Zero-Shot Coordination among LLM Agents

Adrian Hayler[*], **Shashank Reddy Chirra**[*], **Andrei Lupu**,
Johannes Forkel, Bidipta Sarkar, Siheng Feng, Jakob Foerster
BOLD, University of Oxford

Abstract

We study zero-shot coordination (ZSC), where independently developed agents must coordinate at test time. While ZSC has been well studied in the RL literature, far less is known about the performance of LLM agents despite their increasing deployment in such settings. Existing work on LLM coordination often relies on specialised scaffolding that independently developed agents are unlikely to share in practice. Moreover, evaluations on complex environments (e.g., Hanabi) make it difficult to pinpoint the sources of coordination failure. In contrast, we focus on simple, general-purpose scaffolds in minimal environments designed to isolate specific coordination challenges. Our results show that even in these controlled settings, frontier LLM agents struggle to coordinate, largely due to limited understanding of the coordination problem and weak reasoning about their partner's beliefs. While LLMs introduce *semantic information* as an additional axis for coordination, they nonetheless fail to exploit this structure effectively. Towards this, we propose *Coordination-friendly definitions (CFDs)* as a principled approach for enabling robust coordination among LLM agents. Finally, we show that CFDs can be discovered automatically, removing the need for manual engineering. Code is available at `https://github.com/ahayler/zsc-llm-agents`.

1 Introduction

Rapid advances in AI have led to the deployment of autonomous agents in environments that require robust coordination with humans and other independently developed agents [9, 29, 41, 24]. While humans often readily coordinate with unseen partners by reasoning about their behaviour, referred to as *theory of mind (ToM)* [47], autonomous agents struggle to do the same [36, 12]. A classic real-world example is that of autonomous driving, where vehicles developed independently by companies such as Waymo, Tesla, and Wayve must coordinate during on-road interactions [29].

With the rapid growth in large language models (LLMs) capabilities, LLM agents are increasingly deployed in real-world applications that require coordination with other independently developed agents. For example, in AI-assisted coding software such as Cursor, agents powered by different LLM providers are used together to collaboratively generate code [33]. Similar settings also arise in robotics [28, 32, 53] and autonomous driving [27], where LLM agents generate policies as code that must coordinate with independently developed partners at deployment.

This problem is formally known as Zero-shot coordination (ZSC) [24], where agents are trained independently and evaluated together at test time without prior communication or re-training. Although ZSC prohibits agents from sharing policies, it permits prior agreement on a game-agnostic learning rule used during training. The central objective of ZSC is therefore to identify learning rules that encode general coordination principles applicable across a wide range of settings, while avoiding common failure modes. For example, the other-play [24] learning rule ensures that the policy is equivariant with respect to the symmetries of the environment.

[*]Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

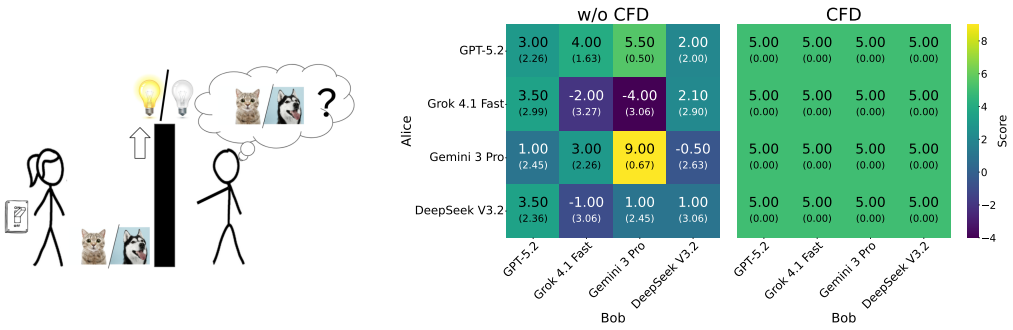


Figure 1: **Cat & Dog coordination game.** *(Left).* Alice observes an animal (cat/dog). She can either bail and end the game for a joint reward of 1, remove the wall for a cost of 5, or use the light bulb to signal for free. Bob either observes the state of the light bulb or the animal if the wall is removed. He can guess the animal (+10/-10) or bail for a reward of 0.5. *(Right)* XP of text agents in the variable LLM setting with and without CFD. Each cell contains the average score of 10 episodes. Adding the CFD leads the agents to consistently pick the safe fallback for a guaranteed reward of $+5$.

