# WRIGHT–GÖDEL MACHINE: MITIGATING HOMOGENIZATION IN SELF-IMPROVING AGENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Recent self-improving agents retain previously discovered implementations in an archive so that improvement can proceed from multiple starting points. However, we find that parent selection can repeatedly favor closely related agents, leaving alternative designs with few opportunities to produce descendants. We refer to this phenomenon as *genetic homogenization*, which can cause the search to miss useful directions for improvement. We characterize it through the effective number of reproductive lineages and parent diversity, and prove that Thompson-sampling parent selection concentrates reproduction on a single lineage in the standard stochastic bandit model, even when every lineage remains in the archive. Inspired by Wright's shifting-balance theory, we propose the Wright–Gödel Machine (WGM). WGM selects parents with a rule based on the Exponential-weight algorithm for Exploration and Exploitation (EXP3), which combines clade statistics with a uniform exploration floor, and uses best arm identification (BAI) to allocate additional evaluations and select the final agent. We prove that WGM gives every lineage a polynomially growing expected number of expansions under any reward sequence. On SWE-bench Verified and Polyglot, WGM discovers agents that outperform state-of-the-art methods under matched evaluation budgets, and these agents transfer to held-out coding tasks and other backbone models. Our code is available at `https://anonymous.4open.science/r/WGM-AD5B`.

## 1 INTRODUCTION

As modern systems grow more complex, the tasks they are used for become harder and more demanding. Agents that serve as long-term assistants to humans are therefore expected to solve harder problems and adapt to a wider range of tasks (Mialon et al., 2024; Zheng et al., 2025). Recursive self-improvement (RSI) offers a way to meet these demands by allowing self-improving agents (SIAs) (Robeyns et al., 2025; Yin et al., 2025) to learn from experience and improve themselves. Usually, SIAs iteratively revise themselves based on feedback from evaluations (Zhang et al., 2024), gradually evolving into stronger agents in specific task. The capabilities they gain can support subsequent improvements (Wang et al., 2023), enabling continued adaptation with less reliance on human intervention. However, naive SIAs (Ou et al., 2025; Zelikman et al., 2023) improve using only the latest evaluation, without reference to earlier designs. This limits their exploration of alternative agent designs and can stall self-improvement before stronger agents are found.

To address this limitation, recent work (Zhang et al., 2026; Wang et al., 2025a) has shifted to archive-based SIAs. An archive-based SIA maintains a collection of agent designs together with their evaluation records. At each expansion, an archived agent is selected as the parent and modified to produce a child, which is then evaluated on coding tasks. Children can be added to the archive alongside their parents even when they do not outperform them (Zhang et al., 2026), so earlier agents remain available for subsequent expansions. Parent–child links form a search tree that records how agent designs develop across generations, and the branches of this tree defined by ancestry, are called *lineages*.

However, archive-based SIAs face a critical bottleneck: **Genetic Homogenization**, in which reproduction concentrates in a few lineages and reduces the diversity of design features inherited by descendants. Keeping an agent in the archive does not guarantee that it will produce descendants. Even as the archive grows, parent selection keeps favoring the few lineages that perform better in current evaluations, leaving other retained lineages with few opportunities to reproduce. Therefore,



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.

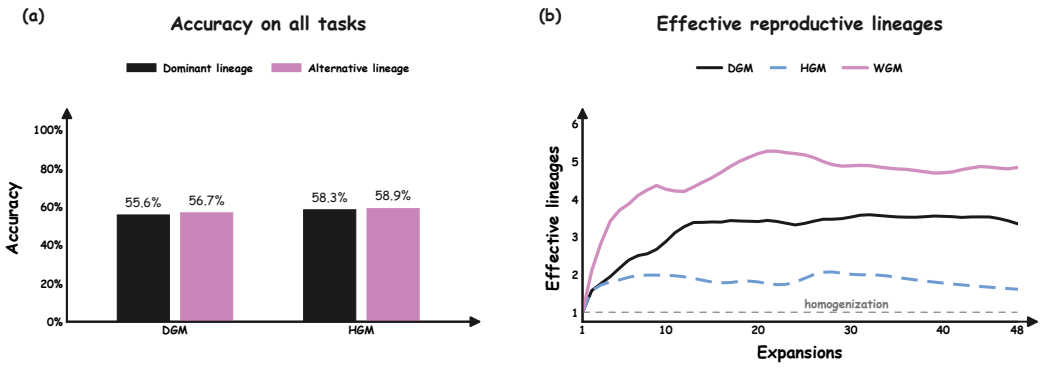



Figure 1: **Reproductive-lineage diversity during self-improvement.** (a) Accuracy of the best agent in the dominant lineage and in the alternative lineages. The alternative lineages contain agents that exceed the best agent of the dominant lineage. (b) Effective number of reproductive lineages $L(T)$ (Eq. 3), computed from the parent-selection histories of the experiments with lineages. $L(T) = 1$ means that all parent selections come from a single lineage. DGM and HGM concentrate reproduction on a few lineages, whereas WGM keeps reproduction spread across more lineages. (See experimental setup in Appendix D.3).

descendants keep inheriting similar designs, which limits the discovery of new directions for improvement. Figure 1 shows the phenomenon at a glance: under the same evaluation budget, in DGM and HGM, the best agent from the alternative lineages achieves accuracy higher than the best agent of the dominant lineage (Figure 1*(a)*), yet these lineages receive few expansion opportunities. Moreover, the effective number of reproductive lineages stays low and approaches one.(Figure 1*(b)*) Such concentration increases the risk of premature convergence (Lehman & Stanley, 2011), as most of the search budget is spent expanding a few lineages while other designs remain underexplored.

To address this challenge, we introduce measures of *active reproductive diversity*, which capture differences among selected parents and how reproduction is spread across lineages. Drawing on population diversity (Rao, 1982), we measure parent diversity as the average difference between selected parents. Inspired by effective species numbers (Hill, 1973), we use the effective number of reproductive lineages to measure how widely parent selections are spread across lineages. Based on these measures, we formalize genetic homogenization and show that selecting parents by current evaluation results concentrates reproduction on a single lineage.

Building on this analysis, we propose the **Wright–Gödel Machine (WGM)**, inspired by Sewall Wright's shifting-balance theory (Wright, 1932), which emphasizes how differences among subpopulations preserve opportunities for further evolutionary improvement. Following this perspective, WGM formulates parent selection as a bandit problem. It uses the Exponential-weight algorithm for Exploration and Exploitation (EXP3) (Auer et al., 2002) to update selection probabilities from initial child evaluations, adjusts them with clade statistics, and reserves a fixed share for uniform exploration. Thus, parent selection reflects the performance of an agent and its descendants while preserving expansion opportunities for alternative lineages. Across the archive, WGM uses best arm identification (BAI) (Kaufmann & Koolen, 2017) to guide additional evaluation and final agent selection. These additional evaluations also update clade statistics and thus make parent selection more reliable under a limited evaluation budget. We show in Theorem 1 that WGM avoids homogenization whenever the archive retains two lineages at a constant fraction.

We evaluate WGM on SWE-bench Verified (Jimenez et al., 2024; Chowdhury et al., 2024) and Polyglot (Gauthier, 2024). Under the same evaluation budgets, WGM discovers agents that outperform DGM (Zhang et al., 2026) and HGM (Wang et al., 2025a). WGM achieves accuracies of 65.0% and 54.7%, which exceed HGM by 3.3 and 4.5 percentage points. The discovered agents also retain their performance advantage on held-out coding tasks and with different backbone models.

To summarize, our contributions are as follows:

- To our knowledge, we are the first to identify and formalize *Genetic Homogenization* in archive-based self-improving agents. We measure it with the effective number of reproductive lineages and parent diversity, and provide a theoretical analysis of its mechanism.
- We introduce **Wright–Gödel Machine (WGM)**, a self-improvement framework which uses EXP3-based rule over clade statistics plus a uniform exploration floor, and uses BAI for additional evaluation and final agent selection to mitigate homogenization. We prove that it avoids homogenization whenever the archive stays balanced, and that it guarantees every lineage a reward-independent, growing expected number of expansions.
- We empirically validate WGM on SWE-bench Verified and Polyglot. The results show that WGM achieves higher accuracy than state-of-the-art methods on both benchmarks under the same evaluation budget, and its discovered agents keep this advantage on held-out tasks and with other backbone models.

## 2 Preliminary and Challenges

### 2.1 Archive-Based Self-Improvement

Archive-based SIAs maintain a collection of agent designs together with their evaluation records. Let $\mathcal{A}_t$ denote the archive before expansion $t$, with $N_t = |\mathcal{A}_t|$. The objective is to discover a high-performing agent within a fixed budget.

At expansion $t$, an archived agent $a_t \in \mathcal{A}_t$ is selected as parent and modified to produce a child $b_t$:

$$a_t \sim \pi_t, \qquad b_t = \Phi(a_t; e_t). \tag{1}$$

Here, $\pi_t$ denotes the selection distribution over $\mathcal{A}_t$, and $\Phi$ is the self-modification operator conditioned on diagnostic evidence $e_t$, such as execution logs from previous tasks.

