ZERO-SHOT COORDINATION IN TEXT OPTIMISATION: DO INDEPENDENTLY OPTIMISED AGENTS WORK TOGETHER?

Anonymous authors
Paper under double-blind review

ABSTRACT

Text optimisation improves an agent by letting a language model repeatedly rewrite text artefacts, such as policy code, action tables, or instructions, against a score. These methods are judged almost entirely by that score. In cooperative tasks this hides a problem known from multi-agent reinforcement learning as zero-shot coordination: agents optimised separately may each perform well, yet fail when paired, because every run settles on its own conventions. We study zero-shot coordination in text optimisation. We run independent optimisation processes across language models and optimisation methods, and pair the resulting agents with partners they never met. A small, exactly solvable signalling game reveals the conventions each run adopts, and a cooperative cooking game tests the same question in a larger domain. We find that optimisation replaces shared focal points with conventions tied to each run's presentation of the task. Scoring candidates against independent samples of the unoptimised model preserves compatibility when those samples agree, and rejecting proposals that are not invariant under the game's symmetries preserves it when they disagree. We argue that partner generalisation should be reported alongside the score.

1 INTRODUCTION

Two tourists who lose each other in Paris and cannot communicate will often both go to the Eiffel Tower. The tower is a focal point (Schelling, 1960): nothing in the rules singles it out, yet both know that the other knows it. Unoptimised language-model agents have coordinated with unfamiliar reinforcement-learning and behaviour-cloned partners in Overcooked and Hanabi (Agashe et al., 2025). We interpret such results through the possibility that a language model brings useful task cues and a shared starting point to coordination. Text optimisation now builds agents by repeatedly rewriting a prompt, instructions or policy code and keeping the changes that raise a score (Agrawal et al., 2026). In the self-play objectives we study, each candidate earns its score with a copy of itself. An optimiser that finds a better meeting place known only to its own copies is rewarded for it, and an agent built this way may fail with a partner built independently. Zero-shot coordination asks exactly this question: whether agents produced by separate runs of the same procedure work together without joint training (Hu et al., 2020). In reinforcement learning the answer is often no, because independent runs settle on different conventions that are equally good in self-play (Hu et al., 2020; 2021).

Figure 1 illustrates the setup. This paper asks whether text optimisation destroys the focal points that language models bring, and how they can be kept. Pairing agents from independent GEPA and evolutionary code search runs, we find that optimisation replaces the shared landmark with a private rule: most runs that leave the landmark move to the option that their own presentation of the task lists first, and runs that saw different presentations no longer agree. Whether the model's unoptimised samples agree with each other, which can be measured before any search, shapes the remedy. For Haiku, whose samples agree, scoring candidates against independent samples keeps runs compatible, as does choosing among archived candidates after the run. For Sonnet and Fable, whose samples split, a constraint that rejects proposals depending on arbitrary labels, a text analogue of Other-Play (Hu et al., 2020), restores compatibility; for Fable, scoring against independently presented samples does too. The constraint costs self-play and, in part, play with the unoptimised model. Hayler et al.

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.

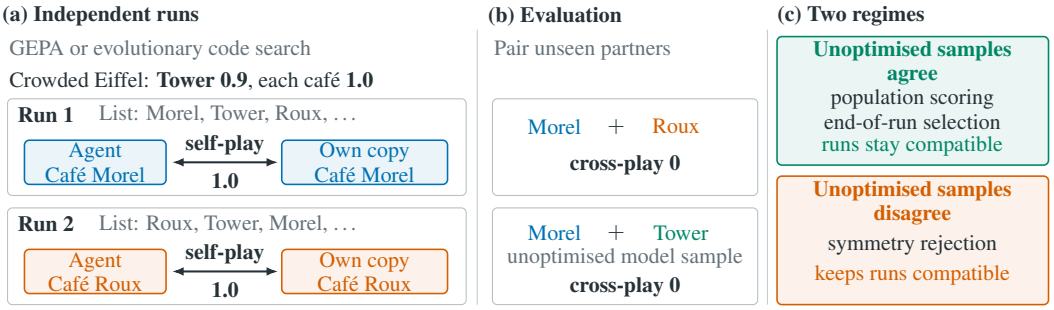

Figure 1: Independent text-optimisation runs score each agent with its own copy, but evaluation pairs agents from different runs and with unoptimised model samples. The crowded Eiffel example shows how different task orders can favour different cafés. In the signalling game, the two regimes indicate which remedies preserve compatibility when unoptimised samples agree or disagree.

(2026) study zero-shot coordination and propose coordination-friendly definitions, and Wang et al. (2026) optimise the prompts of interacting agents jointly; we optimise agents separately and pair each with partners from runs it never encountered.

Our contributions are as follows.

- We operationalise zero-shot coordination for iterative text optimisation, with each independent search run as the unit of evaluation, and measure cross-play both between optimised runs and with the unoptimised model.
- We show that GEPA and evolutionary code search raise self-play while cross-play between independent runs falls, in a worded coordination game for two models and in a signalling game, and we trace the loss to a positional rule that interventions on the presentation control.
- We measure, before any search, whether a model's unoptimised samples agree across presentations, for five models from two providers. In new-run confirmations, population scoring and end-of-run selection preserved cross-play for Haiku, whose samples agree. For Sonnet and Fable, whose samples split, symmetry rejection preserved it; for Fable, population scoring with independently presented partners did as well.
- We translate coordination methods from deep multi-agent reinforcement learning into drop-in changes to GEPA and evolutionary code search: population scoring against unoptimised samples of the model (after fictitious co-play; Strouse et al., 2021), end-of-run selection against such samples, exact symmetry rejection (after Other-Play; Hu et al., 2020), and a fictitious-play update for few-shot adaptation (after Brown, 1951). Section B lists what each analogue keeps and drops.
- We delimit the effect: in a cooperative cooking game optimisation raised both self-play and cross-play, and the best few-shot update rule depended on whether the model falls back to a shared focal point.

2 Related work

Text-search methods improve task scores through proposal and revision (Romera-Paredes et al., 2024; Yang et al., 2024; Fernando et al., 2024; Khattab et al., 2024; Opsahl-Ong et al., 2024; Yuksekgonul et al., 2025). Prior work studies zero-shot language agents with unseen partners and coordination-friendly definitions, while joint prompt optimisation tunes the prompts of interacting agents together (Hayler et al., 2026; Agashe et al., 2025; Wang et al., 2026). Multi-agent benchmarks for language-model agents, such as a competitive bidding, bluffing and bargaining game, rate each agent against the other evaluated models and fixed code-agent baselines (Müller & Müller, 2026). Our endpoint instead pairs agents from separate searches, following the zero-shot coordination setting of Hu et al. (2020). Distinctions from other prompt, population and partner-adaptation methods are given in Section A.11.

3 BACKGROUND

Zero-shot coordination asks whether agents produced independently by the same procedure work together without joint training. Cross-play pairs agents from independent runs, while self-play pairs an agent with its own copy (Hu et al., 2020). Self-play can reward different conventions in different runs. Other-Play uses game symmetries, off-belief learning grounds past actions, and population methods expose agents to more partners (Hu et al., 2020; 2021; Strouse et al., 2021). Text optimisation revises prompts, instructions or code against a score. GEPA uses reflection and a Pareto front, while evolutionary code search maintains a scored archive (Agrawal et al., 2026; Novikov et al., 2025; Sharma, 2025). When an artefact is scored with its own copy, the same coordination question applies. Unoptimised language-model agents have coordinated with unseen partners in Overcooked and Hanabi (Agashe et al., 2025). We interpret a model's shared starting tendencies as a possible coordination resource. Further background is in Section A.9.

4 SETTING

4.1 RUNS AND ARTEFACTS

A *run* is one independent execution of a text-optimisation method on a task. Runs of the same condition share the language model, seed prompt, evaluation budget and method settings. They differ in model and method randomness and, where specified, in the assigned order or labels of the options. Within a run, every valid candidate policy is archived with its score. The policy deployed at a given budget is the highest scoring candidate so far, with ties broken by earliest discovery.

In the main fixed-policy games, the artefact is the policy itself, written as a choice, code or an explicit table from decision points to actions. We evaluate these policies by enumeration without model calls during play. Other experiments in the appendix execute prompts through a model and therefore have execution cost. We do not use model internals. We estimate the unoptimised policy distribution from independent zero-shot samples.

We study two iterative methods and one selection control. Best-of-N draws independent candidates from the unoptimised model and keeps the best one. It measures selection without iterative rewriting. GEPA rewrites candidates after reviewing evaluation traces and keeps a Pareto front over instances (Agrawal et al., 2026). OpenEvolve changes and combines programs from a scored archive (Sharma, 2025). We call it evolutionary code search.

4.2 SELF-PLAY AND CROSS-PLAY

By default each method scores a candidate by letting it play with itself, which we call self-play. In the games we study one artefact describes the behaviour of every player, so text optimisation of this kind is self-play by construction. Cross-play pairs the deployed policy of one run with the deployed policy of another run of the same condition at the same budget. Players draw their actions independently. The gap between self-play and cross-play measures how much a run's success depends on playing with itself. Cross-play with sampled unoptimised policies measures how far a run has moved from the shared prior.

4.3 THE TASK TEXT AS A COORDINATION DEVICE

A text optimiser reads a description of the task, and the description itself carries information that a reinforcement-learning agent never sees. Option names such as "the Eiffel Tower" or "the first lever" come with the model's associations, and the model tends to treat some options as obvious. Independent runs of the same model therefore share a starting point that can serve as a focal point. Whether that focal point survives optimisation is one of our central questions.