ZSC literature has predominantly focused on traditional deep reinforcement learning (RL) training pipelines [24, 23, 40]. More recently, however, the emergence of LLMs has given rise to a new class of autonomous agents [41, 42, 43] that differ substantially from this paradigm. Rather than requiring a large number of environment interactions [14], LLM agents can coordinate zero-shot from a textual description of the environment via in-context learning (ICL), without any task-specific training. These textual descriptions expose an additional axis for coordination through the *semantics* of the environment, a signal that has proven difficult for traditional RL agents to exploit [37]. As illustrated in Figure 5, agents can use semantic cues (e.g., the "red arrow") to coordinate their actions.

We present the first systematic study of LLM agents in the ZSC setting, beginning with a formalisation of two of the most common scaffolds: text agents and code agents. We then introduce a suite of simplified cooperative environments, inspired by prior work [13, 23, 35, 25], each designed to isolate a specific coordination challenge. We augment these environments with text descriptions to support interaction with LLM agents. Together, these environments cover a broad range of challenges an agent is likely to encounter in the real world. Consequently, we posit that strong performance on this suite is **necessary but not sufficient** for robust real-world coordination.

We find that both classes of agents struggle to coordinate reliably in these simplified environments, driven in part by a weak understanding of the underlying coordination challenge and weak ToM abilities. Next, we find that agents have a strong bias toward leveraging *environment semantics* for coordination. This can yield improvements over traditional RL agents that struggle to exploit such cues; however, semantics also introduce coordination strategies that compete with existing ones, and agents that fail to consistently arbitrate between the two see degraded coordination. Overall, our results highlight the need for a new perspective on coordination with LLM agents.

Finally, inspired by the ZSC principle of a shared learning rule among coordinating entities, we introduce the notion of a *Coordination-friendly definition (CFD)*, a natural language specification of general coordination principles shared across agents. This mirrors analogous approaches in other domains, such as AI safety, where high-level rules are codified as a "constitution" to promote adherence to common safety standards [5, 26]. We show that incorporating a CFD improves coordination for both text and code agents across a range of environments, and that meta-learned principles within the CFD can outperform hand-crafted ones. Taken together, these results position CFD as a principled mechanism for enhancing coordination in LLM agents and a promising direction for future research.

2 Background and Notation

Dec-POMDPs We focus on fully cooperative settings which can be represented via the Decentralised Partially Observable Markov Decision Process (Dec-POMDP) framework [8, 46]. A Dec-POMDP $\mathcal{M}$ consists of M agents, each assigned a role $r \in \mathcal{R} = [M]$, a state space $\mathcal{S}$, action space $\mathcal{A}$, per-agent observation spaces $\mathcal{O}_i$, transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$, discount factor $\gamma \in [0, 1]$ and horizon $H \in \mathbb{N}$, which represents the maximum length of an episode. Agents share a common re-

Gemini 3 Pro (Alice)

```
There is no inherent reason to prefer one over the other, which makes this risky.  However, usually,
"Action" (Light On) maps to "Option A" (Cat) and "No Action" (Light Off) maps to "Option B" (Dog).
Let's reconsider the remove_wall option.  A guaranteed 5 is very safe.  But a potential 10 is much
better.  Is there a way to be sure?  No.  But if I am an AI playing with another AI or a rational
player, we likely try to maximize the 10.
```

GPT-5.2 (Bob)

```
Alice deliberately turned the light on as a one-bit signal of which animal is present.  In a
cooperative setting, the natural convention is "light on = dog" (active warning/action), so I should
guess dog to try to secure the +10 joint reward.
```

Figure 2: **Reasoning traces from a failed coordination in the *Cat and Dog* environment.** Gemini 3 Pro and GPT-5.2 independently arrive at incompatible strategies.

