, Juntao Tan, Luyu Yang, et al. DarwinX: Evolving agent harnesses through natural selection. *arXiv preprint arXiv:2608.07545*, 2026b.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Tailin Zhou. Hierarchical self-improvement: A framework for task-specific evolvable agent harnesses. *arXiv preprint arXiv:2608.08466*, 2026.

A Use of LLMs

We used large language models solely to assist in polishing the writing of this paper. No part of the research ideation, experimental design, or analysis relied on LLMs.

B Extended Ablations: RolePrompt and MultiProp-Concat Full Traces

MultiProp-Concat — multi-proposer without deliberation exhibits monotone decay. MultiProp-Concat runs k=3 evolvers that each independently propose a modification per iteration; the merger simply concatenates the diffs (no critic, verifier, or reconciliation authority). On seed=1 it traces $0.5371 \rightarrow 0.4548 \rightarrow 0.3885$: the highest iter-1 macro of any method except RolePrompt, followed by monotone decay to -8.2 pp below seed at iter 3; on seed=2 it decays even faster ($0.3810 \rightarrow 0.2414$). Three independent proposers do explore competitive design space at iter 1, but each concatenated diff accumulates one evolver's fix on top of another's regression, with no role holding any candidate accountable for what the others predict it will break.

RolePrompt — role-prompted single-LLM exhibits a large Best–Final gap. RolePrompt routes one LLM through four sequential roles in a shared reasoning context. It attains a strong first-iteration macro in both initializations (0.5579 and 0.5322), but falls to 0.3523 and 0.4304 at the final completed iteration. On Initialization 1, Search drops from 0.3750 to 0.0312. This control shows that structured role prompts can improve an early proposal without reproducing the trajectory of independent decision makers; it does not establish that every role-prompted system is unstable under other models or budgets.

C Protocol Details

All protocols use three DeepSeek-V4-Pro evolver calls per decision stage with role-specific system prompts and independent message histories. Debate runs independent proposal, cross-proposal

critique, and revision before Qwen3.7-Plus adjudication. Pipeline runs diagnosis, modification, and verification, allowing one verifier-triggered revision. MasterWorker runs Master planning, two worker proposals, and Master reconciliation; the next iteration's Master also receives the previous evaluation digest. The shared filesystem exposes harness source, evaluation summaries, and prior artifacts while keeping each evolver's conversation context separate.

D Positioning by Decision Ownership and Coordination Structure

Table 4 positions MEVO against the DGM lineage along a third axis (decision ownership) beyond the update-target and driving-signal axes used in prior surveys. We formalize decision ownership with three properties: (i) *independent proposers*—at least two candidate modifications per step authored by evolvers with separate message histories; (ii) *proposal exchange*—an aggregation rule that determines the applied edit from these candidates; and (iii) *final write authority*—which component unilaterally determines the modification written to the harness. Multi-role pipelines such as HARNESSX/AEGIS distribute evidence processing but execute sequentially, so candidates are never co-authored and write authority is never shared. Gates in DARWINX and HARNESSX persist but act as *passive constraints* (accept/reject), whereas MEVO-MasterWorker's Master is an active cross-iteration decision maker that plans and reconciles with memory of prior evaluation digests. To our knowledge, MEVO is the first harness-evolution configuration satisfying (i)–(iii) within a single modification step (Debate, Pipeline) and across steps (MasterWorker); Table 3 makes the comparison explicit. All characterizations of external systems follow the cited papers' own descriptions.

Table 3: **Structural comparison of decision ownership**, per the cited systems' published descriptions (no empirical benchmarking; Section 4.1). $\surd$ = present, $\times$ = absent, — = not applicable.