4.4 LABEL REGIMES

To separate meaning from position, we use three label regimes. In the first, named options appear in the same order for every run. In the second, the names stay but order differs across runs. In the third, options have neutral labels such as "spot A" to "spot J", and each run receives an independent label

assignment. Only the third regime matches the label-free setting of Other-Play (Hu et al., 2020); the first two keep shared option names, which that setting excludes. Treutlein et al. (2021) formalise Other-Play's no-shared-labels assumption as label-free coordination and argue that label-freeness alone does not capture the aims of zero-shot coordination. Before cross-play, we map each policy back to the underlying options using its run's stored assignment.

5 The Eiffel Tower problem

Two tourists lose each other in Paris and cannot communicate. Each picks one of ten places to meet. If they pick the same place they meet and receive its payoff; otherwise they receive nothing. In the *crowded tower* variant the Eiffel Tower pays 0.9 and nine quiet cafés each pay 1.0. The game is the lever game of Hu et al. (2020), in which nine levers pay 1.0 and one unique lever pays 0.9, written in words that give the model a real prior. It also tests whether agents use semantic information for coordination, the question studied by Hayler et al. (2026).

The variant is built so that the unoptimised model and a self-play optimiser pull in opposite directions. A model asked where to meet a stranger in Paris is likely to name the Eiffel Tower, and two independent copies that both do so meet and score 0.9. An optimiser that scores a policy against its own copy discovers that any café scores 1.0. If independent runs then choose different cafés, cross-play collapses toward the value of a random match. If they all move to the same café, for example the one that seems second most obvious, coordination survives. Which of these happens, and how quickly, is what we measure. The *open tower* variant separated the model's prior from the label-free solution: the tower paid 1.0, while one unremarkable café uniquely paid 0.9 (Section A.13).

Three further controls guard the interpretation. Payoffs are either stated in the task text or learned only from scores. Each task text exists in three paraphrases with identical content. A freshly invented game with the same structure but different numbers and names checks that results do not come from the model recognising the lever game from the literature. The texts contain no hint about partners, strategies or conventions.

6 Optimisation erodes zero-shot coordination

Protocol. The main grids use twelve independent optimisation runs per condition with a budget of 100 candidate evaluations each; follow-up conditions were first run as pilots of four runs and expanded by a rule fixed in advance, and pilot-only results are marked as exploratory. At every budget, each run deploys its best candidate so far (ties broken by earliest discovery), and we pair the deployed policies of different runs at the same budget. Final cross-play is compared with the run's own first candidate and with an independent panel of fifty unoptimised samples. Intervals are bootstrap intervals over runs; tests permute whole runs, and p-values are Holm-corrected within the declared family.

Before optimisation, meaning coordinates and arbitrary labels do not. Unoptimised models pick the Eiffel Tower almost always when places carry their names, whether or not the list order is shared: cross-play between independent samples is 0.83 to 1.00 for Haiku 4.5, Sonnet 4.6 and Sonnet 5. With neutral labels shuffled per sample, cross-play is 0.09 to 0.14, the level of a random match. When the same model answers twice, it returns the same place, not the same list position, in 84 to 100% of cases. The two findings in the literature, robust coordination with meaningful options and failure with arbitrary labels, therefore appear side by side in one game.

After optimisation, runs abandon the focal point for private conventions. Figure 2 shows cross-play among Sonnet 5 runs in the crowded tower. When places are listed in a shuffled order, 9 of 12 GEPA runs and 11 of 12 OpenEvolve runs leave the tower within the first few evaluations and settle on a café. Because the runs choose different cafés, cross-play falls from 0.90 to 0.12 for both methods (95% intervals $[0.03, 0.23]$ and $[0.03, 0.25]$; Holm-corrected $p = 0.020$ and 0.005 against each run's own first candidate), while self-play rises to 0.97 to 0.99. Best-of-N, which selects among independent samples without iterative rewriting, never leaves the tower and keeps cross-play at 0.90. When the list order is shared, all runs converge on the first café in the list and cross-play among optimised runs returns to 1.00 after a transient drop to about 0.43; cross-play with the unoptimised

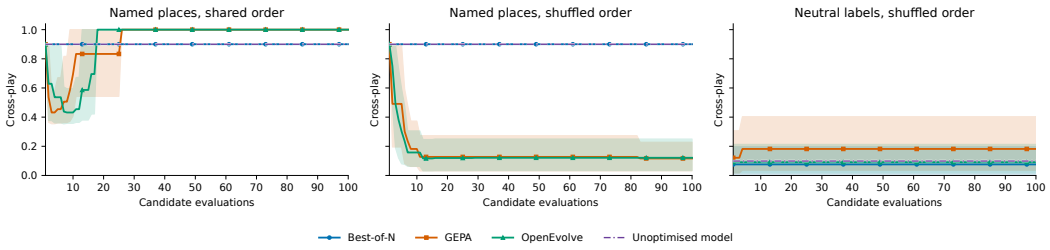

Figure 2: Cross-play among twelve independent Sonnet 5 runs per method in the crowded tower, as a function of candidate evaluations. The y-axis is the mean payoff (0 to 1) when agents from two different runs meet. The dashed line is the unoptimised model. Bands are 95% bootstrap intervals over runs.

model, which still chooses the tower, falls to 0. In the open tower, where the tower already pays the maximum, no run leaves it.

The same erosion with Haiku 4.5. The same grid with Haiku 4.5 (twelve runs per condition) reproduces the erosion under a shuffled order for all three methods (final cross-play 0.08 to 0.11; Holm-corrected $p \leq 0.01$ against each run's own first candidate). Unlike Sonnet 5, Haiku's best-of-N drifts too: its own samples occasionally contain a café, and selecting by self-play then suffices to leave the tower. With payoffs learned only from scores, Haiku 4.5 still drifted, while Sonnet 5 did not propose a café in that condition. Tie-breaking and three prespecified paraphrases gave the robustness checks in Section A.15.

The same pattern without named options. In Tiny Hanabi with Haiku 4.5 and shared labels, every method reaches the same optimal convention class (cross-play 10 out of 10). With labels permuted per run, GEPA and OpenEvolve reach perfect self-play, but cross-play falls to 4.6 and 4.7, below the 6.7 of best-of-N; the gap between self-play and cross-play grows from 0.9 under selection alone to about 5.4 under iterative optimisation. Replacing GEPA's Pareto-front selection by current-best selection leaves cross-play unchanged. Figure 3 and Section 8.1 follow these dynamics over the search and test remedies. In a Lewis signalling game with four or eight objects, an exploratory pilot showed no erosion: the first mapping already earned the maximum self-play score.

7 Mechanism: runs select the first alternative in their own presentation

The drift in Section 6 has a clear pattern: most runs that leave the tower select the first café in their own list. Each run then has a precise choice with its own copy. An independently built partner may see a different list and select another café. We call a selection procedure a transferable coordination principle only if its choices follow relevant public features across the presentations we test. We judge the executed choices and cross-play, rather than how general the instruction sounds. Five analyses test this account.

Runs choose the first alternative in their own presentation. We find a positional choice after optimisation, alongside the semantic coordination questions studied by Hayler et al. (2026). We count a run if it started at the tower, saw named places in an independently shuffled order, and ended at a café. We then record that café's position in the run's own list. Under this rule, 19 of 20 Sonnet 5 runs and 26 of 32 Haiku 4.5 runs chose their first café. The Haiku count covers the main stated-payoff runs; fork parents are reported separately. The broader count of all Haiku runs that left the tower is 30 of 36. All 31 eligible hidden-payoff runs chose the first café. Different lists can produce different choices and lower cross-play. With a shared order, runs often choose the same first café.

Changing the order changes the choice. We gave all runs the same shuffled order in two conditions. Both lists differed from the canonical one and put a different café first. Of the 28 runs that left

the tower, 27 chose the first café in their shared list. Swapping the list changed the café on which agents converged. They coordinated through position and still left the unoptimised model's choice.

Presentation predicts the convention. We restored the complete optimiser state of twelve Haiku runs at 0, 10 and 30 evaluations. We continued each state eight times with fresh randomness. Even at evaluation 0, before any café proposal, 73 to 81% of continuation pairs ended at the same café. Chance agreement was about 11%. Between 79 and 86% chose the parent's café. From evaluation 10 on, agreement was complete. These results link the final convention to the run's presentation.

The objective can matter without a search loop. We added one sentence describing the self-play objective to the unoptimised prompt. We ran no optimisation loop in this control. For Haiku 4.5, 98% of samples chose a café. Nearly all chose the first café in their own list. Cross-play between samples fell from 0.67 to 0.10. For Sonnet 5, choice-only cross-play moved from 0.90 to 0.83. With short reasoning, it stayed at 0.90. Only iterative optimisation moved that model off the tower consistently. In this game, stating the objective sufficed for Haiku, while Sonnet needed search to reach the private option. The incentive and exploration conditions and further interpretation are detailed in Section A.10.

8 Remedies

We translate remedies for zero-shot coordination from reinforcement learning into text optimisation without using model internals, and add one remedy native to text optimisation. Each changes either the score the optimiser receives or the information it sees. Unless stated otherwise, the optimiser is GEPA on the crowded or open tower with places listed in a shuffled order. In the crowded tower, remedies that rewarded the tower kept Sonnet 5 runs compatible, with cross-play rising from 0.06 to 0.90 and self-play falling from 0.98 to 0.90. An exploratory open-tower pilot separated prior anchoring from symmetry-permuted evaluation. A language-model convention critic kept Sonnet 5 tower runs on the shared choice, but its Tiny Hanabi effect did not replicate. An exact symmetry test used as a constraint yielded the remedy tested in Section 8.1. The methods, results and limits of these controls are in Section A.14.