ward function $R : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$. Each agent observes the environment via its observation function $Z_i : \mathcal{S} \to \Delta(\mathcal{O}_i)$. The action space can be written as a product $\mathcal{A} = A_1 \times \cdots \times A_M$, where A_i corresponds to agent i's action space. A (joint) policy π is an M-tuple of local policies $\pi_i : \mathcal{H}_i \to \Delta(A_i)$, where $\mathcal{H}_i$ is the space of possible action-observation histories (AOHs) of player i, each of the form $\tau_t^i = \{o_0^i, a_0^i, \ldots, a_{t-1}^i, o_t^i\}$. The joint return is defined as $J(\pi) = \mathbb{E}_\pi[\sum_{t=0}^{H} \gamma^t R_t]$. We assume access to natural language descriptions $\mathcal{D} = (\mathcal{D}_\mathcal{M}, \mathcal{D}_\mathcal{R}, \{\mathcal{D}_{\mathcal{H}_i}\}_{i\in[M]})$ of the Dec-POMDP. $\mathcal{D}_\mathcal{M} \in \Sigma^*$ is a description of the Dec-POMDP, with Σ^* denoting all strings over a suitable finite alphabet Σ. $\mathcal{D}_{\mathcal{H}_i} : \mathcal{H}_i \to \Sigma^*$ and $\mathcal{D}_\mathcal{R} : \mathcal{R} \to \Sigma^*$ are injective functions, describing each AOH τ_i and role r in natural language. We use the term *role* as a synonym for player index; the distinction is purely terminological and reflects how player identity is communicated to LLM agents via $\mathcal{D}_\mathcal{R}$.

We define an LLM agent as a system that leverages an LLM and in-context learning (ICL) to generate a local or joint policy π for a Dec-POMDP. Crucially, ICL does not involve a separate training phase with gradient-based parameter updates; instead, all the "learning" occurs through conditioning on the prompt during the model's forward pass. We consider two classes of LLM agents as illustrated in Figure 15: **text agents** and **code agents**. Text agents are assigned to a role $r \in \mathcal{R}$, take as input $(\mathcal{D}_\mathcal{M}, \mathcal{D}_\mathcal{R}(r), \mathcal{D}_{\mathcal{H}_i}(\tau_i))$ and return an action a_t^i, whereas code agents take as input $\mathcal{D}_\mathcal{M}$ and return a joint policy π as executable code. Since a joint policy specifies the behaviour of all players simultaneously, code agents require no role assignment via $\mathcal{D}_\mathcal{R}$; the roles, together with their observations and actions, are already specified in $\mathcal{D}_\mathcal{M}$. A worked example of $\mathcal{D}$ for Cat & Dog is given in Appendix D.2. Hence, code agents induce a distribution over joint policies. In contrast, text agents do not produce an explicit joint policy; instead, they act on an implicit local policy π^r assigned to their role. As a result, their behaviour is harder to interpret or modify, and prompting an LLM for each action makes them slower and more expensive to evaluate.

Zero-shot Coordination The standard approach for training fully cooperative multi-agent RL systems is self-play (SP) [50, 54], wherein agents are trained alongside the same counterparts they will encounter at test time. In contrast, the ZSC setting [24] assumes that independently trained policies are paired together at test time without prior communication or adaptation, a setting known as cross-play (XP). To enable coordination, entities are willing to coordinate on (part of) a Dec-POMDP-agnostic learning rule for training their agents. Examples of ZSC learning rules make use of known game symmetries [24] or explicitly control for how agents reason about their partner [23].

3 Problem Formulation

Consider a Dec-POMDP with two LLM agents i and j, and write π_i^r for the local policy of agent i assigned role $r \in \{1, 2\}$. Code agents independently generate joint policies π_i, π_j, with the SP score given as $\mathbb{E}_{\pi_i}[J(\pi_i)]$ and $\mathbb{E}_{\pi_j}[J(\pi_j)]$. Their XP score is defined as

$$J(i, j) = \tfrac{1}{2}\mathbb{E}_{(\pi_i,\pi_j)}\left[J(\pi_i^1, \pi_j^2) + J(\pi_j^1, \pi_i^2)\right] \tag{1}$$

Equation 1 also applies for text agents where assigning agent i to role r induces the local policy π_i^r.