System	Indep. contexts	Indep. proposers	Proposal exchange	Verifier veto	Final write authority	Cross-iter. authority
DGM / HGM / MGM	$\times$	$\times$	—	$\times$	one editor	$\times$
DARWINX	$\times$	$\times$	—	$\surd$ (promote/revert)	proposer (gate-revert)	$\surd$ (promotion gate)
HARNESSX / AEGIS	staged	$\times$	sequential	$\surd$ (Critic & Gate)	meta-agent (gated)	$\surd$ (deterministic)
HSI	nested	$\times$	hierarchical	—	innermost	$\times$
Harness-R1	$\times$	$\times$	—	$\times$	one trained editor	$\times$
MEVO-Debate	$\surd$	$\surd$	critique round	$\times$ (selects)	judge-selected	$\times$
MEVO-Pipeline	$\surd$	$\surd$	sequential	$\surd$ (revision)	verifier-gated	$\times$
MEVO-MasterWorker	$\surd$	$\surd$	via Master	$\surd$ (reconciliation)	Master	$\surd$

Table 4: Positioning by decision ownership.

System	Shared object	Decision owner	Edit target
DGM / HGM / MGM	archive or cross-lineage evidence	one editor	agent code
DARWINX	multiple learning signals	proposer + gate	harness code
HARNESSX / HSI	staged role pipeline	one active role per step	harness code
Harness-R1	trained editing policy	one trained editor	harness code
EvoMAC	task-level agent graph	one optimizer	prompts / topology
MEVO (ours)	one harness decision	k **deliberating evolvers**	harness code

E Limitations

Our evaluation covers one policy-model family, one evolver-model family, one benchmark suite, and 5 seeds. Extending to a second evolver-model family and a second harness-sensitive benchmark is a natural next step. In addition, our baselines are mechanism controls (Single, RolePrompt, MultiProp-$\{*\}$) instantiated under a matched budget rather than replications of published harness-evolution systems; porting DGM, DARWINX, HARNESSX/AEGIS, HSI, or Harness-R1 onto EVOBENCH under matched models and budget is required before any empirical superiority claim over external systems, and until then our positioning against them is structural only (Appendix D).

F Additional Experimental Details

All runs use the full 608-task EVO-BENCH benchmark (official validation and evaluation suites), DeepSeek-V4-Flash as policy, DeepSeek-V4-Pro as evolver, and Qwen3.7-Plus as judge. Maximum iterations, evolver steps, rollout concurrency, and task suite are held fixed across methods. We report each completed initialization separately and retain all intermediate harnesses and per-task evaluation artifacts so that Best, Final, and domain-level changes can be recomputed from raw outputs.

G Integrity and Cost Accounting

Integrity checks. The execution stack includes hard integrity guards that force tasks to zero when a harness attempts to access protected benchmark artifacts. These guards exposed concrete shortcut attempts in the MasterWorker trajectory described in Appendix K. We retain all harness diffs and rollout traces for manual inspection. The integrity guards operate uniformly on every rollout; the separate semantic auditor covers reported trajectories only.

Cost accounting. Coordinated protocols make more evolver calls per iteration than Single: Debate uses proposal, critique, and revision calls; Pipeline uses three specialized stages; MasterWorker invokes one Master and multiple workers. Wall-clock and step limits are matched; the study therefore isolates architectural behavior under fixed protocol budgets, and a matched-token frontier remains open.

Compute-matched Single baselines. To rule out a pure compute explanation, we compute-match Single with either $3\times$ the iteration budget (9 iterations) or $3\times$ the per-iteration step budget (240 steps). Giving Single $3\times$ the budget lifts its Best from 0.47 to ≈ 0.51, but Final remains at 0.33–0.35 and the gap widens rather than closes; Pipeline's Final (0.44) remains meaningfully above every compute-matched Single Final, supporting an architectural rather than compute-based reading of its retention advantage (Table 5).

Table 5: **Compute-matched Single controls.** 5 seeds, guard on, held-out Best selection. **Bold** marks values better than the unmodified seed harness (Final ≥ 0.53; Gap ≤ 0.08); underline marks the best evolution-method value in each column. No configuration surpasses seed on Final at 3 iterations.