8.1 Two regimes in the signalling game

Tiny Hanabi with labels permuted independently for each run has no named focal point, and the payoff table admits a relabelling of signals and actions that maps one optimal convention onto another. Three conventions matter. Two mirror-image conventions earn the maximum self-play score of 10 but score about 0 with each other. A third convention uses only the signal and action that the relabelling leaves fixed. It earns 8 with itself and with any copy of itself, whatever labels a run received; we call a policy *invariant* if its play does not change under the relabelling. We compare three ways of using partners or symmetries against vanilla self-play search. *Population scoring* replaces self-play by the mean score against four frozen unoptimised samples of the same model, drawn with an independent presentation for each sample and run. *Symmetry rejection* discards every proposal that is not invariant under the verified relabellings. *End-of-run selection* changes nothing during search: after the run, it deploys the archived candidate with the highest mean score against a separate panel of four unoptimised samples. None of these uses the other runs or the test partners. Symmetry rejection uses the verified symmetry group, which in this game leaves a single competent convention.

The unoptimised models differ in agreement. Before any search, we sampled 36 policies per model under all twelve label presentations and paired them (Table 8). For Haiku 4.5, two samples score similarly whether or not they saw the same presentation (7.11 and 6.81), and 34 of 36 samples play invariantly. Most Sonnet 5 samples, and all sampled Opus 5.5, Fable 5.1 and Gemini 3.8 Flash policies, pick one of the two maximal conventions according to the order in which labels are presented. Samples that saw the same presentation score 9.1 to 10.0 together, samples that saw different presentations score 4.6 to 4.9, and at most one of 36 samples plays invariantly. We call the first case a *shared prior* and the second a *split prior*.

Model	Method	Cross-play between runs		With fresh base samples		Self-play
		GEPA	OpenEvolve	GEPA	OpenEvolve	
Haiku 4.5	Vanilla	4.64	4.64	5.77	6.34	10
	Population scoring	**7.71**	**7.63**	6.91	7.20	7.5 to 8.3
	End-of-run selection	6.70	7.54	n/a	n/a	n/a
Sonnet 5	Vanilla	4.64	4.64	5.52	5.23	10
	Symmetry rejection	**8.00**	**8.00**	4.12	4.06	8
	End-of-run selection	4.64	4.43	n/a	n/a	n/a

Table 1: Prespecified confirmations in Tiny Hanabi with independently permuted labels, eight new runs per cell, 100 candidate evaluations. Scores range from 0 to 10. Bold marks a Holm-significant improvement over vanilla in the same column. End-of-run selection is computed on the vanilla archives and uses a separate panel of unoptimised samples.

Shared prior: population scoring raised cross-play. In a prespecified confirmation with eight new runs per cell and Haiku 4.5 (Table 1, Figure 3), vanilla search raised self-play to 10 and lowered cross-play between runs from 6.5 at the first candidate to 4.64 for both optimisers. Population scoring kept cross-play at 7.71 with GEPA and 7.63 with OpenEvolve, differences of 3.07 and 2.99 (permutation tests, Holm-corrected $p = 0.008$ for each). The whole-run bootstrap intervals, $[-0.20, 4.90]$ and $[-0.30, 4.88]$, just include zero. Cross-play with fresh unoptimised samples rose by 1.14 and 0.86 (Holm-corrected $p = 0.085$ for each). End-of-run selection on the vanilla archives gave 6.70 and 7.54. An earlier confirmation with partner presentations shared across runs gave the same direction (Section A.6).

Split prior: symmetry rejection preserves cross-play. In a second prespecified confirmation with Sonnet 5 (eight new runs per cell), vanilla GEPA and OpenEvolve both reached self-play 10 with cross-play 4.64, as the runs split between the two maximal conventions. End-of-run selection did not help (4.64 and 4.43), because the unoptimised samples it scores against are themselves split. Symmetry rejection placed all sixteen runs on the invariant convention: cross-play 8.0 for both optimisers, a difference of 3.36 ($[0.69, 5.14]$, Holm-corrected $p = 0.004$ for each). This resembles the trade-off of Other-Play (Hu et al., 2020), although our constraint uses an exact invariance test. Self-play fell from 10 to 8. Cross-play with fresh unoptimised Sonnet samples, which play the maximal conventions, fell from 5.52 to 4.12 for GEPA (Holm-corrected $p = 0.004$) and from 5.23 to 4.06 for OpenEvolve ($p = 0.15$). A confirmation with Fable 5.1, whose samples also split, gave the same result for symmetry rejection with GEPA (cross-play 4.64 to 8.0, Holm-corrected $p = 0.004$). Population scoring with independently presented partners also raised Fable's cross-play, to 7.54 ($p = 0.005$), since against partners that mix both maximal conventions the invariant one is the best response.

Other partner-based remedies. In four-run pilots, the text analogue of Other-Play reached cross-play 8.0 for Haiku 4.5 with both optimisers and for Sonnet 5 with OpenEvolve, but stayed at 5.0 for Sonnet 5 with GEPA. Prior anchoring did not keep Haiku 4.5 runs on the invariant convention. Cross-play reflection, which shows the proposer its failures against training partners, reached 7.96 and 6.96 with Haiku 4.5. With Sonnet 5 and independent partner presentations, it stayed at 4.92 and 5.00 after 64 evaluations, and cross-play with fresh samples fell to 3.44 and 4.22. This matches the split prior: failures against the model's own samples do not single out one convention.

9 Few-shot adaptation

A zero-shot failure matters less if agents recover from it within a few rounds. We played the crowded tower game repeatedly for ten rounds with public outcomes. Each agent adapted between rounds in one of four ways. It kept the raw history in its context, kept a short summary note about the partner, rewrote a partner rule in response to the partner's last choice (*last-move update*), or rewrote the rule as a best response to the partner's empirical choice frequencies (*fictitious-play update*, after Brown (1951)). Notes and rules were limited to 80 words. Ties in the frequency table were broken by the alphabetical order of place names, which both agents see identically; this tie-break is itself

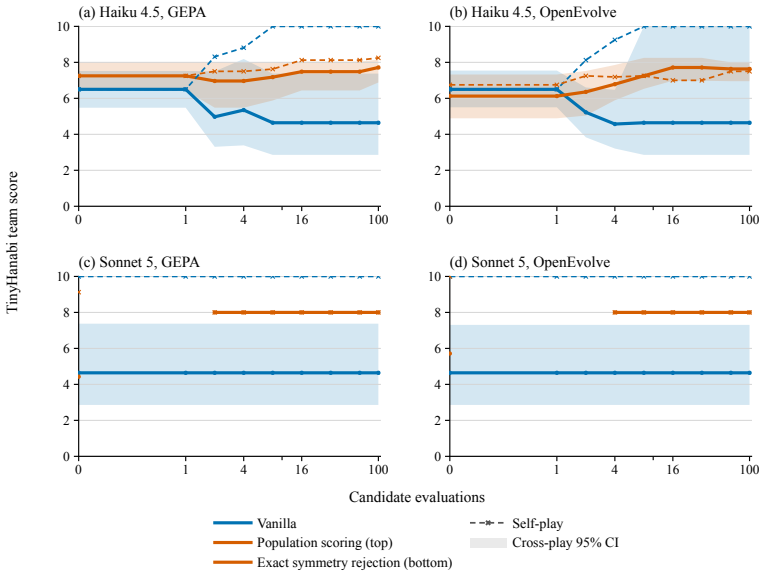

Figure 3: Cross-play between eight independent runs per cell in Tiny Hanabi with permuted labels, as a function of candidate evaluations. Solid lines are cross-play with 95% bootstrap intervals over runs; dashed lines are self-play. Top: Haiku 4.5, vanilla against population scoring. Bottom: Sonnet 5, vanilla against symmetry rejection.

	Mean payoff, rounds 2 to 10				Minus last-move	
Model	Raw	Note	Last move	Fict. play	Raw	Fict. play
Haiku 4.5	0.57	0.60	0.86	0.86	$\mathbf{-0.29}$	0.00
Sonnet 5	0.90	0.81	0.78	0.79	$\mathbf{+0.11}$	$+0.01$
Gemini 3.8 Flash	0.90	0.61	0.65	0.89	$\mathbf{+0.25}$	$\mathbf{+0.24}$

Table 2: Prespecified few-shot confirmation in the repeated crowded tower game. Both agents adapt from incompatible starting cafés. Payoffs range from 0 to 1 per round; Raw is raw history, Fict. play the fictitious-play update. Bold marks differences with Holm-corrected $p < 0.01$ over the six primary contrasts (pair bootstrap). Masked controls without partner history scored 0.00 to 0.08 for Haiku, 0.00 to 0.71 for Sonnet and 0.37 to 0.90 for Gemini.

a shared convention, and breaking ties by first appearance makes symmetric fictitious play cycle. History-masked controls saw no partner choices.

Which update recovers best depends on the model's fallback. In a prespecified confirmation, both agents started from incompatible cafés taken from separate optimisation runs, in 32 pairs per model for Haiku 4.5 and Sonnet 5 and 12 pairs for Gemini 3.8 Flash, with both seat orders and temperature 1.0 (Table 2). Haiku 4.5 recovered fastest with rule updates. Raw history led its agents to swap cafés repeatedly (154 reciprocal swaps in 64 encounters) and lowered the mean payoff in rounds 2 to 10 by 0.29 against the last-move update. Sonnet 5 and Gemini 3.8 Flash returned to the tower even without history: their masked controls reached up to 0.71 and 0.90. For these models the last-move update disrupted that fallback. Raw history beat it by 0.11 and 0.25, and for Gemini the fictitious-play update restored the fallback (+0.24). The fictitious-play update scored similarly to the last-move update for Haiku and Sonnet, and higher for Gemini. All arms ended on a common place in at least 63 of 64 Haiku encounters and 61 of 64 Sonnet encounters; the minimum for Gemini was 21 of 24. An exploratory Haiku pilot at temperature 0 with eight pairs had shown the opposite ordering (Section A.12).