The child is then evaluated on coding tasks. Let $s(a, x) \in \{0, 1\}$ indicate whether agent $a$ solves task $x$. The mean task score of $a$ under a fixed search-task distribution $\mathcal{D}$ is

$$\mu_a = \mathbb{E}_{x\sim\mathcal{D}}[s(a, x)]. \tag{2}$$

We denote the numbers of successful and failed task evaluations of agent $a$ before expansion $t$ by $S_{a,t}$ and $F_{a,t}$, respectively. If the child is admitted, the archive becomes $\mathcal{A}_{t+1} = \mathcal{A}_t \cup \{b_t\}$, and its evaluation records are stored alongside those of existing agents. Admission does not require the child to outperform its parent, and earlier agents remain available for subsequent expansions. Parent–child links form a search tree whose branches, defined by ancestry, are called lineages (formalized in Definition 1). A lineage can contain multiple agents with different designs.

Existing methods select parents based on the evaluation results of the agents themselves or of their descendants (Zhang et al., 2026; Wang et al., 2025a). The resulting selection distribution $\pi_t$ determines how expansion opportunities are allocated across lineages.

### 2.2 Challenges in Archive-Based Self-Improvement

Archive-based SIAs face a critical bottleneck: **Genetic Homogenization**, in which reproduction concentrates in a few lineages and their descendants inherit increasingly similar design features. Keeping an agent in the archive does not guarantee that it will produce descendants. Parent selection may repeatedly favor lineages that perform better in current evaluations, leaving other retained lineages with few opportunities to reproduce. The archive can therefore keep growing while the search repeatedly extends similar designs.

This matters under a finite search budget. An agent with a low current evaluation score may still produce stronger descendants after several modifications, but if its lineage is rarely selected, those modifications are never attempted. Concentrating the budget on a few lineages therefore increases the risk of premature convergence (Lehman & Stanley, 2011): other designs remain insufficiently explored, and the search is less likely to discover new directions for improvement.

Therefore, the core challenge is to maintain *active reproductive diversity* during self-improvement. We consider both design differences among selected parents and how widely reproduction is spread

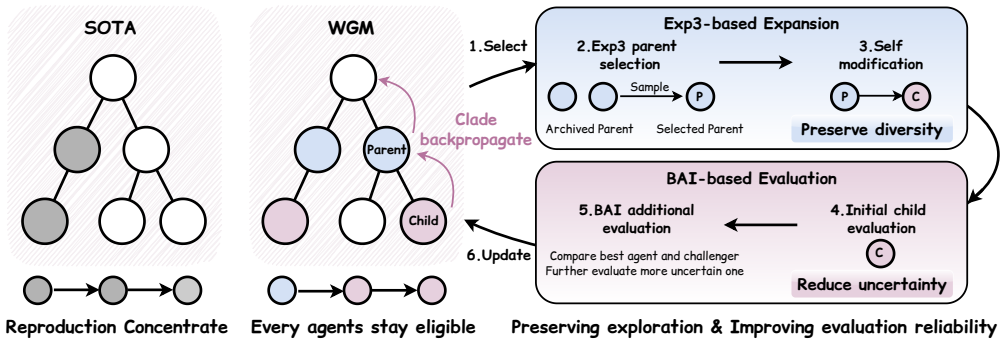



Figure 2: Overview of WGM. EXP3 selects parents using weights adjusted by clade statistics, while a uniform exploration floor keeps every archived agent eligible. BAI allocates additional evaluations across the archive and selects the final agent. Initial evaluations of new children update their parents' EXP3 weights, and both initial and additional evaluations update clade statistics for subsequent parent selection.

across lineages. These aspects are distinct: agents within the same lineage may have different designs, while agents from different lineages may inherit similar features. Section 3.2 formalizes these quantities and examines when parent selection leads to homogenization.

# 3 METHODOLOGY

In this section, we present the Wright–Gödel Machine (WGM), an archive-based self-improvement framework designed to mitigate genetic homogenization. We first restate the problem (Section 3.1), then formalize homogenization and show how parent selection leads to it (Section 3.2). We then introduce WGM's parent selection based on the Exponential-weight algorithm for Exploration and Exploitation (EXP3) with an exploration floor, together with its reproduction guarantee (Section 3.3), followed by best arm identification (BAI) for additional evaluation and final-agent selection (Section 3.4). Next, we present the overall WGM framework that integrates both components.

## 3.1 PROBLEM REFORMULATION

Archive-based self-improvement aims to discover a high-performing agent within a fixed evaluation budget. Existing methods select parents based on the evaluation results of agents or their descendants. However, keeping an agent in the archive does not guarantee that it will produce descendants. Even as the archive grows, parent selection keeps favoring a few lineages, so descendants inherit similar designs and other lineages lose their opportunities to reproduce. This will increase the risk of premature convergence. We formalize and analyze this phenomenon in Section 3.2.

## 3.2 HOMOGENIZATION IN SELF-IMPROVEMENT

To characterize genetic homogenization, we measure active reproductive diversity through design differences among selected parents and the shares of parent selections across lineages.

**Definition 1** (Lineage)**.** Let $a_0$ be the initial agent. For an archived agent $a$, its founder $\phi(a)$ is the unique child of $a_0$ on the path from $a_0$ to $a$ in the search tree, and $\phi(a_0) = a_0$. The lineage of $a$ is identified with its founder, and lineage $c$ is the set $\{a : \phi(a) = c\}$.

Let $n_c(T)$ count the parent selections from lineage $c$ among the first $T$ expansions, and $q_c(T) = n_c(T)/T$. When selecting two parents from these expansions $T$, the probability that they belong to the same lineage is $\sum_{c=1}^{C} q_c(T)^2$. Inspired by effective species numbers (Hill, 1973), we define the *effective number of reproductive lineages* as the inverse of this probability:

$$L(T) = \frac{1}{\sum_c q_c(T)^2}. \tag{3}$$

$L(T)$ expresses the observed shares as an equivalent number of equally selected lineages and equals one when all selected parents come from a single lineage. Adding agents to the archive without selecting them as parents does not change $L(T)$.

However, parents within a lineage can differ, and parents from different lineages can converge to similar designs. Therefore, we also measure design differences. Let $d(a, b) \in [0, 1]$ be a symmetric difference between agent designs. Following the use of average differences to measure population diversity (Rao, 1982), we define *parent diversity* as

$$D(T) = \frac{1}{T^2} \sum_{s=1}^{T} \sum_{t=1}^{T} d(a_s, a_t). \tag{4}$$

$D(T)$ is the expected difference between two parents. It lies in $[0, 1]$ and equals zero when every pair of selected parents has no difference under $d$.

**Definition 2** (Genetic homogenization)**.** A self-improvement process undergoes genetic homogenization if $L(T) \xrightarrow{p} 1$ as $T \to \infty$, or equivalently if $\max_c q_c(T) \xrightarrow{p} 1$.

**Thompson sampling homogenizes.** Recent archive-based methods such as HGM select parents by Thompson sampling on clade statistics. To examine this selection principle, we analyze Beta–Bernoulli Thompson sampling over lineages in the standard stochastic bandit model.

**Assumption 1** (Stationary lineage rewards)**.** Each selection of lineage $c$ yields an independent Bernoulli reward with mean $\theta_c \in (0, 1)$, and there is a unique best lineage $c^* = \arg\max_c \theta_c$.

**Proposition 1** (Thompson sampling homogenizes)**.** *Under Assumption 1, Beta–Bernoulli Thompson sampling over lineages satisfies $\mathbb{E}[n_c(T)] = O(\log T)$ for every $c \neq c^*$, where $c^*$ is the best lineage. Consequently, $\mathbb{E}\big[1 - q_{c^*}(T)\big] = O\Big(\frac{\log T}{T}\Big)$, $q_{c^*}(T) \xrightarrow{p} 1$, $L(T) \xrightarrow{p} 1$.*

Proposition 1 (proof in Appendix B.1) shows that Thompson sampling, which is designed to maximize cumulative reward, gives non-leading lineage only $O(\log T)$ expected expansions. Since the total number of expansions grows linearly in $T$, the share of reproduction received by these lineages is $O(\log T/T)$ and vanishes as the search continues. However, in self-improvement, a lineage with low current score may still produce stronger descendants after several modifications. This motivates an allocation that does not vanish with a lineage's current reward.

### 3.3 EXP3-BASED PARENT SELECTION

As discussed in Section 3.2, keeping alternative lineages in the archive does not ensure that they receive expansion opportunities. Therefore, we use an EXP3-based rule (Auer et al., 2002) with clade statistics to guide parent selection with evaluation feedback while maintaining a fixed minimum selection probability over the archived agents. This allows lineages with lower observed rewards to keep producing descendants.

The archive starts as a tree containing only the initial agent, and each admitted child becomes a new node connected to its parent. Let $\mathcal{C}_t(a)$ contain $a$ and all its archived descendants. We aggregate their evaluation records into clade statistics

$$S^C_{a,t} = \sum_{v \in \mathcal{C}_t(a)} S_{v,t}, \qquad F^C_{a,t} = \sum_{v \in \mathcal{C}_t(a)} F_{v,t}. \tag{5}$$

We define the smoothed clade success rate as

$$\rho_{a,t} = \frac{S^C_{a,t} + 1}{S^C_{a,t} + F^C_{a,t} + 2}. \tag{6}$$

The pseudocounts give an unevaluated clade a score of $1/2$ and ensure that $\rho_{a,t} > 0$.