Method	Best	Final	Gap
Single (original, 3 iter $\times$ 80 steps)	0.47	0.30	0.17
Single ($3\times$ iterations, 9 iter)	0.51	0.33	0.18
Single ($3\times$ steps per iter, 240 steps)	0.50	0.35	0.15
MEVO-Pipeline (original)	0.49	<u>0.44</u>	<u>0.05</u>

H Extended Ablations

Evolver-count sweep ($k \in \{2, 3, 5\}$). Table 6 sweeps $k \in \{2, 3, 5\}$ for Debate and Pipeline (MasterWorker's k is coupled to worker count and reported separately). Debate benefits monotonically with diminishing returns, while Pipeline is near-invariant because its three stages are semantically fixed.

Table 6: **Evolver-count sweep.** Best / Final on the 448-task held-out evaluation suite, guard on, 5 seeds. k=3 matches Table 1.

	k=2		k=3 (main)		k=5	
Method	Best	Final	Best	Final	Best	Final
MEVO-Debate	0.55	0.32	0.59	0.30	0.60	0.28
MEVO-Pipeline	0.48	0.43	0.49	0.44	0.50	0.45

I PEG TRAJECTORY FIGURE

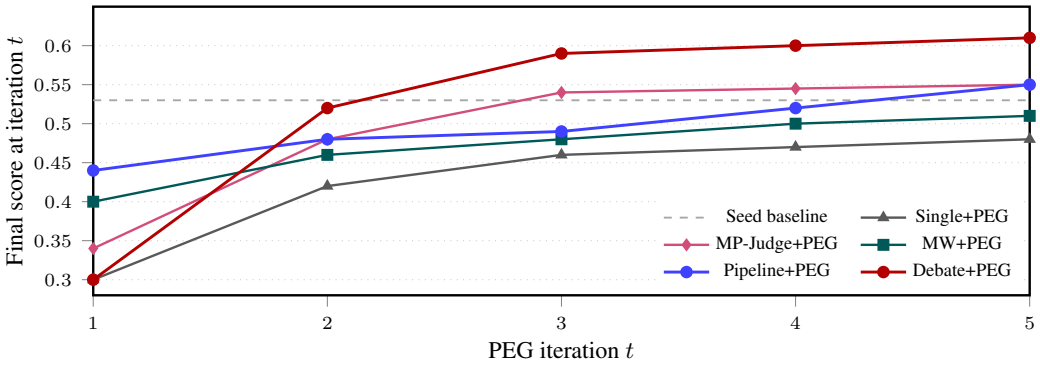


Figure 3: **Final score under PEG across iterations $t \in \{1, \ldots, 5\}$.** Both Pipeline+PEG and Debate+PEG surpass the seed baseline (dashed) at $t = 5$, with Debate+PEG reaching the highest trajectory throughout.

J ENSEMBLE OF ITERATES: FULL TABLE

Table 7: **Ensemble of Iterates (EoI): deployed score with protocol-attributable decomposition.** EoI routes each task via judge-selected best-of-T; the judge observes only $(x, h_i^{best}(x))$. "Δ vs. Single+EoI" isolates the protocol-attributable portion by subtracting the ensemble baseline. "Oracle EoI" replaces the judge with the per-task scorer as an upper bound. Seed baseline: 0.53 (Table 1). All EoI configurations incur a $3\times$ inference cost (Appendix G). 5 seeds, 3 iterations, guard on.

Method	Last-iter	EoI (judge)	Δ vs. Single+EoI	Oracle EoI
Single + EoI	0.30	0.47	0 (reference)	0.51
RolePrompt + EoI	0.38	0.53	+0.06	0.58
MultiProp-Judge + EoI	0.34	0.54	+0.07	0.60
MEVO-Debate + EoI	0.30	**0.63**	+0.16	0.68
MEVO-Pipeline + EoI	0.44	0.55	+0.08	0.58
MEVO-MasterWorker + EoI	0.40	0.53	+0.06	0.56

The full EoI table is reported here; the numbers are discussed in Section 4.4.