Optimisation changed adaptation little for Haiku. We also optimised the instruction of a history-capable Haiku 4.5 agent with GEPA-style and OpenEvolve-style search, under vanilla self-

play or an adaptation-aware objective, with four searches per cell. We deployed the first and final instruction against stationary partners. Across sixteen combinations of search method, objective and adaptation mechanism, the area under the payoff curve changed by between 0.00 and -0.06. In five combinations the uncorrected interval lay below zero. Final instructions exceeded the per-call token limit in 168 rounds against 10 for first instructions; scoring those rounds as misses raises the loss to -0.20 (Section A.7).

10 BEYOND THE TOWER GAME

A cooperative cooking game. In a Python reimplementation of Overcooked's Coordination Ring layout (Carroll et al., 2019), Opus 5.5 wrote complete Python policies with a 200 ms decision limit. Evaluation used varied held-out starts and both seats. Four vanilla GEPA runs with twelve proposals each raised self-play from 66 to 318 points per episode and cross-play from 115 to 314, with final pair scores from 294 to 320. We observed no payoff-equivalent conventions separating competent runs. Population scoring against four unoptimised Opus policies reached similar cross-play between runs (299) and higher cross-play with eight separate unoptimised policies (197 against 161 for vanilla). Population runs shared one training pool, so their cross-play is not independent; four runs per arm do not support a test. Fifteen of 100 model calls reached the $32{,}000$-token output limit and counted as invalid proposals.

A sequential signalling game. Signal Relay extends the Cat–Dog game of Hu et al. (2021) with hidden hands, changing stacks and limited clue tokens over twelve turns. Players choose between arbitrary free codes and costly literal reveals; Section A.2 gives details. Exploratory diagnostics found presentation-dependent starting conventions in Haiku 4.5, Sonnet 5 and Gemini 3.8 Flash. Four Haiku runs per arm showed that vanilla optimisation and population scoring left the existing split, while exact symmetry rejection produced reveal-only policies with self-play and cross-play 4.97 out of six, costing about 0.4 in self-play.

11 LIMITATIONS

The main evidence comes from small games whose payoffs we evaluate exactly. In the crowded tower, the remedies that keep agents at the tower often reward that known answer. Symmetry rejection needs a verified symmetry group, which may be hard to find in a new task. In the larger games we tested, Overcooked showed no loss of cross-play, and in Signal Relay the split already existed before search. The diagnostic covers two providers, and the remedy confirmations so far use Anthropic models. Our partner panels are base-model samples, not trained policy lineages, and our K-level procedure is sequential rather than joint (Section B). In earlier stages, runs shared some partner presentations (Section A.6); the confirmations reported in Table 1 use independent presentations, except the Sonnet 5 cross-play with fresh base samples. Several comparisons rest on four-run pilots, which we mark as exploratory.

12 CONCLUSION

Text optimisation scores an agent with copies of itself, yet deployed agents meet partners built elsewhere. Language models share focal points, and in two coordination games search destroyed them: independent GEPA and evolutionary code search runs raised self-play and lost compatibility, because each run adopted a convention singled out by its own presentation of the task. Text analogues of deep reinforcement learning methods restored compatibility, and which one worked followed the unoptimised model. Scoring against base samples helped Haiku, whose samples agreed, and Fable, whose samples split; symmetry rejection helped Sonnet and Fable, at a cost in self-play. Few-shot recovery depended on whether the model falls back to a focal point. Open questions are invariance without a known symmetry group and large sequential tasks.

AI USE STATEMENT

Language models are both the subject and a tool of this study. As the subject, Claude Haiku 4.5, Sonnet 5, Opus 5.5 and Fable 5.1 (Anthropic) and Gemini 3.8 Flash (Google) generated all optimised

artefacts and adaptation decisions (Section 4). As a tool, language-model agents (Claude Code with Claude Opus 5.5, and OpenAI Codex with GPT-6 models) were used extensively under the authors' direction: for research methodology and experiment design, including proposing hypotheses, methods such as the remedies tested here, and the prespecified protocols; for implementing the experiment code and tests; for running and monitoring experiments; for statistical analysis and the interpretation of results; for literature search and citation verification; for creating figures; for drafting and editing the text; and for simulated reviews of drafts. The authors set the research questions and made the key decisions. LLM-generated code was checked with unit tests and independent LLM code reviews. Reported numbers were cross-checked against the run archives by independent LLM audits, and every citation was checked for existence and for support of the cited claim by an automated pipeline that reads the cited papers. The authors take full responsibility for the content.

Ethics statement

This work studies coordination among language-model agents in games. It involves no human subjects and no personal data. A possible negative use is that the analysis could help an optimiser select agents that coordinate only with copies of themselves; our recommendation to report cross-play with independent partners works against that use.

Reproducibility statement

Sections A.1, A.3 and A.4 give the game definitions, method settings and statistical procedures, and Section A.5 reports the compute used. Eiffel and Tiny Hanabi fixed policies are evaluated exactly. Signal Relay uses a fixed bank of 512 held-out deals, so its reported cross-play values can be recomputed from archived candidates without model calls. We will release the code, configuration files, prompts and run archives.

References

Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. LLM-Coordination: Evaluating and analyzing multi-agent coordination abilities in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 8053–8072. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-naacl.448. URL https://aclanthology.org/2025.findings-naacl.448/. 1, 2, 3, 19

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, 2026. 1, 3, 19

Juyang Bai and Laixi Shi. MAS-PromptBench: When does prompt optimization improve multi-agent LLM systems?, 2026. URL https://arxiv.org/abs/2606.23664. 20

Nolan Bard, Jakob N. Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H. Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, Iain Dunning, Shibl Mourad, Hugo Larochelle, Marc G. Bellemare, and Michael Bowling. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020. doi: 10.1016/j.artint.2019.103216. 14

George W. Brown. Iterative solution of games by fictitious play. In Tjalling C. Koopmans (ed.), *Activity Analysis of Production and Allocation*, pp. 374–376. Wiley, New York, 1951. 2, 7

Colin F. Camerer, Teck-Hua Ho, and Juin-Kuan Chong. A cognitive hierarchy model of games. *The Quarterly Journal of Economics*, 119(3):861–898, 2004. doi: 10.1162/0033553041502225. 20, 22

Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for

human-AI coordination. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://papers.nips.cc/paper_files/paper/2019/file/f5b1b89d98b7286673128a5fb112cb9a-Paper.pdf. 9, 20

Rujikorn Charakorn, Poramate Manoonpong, and Nat Dilokthanakul. Generating diverse cooperative agents by learning incompatible policies. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=a7vLnGKGIjY. 20

Brandon Cui, Hengyuan Hu, Luis Pineda, and Jakob Foerster. K-level reasoning for zero-shot coordination in hanabi. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/4547dff5fd7604f18c8ee32cf3da41d7-Abstract.html. 20, 22

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. 2, 19

Johannes Forkel, Constantin Ruhdorfer, Michael Beukman, Andreas Bulling, and Jakob Foerster. High entropy regularization leads to symmetry-equivariant policies in Dec-POMDPs, 2025. 18

Lucheng Fu, Ye Yu, Yiyang Wang, Yiqiao Jin, Haibo Jin, B. Aditya Prakash, and Haohan Wang. TextReg: Mitigating prompt distributional overfitting via regularized text-space optimization, 2026. URL https://arxiv.org/abs/2605.21318. 20

Jianxin Gao, Tianyi Yu, Linna Deng, Runze Li, and Zining Wang. Testing interchangeability in LLM agent teams, 2026. 19

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665–673, 2020. doi: 10.1038/s42256-020-00257-z. URL https://doi.org/10.1038/s42256-020-00257-z. 20

Adrian Hayler, Shashank Reddy Chirra, Andrei Lupu, Johannes Forkel, Bidipta Sarkar, Siheng Feng, and Jakob Foerster. Zero-shot coordination among LLM agents. In *ICLR 2026 Workshop on Multi-Agent Learning and Its Opportunities in the Era of Generative AI*, 2026. URL https://openreview.net/forum?id=HHPbQlyA7Y. 1, 2, 4, 5, 19, 21

Hengyuan Hu and Dorsa Sadigh. Language instructed reinforcement learning for human-AI coordination. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, 2023. 18

Hengyuan Hu, Adam Lerer, Alex Peysakhovich, and Jakob Foerster. "Other-Play" for zero-shot coordination. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 2020. 1, 2, 3, 4, 7, 15, 18, 20, 22

Hengyuan Hu, Adam Lerer, Brandon Cui, David Wu, Luis Pineda, Noam Brown, and Jakob Foerster. Off-belief learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 2021. 1, 3, 9, 14, 18, 20, 22

Athul Paul Jacob, David J. Wu, Gabriele Farina, Adam Lerer, Hengyuan Hu, Anton Bakhtin, Jacob Andreas, and Noam Brown. Modeling strong and human-like gameplay with KL-regularized search. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 2022. 18, 22

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan A, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into state-of-the-art pipelines. In *International Conference on Learning Representations*, 2024. 2, 19

Victoria Krakovna, Jonathan Uesato, Vladimir Mikulik, Matthew Rahtz, Tom Everitt, Ramana Kumar, Zac Kenton, Jan Leike, and Shane Legg. Specification gaming: the flip side of AI ingenuity. Google DeepMind blog, 2020. URL https://deepmind.google/blog/specification-gaming-the-flip-side-of-ai-ingenuity/. 20

David K. Lewis. *Convention: A Philosophical Study*. Harvard University Press, 1969. 14, 20

Keane Lucas and Ross E. Allen. Any-play: An intrinsic augmentation for zero-shot coordination. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 2022. 20

Andrei Lupu, Brandon Cui, Hengyuan Hu, and Jakob Foerster. Trajectory diversity for zero-shot coordination. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7204–7213, 2021. URL https://proceedings.mlr.press/v139/lupu21a.html. 20

Judith Mehta, Chris Starmer, and Robert Sugden. Focal points in pure coordination games: An experimental investigation. *Theory and Decision*, 36(2):163–185, 1994. doi: 10.1007/BF01079211. 20