We maintain a weight $w_{a,t} > 0$ for each agent $a$, initialized to one when $a$ enters the archive, and normalize the products $w_{a,t}\rho_{a,t}$ to obtain

$$\widetilde{\pi}_{a,t} = \frac{w_{a,t}\rho_{a,t}}{\sum_{u \in \mathcal{A}_t} w_{u,t}\rho_{u,t}}. \tag{7}$$

Larger values of $w_{a,t}\rho_{a,t}$ give higher selection probabilities. We reserve a fixed exploration share $\gamma \in (0,1)$ by choosing uniformly from $\mathcal{A}_t$ with probability $\gamma$. With remaining probability $1-\gamma$, we sample according to $\widetilde{\pi}_t$. Uniform selection assigns probability $1/N_t$ to each agent. The resulting parent selection distribution is

$$\pi_{a,t} = (1-\gamma)\widetilde{\pi}_{a,t} + \frac{\gamma}{N_t} = (1-\gamma)\frac{w_{a,t}\rho_{a,t}}{\sum_{u\in\mathcal{A}_t} w_{u,t}\rho_{u,t}} + \frac{\gamma}{N_t}. \tag{8}$$

At expansion $t$, we draw the parent $a_t \sim \pi_t$. Thus, expansion feedback guides selection while every archived agent retains a selection probability of at least $\gamma/N_t$.

After selecting $a_t$, we modify it to produce one child $b_t$ and evaluate the child on a batch of $M$ search tasks $\mathcal{X}_t = (x_{t,1}, \ldots, x_{t,M})$. The child's initial performance provides feedback on the outcome of expanding its parent, and we define the expansion reward

$$r_t = \frac{1}{M}\sum_{j=1}^{M} s(b_t, x_{t,j}). \tag{9}$$

An expansion that fails to produce an executable child receives zero reward. We then use this reward to update the weights, correcting for unequal selection probabilities with an importance-weighted reward. For each $a \in \mathcal{A}_t$,

$$\widehat{r}_{a,t} = \frac{r_t \mathbf{1}\{a_t = a\}}{\pi_{a,t}}, \qquad w_{a,t+1} = w_{a,t}\exp(\eta_t \widehat{r}_{a,t}), \qquad \eta_t = \frac{\gamma}{N_t}. \tag{10}$$

We prioritize agents with larger values of $w_{a,t}\rho_{a,t}$. Both initial and additional evaluations update the clade statistics of the evaluated agent and all its ancestors.

**Reproduction guarantee.** The exploration floor in Eq. 8 gives every archived agent a selection probability of at least $\gamma/N_t$. We show that this floor passes the diversity present in the archive on to the selected parents, under the following assumption.

**Assumption 2** (Archive diversity)**.** After some expansion $m$, (i) two fixed distinct lineages each account for at least a $\kappa$ fraction of the archive, and (ii) every archived agent remains, on average, at least $\delta$ distant from the archive.

**Theorem 1** (Avoid homogenization under WGM)**.** *Under Assumption 2 and the parent-selection rule in Eq. 8, as $T \to \infty$, $L(T) \not\xrightarrow{p} 1$ and $D(T) \not\xrightarrow{p} 0$.*

Theorem 1 shows that WGM does not undergo genetic homogenization under Assumption 2; see Appendix B for the proof and a detailed discussion.

### 3.4 BAI-based Evaluation and Agent Selection

Final-agent selection and EXP3 address different statistical objectives. EXP3 allocates expansion opportunities using noisy feedback from initial child evaluations, but it is not designed to identify the best agent in a growing archive. Since each new agent is initially evaluated on only a small batch of search tasks, directly returning the agent with the highest observed score can favor agents with uncertain estimates. Therefore, we use additional evaluations to reduce the uncertainty that matters most for the final selection.

Inspired by confidence-interval comparisons in BAI-MCTS (Kaufmann & Koolen, 2017), we maintain a confidence interval $[\ell_a, u_a]$ for the task success probability $\mu_a$ of each archived agent $a$, using the agent's own evaluation records. We use Wilson score intervals, which remain informative for newly created agents with few evaluations (Appendix C).

After each expansion, we compare agents in the current archive $\mathcal{A}$. We choose the agent with the highest lower confidence bound as the provisional leader $h$, and the remaining agent with the highest upper confidence bound as its challenger $k$:

$$h = \arg\max_{a\in\mathcal{A}} \ell_a, \qquad k = \arg\max_{a\in\mathcal{A}\setminus\{h\}} u_a. \tag{11}$$

Table 1: **Self-Improving Capability Comparison.** Accuracy (%) of the final best agents and average number of task evaluations per expansion during self-improvement under a budget of 800 task evaluations.

| Method | SWE-Verified-60 | | Polyglot | |
|---|---|---|---|---|
| | Acc. (%) ↑ | Evals/Exp. | Acc. (%) ↑ | Evals/Exp. |
| DGM | 58.3 | 26.7 | 47.1 | 19.5 |
| HGM | 61.7 | 8.3 | 50.2 | 6.3 |
| **WGM (Ours)** | **65.0** | 5.7 | **54.7** | 5.4 |

The leader has the strongest supported performance, whereas the challenger is the alternative that could most plausibly outperform it. We give one additional evaluation to the candidate with the wider confidence interval:

$$a_{\text{eval}} = \arg\max_{a \in \{h,k\}} (u_a - \ell_a). \tag{12}$$

To see why this comparison is useful, consider the event that all confidence intervals contain their corresponding means. For every $a \neq h$, the definition of the challenger gives

$$\mu_a - \mu_h \leq u_a - \ell_h \leq u_k - \ell_h. \tag{13}$$

Consequently, if $\ell_h \geq u_k$, then $h$ is the best agent in $\mathcal{A}$ on this event. Otherwise, $\max\{0, u_k - \ell_h\}$ quantifies the unresolved uncertainty in the current selection. Evaluating $h$ or $k$ targets one of the two intervals that determine this certificate, and choosing the wider interval prioritizes the less precisely evaluated candidate.

The additional outcome updates the evaluated agent's confidence interval and the clade statistics. During search, the updated clade success rates also affect subsequent parent selection through Eq. 8. The contribution of BAI is evaluated in Section 4.2.3.

### 3.5 Overall Framework

Based on the above analysis, WGM combines EXP3-based parent selection with BAI-based evaluation, as illustrated in Figure 2. EXP3 determines which agent to expand, while BAI determines which agent to evaluate further. WGM alternates between agent expansion and candidate evaluation.

During the expansion stage, EXP3 selects a parent from the archive using weights adjusted by clade statistics, while reserving a fixed share of probability for uniform exploration, and the parent is modified to produce a child. During the evaluation stage, the child first receives an initial evaluation, which provides the reward that updates the parent's EXP3 weight. Admitted children are retained with their evaluation records alongside their parents. BAI then identifies the current leader, the agent with the highest lower confidence bound, and its challenger, and the candidate with the wider confidence interval receives one additional task evaluation. Both initial and additional evaluations update clade statistics along the path to the root, which affects the next parent selection. WGM repeats this process while reserving part of the evaluation budget for final selection. Once expansion stops, BAI uses the reserved budget to compare archived agents and returns the current leader. Detailed procedures are provided in Algorithm 1 in Appendix C.

## 4 Evaluation

This section evaluates the performance of WGM through extensive experiments. We compare WGM with existing self-improving agent methods, examine its generalization across datasets and LLMs, and assess the contributions of EXP3 and BAI through ablation studies. All experiments are repeated three times with different random seeds, and we report the mean results. Performance is summarized in tables, where the best results are highlighted in bold.

Table 2: **Generalization on SWE-Lite with GPT-5.6 and Claude Opus 5.5.** Accuracy (%) of SWE-agent and the final agents discovered by HGM and WGM on SWE-bench Verified with GPT-5.6 and Claude Opus 5.5. We evaluate the same agents with both execution models under the Filtered and Standard settings.

| Coding Agents | SWE-Lite Filtered (%) | SWE-Lite Standard (%) |
|---|---|---|
| SWE-agent + GPT-5.6 | 50.4 | 58.7 |
| HGM's Best-belief Agent + GPT-5.6 | 51.2 | 59.3 |
| **WGM's Final Agent + GPT-5.6** | **51.9** | **60.4** |
| SWE-agent + Claude Opus 5.5 | 52.3 | 60.7 |
| HGM's Best-belief Agent + Claude Opus 5.5 | 53.0 | 61.4 |
| **WGM's Final Agent + Claude Opus 5.5** | **53.6** | **62.1** |

### 4.1 Experiment Setup

**Benchmarks.** We evaluate WGM on SWE-bench Verified and SWE-Lite (Jimenez et al., 2024) for software engineering tasks, and Polyglot (Gauthier, 2024) for coding across multiple programming languages, following HGM (Wang et al., 2025a). The main comparison and ablation experiments use SWE-Verified-60, the same 60-task subset of SWE-bench Verified used by HGM (Wang et al., 2025a), and Polyglot. For generalization, we evaluate agents discovered on the full SWE-bench Verified benchmark on SWE-Lite. We report accuracy as the percentage of tasks successfully solved.

**Baselines.** To validate our hypothesis that more effective parent selection and more reliable agent evaluation lead to better final agents, we evaluate WGM against two state-of-the-art self-improving coding agent methods: DGM and HGM, as detailed in Appendix D.1.