We consider three evaluation variants. **Self-play** (SP) pairs a text agent with itself; for code agents, it is the score of the generated joint policy. The **fixed LLM** setting (XP_{seed}) applies only to code agents,

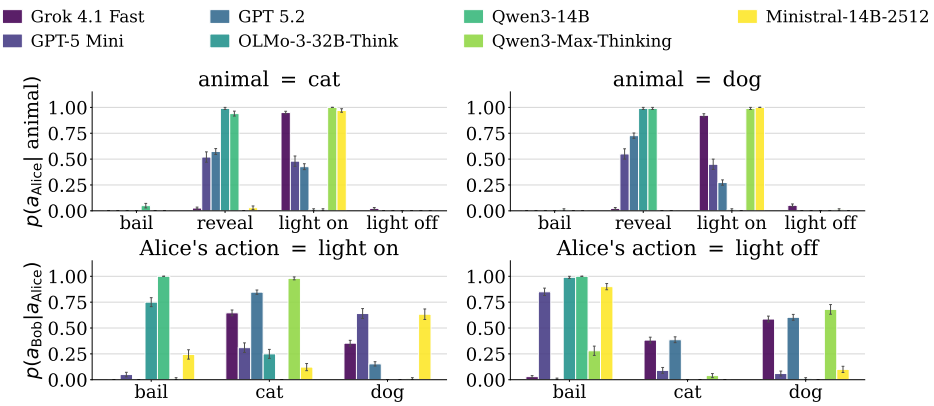


Figure 3: **Local policies induced by text agents on Cat & Dog.** Most agents induce degenerate signalling policies; e.g, Qwen3-Max-Thinking and Ministral-14B-2512 turn the light on regardless of the observed animal. Qwen3-14B and OLMo-3-32B-Think induce the ZSC-optimal policy.

which induce a distribution over joint policies: two policies are sampled independently from the same agent and evaluated under XP, measuring compatibility within a single distribution. The **variable LLM** setting (XP_{LLM}), applicable to both code and text agents, instead pairs two distinct LLM agents under XP. A schematic of the evaluation regimes is provided in Figure 16 in the Appendix.

We treat a single ICL generation as the analogue of a training run in tabula-rasa RL. For code agents, one generated joint policy therefore corresponds to one self-play training run, and pairing independently generated joint policies mirrors standard XP evaluation [24]. For text agents, which only induce local policies, the distinction is less clear-cut. We refer to pairing two instances of the same LLM with the same prompt (but different roles) as SP, reflecting that providers may optimise models to work well with instances of themselves, e.g., as subagents within a shared harness. As we show in Figure 4, even such local policies are not guaranteed to be mutually compatible.

4 ZSC performance of LLM agents

It is natural to ask whether LLM agents coordinate effectively out of the box, rendering further study unnecessary. Towards this, we conduct experiments across nine diverse coordination problems. Many of these are adapted from the ZSC literature, including Cat & Dog [23], Tiny Hanabi [13] and a variant of the lever game [23, 55]. Our choice of environments is motivated by two reasons: (a) each environment **isolates a core coordination capability** under ZSC evaluation in simplified settings, enabling a clean assessment of whether LLM agents exhibit the targeted behaviour; and (b) collectively, they span a diverse set of challenges that arise in real-world coordination. These include, for example, selecting a safe fallback in the absence of shared conventions, forming conventions at test time, and achieving coordinated symmetry breaking. A detailed description of the environments and the challenges they capture is provided in Appendix D. We emphasise that strong performance across these environments constitutes a **necessary but not sufficient condition** for effective real-world coordination. For instance, the narrow tunnel environment isolates the problem of tie-breaking, an issue that has been observed in deployed autonomous driving systems such as Waymo [12]. If LLM agents are used to generate navigation policies for autonomous vehicles [27], failures in this environment would likely translate to analogous failures in the real world.