Robert Müller and Clemens Müller. Cattle trade: A multi-agent benchmark for LLM bluffing, bidding, and bargaining. In *ICLR 2026 Workshop on Multi-Agent Learning and Its Opportunities in the Era of Generative AI*, 2026. URL https://arxiv.org/abs/2605.14537. 2

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav 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, 2025. 3, 19

Krista Opsahl-Ong, Michael J. Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9340–9366, 2024. doi: 10.18653/v1/2024.emnlp-main.525. URL https://aclanthology.org/2024.emnlp-main.525/. 2, 19

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. doi: 10.1038/s41586-023-06924-6. 2, 19

Thomas C. Schelling. *The Strategy of Conflict*. Harvard University Press, 1960. 1, 20

Asankhaya Sharma. OpenEvolve: an open-source evolutionary coding agent. Software repository, 2025. URL https://github.com/algorithmicsuperintelligence/openevolve. Version 0.3.2. 3, 19

DJ Strouse, Kevin R. McKee, Matt Botvinick, Edward Hughes, and Richard Everett. Collaborating with humans without human data. In *Advances in Neural Information Processing Systems*, 2021. 2, 3, 18, 22

Johannes Treutlein, Michael Dennis, Caspar Oesterheld, and Jakob Foerster. A new formalism, method and open issues for zero-shot coordination. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10413–10423. PMLR, 2021. URL https://proceedings.mlr.press/v139/treutlein21a.html. 4

Zhexuan Wang, Xuebo Liu, Li Wang, Zifei Shan, Yutong Wang, Zhenxi Song, and Min Zhang. MASPO: Joint prompt optimization for LLM-based multi-agent systems. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.06623. 2, 19

Maksymilian Wolski, Nicholas Hoernle, Johannes Forkel, and Jakob Foerster. Is inter-seed cross-play enough? evaluating the robustness of zero-shot coordination algorithms to implementation details, 2026. 18

Table 3: Tiny Hanabi payoff vectors by cards and signal.

c_A	c_B	$a_1 = 0$	$a_1 = 1$	$a_1 = 2$
0	0	$(10, 0, 0)$	$(4, 8, 4)$	$(10, 0, 0)$
0	1	$(0, 0, 10)$	$(4, 8, 4)$	$(0, 0, 10)$
1	0	$(0, 0, 10)$	$(4, 8, 4)$	$(0, 0, 10)$
1	1	$(10, 0, 0)$	$(4, 8, 4)$	$(10, 0, 0)$

Peng Xia, Rujun Han, Zifeng Wang, Yanfei Chen, Yufan Zhuang, Yoonho Lee, Chengsong Huang, Han Yu, Zhongying CuiZhu, Yifei Ming, Huaxiu Yao, Burak Gokturk, Tomas Pfister, and Chen-Yu Lee. RRSI: Regularized recursive self-improvement of agent harnesses, 2026. 21

Xue Yan, Jiaxian Guo, Xingzhou Lou, Jun Wang, Haifeng Zhang, and Yali Du. An efficient end-to-end training approach for zero-shot human-AI coordination. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/hash/07a363fd2263091c2063998e0034999c-Abstract.html`. 20

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*, 2024. 2, 19

Chao Yu, Jiaxuan Gao, Weilin Liu, Botian Xu, Hao Tang, Jiaqi Yang, Yu Wang, and Yi Wu. Learning zero-shot cooperation with humans, assuming humans are biased. In *International Conference on Learning Representations*, 2023. 20

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639:609–616, 2025. doi: 10.1038/s41586-025-08661-4. 2, 19

Rui Zhao, Jinming Song, Yufeng Yuan, Haifeng Hu, Yang Gao, Yi Wu, Zhongqian Sun, and Yang Wei. Maximum entropy population-based training for zero-shot human-ai coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023. doi: 10.1609/aaai.v37i5.25758. 20

A Appendix

A.1 Game definitions and label regimes

In the Eiffel game, two players independently chose one of ten places and received a shared payoff only when their choices matched. The places were Café Girard, Café Vasseur, Café Morel, Café Brunet, Café Perrin, Café Roux, the Eiffel Tower, Café Fournier, Café Collin, and Café Gauthier. In the crowded variant, matching at the tower paid 0.9 and matching at any café paid 1; in the open variant, matching at Café Morel paid 0.9 and matching at any other place, including the tower, paid 1. Mismatches paid 0. The game used named labels in a shared or shuffled order, neutral shuffled labels (`spot A` through `spot J`), or a fixed order shuffled by a presentation seed. The core Eiffel grid used shuffled named labels.

Tiny Hanabi used two uniformly drawn binary cards. Player A observed c_A and sent one of three signals a_1; player B observed a_1 and c_B and chose a_2. Table 3 gives the shared payoff. Each entry lists payoffs for $a_2 = 0, 1, 2$ in that order. The payoff tensor admitted eight payoff-preserving relabellings of cards and actions. The two mirror conventions encoded the two A cards with signals 0 and 2 in opposite orders, then decoded the signal and B's card to select action 0 for equal cards and action 2 for unequal cards. Each earned 10 in self-play. The invariant convention sent signal 1 and chose action 1 at every B information state, earning 8. These were the three focal deterministic convention classes. The shared regime kept the canonical labels; the permuted regime independently shuffled the two card labels and three action labels, giving $2!\,3! = 12$ possible presentations. The core Tiny Hanabi grid used the permuted regime.

A.2 SIGNAL RELAY

Signal Relay is a stateful extension of the Cat–Dog game of Hu et al. (2021). It keeps the choice between a free signal with arbitrary meaning and a costly literal message, uses two free codes, and adds changing hands, stacks, lives and clue tokens. This signalling problem follows Lewis (1969); the card and resource mechanics draw on Hanabi (Bard et al., 2020). We built it to test whether independently optimised policies remain compatible when actions have consequences across turns, while keeping the policies small enough for an exact symmetry test. The restricted policy interface tests convention choice in a sequential game, rather than general planning from a full history.

Rules. Two players hold two cards each and see their partner's cards but not their own. The shuffled deck has 16 cards: four rank-1 cards, three rank-2 cards and one rank-3 card in each of two colours, red and blue. Each colour has a stack that starts at zero. The players start with three shared lives and two shared clue tokens, with a token cap of two. Played and discarded cards, stack heights, lives, tokens, turn number and messages are public.

Players alternate as actor for at most twelve turns. Before each move, the other player sees the actor's hand and sends one of five messages. Codes A and B are free and mechanically identical, with no prescribed meaning. The message `pass` is also free and asserts nothing. The message `reveal0` or `reveal1` costs one token and recommends playing the indicated slot; it is legal only if a token is available and that card is currently playable. The actor then chooses `play0`, `play1`, `discard0` or `discard1`. A play succeeds only when the card's rank is exactly one above its colour's stack, which then rises by one. A failed play discards the card and loses one life without restoring a token. A discard restores one token up to the cap. The removed slot is refilled from the deck without moving the other slot. The twelve cards left after the initial deal provide all replacements, so both hands contain two cards at every decision. Play ends after twelve actor turns, loss of all three lives, or completion of both stacks. The shared reward counts successful placements and ranges from zero to six.

Table policy and presentation. One deterministic table controls both roles and both players. The signaller receives only the token count $t \in \{0, 1, 2\}$ and a playable mask $m \in \{0, 1, 2, 3\}$. Mask 0 means neither actor slot is playable, 1 means only slot 0, 2 means only slot 1, and 3 means both. The actor receives only the message and the token count after any message cost. It never receives its hidden cards or the mask. The table therefore has 27 information states: twelve signal entries and fifteen move entries. It has no memory of earlier turns.

The artefact is a single Python literal assignment, `POLICY = {...}`, containing a 3×4 signal array indexed by token count and mask, and three move dictionaries with five message keys each. All entries are required, including entries not reached in play. A fixed parser accepts literal data only and rejects executable code, duplicate keys, missing or extra entries, unknown actions and illegal reveals. A trusted adapter executes table lookups without model calls during play. Invalid proposals consume the proposal budget and are not repaired.

The prompt displays the free codes as `amber` and `violet`. Their display order and their name-to-code mapping are drawn independently for each run. Thus the first displayed name can denote either internal code. We archive the presentation map and both the raw and canonical tables, mapping displayed names back to A and B before evaluation.

Certified symmetries and exact rejection. The certified subgroup is $G \cong C_2^3$, the eight combinations of three independent swaps: exchange A and B; exchange red and blue throughout the deck, hands and stacks; and exchange player identities together with the initial actor. These transformations preserve transitions and reward. We do not claim that this subgroup contains every game symmetry. Colour and seat swaps leave the abstract table unchanged. The code swap exchanges signal outputs A and B and exchanges their input entries in each move dictionary, while fixing slots, reveals and actor actions.

For a parsed canonical policy π, exact rejection accepts only if $g\pi = \pi$ for every $g \in G$, comparing all 27 entries. Equivalently, no signal entry may output A or B, and the move responses to A and B must agree at every token count. A rejected proposal receives the specific changed entry in its displayed names. Exact rejection disables the free code channel by construction. Literal reveals

and pass remain available, but invariance alone does not ensure competent play. This deterministic restriction is stronger than the symmetry-based training objective of Other-Play (Hu et al., 2020).

Scripted design check. Before model optimisation, we evaluated two opposite code policies and a literal policy on 512 fixed held-out deals with both initial actors. AB interprets A as slot 0 and B as slot 1; BA reverses these meanings. The code policies pass when neither slot is playable. The literal policy reveals a playable slot when a token is available and otherwise passes; its actor follows reveals and discards on pass. The prespecified primary variant chooses slot 0 when both cards are playable and discards slot 0 on pass. Its AB and BA self-play scores were both 5.320, and each mixed pairing scored 1.368. The mixed-play drop was 3.952, above the required 0.8, and the self-play difference was zero, below the allowed 0.3. Literal self-play was 4.971, above the required 1.5. All three design thresholds passed. These are scripted properties, not evidence of optimisation-induced loss.