### 4.2 Performance Evaluation

#### 4.2.1 Self-Improving Capability Comparison.

To assess whether preserving expansion opportunities leads to more effective self-improvement, we compare WGM with HGM and DGM on SWE-Verified-60 and Polyglot. All methods start with the same initial agent and use GPT-5.5 for both self-modification and task execution. Each method receives a budget of 800 task evaluations (other hyperparameters are listed in Appendix C.1). We report the accuracy of the final agent returned by each method using its own selection procedure. Since the methods divide the same budget differently between creating new agents and evaluating existing ones, we also report the average number of task evaluations per expansion.

**Evaluation Results.** Table 1 shows that WGM achieves the highest accuracy on both benchmarks. On SWE-Verified-60, WGM reaches 65.0%, exceeding HGM and DGM by 3.3 and 6.7 percentage points, respectively. On Polyglot, WGM achieves 54.7%, improving over HGM and DGM by 4.5 and 7.6 percentage respectively. These gains show that WGM finds more capable final agents across both benchmarks under the same evaluation budget. WGM spends 5.7 and 5.4 task evaluations per expansion on the two benchmarks, creating a comparable number of agents with HGM. The larger difference lies in how these expansions are allocated: WGM spreads them across more lineages while HGM concentrates them on a few. This suggests that WGM's gain comes mainly from preserving expansion opportunities for alternative lineages rather than expanding more often.

#### 4.2.2 Generalization to Different LLMs and Datasets.

To examine whether WGM improves general coding ability rather than overfitting to SWE-Verified, we run HGM and WGM on the full SWE-bench Verified benchmark (500 tasks) with 8,000 task evaluations per method. Both methods use the same initial agent and GPT-5.5 configuration. Given the substantially higher cost, we compare WGM with HGM, the strongest baseline in our matched-budget experiments. To assess generalization across datasets and LLMs, we compare their final agents with SWE-agent on SWE-Lite using GPT-5.6 and Claude Opus 5.5 for task execution. Per-

Table 3: **Ablation Study.** Accuracy (%) under the same evaluation budget.

| Method | SWE-Verified-60 (%) | Polyglot (%) |
|---|---|---|
| HGM | 61.7 | 50.2 |
| WGM (w/o BAI) | 63.3 | 52.4 |
| WGM (w/o EXP3) | 62.2 | 51.1 |
| **WGM (Ours)** | **65.0** | **54.7** |

formance is reported in two settings: (i) Filtered (207 tasks excluding the overlap) and (ii) Standard (all 300 tasks).

**Evaluation Results.** Table 2 summarizes the results. With GPT-5.6, WGM's final agent achieves 51.9% and 60.4% in the two settings. With Claude Opus 5.5, the same WGM agent achieves 53.6% and 62.1%. WGM thus achieves the highest accuracy in every setting. Its advantage is preserved when the execution model changes, including to a model from a different family. This indicates that the improvements found by WGM are not specific to the self-improvement set or to the backbone model used during search.

#### 4.2.3 Ablation Study.

To isolate the contributions of WGM under different parent selection and evaluation strategies, we conduct ablation experiments on SWE-Verified-60 and Polyglot by comparing multiple combinations of these strategies. Since this ablation focuses on our parent selection and evaluation design, we use HGM as the reference method because it shares the same self-modification operator and clade statistics as WGM. We hypothesize that the performance of WGM primarily arises from more effective parent selection and reliable agent evaluation.

To separately examine the roles of EXP3 and BAI, we construct the following WGM variants under the same evaluation budget: (i) WGM (w/o BAI): we do not use BAI for additional evaluation or final agent selection, and use the same final-selection rule as HGM, while keeping all other components unchanged. (ii) WGM (w/o EXP3): we replace the EXP3 weights in parent selection with HGM's Thompson sampling over clade statistics, while keeping all other components unchanged. (iii) HGM: a reference baseline to assess the overall effectiveness of WGM without either component. All variants use the same settings.

**Evaluation Results.** Table 3 shows that both EXP3 and BAI contribute to final performance: removing either component lowers accuracy on both benchmarks, and the full WGM achieves the best results. These results suggest that WGM's gains do not come merely from the archive-based framework itself, but from more effective parent selection and more reliable agent evaluation.

## 5 Conclusion

In this work, we identify a limitation of archive-based self-improving agents: parent selection may repeatedly favor closely related agents even as the archive grows, leaving alternative designs with few opportunities to produce descendants. We term this phenomenon genetic homogenization, which can cause the search to miss useful directions for improvement, and characterize it through the effective number of reproductive lineages and parent diversity. We theoretically show that Thompson-sampling parent selection concentrates reproduction in one lineage even when every lineage remains in the archive, whereas WGM's exploration floor gives every lineage a reward-independent, polynomially growing expected number of expansions. Building on this analysis and drawing inspiration from Wright's shifting-balance theory, we propose the Wright–Gödel Machine (WGM), which uses EXP3 with uniform exploration for parent selection and BAI for additional evaluation and final agent selection. Experiments on SWE-bench Verified and Polyglot show that WGM discovers stronger agents than HGM and DGM under matched evaluation budgets, and the resulting agents keep their advantage on held-out tasks and with other backbone models. Together, these results suggest that preserving opportunities to expand alternative designs, combined with reliable candidate evaluation, improves the agents discovered within a fixed search budget.

## AI USE STATEMENT

AI tools were used to aid and polish the writing of this manuscript. Specifically, they were used to improve grammar, clarity, readability, and academic phrasing, as well as to refine the presentation of author-written text. All AI-assisted edits were reviewed and verified by the authors.

## REFERENCES


Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pp. 39–1. JMLR Workshop and Conference Proceedings, 2012.

Shipra Agrawal and Navin Goyal. Near-optimal regret bounds for thompson sampling. *Journal of the ACM (JACM)*, 64(5):1–24, 2017.

Jean-Yves Audibert, Sébastien Bubeck, and R´emi Munos. Best arm identification in multi-armed bandits. In *Annual Conference Computational Learning Theory*, 2010. URL https://api.semanticscholar.org/CorpusID:8195078.

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multi-armed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

Herbie Bradley, Andrew Dai, Hannah Benita Teufel, Jenny Zhang, Koen Oostermeijer, Marco Bellagente, Jeff Clune, Kenneth O. Stanley, Grégory Schott, and Joel Lehman. Quality-diversity through ai feedback. *ArXiv*, abs/2310.13032, 2023. URL https://api.semanticscholar.org/CorpusID:264405960.

Alexandra Carpentier and Michal Valko. Simple regret for infinitely many armed bandits. *ArXiv*, abs/1505.04627, 2015. URL https://api.semanticscholar.org/CorpusID:1296488.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Neural Information Processing Systems*, 2011. URL https://api.semanticscholar.org/CorpusID:6002655.

Neil Chowdhury, James Aung, Chan Jun Shern, Oliver Jaffe, Dane Sherburn, Giulio Starace, Evan Mays, Rachel Dias, Marwan Aljubeh, Mia Glaese, Carlos E. Jimenez, John Yang, Leyton Ho, Tejal Patwardhan, Kevin Liu, and Aleksander Madry. Introducing SWE-bench verified, August 2024. URL https://openai.com/index/introducing-swe-bench-verified/.

Jeff Clune. Ai-gas: Ai-generating algorithms, an alternate paradigm for producing general artificial intelligence. *arXiv preprint arXiv:1905.10985*, 2019.

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *Computers and Games*, 2006. URL https://api.semanticscholar.org/CorpusID:16724115.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. First return, then explore. *Nature*, 590(7847):580–586, 2021.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *International Conference on Machine Learning*, 2023. URL https://api.semanticscholar.org/CorpusID:263310323.

Paul Gauthier. o1 tops aider's new polyglot leaderboard, December 2024. URL https://aider.chat/2024/12/21/polyglot.html. Accessed: 2026-09-09.

Irving John Good. Speculations concerning the first ultraintelligent machine. In *Advances in computers*, volume 6, pp. 31–88. Elsevier, 1966.

Mark O. Hill. Diversity and evenness: A unifying notation and its consequences. *Ecology*, 54(2): 427–432, 1973. doi: 10.2307/1934352.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

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*, volume 2024, pp. 54107–54157, 2024.

Zohar S. Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *International Conference on Machine Learning*, 2013. URL https://api.semanticscholar.org/CorpusID:14309288.

Emilie Kaufmann and Wouter Koolen. Monte-carlo tree search by best arm identification. *Advances in Neural Information Processing Systems*, 30, 2017.

Emilie Kaufmann, Nathaniel Korda, and Rémi Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *International conference on algorithmic learning theory*, pp. 199–213. Springer, 2012.

Levente Kocsis and Csaba Szepesvari. Bandit based monte-carlo planning. In *European Conference on Machine Learning*, 2006. URL https://api.semanticscholar.org/CorpusID:15184765.

Joel Lehman and Kenneth O Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011.

Joel Lehman, Jonathan Gordon, Shawn Jain, Kamal Ndousse, Cathy Yeh, and Kenneth O Stanley. Evolution through large models. In *Handbook of evolutionary machine learning*, pp. 331–366. Springer, 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*, volume 2024, pp. 9025–9049, 2024.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Alexander Novikov, Ngân Vu, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav M. Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery. *ArXiv*, abs/2506.13131, 2025. URL https://api.semanticscholar.org/CorpusID:278658695.