To ensure a fair evaluation, we include an explanation of the XP evaluation framework used, as well as provide explicit instructions to the agents to optimise their policies for high XP scores. The prompt templates used for both types of agents are reported in Appendix G.3. While we evaluate agents across all environments, we conduct a detailed analysis on the Cat and Dog toy environment [23] and a simplified variant of Hanabi [6], a widely used coordination benchmark. As we show later, coordination failure modes observed in Cat and Dog persist in the more complex game of Hanabi as well. In Cat & Dog, as shown in Figure 1, there exist two incompatible conventions (light on $\Rightarrow$ cat and light on $\Rightarrow$ dog), both of which achieve the maximum reward of 10. However, without any prior communication, the (ZSC-optimal) strategy is to remove the wall, yielding a reward of 5. For code agents, we employ SOTA reasoning models, namely Grok 4.1 Fast, GPT-5.2, Gemini 3 Pro, Claude

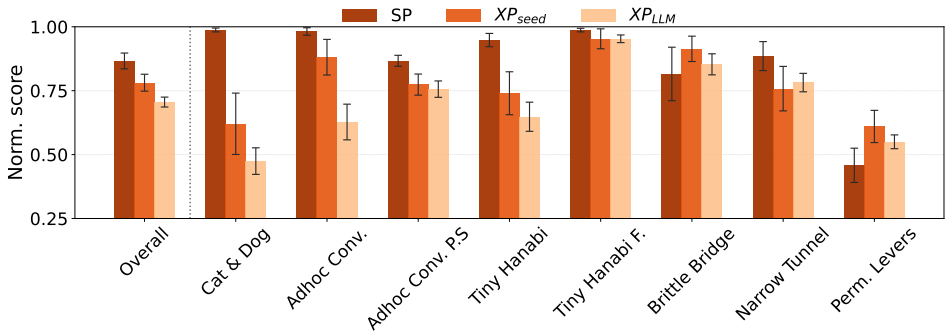


Figure 4: **SP and XP scores of code agents, aggregated across LLMs.** Code agents exhibit a clear SP-XP gap, indicating that independently generated policies use incompatible strategies.

Opus 4.5 and DeepSeek V3.2. Due to cost reasons, we evaluate text agents in the SP setting across all games using GPT-5.2 and Grok 4.1 Fast, and restrict the variable LLM setting to Cat & Dog.

Text agents As shown in Figure 1, text agents coordinate poorly in Cat and Dog, resulting in XP scores well below the safe guaranteed reward of 5 obtained by removing the wall. We find that Gemini 3 Pro consistently adopts the convention *light on* $\Rightarrow$ *cat*, enabling it to coordinate with itself. Since this convention is arbitrary, it leads to poor coordination with other agents. An example failure case is shown in Figure 2, which records the reasoning traces from an episode between Gemini 3 Pro and GPT 5.2. When acting as Alice, the Gemini agent *recognises* that any coordination convention is *arbitrary*, yet nevertheless selects one to avoid the -5 penalty and subsequently rationalises this choice as "natural". In contrast, when acting as Bob, the GPT agent independently rationalises the opposite convention to be "natural"; importantly, it does not consider bailing as an option.

Next, we estimate the empirical local policies of text agents by repeatedly prompting them 100 times from a fixed game state, as illustrated in Figure 3. As Alice, we observe that Grok, Ministral, and Qwen3-Max consistently turn the light on regardless of the observed animal, yielding a degenerate signalling policy. GPT models exhibit similar behaviour, though with lower probability, opting to reveal in the remaining cases. We observe similarly degenerate behaviour as Bob. Ministral tends to bail when the light is off but guesses "dog" with high probability when the light is on, whereas Qwen3-Max guesses "cat" when the light is on and "dog" otherwise. Overall, their behaviour is largely arbitrary and degenerate. Surprisingly, smaller models such as Qwen3-14B and OLMo-32B-Think recover the optimal ZSC policy: revealing as Alice and bailing when observing a signal as Bob, despite being weaker than frontier models like Grok-4.1-Fast, GPT-5.2, and Qwen3-Max-Thinking at mainstream tasks such as Math and Coding. Notably, Qwen3-Max-Thinking ($\sim$1T parameters) performs significantly worse than its smaller counterpart, Qwen3-14B. We observe a similar pattern for smaller models such as Grok 4.1 Fast and GPT-5 mini (see Table 3 in the Appendix), where increased reasoning effort correlates with lower coordination performance due to more frequent hallucinations of a shared convention. However, the relationship is mixed: for frontier models such as Claude Opus 5 and Grok 4.5, higher reasoning effort increases the reveal rate, whereas the newer GPT-5.6 Sol reveals far less often than GPT-5.2 (Appendix F.6). Greater model capability or reasoning effort therefore does not reliably translate into stronger ZSC.