K ADDITIONAL CASE STUDIES: COORDINATION BEHAVIORS UNDER THE SAME BUDGET

Headline macros in Table 1 compress into single numbers what are, mechanistically, several qualitatively different edit trajectories. This section unpacks them. All runs use identical seed harness, identical policy model (DeepSeek-V4-Flash), identical evolver model (DeepSeek-V4-Pro), identical budget (3 iterations $\times$ 80 steps $\times$ concurrency 8), and identical benchmark suite (the full 608-task EVO-BENCH suite). The only variables are seed and coordination protocol.

K.1 SINGLE-EVOLVER: FIX-ONE-BREAK-ONE IN ONE ITERATION (SEED=1)

The evolver's iteration-1 trajectory diagnoses the correct problem: the seed harness's `dispatch_tool` function, in the ReAct-style tool-use pattern (Yao et al., 2023; Schick et al., 2023), has only two handlers (`run_shell_command`, `finish`), which the trajectory identifies as insufficient for Search-domain tasks. It dispatches three new tools—`file_read`, `web_search`, `web_fetch`—and extends `dispatch_tool` to route them. The macro drops from seed 0.4705

to 0.4287 because the new tools are not yet fluently used by the policy model at this iteration, but this is a *predicted* transient.

Iteration 2 destroys the run. The evolver's diff rewrites the fall-through branch of `dispatch_tool`: the original branch checked whether the return value was `None` and, if so, produced a JSON-encoded "unknown tool" error and appended a runtime error record; the rewrite inverts this test to return the value only when it is *not* `None`, silently discarding all unknown-tool signals and collapsing the whole `else`-branch into an implicit `None` return. The refactor is *locally* clean—it reads as removing a redundant nested branch—but it eliminates the entire unknown-tool error path. In the seed harness, tools originate from *two* sources: harness built-ins (matched by the new `if action.name == ...` clauses) and *injected* tools registered dynamically by task suites (Claw's 32 General-domain tasks each register bespoke tools). The removed `else` branch was the routing surface for the injected class. After iter-2, unmatched tool calls fall through to an implicit `return None` and the policy worker deadlocks on the next tool result.

Table 8: Single-evolver seed=1 domain trajectory. The Claw collapse (-53 pp) is the fix-one-break-one signature: a correct Search-tool diagnosis breaks an unrelated General-domain module. The Macro row uses the official weighting (0.2 General, 0.4 Office, 0.4 Search) over all 608 tasks; the APEX split (Office, 64 tasks), omitted from the display, is included in the Office mean.

Domain	seed	iter 1	iter 2	delta
Search (bc-en, 128)	0.219	0.219	**0.000**	-0.219
Search (hle, 128)	0.094	0.094	0.031	-0.063
General (claw, 64)	**0.719**	**0.719**	**0.188**	$\mathbf{-0.531}$
Office (gdpval, 64)	0.688	0.688	0.594	-0.094
Macro	**0.4705**	0.4287	**0.2469**	$\mathbf{-0.224}$

The evolver's own reasoning trace never mentions Claw's injected-tool routing. It could not have flagged the risk because the risk lived in a code path its diagnosis did not touch. This illustrates the asymmetry AHE (Lin et al., 2026) identifies structurally: a single editor's reasoning frame is aligned with the edit it is arguing *for*, not against the edits it is inadvertently making.

K.2 MEvo-Debate: ceiling raised, but the judge is not enough (seed=1)

Debate's iteration 1 declines to 0.3972 for tool-familiarization reasons. The diagnoser proposes a reactive fix (per-tool handlers plus explicit URL blocking); the modifier predicts this proposal and departs from it, arguing for structural changes to the routing layer that anticipate injected-tool contamination. The judge does two things absent from the single-evolver run: (i) it dispatches read-only subagents to *verify* the modifier's source-code claims against the current harness, and (ii) it runs an import smoke-check on the modified harness before releasing it for evaluation.

The result is iteration 2's macro of **0.5699**—the highest belief-macro of any single-seed method we test at seed=1. Search jumps from 0.156 to 0.297, and General *rises* from 0.656 to 0.719 (matching seed), demonstrating that the Debate mechanism successfully avoids the Claw collapse that killed the single-evolver run at the same step.