Yixin Ou, Wangchunshu Zhou, Shengwei Ding, Long Li, Jialong Wu, Tiannan Wang, Jiamin Chen, Shuai Wang, Xiaohua Xu, Ningyu Zhang, et al. Symbolic learning enables self-evolving agents. *AI Open*, 2025.

Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

C Radhakrishna Rao. Diversity and dissimilarity coefficients: a unified approach. *Theoretical population biology*, 21(1):24–43, 1982.

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 JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Jürgen Schmidhuber. Evolutionary principles in self-referential learning, or on learning how to learn: The meta-meta-. hook. 1987. URL https://api.semanticscholar.org/CorpusID:264351059.

Jürgen Schmidhuber. Godel machines: Self-referential universal problem solvers making provably optimal self-improvements. *arXiv preprint cs.LO/0309048*, 2003.

Jürgen Schmidhuber, Jieyu Zhao, and Marco Wiering. Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. *Machine Learning*, 28(1): 105–130, 1997.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285–294, 1933. URL https://api.semanticscholar.org/CorpusID:120462794.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Wen-Yi Wang, Piotr Pikekos, Nan-Bo Li, Firas Laakom, Yi-Meng Chen, Mateusz Ostaszewski, Mingchen Zhuge, and Jurgen Schmidhuber. Huxley-gödel machine: Human-level coding agent development by an approximation of the optimal self-improving machine. *ArXiv*, abs/2510.21614, 2025a. URL https://api.semanticscholar.org/CorpusID:282384340.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. Openhands: An open platform for ai software developers as generalist agents. In *International Conference on Learning Representations*, volume 2025, pp. 65882–65919, 2025b.

Yizao Wang, Jean-Yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. In *Neural Information Processing Systems*, 2008. URL https://api.semanticscholar.org/CorpusID:11166721.

Edwin B Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

S. Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proceedings of the XI International Congress of Genetics*, 8:209–222, 1932.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Demystifying llm-based software engineering agents. *Proceedings of the ACM on Software Engineering*, 2(FSE):801–824, 2025.

John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652, 2024.

Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursively self-improvement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27890–27913, 2025.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (stop): Recursively self-improving code generation. *arXiv preprint arXiv:2310.02304*, 2023.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5348–5375, 2024.

Junhao Zheng, Xidi Cai, Qiuke Li, Duzhen Zhang, ZhongZhi Li, Yingying Zhang, Le Song, and Qianli Ma. Lifelongagentbench: Evaluating llm agents as lifelong learners. *arXiv preprint arXiv:2505.11942*, 2025.

# A Related Work

## A.1 Self-Improving Agents

The idea of machines that design more capable successors dates back to Good (1966). Schmidhuber (1987) introduced self-referential learning systems that generate and evaluate modified versions of themselves, and the Success-Story Algorithm (Schmidhuber et al., 1997) keeps only those self-modifications that are followed by higher long-term reward. The Gödel Machine (Schmidhuber, 2003) rewrites its own code whenever it can prove that the rewrite increases expected utility, which gives an optimal but impractical account of self-improvement.

Large language models have made self-modification practical. Coding agents such as SWE-agent (Yang et al., 2024), OpenHands (Wang et al., 2025b), and Agentless (Xia et al., 2025) let models operate in real software environments, and a growing line of work lets such agents revise their own prompts, tools, or code from evaluation feedback (Zelikman et al., 2023; Yin et al., 2025; Zhang et al., 2024; Ou et al., 2025; Robeyns et al., 2025). Most of these methods improve a single current agent, so an unproductive modification can stall later progress. Archive-based methods instead keep every discovered agent available for future expansion. ADAS (Hu et al., 2025) maintains an archive of agent designs from which a meta agent proposes new ones. DGM (Zhang et al., 2026) samples parents in proportion to a sigmoid-scaled benchmark score together with a bonus for agents with few children, and HGM (Wang et al., 2025a) replaces the agent's own score with a clade-level estimate of metaproductivity and selects parents by Thompson sampling. These methods differ in how they score a candidate parent. WGM also uses clade statistics to score candidate parents, and in addition controls how the expansion budget is distributed across lineages.

## A.2 Diversity in Evolutionary and Open-Ended Search

Loss of diversity is a long-standing concern in evolutionary computation. Novelty search (Lehman & Stanley, 2011) rewards behavioral novelty instead of the objective to avoid deceptive local optima, and quality-diversity methods such as MAP-Elites (Mouret & Clune, 2015; Pugh et al., 2016) keep the best solution in each cell of a behavior-descriptor grid. Go-Explore (Ecoffet et al., 2021) maintains an archive of stepping stones and favors returning to less-visited ones; DGM's parent selection is inspired by this design. Open-endedness research more broadly argues that a search should keep accumulating diverse stepping stones rather than converge (Clune, 2019). With language models as mutation operators, these ideas have been carried to program and prompt search (Lehman et al., 2023; Fernando et al., 2023; Bradley et al., 2023). FunSearch (Romera-Paredes et al., 2024) keeps an island-based program database and periodically reseeds its weakest islands, and AlphaEvolve (Novikov et al., 2025) evolves programs from a similar evolutionary database.

WGM draws on the same concern but addresses a different stage of the search. Quality-diversity and novelty methods control what is stored or rewarded, typically through hand-designed behavior descriptors. In archive-based self-improvement, the bottleneck is which of agents are allowed to reproduce. This is closer to Wright's shifting-balance theory (Wright, 1932), where partially isolated subpopulations keep evolving in parallel so that the population as a whole can move beyond a single peak. We measure the resulting reproductive diversity with the effective number of lineages and parent diversity, following diversity indices from ecology (Hill, 1973; Rao, 1982).

## A.3 Bandit Methods for Selection and Evaluation

Parent selection and candidate evaluation in archive-based self-improvement can both be framed as bandit problems. Stochastic bandit algorithms such as Thompson sampling (Thompson, 1933; Agrawal & Goyal, 2012; Chapelle & Li, 2011) minimize cumulative regret, and do so by pulling suboptimal arms only $O(\log T)$ times. This is the right behavior when arms are fixed and rewards are stationary, but when arms are lineages whose value depends on descendants not yet created, it is exactly the concentration that produces homogenization. EXP3 (Auer et al., 2002) was designed for nonstochastic rewards and mixes its exponential weights with uniform exploration. WGM uses this mixture as an explicit selection floor, so every archived agent keeps a guaranteed share of expansions while clade statistics still steer the rest.

For evaluation, fixed-budget best arm identification (Audibert et al., 2010; Karnin et al., 2013) targets the quality of the final choice rather than cumulative reward. Monte-Carlo tree search and UCT (Coulom, 2006; Kocsis & Szepesvari, 2006) alternate selection and expansion over a growing tree, and BAI-MCTS (Kaufmann & Koolen, 2017) compares confidence intervals to identify the best action; WGM's leader–challenger evaluation follows this comparison. Infinite-armed bandits (Wang et al., 2008; Carpentier & Valko, 2015) model the continual arrival of new arms, as in an archive that grows with each expansion, but they treat new arms as independent draws and ignore the ancestry among agents that clade statistics exploit.

# B PROOFS

We analyze the sequential form of Algorithm 1. Adaptive expansions are indexed by $t \geq t_0 = 5P + 1$; each draws its parent from Eq. 8, and the admission of $b_t$ is settled before $a_{t+1}$ is drawn. Let $\mathcal{F}_{t-1}$ denote the history before $a_t$ is drawn, so that $\mathcal{A}_t$ and $N_t$ are $\mathcal{F}_{t-1}$-measurable. Since initial expansion attempts are counted and each expansion adds at most one agent, $N_t \leq t$ (Appendix C). Asynchronous execution is treated at the end of Appendix B.4.

## B.1 PROOF OF PROPOSITION 1

Under Assumption 1, Beta–Bernoulli Thompson sampling with $\mathrm{Beta}(1,1)$ priors satisfies, for every $\epsilon > 0$ and $c \neq c^*$,

$$\mathbb{E}[n_c(T)] \leq \frac{(1+\epsilon)\ln T}{\mathrm{kl}(\theta_c, \theta_{c^*})} + O\left(\frac{1}{\epsilon^2}\right), \qquad \mathrm{kl}(p,q) = p\ln\frac{p}{q} + (1-p)\ln\frac{1-p}{1-q}, \tag{B.1}$$

(Agrawal & Goyal, 2017); see also Kaufmann et al. (2012). The $O(\epsilon^{-2})$ term hides constants that depend on $\theta$. Hence $\mathbb{E}[n_c(T)] = O(\log T)$. Since the set of lineages is fixed, $\mathbb{E}[1 - q_{c^*}(T)] = \frac{1}{T}\sum_{c \neq c^*} \mathbb{E}[n_c(T)] = O(\log T/T)$, and Markov's inequality gives $q_{c^*}(T) \xrightarrow{p} 1$. Since $\sum_c q_c^2 \geq q_{c^*}^2$, it follows that $L(T) \xrightarrow{p} 1$.