Next, we examine how semantics, specifically, textual descriptions of the environment, affect the agents' local policies. To this end, we consider two reformulations of the Cat & Dog game. First, we rename the signalling actions from "light on/off" to "red/blue light." Second, we translate the entire environment description $\mathcal{D}$ into German. The latter reflects settings where coordinating agents may access the same environment (e.g., a website) in different languages. Note that the payoffs in each of these cases is unchanged. We find that semantics can significantly influence the agents' local policies as shown in Figure 18 (Top-Right and Bottom-Left) in the Appendix. For instance, GPT models become more grounded, choosing to reveal with higher probability when we relabel the actions to red light and blue light. In contrast, Qwen3-14B and OLMo-3-32B exhibit a reduced tendency to reveal, with Qwen3-14B increasingly opting to bail as Alice-an effect that is further amplified under the German relabeling. Meanwhile, Grok, Ministral, and GPT-5-Mini continue to exhibit degenerate policies across these modified environments. These results indicate that the **models are unable to robustly reason about the game and are arbitrarily influenced by *semantics***.

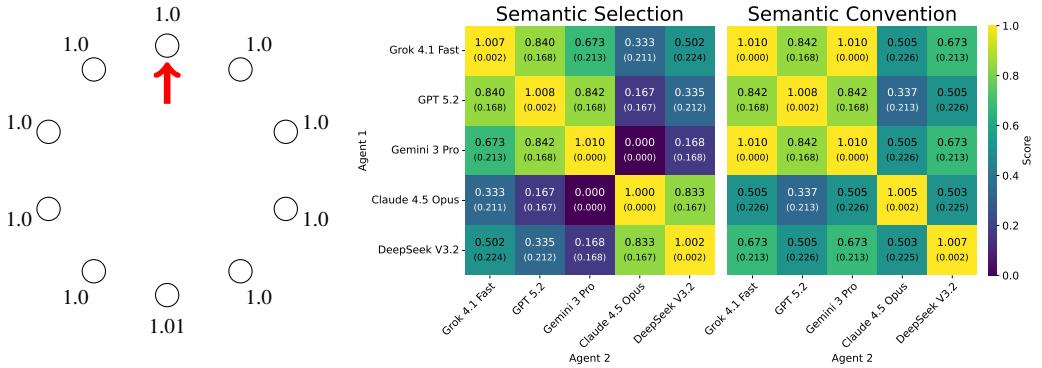


Figure 5: *(Left)* **Semantic Selection game.** Agents could either coordinate on the semantic focal point (top) or Dec-POMDP-based focal point (bottom). *(Right)* **XP scores of code agents on both games.** The SP-XP gap reveals that agents inconsistently choose between the two strategies.