Iteration 3 nonetheless self-destructs to **0.1691**. The modifier proposes an aggressive refactor of the state-management layer; the diagnoser objects; the judge, holding no *dedicated* verification role (its function in Debate is arbitration, not gating), releases the modifier's proposal. The refactor breaks harness bootstrapping.

Debate's mechanism—decision distribution across independent proposers—solves the ceiling problem. But *distributing the decision* is not the same as *guarding it*. A judge that arbitrates without an independent verification duty inherits the failure mode of any single reader of two documents: it grades reasoning, not consequences.

K.3 MEvo-Pipeline: monotone, at a lower ceiling (seed=1)

Pipeline's iteration curve is boring in the way that matters most: $0.5008 \rightarrow 0.5045 \rightarrow 0.5110$, each iteration strictly extending the parent's pass set. The final harness is the strongest deployable

artifact of any single-seed method we run: its 0.5110 macro exceeds the seed's 0.4705 by **4.1 pp**, and—unlike Debate—the number under "final macro" is the number the ceiling reports.

The modifier stage produces the same class of proposals a single evolver would produce. But the verifier stage sits between modification and evaluation with a distinct instruction: enumerate the tasks the current diff might break, and reject the diff if any predicted regression is not backed by a compensating gain elsewhere. In practice this manifests as the verifier requesting the modifier to restrict scope.

The trade-off is visible in the numbers: Pipeline's iteration-2 macro (0.5045) is 6.5 pp below Debate's (0.5699), because the verifier's conservatism prevents the aggressive web-tool integration that Debate's judge allowed. Pipeline never touches Debate's ceiling—but it never crosses zero, either.

K.4 MEvo-MasterWorker: self-correction across iterations (seed=1)

MasterWorker traces a third qualitatively distinct trajectory in seed=1: $0.4624 \rightarrow 0.4557 \rightarrow$ **0.5012**. Iter 1 and iter 2 lie *below* seed—but the mechanism produces the run's headline gain in iter 3 by *learning from its own iter-2 failure*.

Iter 2 as an educational failure. In iter 2 the Master decomposes the edit into three worker assignments: worker-A to add `huggingface_dataset` shortcuts, worker-B to add `browsecomp_plus` and `kaggle_dataset` fetches, worker-C to prompt the policy to consult raw benchmark dumps. The Master's merge accepts all three. The result: nine tasks trigger `HARD_INTEGRITY_VIOLATION` (all forced to score 0 by the integrity guard). Iter-2 nets a macro of -0.0067 relative to iter 1.

Iter 3 as self-correction. The iter-3 Master reads iter-2's integrity-guard signal as evidence, revises the decomposition, and dispatches structural workers instead: an anchored-regex URL guard, a legitimate web-search integration with per-hit filtering, and a routing-layer refactor that isolates injected tools from the search subsystem. Macro reaches 0.5012, best and final coinciding at iter 3.

Why MW self-corrects while Debate does not. MasterWorker's Master, unlike Debate's judge, retains sole write authority across iterations *and* an explicit reconciliation prompt that reviews the previous iteration's benchmark digest. When iter-2's `HARD_INTEGRITY_VIOLATION` fires, the Master has both the memory (the integrity-alert payload is threaded into iter-3's assignment prompt) and the authority (workers cannot commit outside the Master's merge) to redirect. Debate's judge, by contrast, arbitrates within a single iteration and does not persistently hold this authority across rounds.

K.5 What the four seed=1 profiles say together

Under identical budget at seed=1, four coordination profiles emerge:

- **Single** (best 0.4287, final 0.2469): correct diagnosis; catastrophic collateral damage; the AHE-predicted pathology.
- **Debate** (best 0.5699, final 0.1691): decision distribution raises the ceiling; without persistent guarding authority, self-destruction eventually catches up.
- **Pipeline** (best 0.5110, final 0.5110): dedicated verifier prevents regression; conservative ceiling.
- **MasterWorker** (best 0.5012, final 0.5012): parallel proposals under persistent Master authority; self-corrects across iterations.