**Parent diversity.** For a lineage $c$, let $\varepsilon_c(T) = \max\{d(a_s, a_t) : s, t \leq T,\ a_s, a_t \in c\}$ be the spread among its selected parents. Pairs of parents inside $c$ contribute at most $\varepsilon_c(T)$ to Eq. 4, and all other pairs have total weight $1 - q_c(T)^2 \leq 2(1 - q_c(T))$, so

$$D(T) \ \leq\ \varepsilon_c(T) + 2\big(1 - q_c(T)\big). \tag{B.2}$$

Taking $c = c^*$ and expectations gives $\mathbb{E}[D(T)] \leq \mathbb{E}[\varepsilon_{c^*}(T)] + O(\log T/T)$. Under Thompson sampling, parent diversity therefore approaches at most the spread within the best lineage.

**Relation to HGM.** HGM scales both Beta parameters by a time-increasing factor $\lambda_t$ to sharpen the posterior late in the search, and it selects among agents using clade statistics. Proposition 1 takes $\lambda_t \equiv 1$ and selects among lineages, so it characterizes the selection principle underlying HGM. Assumption 1 is the setting that Thompson sampling is designed for; Theorem 1 and Proposition 2 do not use it.

## B.2 PROOF OF THEOREM 1

Formally, Assumption 2 states that there exist an expansion $m$ and two lineages $c_1 \neq c_2$, possibly depending on the run, such that for all $t \geq m$,

$$\frac{|\mathcal{A}_t \cap c_i|}{N_t} \geq \kappa \ \text{ for } i = 1, 2, \qquad \frac{1}{N_t}\sum_{a' \in \mathcal{A}_t} d(a, a') \geq \delta \ \text{ for every } a \in \mathcal{A}_t. \tag{B.3}$$

For every $t \geq t_0$, Eq. 8 with positive weights $w_{a,t}\rho_{a,t}$ gives

$$\Pr(a_t = a \mid \mathcal{F}_{t-1}) = \pi_{a,t} \geq \frac{\gamma}{N_t} \geq \frac{\gamma}{t} \qquad \text{for all } a \in \mathcal{A}_t. \tag{B.4}$$

No other property of the weights or clade statistics is used, so the result holds for any reward sequence, including non-stationary and descendant-dependent rewards. Condition (i) yields the bound on $L(T)$, and condition (ii) yields the bound on $D(T)$.

**Lineage shares.** Index lineages by the expansion at which they are founded; there are countably many. For each such lineage $k$, the increments $\mathbf{1}\{a_t \in k\} - \Pr(a_t \in k \mid \mathcal{F}_{t-1})$ are bounded martingale differences, so the strong law for martingale differences gives

$$\frac{1}{T}\sum_{t\le T}\Big(\mathbf{1}\{a_t \in k\} - \Pr(a_t \in k \mid \mathcal{F}_{t-1})\Big) \to 0 \qquad \text{almost surely, simultaneously for all } k. \tag{B.5}$$

Fix an outcome on which Assumption 2 holds, outside the null set of Eq. B.5. For $t \ge m$, summing Eq. B.4 over $\mathcal{A}_t \cap c_i$ gives $\Pr(a_t \in c_i \mid \mathcal{F}_{t-1}) \ge \gamma\kappa$, and hence

$$\liminf_{T\to\infty} q_{c_i}(T) \ge \gamma\kappa, \qquad i = 1, 2. \tag{B.6}$$

Since the two lineages each hold at least a $\kappa$ fraction of the archive, $\gamma\kappa \le \kappa \le 1/2$.

**Effective number of lineages.** Fix $\eta \in (0, \gamma\kappa)$ and set $\sigma = \gamma\kappa - \eta \le 1/2$. By Eq. B.6, $q_{c_1}(T), q_{c_2}(T) \ge \sigma$ for all large $T$. The map $q \mapsto \sum_c q_c^2$ is convex, so over the set $\{q : q_{c_1}, q_{c_2} \ge \sigma\}$ it is maximized at a vertex. This set is a simplex whose vertices place $\sigma$ on each of $c_1, c_2$ and the remaining $1-2\sigma$ on a single lineage ($c_1$, $c_2$, or a third one). The corresponding values are $\sigma^2+(1-\sigma)^2$ and $2\sigma^2 + (1-2\sigma)^2$, whose difference is $2\sigma(1-2\sigma) \ge 0$. Hence $\sum_c q_c(T)^2 \le 1 - 2\sigma + 2\sigma^2$, and letting $\eta \downarrow 0$ along a countable sequence gives

$$\liminf_{T\to\infty} L(T) \ge \frac{1}{1 - 2\gamma\kappa + 2\gamma^2\kappa^2} > 1, \tag{B.7}$$

where the last inequality uses $0 < \gamma\kappa \le 1/2$.

**Parent diversity.** Let $Z_t = \sum_{s<t} d(a_s, a_t) \in [0, t-1]$, so that $D(T) = \frac{2}{T^2}\sum_{t\le T} Z_t$. Every earlier parent $a_s$ with $s < t$ lies in $\mathcal{A}_t$ and is $\mathcal{F}_{t-1}$-measurable. Hence, for $t \ge m$, Eq. B.4 and condition (ii) give

$$\mathbb{E}[Z_t \mid \mathcal{F}_{t-1}] = \sum_{s<t}\sum_{a\in\mathcal{A}_t} \pi_{a,t}\, d(a_s, a) \ge \sum_{s<t} \frac{\gamma}{N_t} \sum_{a\in\mathcal{A}_t} d(a_s, a) \ge \gamma\delta\,(t-1). \tag{B.8}$$

The differences $\xi_t = Z_t - \mathbb{E}[Z_t \mid \mathcal{F}_{t-1}]$ satisfy $|\xi_t| \le t$, so $\sum_t \mathbb{E}[\xi_t^2]/t^4 < \infty$. The martingale $\sum_t \xi_t/t^2$ therefore converges almost surely, and Kronecker's lemma gives $T^{-2}\sum_{t\le T}\xi_t \to 0$. Combined with Eq. B.8,

$$\liminf_{T\to\infty} D(T) \ge \liminf_{T\to\infty} \frac{2}{T^2}\sum_{t=m}^{T} \gamma\delta\,(t-1) = \gamma\delta. \tag{B.9}$$

**Conclusion.** Eqs. B.7 and B.9 hold almost surely, so in particular $L(T) \not\xrightarrow{p} 1$ and $D(T) \not\xrightarrow{p} 0$.

### B.3 Discussion of Assumption 2

**A condition on the archive is necessary.** If the archive holds a single lineage, then $L(T) = 1$ under every selection rule; if all archived agents are identical in the properties measured by $d$, then $D(T) = 0$ under every selection rule. No non-homogenization guarantee can therefore hold for all archives.

**The condition is mild and checkable.** It requires only two lineages at a constant fraction, not balance across all lineages, and it places no condition on the rewards or on the selection rule. Part (i) alone yields the bound on $L(T)$, and part (ii) alone the bound on $D(T)$. Both parts refer to the archive only, so they can be computed from the archive log with the procedure in Appendix D.4.

**It separates WGM from Thompson sampling.** Under Thompson sampling, a lineage's selection probability depends only on its rewards, so reproduction concentrates on one lineage however diverse the archive is. Under Eq. 8, each lineage's selection probability is at least $\gamma$ times its archive share. Theorem 1 thus shows that WGM passes archive-level diversity on to reproduction.

**Without the assumption.** Assumption 2 is a property of the realized archive and is not implied by Eq. 8. Appendix B.4 shows that, even without it, every lineage receives a polynomially growing expected number of expansions.

### B.4 Reproduction Floor without Assumption 2

The following result requires no assumption on the archive or on the rewards. It uses only that self-modification within the lineage of interest does not always fail.

**Assumption 3** (Non-degenerate expansion)**.** For a lineage $c$, there exists $\alpha_c \in (0, 1]$ such that, for every $t$ with $a_t \in c$, the child $b_t$ is admitted to the archive with probability at least $\alpha_c$ conditional on $\mathcal{F}_{t-1}$ and $a_t$.

**Proposition 2** (Reward-independent reproduction floor)**.** *Let parents be selected by Eq. 8 with any positive weights, and hence for any reward sequence. Fix an adaptive expansion $s$, and let $\mathcal{F}_{s-1}$ be the history before $a_s$ is drawn.*

*(i) Every agent $a \in \mathcal{A}_s$ is expanded infinitely often almost surely, and $\mathbb{E}[n_a(T) \mid \mathcal{F}_{s-1}] \geq \gamma \ln \frac{T+1}{s}$.*

*(ii) Every lineage $c \neq a_0$ with $\mathcal{A}_s \cap c \neq \emptyset$ that satisfies Assumption 3 satisfies*

$$\mathbb{E}[n_c(T) \mid \mathcal{F}_{s-1}] \;\geq\; \frac{1}{\alpha_c}\left[\Big(\frac{T+1}{s}\Big)^{\alpha_c\gamma} - 1\right] = \Omega\big(T^{\alpha_c\gamma}\big). \tag{B.10}$$

*(iii) Consequently, for every lineage $c' \neq c$, $\mathbb{E}[1 - q_{c'}(T) \mid \mathcal{F}_{s-1}] = \Omega\big(T^{\alpha_c\gamma-1}\big)$, whereas Thompson sampling in Proposition 1 gives $\mathbb{E}[1 - q_{c^*}(T)] = O(\log T/T)$.*

Proposition 2 bounds the rate at which reproduction can concentrate; Theorem 1 gives the condition under which concentration is excluded. The separation from $O(\log T)$ is asymptotic: bound (ii) improves on bound (i) applied to the founder once $\alpha_c\gamma \ln(T/s) \gtrsim 1$.