As discussed in Section 1, successful coordination requires theory of mind (ToM) [47], since agents must reason about which conventions, if any, their partners are likely to adopt. In the absence of an established convention, a well-calibrated agent should maintain uncertainty over how its partner will interpret a signal [23]; selecting the ambiguous signal anyway reflects a failure of this reasoning. We note, however, that failures in Cat & Dog do not by themselves causally separate weak ToM from a failure to recognise that no shared convention exists, and we view both as contributing factors. We observed that the agents are strongly averse to incurring the penalty for revealing, which pushes them to adopt an arbitrary signalling convention (also seen in Figure 2). Moreover, agents rationalise their chosen convention as the unique "natural" convention, something that the Gemini agent exploits for successful coordination (Figure 1). We test whether this is a general strategy that other agents can also exploit. For this, we add the following instruction to the prompt: "You are playing a cooperative game with an identical instance of yourself. The same language model is used for both players, but each instance is assigned a different role in the game." If the model has a consistent notion of what it considers a "natural" convention, it should be able to coordinate on that convention and thereby achieve the maximum reward. That said, as shown in Figure 18 (Bottom-Right) in the Appendix, agents are unable to exhibit such capabilities. Instead, coordination performance deteriorates. For example, GPT-5.2, which initially revealed in about 75% of cases, now adopts a degenerate policy of always turning the light on, as Alice, revealing in only about 5% of cases. Similarly, Qwen3-14B and OLMo-32B-Think reduce their reveal rates to around 50% when observing a dog, down from over 95%. As Bob, they also shift toward guessing the animal more frequently, rather than bailing as they did previously. These results suggest that **agents behave as though a consistent internal notion of a "natural" convention exists; however, in practice, no such consistency is observed**. Finally, we find that coordination failures in text agents are not limited to the Cat & Dog environment. As summarised in (Table 1 in the Appendix), Grok 4.1 Fast and GPT 5.2 exhibit high failure rates in $7/9$ tasks. Since these simplified environments are designed to isolate coordination failure modes, we examine whether the same failure modes persist in more complex settings. This helps validate our claim that strong performance in simplified environments is **necessary but not sufficient** for real-world coordination. We evaluate text agents on a 3x3 variant of Hanabi [6], of which Cat & Dog serves as an abstraction. The game mechanics are described in Appendix D in the Appendix, and results are shown in Figure 20 in the Appendix. We find that most failures stem from mismatched assumptions of shared conventions, closely mirroring our findings in Cat & Dog. For instance, a predominant failure mode arises when the clue receiver assumes the "Good Touch Principle" (i.e., that all hinted cards will be played), while the clue giver does not (Figure 32 in the Appendix).

Code agents We evaluate the SP and XP performance of code agents both in the fixed LLM ($\text{SP}, \text{XP}_{\text{seed}}$) and variable LLM setting (XP_{LLM}) and report the aggregated results in Figure 4, providing detailed results in Appendix F.3. Similar to text agents, code agents although *aware* of the safe reveal strategy are averse to the penalty associated with revealing. Adding to this we observe a strong bias toward adopting conventions based on *semantics*, primarily via *lexicographical ordering*. Yet, the agents frequently fail to agree on the ordering itself. For instance, assumptions such as cat$<$dog and on$<$off make mappings like cat $\rightarrow$ light on a common choice. However, many agents instead interpret light off/on as a binary signal, equating it with $0/1$, which yields light off=cat resulting in

Hand-crafted CFD

An action/policy is *not coordination-friendly* if it satisfies *any* of the following:

- It assumes the existence of arbitrary conventions. To assess this, ask:
  - Is it using a signalling convention that could be replaced by a different, equivalent convention?
  - Is there a role assignment that could be flipped? (This applies when two players are symmetric.)
- *Note:* Most actions/policies rely on some convention-judge whether that convention is arbitrary or essential.

Otherwise, the action/policy is coordination-friendly.

Before suggesting an action/policy, evaluate the criteria above; if any apply, the action should be avoided.

Meta-learned CFD

Prefer actions that would still be reasonable under several plausible partner interpretations of the shared observation, and avoid actions that are good only if a partner infers one specific unstated rationale.

Figure 6: **Hand-crafted and Meta-learned CFDs.**

coordination failure. More broadly, lexicographical ordering is a brittle ZSC coordination rule. For example, if agents use textual descriptions from different languages (as discussed in Figure 4), the ordering may invert. For example, *Hund* (dog) precedes *Katze* (cat) in German.

Similar coordination failures are observed across the remaining environments. This stems from code agents generating joint policies that achieve high self-play returns but coordinate poorly with other independently generated policies. Across both fixed and variable LLM settings, we observe a significant gap relative to SP returns (see Figure 4). Qualitative analysis (Appendix H) further indicates that these policies are brittle and sheds light on the dominant failure modes across environments. Although the evaluation settings of code and text agents are not directly comparable, we qualitatively find that code agents are markedly more likely to resort to such brittle policies, while text agents exhibit more robust reasoning. Overall, our experiments demonstrate that **both sets of agents perform poorly across the evaluated environments**. This finding is particularly important in light of recent work deploying both text and code agents in real-world ZSC settings [32, 53, 27].