Tie slot	Discard slot	AB self	BA self	AB with BA	BA with AB	Literal self
0	0	5.320	5.320	1.368	1.368	4.971
0	1	5.373	5.373	1.340	1.340	5.021
1	0	5.375	5.375	1.389	1.389	5.067
1	1	5.319	5.319	1.387	1.387	4.950

Table 4: Signal Relay scripted checks. The four tie and discard variants were fixed before scoring; none was selected after inspecting scores. Each pairing averages both initial actors.

Exploratory base diagnostics. Before search, we sampled 64 valid independent step-zero policies per model, half with the first listed code name denoting A and half denoting B. Evaluation used the held-out bank of 512 deals with both initial actors. Table 5 reports cross-play between distinct base samples with the same or different presentation in this sense. Reveal share is the fraction of messages that are literal reveals. Starting conventions depended on presentation in all three models.

The classification rule, fixed before pilot outcomes, calls a model's prior shared only if different-presentation cross-play is at least same-presentation cross-play minus 0.5, and at least 75% of samples use the same canonical AB or BA convention. Otherwise it is split. All three models were classified as split. These thresholds are descriptive rules, not significance tests, and these measurements concern unoptimised policies only.

Model	Same presentation	Different presentation	Reveal share
Haiku 4.5	5.29	4.62	0.38
Sonnet 5	4.98	2.92	0.15
Gemini 3.8 Flash	5.26	2.97	0.16

Table 5: Exploratory Signal Relay base diagnostics before optimisation. Cross-play scores are out of six. Each model has 32 valid policies per presentation class.

During development, a presentation bug made the first listed name always denote the same code. All 64 Haiku samples then used that name for slot 0. The bug was fixed by drawing list order and code mapping independently. This exploratory observation is consistent with selection by list position in Section 7, but does not by itself distinguish position from code identity.

Exploratory protocol. The Haiku comparison uses GEPA with four independent runs per arm, read at sixteen proposals. Vanilla scores each candidate with its own copy. Population scoring averages both seat orders against four frozen step-zero policies from the same model, sampled separately for each run. Exact symmetry rejection uses the vanilla score after the invariant-table test above. Proposals are scored on 64 fixed training deals per run with both initial actors. Each run deploys its highest scoring valid candidate so far, breaking ties by earliest discovery. Invalid and rejected proposals count toward the budget.

Final evaluation uses a shared bank of 512 held-out deals, frozen before search, with both initial actors. We measure self-play S, cross-play between distinct runs C_{ind}, and cross-play with each run's separate fresh base panel C_{base}, using the same deals for all comparisons and both seat orders

for cross-play. Bootstrap intervals resample whole runs. Training, selection and final-test partners are separate. The corrected v3 seed scheme derives separate seeds for presentations, deals and partner panels with SHA-256 and checks that independent runs do not share presentation seeds.

A.2.1 Exploratory results

The Haiku runs stopped at 18 to 24 proposals because of a token cap. Proposal 16 was the last checkpoint every run reached. Table 6 reports the exploratory comparison. Four runs per arm do not support a test.

Arm	Self-play	Cross-play between runs
Vanilla	5.37	4.85 (4.26 to 5.37)
Population scoring	5.33	5.02 (4.63 to 5.36)
Exact symmetry rejection	4.97	4.97

Table 6: Exploratory Haiku Signal Relay results at proposal 16, with four runs per arm. Scores are out of six on 512 held-out deals with both initial actors. Parentheses give bootstrap intervals where reported.

Vanilla cross-play with fresh base policies was 5.01, and its cross-play between runs at proposal zero was 4.88. Vanilla and population runs each split three to one between the two free codes used when no clue token remained. Pairs with matching codes scored about 5.3, and pairs with different codes scored about 4.3. In this exploratory Haiku pilot, the split was already present in the base model; the pilot did not detect a change in it, and population scoring against base partners did not remove it. All four exact rejection runs converged to the reveal-only policy. Rejection removed the split at a self-play cost of about 0.4 relative to vanilla.

For Sonnet 5 and Gemini 3.8 Flash, exploratory single-run tests with eight proposals per arm showed the same pattern for exact rejection, with self-play 4.97 and a reveal-only policy. These single runs do not measure cross-play between runs.

A.3 Optimisation methods and settings

Sonnet 5 ran with its provider-default adaptive thinking, and thinking tokens counted towards the output limit. Each run began with a model response to the public task and output contract. The Eiffel prompt described the ten locations, the matching rule and, when disclosed, their scores, and requested one displayed place. The Tiny Hanabi prompt gave the four card pairs and their three payoff vectors per signal and requested a complete JSON action table for both players. This seed response was the step-zero candidate. Both optimisers then used a budget of 100 candidate evaluations per run and temperature 1.0. The core pilot planned four runs per cell, and cells cut short by token caps are reported as incomplete; the Haiku population and Sonnet symmetry confirmations used eight. Haiku 4.5 used a 4,096-token response limit. Sonnet 5 used 16,000 in the core pilot and 32,000 in the core completion and confirmation stages.

GEPA 0.1.4 used its Pareto candidate selector and a one-example shuffled batch sampler. OpenEvolve 0.3.2 used one island, a population and archive capacity of 1,000 each, full-program proposals, and sequential evaluations. The default objective was self-play. Population scoring (`fcp_style`) instead averaged the two seat orders against four independently sampled step-zero training partners; it did not construct a growing fictitious self-play archive. Exact symmetry rejection tested whether the parsed policy was unchanged by every member of the eight-element Tiny Hanabi payoff group, or by generators swapping each equal-payoff Eiffel option with a reference option. A noninvariant proposal received a rejection verdict and a relabelling explanation.

At each checkpoint, the deployed candidate was the valid proposal with the highest recorded objective score so far; the earliest proposal broke ties. A separate terminal analysis reranked all valid candidates against the selection panel, again breaking ties by earliest discovery, and evaluated that choice on the final-test panel. Each run had three separate panels of four step-zero samples: training for panel-based objectives, selection for terminal reranking, and final test for base-partner evaluation.

Table 7: Recorded model tokens for the listed stages (millions), as of 25 September. Core and families stages were measured from per-run call logs, the other stages from the shared token ledger. Later few-shot and Signal Relay stages are excluded.

Stage	Model	Tokens
Core pilot (Eiffel and Tiny Hanabi, both optimisers)	Haiku 4.5, Sonnet 5	26.7
Core completion (Tiny Hanabi, OpenEvolve)	Haiku 4.5, Sonnet 5	16.0
Families expansion (vanilla, K-level, cognitive hierarchy)	Haiku 4.5	10.0
Haiku population confirmation	Haiku 4.5	6.0
Sonnet symmetry confirmation	Sonnet 5	7.4
Sonnet cross-play reflection (v3b, reported at 64 evaluations)	Sonnet 5	8.0
Overcooked, vanilla and population	Opus 5.5	2.1
Few-shot pilot (including an invalid first run)	Haiku 4.5	2.4
Few-shot before-and-after experiment	Haiku 4.5	13.7

A.4 Statistical analysis

Cross-play averaged directed payoffs between distinct optimisation runs, with both seat orders represented. Confidence intervals resampled whole runs with replacement (2,000 draws); repeated copies of a sampled run did not create self-pairs. Treatment comparisons permuted whole-run labels, exactly when the assignment count was at most 5,000 and by sampled assignments otherwise. Paired endpoint tests swapped the initial and final labels within whole runs. Holm correction was applied within each declared primary comparison family, including the two-optimiser confirmation families. Analyses used the same best-so-far rule at synchronous candidate counts $0, 1, 2, 4, 8, 16, 32, 64, 100$; a candidate absent at a checkpoint remained missing rather than being replaced by a later one.

A.5 Recorded compute

Table 7 reports the recorded tokens per stage, including partner construction, rejected proposals and invalid calls. As of 26 September, the shared token ledger recorded 261.4 million Haiku 4.5, 112.4 million Sonnet 5, 5.4 million Opus 5.5, 5.4 million Gemini 3.8 Flash and 1.3 million Fable 5.1 tokens across the project, including development and smoke tests. The stage table is a snapshot of the listed stages and excludes later few-shot and Signal Relay work.

A.6 Partner-presentation overlap

The completed full stages used consecutive run seeds, while the original panel builder added the split offset and sample index to each run seed. Presentation seeds therefore repeated across runs. The audit found that between 87.3% and 93.8% of samples in the listed full core and confirmation stages shared a presentation seed with a sample from another run in the same split. Tiny Hanabi had only 12 distinct label permutations, so independent seeds could still yield the same displayed mapping. The defect was the repeated seed itself.

The zero-token presentation test measured mean base-sample cross-play of 9.30 for matching presentations and 4.67 for different presentations with Sonnet 5. The corresponding Haiku 4.5 values were 6.81 and 6.79. The Haiku presentation dependence was negligible, so we retained the Haiku results, as were Sonnet vanilla and exact symmetry-rejection cross-play between independently optimised runs: neither objective trained against the panels. We did not report Sonnet prior-anchoring, population-scoring, or cross-play-reflection effects as independent-run operator effects, because their training partners were correlated across runs. Sonnet final-test base-partner and terminal-reranking estimates also depended on the stage's presentation mix, so their intervals were not presented as independent-panel intervals. The corrected builder derived each presentation seed from SHA-256 of the run seed, split, and sample index, versioned the seed scheme in the panel cache identity, and checked cross-run seed uniqueness before model calls.