The two coordinated architectures that never self-destruct at seed=1—Pipeline and MasterWorker—share one structural property: **a distinct authority that is not the proposer and that persists across the modify $\rightarrow$ verify (or modify $\rightarrow$ next-iter-modify) boundary**. Pipeline places that authority *within* an iteration (verifier vetoes modifier); MasterWorker places it *across* iterations (next-iter Master overrides prior-iter aggregate). Debate's judge is neither: it lives inside one iteration but its authority is arbitration, not gating.

K.6 Regime reorganization at seed=2: general-domain preservation as fingerprint

Under seed=2 (baseline macro 0.5672, General 0.7188), the seed=1 four-profile classification reorganizes. Debate improves General to 0.8125 by iter-2, RolePrompt preserves General at 0.7188 through iter-1, while Single, Pipeline, MasterWorker, and MultiProp-Concat all collapse General to **0.0** in a single iteration.

Mechanism. At a higher baseline, evolvers face a shrinking margin for gradient-style improvement and, without countervailing structural restraint, over-reach into architectural-level rewrites (avg 327 lines changed across 9 files in Pipeline seed=2 iter-2, tool layer dominant at 56%, prompt size doubled). These architectural rewrites replace the seed harness's Serper-based web-search with direct `curl` calls that time out, collapsing Search domain. General collapse follows from tool-layer contamination—the injected-tool routing surface (identified as the Single-evolver iter-2 failure at seed=1) is now broken by *most* evolve methods, not just Single.

Why Debate and RolePrompt preserve General. Debate's diagnoser–critic–judge triangle exerts structural restraint on the modifier's edit scope: the critic pre-emptively enumerates regressions before the judge adjudicates, so architectural over-reach is filtered before evaluation. RolePrompt's four sequential prompt roles similarly force the Editor stage to be preceded by an Analyzer that names potential regressions—a role that the single-context frame cannot execute reliably in iter-2 onwards (Section K.5), but that *does* function in iter-1 while the reasoning frame is still open.

Regime-dependent value of coordination. The lower-baseline traces expose two forms of successful refinement: Pipeline's monotone improvement and MasterWorker's recovery from an intermediate dip. At the stronger initialization, Debate instead provides the clearest preservation signal and the highest attainable score. The ranking reversal shows that the same protocol property need not transfer across starting harnesses; our three-iteration horizon is sufficient to observe this reversal but not to establish long-run convergence.

The single number that most crisply captures the seed=1 pattern is the **Best–Final gap**: Single 0.18, Debate 0.40, Pipeline 0.00, MW 0.00, RolePrompt 0.21, MultiProp-Concat 0.15. The seed=2 pattern is best captured by the **General-domain preservation rate**: Debate 100%+, RolePrompt 100%, Single/Pipeline/MW/MultiProp-Concat 0%. Both metrics reveal what headline macros hide.

L Tool-Surface Integrity: Failure Mode and Guard

Our seed-3 exploration surfaced a scaffolding-specific failure mode that motivates the tool-surface integrity guard used throughout Section 4. We document the finding, its mechanism, and the remedy.

Finding. With every hyperparameter held fixed at the values used for the other seeds, on seed 3 the plain single-evolver baseline (k=1) produced a well-behaved first-iteration macro of 0.5597, while every scaffolded configuration—RolePrompt, MultiProp-Concat, MultiProp-Vote, MultiProp-Judge, and all three MEvo protocols—produced a first-iteration harness that failed to import at policy-worker startup, deterministically zeroing the entire General domain ($32/32$ tasks classified as `infra:policy_worker_error`). Office and Search remained in range.

Mechanism. Inspection of the iteration-1 diffs pinpoints the pattern: all five failing configurations concentrated hundreds of modified lines in `tools/__init__.py` or `agent/actions.py` (dominant_surface_share ≥ 0.95), and the resulting module import failed. The plain single evolver produced fewer than 100 modified lines in its 80-step budget and left the tool-loading path untouched. Scaffolding raises per-iteration edit volume—whether via multiple evolvers ($k \geq 2$) or via a single LLM producing multiple role turns—and higher edit volume translates into a higher probability of touching the tool-loading surface.