*Proof.* Fix $s \geq t_0$, write $\mathbb{E}_s[\cdot] = \mathbb{E}[\cdot \mid \mathcal{F}_{s-1}]$, and let $\beta = \alpha_c\gamma \in (0, 1]$.

*(i)* Agent $a \in \mathcal{A}_s$ stays in the archive forever. By Eq. B.4, $\sum_{t\geq s} \Pr(a_t = a \mid \mathcal{F}_{t-1}) \geq \gamma \sum_{t\geq s} 1/t = \infty$, so $a$ is selected infinitely often almost surely by Lévy's extension of the Borel–Cantelli lemma. Moreover, $\mathbb{E}_s[n_a(T)] \geq \sum_{t=s}^{T} \mathbb{E}_s[\Pr(a_t = a \mid \mathcal{F}_{t-1})] \geq \gamma \sum_{t=s}^{T} 1/t \geq \gamma \ln \frac{T+1}{s}$.

*(ii)* Let $X_t = |\mathcal{A}_t \cap c|$, so $X_s \geq 1$. Since $c \neq a_0$, an admitted child of a parent in $c$ joins $c$ (Definition 1), and the archive never shrinks. By Assumption 3 and Eq. B.4, for $t \geq s$,

$$\begin{aligned}\mathbb{E}[X_{t+1} \mid \mathcal{F}_{t-1}] \;&\geq\; X_t + \alpha_c \,\Pr(a_t \in c \mid \mathcal{F}_{t-1}) \\ &\geq\; X_t + \alpha_c\gamma\frac{X_t}{N_t} \;\geq\; X_t\Big(1 + \frac{\beta}{t}\Big).\end{aligned} \tag{B.11}$$

Taking $\mathbb{E}_s$, iterating, and applying Bernoulli's inequality $(1 + 1/u)^\beta \leq 1 + \beta/u$ for $\beta \in [0, 1]$ gives

$$\mathbb{E}_s[X_t] \;\geq\; \prod_{u=s}^{t-1}\Big(1 + \frac{\beta}{u}\Big) \;\geq\; \prod_{u=s}^{t-1}\Big(\frac{u+1}{u}\Big)^\beta = \Big(\frac{t}{s}\Big)^\beta. \tag{B.12}$$

Summing the selection probabilities and using Eqs. B.4 and B.12,

$$\begin{aligned}\mathbb{E}_s[n_c(T)] &\geq \gamma \sum_{t=s}^{T} \frac{\mathbb{E}_s[X_t]}{t} \geq \gamma s^{-\beta} \sum_{t=s}^{T} t^{\beta-1} \\ &\geq \gamma s^{-\beta} \int_s^{T+1} x^{\beta-1} dx = \frac{1}{\alpha_c}\Big[\Big(\frac{T+1}{s}\Big)^\beta - 1\Big],\end{aligned} \tag{B.13}$$

where the sum-to-integral step uses that $x^{\beta-1}$ is nonincreasing.

*(iii)* For $c' \neq c$, $1 - q_{c'}(T) \geq q_c(T) = n_c(T)/T$. Taking $\mathbb{E}_s$ and applying (ii) completes the proof.

**Asynchronous execution.** Suppose each child selected at expansion $u$ is admitted, if at all, before expansion $u + P$. Let $g(t) = \mathbb{E}_s[X_t]$. Every child selected at some $u \leq T - P - 1$ is settled before expansion $T$, so

$$g(T) \;\geq\; 1 + \beta \sum_{u=s}^{T-P-1} \frac{g(u)}{u}. \tag{B.14}$$

Let $K = s^{-\beta}/(1 + \beta P/s)$. We show $g(T) \geq KT^\beta$ by induction on $T$.

- *Base case,* $s \le T \le s + P$. Here $KT^\beta \le (1 + P/s)^\beta/(1 + \beta P/s) \le 1 \le g(T)$ by Bernoulli's inequality.
- *Inductive step,* $T > s + P$. By the induction hypothesis and the integral bound, $g(T) \ge 1 + K\big((T - P)^\beta - s^\beta\big)$. Moreover, $T^\beta - (T - P)^\beta \le \beta P(T - P)^{\beta-1} \le \beta P s^{\beta-1}$ and $1 - Ks^\beta = K\beta P s^{\beta-1}$, so $g(T) \ge KT^\beta$.

Summing as above gives $\mathbb{E}_s[n_c(T)] \ge \frac{1}{\alpha_c(1+\beta P/s)}\big[\big(\frac{T+1}{s}\big)^\beta - 1\big]$, which has the same exponent as the sequential bound. The proof of Theorem 1 also carries over, since it uses only Eq. B.4 at the time each parent is drawn.

**Tightness.** The exponent $\beta = \alpha_c\gamma$ cannot be improved in general. Suppose the reward-driven component places all but a summable amount of mass on a lineage $c' \ne c$ (weights stay positive, with $\sum_t \epsilon_t < \infty$ mass elsewhere). Suppose also that children of $c'$ are always admitted and children of $c$ are admitted with probability exactly $\alpha_c$. Then $N_t = \Theta(t)$, and $\mathbb{E}[X_t]$ grows as $\Theta(t^\beta)$, so the archive share of $c$ decays as $\Theta(t^{\beta-1})$. In this example, a simulation with $\gamma = 0.3$ and $\alpha_c = 1$ matches the bound in (ii) to within 10% at $T = 10^3$.

## C ALGORITHM

Algorithm 1 presents the procedure of WGM. We initialize the search by expanding the initial agent $a_0$ five times in each of $P$ parallel processes following HGM. All initial expansions start from $a_0$. We wait for these expansions and their initial evaluations to finish before adaptive selection begins, avoiding a bias toward agents whose expansions finish first.

Let $B_{\mathrm{tot}}$ denote the total budget of task evaluations, and reserve $B_{\mathrm{fin}}$ of them for the final selection stage of Algorithm 1. Initialization and adaptive search share $B_{\mathrm{tot}} - B_{\mathrm{fin}}$, while final selection uses $B_{\mathrm{fin}}$. Every task evaluation is charged to its stage budget, including evaluations of failed attempts. We assume that the budget covers initialization while preserving $B_{\mathrm{fin}}$ for final evaluation. Budget checks enforce per-operation cost limits: a full adaptive iteration must fit one expansion, up to $M$ initial task evaluations, and one additional BAI evaluation without using the reserved budget.

The forced initial expansions provide evaluation records but do not trigger updates to EXP3 weights. After initialization, all archived agents receive weight one, and their own and clade statistics are initialized from stored evaluation records. Adaptive expansion starts at $t = 5P + 1$, counting the initial expansion attempts. This preserves $N_t \le t$. During adaptive search, the initial evaluation of each child provides exactly one update to its parent's EXP3 weight. For each archived agent, both initial and additional evaluations update its own statistics and the clade statistics along its path to the root. Additional BAI evaluations update the confidence bounds and affect subsequent parent selection through the updated clade statistics. All evaluations follow the search-task distribution $\mathcal{D}$ in Eq. 2. All ties are broken uniformly at random. Only executable children that satisfy the archive admission rule are retained.

**BAI details.** For an agent with $n_a = S_a + F_a$ stored outcomes and $\hat{\mu}_a = S_a/n_a$, the confidence bounds are the Wilson score interval (Wilson, 1927)

$$[\ell_a, u_a] = \frac{1}{1 + z^2/n_a}\left[\hat{\mu}_a + \frac{z^2}{2n_a} \mp z\sqrt{\frac{\hat{\mu}_a(1 - \hat{\mu}_a)}{n_a} + \frac{z^2}{4n_a^2}}\right], \tag{C.1}$$

computed from all stored outcomes of the agent, with $[\ell_a, u_a] = [0, 1]$ for an unevaluated agent; we use $z = 1.96$. Unlike Hoeffding-type bounds, Wilson intervals remain informative when $n_a$ is small and $\hat{\mu}_a$ is close to 0 or 1, which is the typical regime for newly created agents. They are per-agent nominal intervals, and we apply no multiplicity correction across the growing archive. Eq. 13 is therefore a certificate on the event that all intervals cover their means, and we use the leader–challenger rule as an evaluation-allocation rule rather than claiming a PAC identification guarantee. The rule is run with a fixed budget and no early stopping. During search, each adaptive iteration spends one additional evaluation. After expansion stops, BAI continues on the frozen archive until $B_{\mathrm{fin}}$ is exhausted and returns the agent with the highest lower confidence bound.