A.7 Few-shot invalid rounds

The supplied few-shot analysis contained 178 invalid decision rounds among 6,790 analysed rounds: 10 at step zero and 168 at the final checkpoint. It recorded 698 over-ceiling calls among 25,795 calls; an over-ceiling update or candidate call was not itself an invalid decision round. When a prompt exceeded the per-call token limit, the attempted call produced no model response and the decision was marked invalid. The primary few-shot payoff and area-under-curve summaries excluded invalid decision rounds. A separate sensitivity calculation assigned those rounds zero payoff, while resampling paired blocks of run and partner rather than individual rounds or calls.

A.8 Prior-agreement diagnostic

For each model we sampled 36 unoptimised Tiny Hanabi policies, three under each of the twelve label presentations, with the same prompt and parser as the partner panels. All samples were valid. Table 8 reports the mean exact cross-play between pairs of samples that saw the same presentation and pairs that saw different presentations, and the number of samples whose play is unchanged under the relabelling group. Opus, Fable and Gemini split evenly (18 and 18) between the two maximal conventions. A first Haiku confirmation, run before the presentation-seed fix of Section A.6, gave population scoring 8.0 against vanilla 4.29 with GEPA (Holm-corrected $p = 0.004$) and 8.0 against 7.50 with OpenEvolve ($p = 0.82$), where six of eight vanilla runs reached the same maximal convention.

Table 8: Prior agreement of five unoptimised models in Tiny Hanabi (scores from 0 to 10).

Model	Same presentation	Different presentation	Behaviourally invariant	Self-play 10
Haiku 4.5	7.11	6.81	34/36	2/36
Sonnet 5	9.10	4.94	1/36	31/36
Opus 5.5	10.00	4.55	0/36	36/36
Fable 5.1	10.00	4.55	0/36	36/36
Gemini 3.8 Flash	10.00	4.55	0/36	36/36

A.9 Background and prior methods

A.9.1 Zero-shot coordination

In a cooperative game all players share one payoff. Many such games have several equally good ways to play, which we call conventions. A convention is useful only if the partners agree on it. Zero-shot coordination asks for agents that agree with partners they have never played with, where the partners were produced independently by the same procedure (Hu et al., 2020). The standard measurement pairs agents from independent training runs and compares their joint score, called cross-play, with the score each agent obtains with a copy of itself, called self-play.

Self-play training tends to fail this test. When several conventions are equally good, nothing in the self-play objective prefers one of them, and each run settles on whichever it reaches first. Runs then break the tie in different ways and fail when paired. Other-Play removes the incentive to rely on arbitrary choices by training against copies of the agent whose view is relabelled by the symmetries of the game (Hu et al., 2020). Off-belief learning prevents actions from carrying arbitrary hidden meaning (Hu et al., 2021). Population methods train a single agent to work with many partners (Strouse et al., 2021). Regularising a policy toward a shared prior, such as a model of human play or a language model, is a further route to compatible behaviour (Jacob et al., 2022; Hu & Sadigh, 2023), and, under exact tabular policy-gradient dynamics, sufficiently strong entropy regularisation makes independent runs converge to the same symmetry-equivariant policy (Forkel et al., 2025). Wolski et al. (2026) test whether inter-seed cross-play understates cross-implementation difficulty and find no gap for Other-Play.

A.9.2 TEXT OPTIMISATION

Text optimisation improves an agent without changing model weights. A language model proposes a prompt, instructions or policy code. A score and past traces guide later proposals. Methods differ in what they edit and which candidates they keep. GEPA reflects on evaluation traces and keeps a Pareto front of candidates that are best on at least one instance (Agrawal et al., 2026). Evolutionary code search keeps a scored archive and asks the model to change or combine programs. Our implementation follows OpenEvolve, an open version of the AlphaEvolve approach (Novikov et al., 2025; Sharma, 2025). Best-of-N draws independent candidates and keeps the best. It serves as our selection control.

These methods are usually evaluated by the score they optimise. When the artefact describes behaviour in a cooperative game and the score is obtained by letting the artefact play with itself, text optimisation is self-play in the sense above. The question of zero-shot coordination therefore applies to it directly.

A.9.3 LANGUAGE-MODEL AGENTS AND COORDINATION

Language-model agents start with a language prior that may provide common cues. Unoptimised agents have coordinated with unfamiliar reinforcement-learning and behaviour-cloned partners in Overcooked and Hanabi (Agashe et al., 2025). We interpret shared task cues as one possible reason, rather than a mechanism established by that study. When a team of language-model agents builds up its own conventions in memory over repeated play, swapping members between teams raises the cost of coordination (Gao et al., 2026). We study private conventions that arise when an optimisation loop only scores an agent against its own copy.

A.10 ADDITIONAL POSITIONAL-MECHANISM ANALYSES

Drift needs an incentive and exploration. The positional rule explains where runs go. A run moves only if a private option scores higher in self-play and search reaches it. In the open variant, the tower already scores highest and no run moves. Haiku sometimes samples a café without iterative rewriting, so even best-of-N selection drifts. Sonnet 5 does not sample a café in that condition. Iterative rewriting moves it. When agents must learn payoffs from scores, Haiku still drifts, but later. Cross-play falls from 0.90 to 0.14 and 0.07 for GEPA and OpenEvolve in eight new runs each, with $p = 0.015$ and 0.009. Sonnet 5 proposes no café in that setting and shows no drift. In these experiments, a strong prior preserves coordination when search does not reach the better private option.

Implication. In the crowded game, optimisation selects a rule that depends on each run's displayed order. Agents with different lists can therefore disagree. The open game gives no self-play incentive to leave the tower. In that game, our earlier experiments show little drift. The evidence supports this mechanism when a higher paying private option is available and search reaches it. It does not show that text optimisation always favours private rules.

A.11 RELATED WORK IN DETAIL

FunSearch evolves code with an automatic evaluator (Romera-Paredes et al., 2024). OPRO proposes solutions from earlier candidates and scores (Yang et al., 2024), and PromptBreeder evolves both task prompts and mutation prompts (Fernando et al., 2024). DSPy and MIPRO optimise instructions and examples across language-model programs (Khattab et al., 2024; Opsahl-Ong et al., 2024). TextGrad uses language feedback to revise text variables (Yuksekgonul et al., 2025). These methods establish ways to improve a task score. We ask whether independent searches using such scores leave agents compatible with new partners. GEPA (Agrawal et al., 2026) is our reflective search instance; our other iterative instance is evolutionary code search (Novikov et al., 2025; Sharma, 2025). Wang et al. (2026) jointly optimise interacting prompts. Our agents are optimised in separate runs and paired only at evaluation.

Hayler et al. (2026) study zero-shot coordination among language-model agents and propose coordination-friendly definitions. We test what happens to cross-play as independent text searches revise agents. Agashe et al. (2025) test language models with unfamiliar partners but do not follow

this search process. Bai & Shi (2026) study prompt optimisation inside fixed multi-agent workflows. Their unit is a workflow; ours is a pair of agents from separate searches. Fu et al. (2026) regularise prompt updates against task distribution shifts. Our shift is the partner and, in some games, its presentation.

Other-Play averages over known symmetries (Hu et al., 2020). Off-belief learning grounds past actions under a fixed policy (Hu et al., 2021). Cui et al. (2021) train levels of response together for Hanabi, while Camerer et al. (2004) model players who respond to mixtures of less sophisticated players. Our text analogues use exact small-game scores, frozen levels, and sampled base policies. They do not inherit the original training guarantees. TrajeDi builds a diverse population using trajectory differences (Lupu et al., 2021); MEP uses a population entropy reward (Zhao et al., 2023); and LIPO deliberately learns incompatible policies to cover more partner behaviours (Charakorn et al., 2023). We test whether a fixed training panel helps text search, but base samples do not reproduce those trained populations. HSP models hidden human preferences (Yu et al., 2023). Any-Play targets partners trained by different algorithms (Lucas & Allen, 2022), and E3T learns to adapt with a partner model (Yan et al., 2023). Our main endpoint is immediate play between independently optimised agents, before online adaptation.

Carroll et al. (2019) show that self-play strength need not give strong human partnerships in Overcooked. Our fixed-choice games expose the chosen convention directly; they do not establish the same effect in Overcooked. Schelling introduced focal choices in coordination games (Schelling, 1960). Mehta et al. (1994) test how people use salient labels, and Lewis (1969) analyses how conventions support repeated coordination. Our tower game makes a familiar label salient, then gives a private alternative a higher self-play payoff. This is related to specification gaming and shortcut learning, where measured success may miss the broader goal (Krakovna et al., 2020; Geirhos et al., 2020). Here the measured self-play score can improve while compatibility with an unfamiliar partner falls.

A.12 Exploratory few-shot recovery pilot

Recovery depends on how agents adapt. In an exploratory pilot, we paired eight archived agents from separate optimisation runs, each committed to a different café, and forced their first choice. When both agents adapted, raw history led 15 of 16 encounters to end on the Eiffel Tower, with a mean payoff of 0.91 after eight observed rounds. A frozen instruction with an editable partner rule, rewritten after every round, ended on the tower in one of 16 encounters. Seven encounters never settled, and pairs swapped conventions 4.4 times on average against 2.5 under raw history. All three mechanisms beat their masked controls (mean gain over rounds 2 to 10 between $+0.39$ and $+0.55$). Against a partner that never changed, raw history followed the partner within one or two rounds. When both agents rewrite explicit partner rules, each chases the other's last choice, while raw history lets the pair return to the shared focal point.

A.13 The open-tower contrast

In the crowded tower the model's prior and the label-free solution coincide: the Eiffel Tower is both the obvious place and the unique option that does not depend on labels. Remedies that pull toward the prior and remedies that remove label dependence therefore predict the same outcome. The *open tower* variant separates them. There the Eiffel Tower pays 1.0 together with eight cafés, and one unremarkable café is the unique option paying 0.9. The prior still points to the tower, while the label-free solution is now the unremarkable café.