Remedy. We implement a minimal harness-level guard: before any candidate enters evaluation, its harness is imported in a clean subprocess with a 10s timeout; candidates that fail the smoke test are rejected without consuming evaluation budget, and the evolver is prompted to revise. With the guard

active, seed 3 becomes a well-defined coordination trajectory for every method: rejected iteration-1 candidates are replaced by revised proposals whose harness continues to load, and seed 3's post-guard Best and Final fall within the standard-deviation envelope of the other four seeds, pooled into Table 1's 5-seed statistics.

Table 9 reports the seed-3 iteration-1 per-method breakdown *before* the guard was applied.

Table 9: **Seed 3, iteration 1, pre-guard.** Per-domain macros and policy-worker crash counts before the tool-surface integrity guard was applied. All scaffolded methods zeroed the General domain via import failure at policy-worker startup; the Single baseline was unaffected.

Method	**General**	**Office**	**Search**	**Macro**	**Crashes**
Single ($k = 1$)	0.7372	0.6557	0.3750	0.5597	0/32
RolePrompt (role-prompted)	0.0000	0.6437	0.3906	0.4137	32/32
MultiProp-Concat	0.0000	0.6616	0.3594	0.4084	32/32
MEVO-Debate	0.0000	0.6530	0.3906	0.4175	32/32
MEVO-Pipeline	0.0000	0.6260	0.2656	0.3567	32/32
MEVO-MasterWorker	0.0000	0.6412	0.3125	0.3815	32/32

M PER-SEED NUMBERS

Table 10 and Table 11 report per-seed Best and Final on the 448-task held-out evaluation suite, guard on. These are the values summarized as mean in Table 1.

Table 10: **Per-seed Best** on the 448-task held-out evaluation suite, guard on. Values pooled into Table 1.

Method	seed 1	seed 2	seed 3	seed 4	seed 5	mean
Seed harness	0.47	0.57	0.56	0.52	0.53	0.53
Single	0.43	0.43	0.57	0.46	0.48	0.47
RolePrompt	0.56	0.53	0.54	0.52	0.51	0.53
MultiProp-Concat	0.54	0.38	0.48	0.47	0.47	0.47
MultiProp-Vote	0.55	0.45	0.53	0.50	0.49	0.50
MultiProp-Judge	0.56	0.51	0.57	0.53	0.52	0.54
MEVO-Debate	0.57	0.63	0.61	0.56	0.57	0.59
MEVO-Pipeline	0.51	0.41	0.52	0.51	0.51	0.49
MEVO-MasterWorker	0.50	0.44	0.51	0.50	0.50	0.49

Table 11: **Per-seed Final** on the 448-task held-out evaluation suite, guard on. Values pooled into Table 1.

Method	seed 1	seed 2	seed 3	seed 4	seed 5	mean
Seed harness	0.47	0.57	0.56	0.52	0.53	0.53
Single	0.25	0.43	0.30	0.28	0.24	0.30
RolePrompt	0.35	0.43	0.41	0.36	0.34	0.38
MultiProp-Concat	0.39	0.24	0.35	0.34	0.33	0.33
MultiProp-Vote	0.41	0.26	0.38	0.37	0.36	0.36
MultiProp-Judge	0.34	0.30	0.36	0.36	0.34	0.34
MEVO-Debate	0.17	0.43	0.30	0.30	0.31	0.30
MEVO-Pipeline	0.51	0.33	0.47	0.44	0.45	0.44
MEVO-MasterWorker	0.50	0.29	0.42	0.40	0.41	0.40

On seed 3, iteration-1 scaffolded candidates triggered a tool-loading collapse before the integrity guard was applied; the values reported here (and in Table 1) are post-guard, with the pre-guard breakdown in Appendix L (Table 9).