**Algorithm 1** Wright–Gödel Machine (WGM)

**Input:** initial agent $a_0$, exploration share $\gamma \in (0,1)$, batch size $M$, number of processes $P$, total budget $B_{\mathrm{tot}}$, reserved final budget $B_{\mathrm{fin}}$
Expand $a_0$ five times in each of $P$ processes; evaluate each executable child on $M$ tasks; admit children with parent $a_0$
$t \leftarrow 5P+1$; for all $a \in \mathcal{A}_t$: $w_{a,t} \leftarrow 1$, initialize $S_a, F_a, [\ell_a, u_a]$ and clade statistics $S_a^C, F_a^C$
**while** a full iteration fits the budget outside $B_{\mathrm{fin}}$ **do**
  $N_t \leftarrow |\mathcal{A}_t|$, $\eta_t \leftarrow \gamma / N_t$; compute $\rho_{a,t}$ and $\pi_{a,t}$ by Eqs. 6 and 8
  Sample $a_t \sim \pi_t$; generate $b_t \leftarrow \Phi(a_t; e_t)$
  $r_t \leftarrow 0$; **if** $b_t$ is executable **then** evaluate $b_t$ on $\mathcal{X}_t$ and set $r_t$ by Eq. 9
  Update $w_{a_t,t+1} \leftarrow w_{a_t,t} \exp(\eta_t r_t / \pi_{a_t,t})$; keep $w_{a,t+1} \leftarrow w_{a,t}$ for $a \neq a_t$
  $\mathcal{A}_{t+1} \leftarrow \mathcal{A}_t$
  **if** $b_t$ is executable and satisfies the admission rule **then**
    Add $b_t$ with parent $a_t$ and $w_{b_t,t+1} \leftarrow 1$; add its outcomes to the clade statistics of $b_t$ and all its ancestors
  **end if**
  BAISTEP($\mathcal{A}_{t+1}$); $t \leftarrow t+1$
**end while**
Freeze the archive $\mathcal{A} \leftarrow \mathcal{A}_t$
**while** $B_{\mathrm{fin}}$ permits another task evaluation **do**
  BAISTEP($\mathcal{A}$)
**end while**
**return** $\arg\max_{a \in \mathcal{A}} \ell_a$

**function** BAISTEP($\mathcal{A}$)
  $h \leftarrow \arg\max_{a \in \mathcal{A}} \ell_a$, $k \leftarrow \arg\max_{a \in \mathcal{A} \setminus \{h\}} u_a$, $a_{\mathrm{eval}} \leftarrow \arg\max_{a \in \{h,k\}} (u_a - \ell_a)$ ▷ $a_{\mathrm{eval}} \leftarrow h$ if $|\mathcal{A}| = 1$
  Evaluate $a_{\mathrm{eval}}$ on one search task with outcome $y \in \{0,1\}$
  Update $S_{a_{\mathrm{eval}}}, F_{a_{\mathrm{eval}}}$ and $[\ell_{a_{\mathrm{eval}}}, u_{a_{\mathrm{eval}}}]$ by Eq. C.1; add $y$ to the clade statistics of $a_{\mathrm{eval}}$ and all its ancestors
**end function**

### C.1 Hyperparameters

We follow the experimental setup of HGM (Wang et al., 2025a): all methods start from the same initial agent, use GPT-5.5 for both self-modification and task execution, receive a budget of 800 task evaluations, and are run with three random seeds. For WGM, each new child is evaluated on $M = 3$ search tasks. BAI spends one additional evaluation per iteration during search, and $B_{\mathrm{fin}} = 260$ evaluations are reserved for final selection. Confidence intervals use $z = 1.96$.

## D Additional Experimental Details

### D.1 Baselines

We compare WGM with two state-of-the-art archive-based self-improving coding agent frameworks, DGM and HGM. These methods mainly differ in how they select parents for modification and allocate evaluations among archived agents:

- **DGM:** Maintains an archive of previously discovered agents and selects a parent according to its evaluation performance and the number of children it has produced. Specifically, DGM combines a sigmoid-scaled performance score with a bonus for agents with fewer children, encouraging the search to revisit less-expanded agents.
- **HGM:** Estimates the potential of an archived agent using evaluation statistics aggregated over its clade, including its descendants. It applies Thompson sampling to select parents based on these clade-level statistics and adaptively allocates computation between agent modification and evaluation.
- **WGM (Ours):** Uses an EXP3-based parent-selection rule that combines clade-level statistics with a fixed uniform exploration probability. This exploration floor preserves expansion

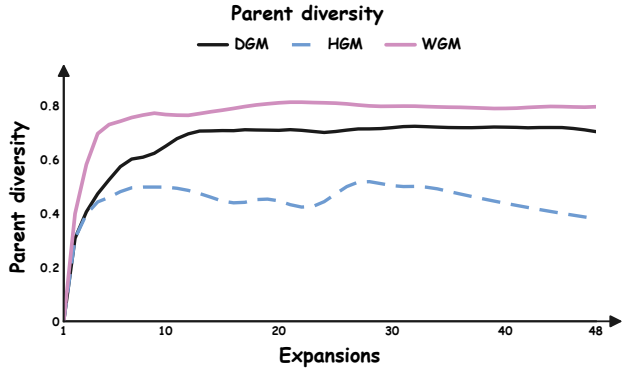



Figure 3: **Lineage parent diversity during self-improvement.** Parent diversity $D(T)$ computed with the lineage indicator. The curves use the same search histories as Figure 1*(b)*.

opportunities for all archived agents. WGM further uses best arm identification (BAI) to allocate additional evaluations and select the final agent.

### D.2 Datasets

We evaluate the compared methods on three software engineering benchmarks:

- **SWE-bench Verified:** Contains 500 human-validated software engineering tasks derived from real-world GitHub issues. The main comparison and ablation experiments use SWE-Verified-60, the same 60-task subset adopted by HGM, while the full benchmark is used for self-improvement in the generalization experiment.
- **SWE-Lite:** Contains 300 software engineering tasks and is used to evaluate the generalization of the discovered agents. We report results on both Standard setting with all 300 tasks and Filtered setting with 207 tasks that do not overlap with the self-improvement set.
- **Polyglot:** Evaluates coding agents across multiple programming languages. We use it in the main comparison and ablation experiments to examine whether the methods generalize beyond repository-level Python tasks.

### D.3 Additional Experiments Setup

**Figure 1*(a)*.** We run DGM and HGM on SWE-Verified-60 with the same initial agent and a budget of 200 task evaluations. In each run, the dominant lineage is the lineage with the most parent selections, and all other lineages are alternative lineages. For each group, we report the accuracy of its best agent. Because these runs use a smaller budget than the main experiments, the accuracies are lower than the final-agent accuracies in Table 1.

**Figure 1*(b)*and 3.** These figures compare how reproduction is distributed at the same number of expansions. They are computed from the parent-selection logs. Since DGM spends more task evaluations per expansion, a budget of 800 evaluations yields fewer than 48 DGM expansions. We therefore extended the DGM run until it reached 48 expansions. For HGM and WGM, we show the first 48 expansions of their evaluation runs.

### D.4 Lineage Statistics and Parent Diversity

**Lineages.** Lineage labels follow Definition 1. We reconstruct the search tree from the archive log, in which each admitted agent records its parent, and label each agent by the child of $a_0$ on its root path. For every parent-selection event $t$ in the log, we record the lineage of $a_t$ and compute $n_c(T)$, $q_c(T)$, and $L(T)$ (Eq. 3) for $T = 1, 2, \ldots$. Expansions that fail to produce an admitted child still count as selection events, since they consume an expansion opportunity. The same code is applied to the logs of all methods.

**Parent diversity.** Section 3.2 defines parent diversity $D(T)$ for a general difference $d$ between agents. In our experiments, we instantiate $d$ with the lineage indicator $d_{\text{lin}}(a, b) = \mathbb{1}\{\phi(a) \neq \phi(b)\}$, so that

$$D_{\text{lin}}(T) = \frac{1}{T^2} \sum_{s=1}^{T} \sum_{t=1}^{T} d_{\text{lin}}(a_s, a_t) = 1 - \sum_{c} q_c(T)^2 = 1 - \frac{1}{L(T)}. \tag{D.1}$$

$D_{\text{lin}}(T)$ is the probability that two parents drawn from the first $T$ expansions come from different lineages. It describes the same lineage shares as $L(T)$ on a $[0, 1]$ scale, as the Gini–Simpson and inverse Simpson indices do in ecology (Hill, 1973).

The difference $d$ in Eq. 4 admits several instantiations. A code-level distance, such as the token edit distance between agent scaffolds, depends on how source files are tokenized and compared, and can be dominated by incidental edits, such as added logging or reformatting, that self-modification often introduces. A behavioral distance, such as the fraction of tasks on which two agents disagree, requires every agent to be evaluated on a common set of tasks; under our budget, each child is evaluated on only $M = 3$ search tasks, so such a distance would need additional evaluations. We therefore use lineage membership, which is the unit on which parent selection acts: every descendant inherits its founder's modifications, so lineage identity records shared design ancestry. Lineage membership is a coarse notion of design difference, since it treats parents within a lineage as identical. Other distances can be substituted into Eq. 4 without changing the analysis in Appendix B, and we leave them to future work.

The calculation uses the parent-selection histories from the original search runs and requires no additional task evaluations. Figure 3 reports $D_{\text{lin}}(T)$ through the search. For HGM it decreases as parents are increasingly drawn from the same lineage, whereas WGM keeps it higher over the observed horizon.

### D.5 Other Details

All self-improvement experiments use GPT-5.5 for both agent self-modification and task execution. The cross-model generalization experiments additionally evaluate the discovered agents with GPT-5.6 and Claude Opus 5.5. Overall, we spent approximately $12,000 USD to produce all reported experimental results.