A.14 Tower-game remedies and critic pilots

Remedies that change the score. *Prior anchoring* subtracts from self-play a penalty proportional to the total-variation distance between a candidate's choice and a panel of unoptimised samples. *Symmetry-permuted evaluation*, the analogue of Other-Play (Hu et al., 2020), scores a candidate against copies of itself under the symmetries of the game. *Archive scoring* replaces self-play by the score against a fixed pool of partners, either unoptimised samples or a pool that favours one café. In the crowded game, several remedies reward the known answer: the tower is both the model's prior choice and the unique option fixed by the game's symmetries. Each remedy that rewards the tower

removes erosion for Sonnet 5 (cross-play among runs $0.06 \rightarrow 0.90$, twelve runs per condition), at the price of self-play ($0.98 \rightarrow 0.90$). For Haiku 4.5, anchoring behaves as a threshold: a penalty weight of 0.1 exactly offsets the café's self-play gain and runs still drift, while weights of 0.3 and above keep every run on the tower.

Anchoring and Other-Play separate in the open tower. In the crowded tower the prior's choice and the label-free solution coincide, so anchoring and symmetry-permuted evaluation cannot be told apart. In the open tower they differ: the tower remains the prior's choice, while the only option fixed by the payoff symmetry is the unremarkable café paying 0.9. In an exploratory four-run Haiku 4.5 pilot, symmetry-permuted evaluation moved every run to that café, whereas anchoring kept every run on the tower. With Sonnet 5, the tested configuration never proposed the rewarded café in 1,200 proposals, although the tower scored only $1/9$ under permuted evaluation. This score change did not move Sonnet's strong prior in that configuration. The observed limit is proposal coverage; the record does not isolate whether more detailed feedback would change it.

Remedies that change the information. Telling the optimiser that its agent will meet an independently produced partner halves the erosion for Sonnet 5 (cross-play 0.40 against 0.08 for a matched generic instruction) and has no effect for Haiku 4.5. We therefore designed a remedy that supplies a reason. Inspired by regularised self-improvement of agent harnesses (Xia et al., 2026), a *convention critic* screens every proposal before it is scored. Its rubric follows the aim of the coordination-friendly definitions proposed by Hayler et al. (2026); here it is enforced on proposals inside the optimisation loop. It is told only that an independently written partner may see the options in a different order and under different labels, and it rejects a proposal, with an explanation returned to the optimiser, if the proposal's gain depends on something the shared content does not single out. On a calibration set the critic rejected the first café of a run's own list, an arbitrary café and a letter label, and allowed the Eiffel Tower and the unique 0.9 option in both the named and the neutral game (30 of 30 verdicts correct). In eight new runs per condition with Sonnet 5, the critic kept every run on the tower and raised cross-play from 0.14 in a matched control to 0.90 (uncorrected permutation $p = 0.0002$).

Limits of the critic. In Tiny Hanabi with permuted labels, a pilot of four runs suggested a large effect, which did not replicate in eight new runs (cross-play 5.4 against 4.3, $p = 0.42$). An audit showed why: the critic accepted signalling codes such as "signal 1 means card 0" as "supported by the payoff matrix", although a symmetry of the game makes the reverse code equally good. A language-model critic recognises arbitrariness that is visible, such as list positions, letters and equally described options, but not arbitrariness hidden in the symmetries of a payoff table.

Constraints, not reasons. To separate a reason from a constraint, we built a critic that needs no language model. It tests each proposal exactly against the verified symmetries of the game and, if the proposal is not invariant, states which relabelling makes it arbitrary and which invariant alternatives exist. Given only as information, with the score unchanged, this explanation had no measurable effect in a four-run Tiny Hanabi pilot with Haiku 4.5 (cross-play 3.3). Used as a constraint that rejects non-invariant proposals, the same test is the symmetry-rejection method of Section 8.1.

A.15 Hidden payoffs and robustness

Drift does not require stated payoffs. If payoffs are never written in the task and are only learned from scores, Haiku 4.5 still drifts: in eight new runs per method, cross-play among runs falls from 0.90 to 0.14 for GEPA and to 0.07 for OpenEvolve ($p = 0.015$ and 0.009), although runs leave the tower later (evaluation 2 to 40 instead of 1). Sonnet 5 under hidden payoffs never proposes a café and never drifts. Drift therefore requires two ingredients: information that a better private convention exists, and exploration that reaches it. A strong prior protects coordination only while the better alternative stays unseen.

Robustness. Final cross-play is unchanged when ties among equally scored candidates are broken by the latest discovery or at random instead of the earliest. The erosion is also robust to the wording of the task: in three prespecified paraphrases, all twelve GEPA runs with Sonnet 5 left the tower (cross-play 0.00 to 0.17).

B FIDELITY OF THE METHOD ANALOGUES

Table 9 states which part of each method we transfer. We do not claim that the text analogues inherit the guarantees of their reinforcement learning sources. Candidate evaluations use the exact game rules in the fixed-policy games. A sample from the base model is a training partner, not a trained policy lineage.

Method	Original method	Text analogue	Privileged information	Extra cost
Prior anchoring	KL penalty toward a reference policy in piKL (Jacob et al., 2022)	Total-variation penalty to sampled base choices	Base sample panel	Panel calls and scoring
Other-Play	Train under known game symmetries (Hu et al., 2020)	Score candidates after verified relabellings	Exact symmetry group	More game evaluations
Population scoring	Fictitious Co-Play best-responds to independently seeded self-play partners and their past checkpoints (Strouse et al., 2021)	Score against frozen step-0 samples	Training partner panel	Partner generation and pair scores
Cross-play reflection	No single original; partner training is the closest link	Show the proposer failures against a frozen partner panel	Training partners and verified relabellings	Partner evaluations and reflection calls
Symmetry rejection	Other-Play changes the objective (Hu et al., 2020)	Reject candidates that fail an exact invariance test	Exact symmetry group	Invariance checks and lost proposals
OBL	Ground past actions under a fixed policy; continue with improved policy (Hu et al., 2021)	Exact grounded beliefs for past actions; candidate continuation	Fixed grounding policy and game model	Extra game scores
K-level	Train levels together; each responds to the lower level (Cui et al., 2021)	Train frozen levels in sequence with a split proposal budget	Level-zero policy	Separate search at each level
Cognitive hierarchy	Respond to a mixture of lower levels (Camerer et al., 2004)	Score each level against frozen lower-level mixture	Level-zero policy and mixture weights	Separate search and mixture scores

Table 9: Method fidelity and costs. The OBL text analogue is exploratory. The K-level and cognitive-hierarchy analogues are described here for method fidelity; their exploratory results are in the labbook archive and are not reported as findings.

C OFF-TASK PROMPT OPTIMISATION

Self-play is one kind of optimisation pressure. Developers also optimise agents for jobs they do alone, such as analysing a financial statement. A specialist may later need to work with an agent from another developer. The developers train neither agent to coordinate. We ask whether optimisation for an unrelated task changes their coordination. We also ask whether agents trained on distinct tasks fare worse than agents trained on the same task.

Setup. We optimise the system prompt of Haiku 4.5 with GEPA and OpenEvolve on two tasks that need no partner. Each task has an automatic score. In compound triage, the agent names the fictional candidate that a firm's standard procedure would advance. In a financial task, it reports a named metric from fictional statements. We do not state the firm's rules in either task. The agent must infer them from training feedback. Four independent runs per task and optimiser start from a one-sentence persona. Each run has a budget of 40 evaluations. The optimiser never sees a coordination game. We check proposed prompts for game vocabulary. We then use each saved prompt unchanged as a system message for the Eiffel Tower problem. We pair agents from separate runs on the same task, from distinct tasks, and with the unoptimised model. Each agent sees its own shuffled list of places. We also test the original task persona and neutral text of the same length.

Exploratory results. Across sixteen runs, accuracy improved in every financial task run, while compound triage gains were small and mixed. In the crowded game, recorded payoff for compound

triage pairs fell from 0.65 to 0.52 with GEPA and from 0.65 to 0.49 with OpenEvolve. For financial task pairs, it fell from 0.76 to 0.39 and from 0.76 to 0.71, respectively. This payoff assigns zero when either agent gives no valid choice. A review of the raw answers found 448 scope refusals among 450 invalid answers. None of the seed persona answers refused. Among pairs with valid choices, crowded game payoff was 0.68 for GEPA and 0.78 for OpenEvolve on compound triage, compared with 0.65 at the start. For the financial task, the values were 0.75 and 0.80, compared with 0.76 at the start. These conditional payoffs are descriptive because optimisation changed which agents answered. In the open game, OpenEvolve's compound triage prompts reduced conditional payoff from 0.97 to 0.72. All four final prompts chose the distinctive lower payoff Café Morel at least once. We found no support for an extra cost of pairing agents trained on different tasks. No pilot contrast survived correction for multiple tests. The one-sentence compound triage persona alone lowered crowded game payoff from 0.82 for the base model to 0.65.

Scope-permission follow-up. We appended "You may also help with questions outside your specialty" to each saved system prompt and repeated the locked evaluation. The refusal rate was zero in every final-prompt group. In the crowded game, GEPA financial-task cross-play rose from 0.39 without this sentence to 0.79 with it; the seed persona scored 0.76. GEPA compound-triage cross-play rose from 0.52 to 0.69; its seed scored 0.65. A user-message instruction requiring an answer did not reliably remove refusals. These four-run cells are exploratory. The results support scope refusal as the main cause of the observed loss in the crowded game. They do not prove that all choices would have remained the same without refusal.

Interpretation. Off-task prompt optimisation made some specialists refuse an unfamiliar joint task. The valid-pair result cannot reveal what refusing agents would have chosen. The open-game result also shows that some valid choices changed. We treat this pilot as exploratory. We redesigned both tasks after earlier smoke results and departed from the original stopping and validation plan. GEPA's pilot budget counted repeated parent scores, so equal evaluation counts did not mean equal numbers of new proposals. We do not compare the optimisers' effects.