5 Semantics and Coordination

LLM agents rely on textual descriptions of the Dec-POMDP to access the environment, thereby introducing *semantics* as an additional axis for coordination. Semantics have important implications for coordination, enabling agents to leverage semantic relationships between observations and actions that have been difficult for traditional RL methods to exploit [37]. Moreover, semantics introduce new coordination strategies that compete with those grounded in the structure of the Dec-POMDP. As discussed in the previous section, LLM agents exhibit a strong bias toward leveraging semantic features for coordination; here, we investigate this more systematically, using environments specifically designed to isolate the role of semantics in coordination.

Exploiting semantics for coordination Code agents achieve near-perfect XP returns on Tiny Hanabi Fixed, despite its close similarity to the Cat & Dog setting, which features a safe fallback and two symmetric but incompatible Nash equilibria. Inspection of the reasoning traces reveals that agents independently converge on a unique convention induced by the *lexicographic* mapping $(A \mapsto 1, C \mapsto 2)$ (see Figure 34 in the Appendix). Traditional ZSC algorithms, which cannot exploit such semantic structure, instead default to the safe fallback and incur lower rewards [13, 23].

To further probe this ability, we evaluate text and code agents on the Hint–Guess game [37] (Appendix D), a task where success requires exploiting semantic relationships between observations and actions and where agents can coordinate through human-like reasoning patterns such as mutual exclusivity [38] and implicature [17] (Figure 14 in the Appendix). Text agents perform strongly in this environment (Figure 20 in the Appendix), matching the human baselines reported in [37] and adopting the same coordination strategies as human players (Table 8 in the Appendix); however, they occasionally suffer coordination failures that are unlikely to arise in human play (Figure 41

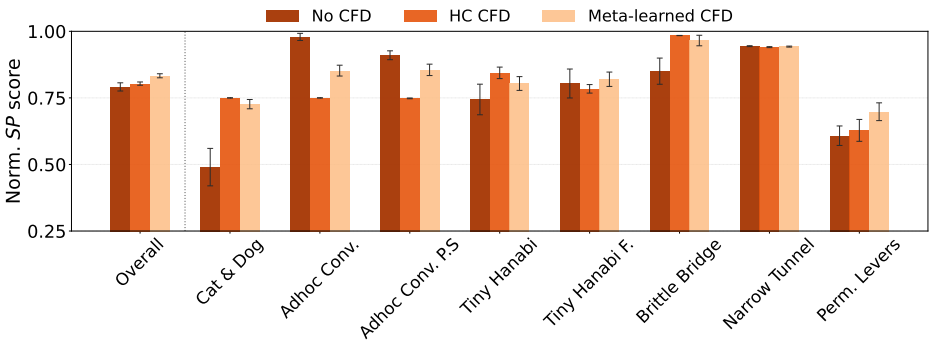


Figure 7: **Self-play performance of Grok 4.1 Fast as text agent.** Incorporating the meta-learned CFD outperforms the hand-crafted CFD, which in turn improves over the baseline.

in the Appendix). Code agents, in contrast, perform poorly (Figure 20 in the Appendix): rather than grounding coordination in semantic features, they fall back on arbitrary conventions such as lexicographic sorting or positional indexing (Figure 43 in the Appendix), and none of the resulting policies successfully leverage mutual exclusivity or implicature. This is consistent with our findings in Figure 4, where code agents perform worse than text agents across the range of coordination tasks.

Semantics introduce competing strategies Conversely, code agents achieve lower XP scores in the Tiny Hanabi environment, despite the presence of a unique optimal strategy $(A \mapsto 2, C \mapsto 1)$. Analysis of the reasoning traces reveals that semantics introduce an additional coordination strategy based on lexicographic ordering, as discussed above, albeit with a lower joint return. The resulting competition between these strategies degrades XP performance, whereas traditional ZSC algorithms, which do not exploit such semantic structure, are not affected [13].

To further investigate this competition, we introduce the *Semantic Selection Game* (Figure 5) and the *Semantic Convention Game* (Appendix D). Broadly, agents must choose between two competing and incompatible coordination strategies: coordinating on the uniquely SP-optimal joint policy, or coordinating based on the semantic relationship between the observations and actions. For example, as shown in Figure 5, agents can either coordinate by selecting the lever with reward 1.01 or by choosing the lever marked with a "big red arrow". Figure 5 demonstrates that the LLM agents fail to resolve this competition consistently, resulting in low XP scores. We find that Gemini 3 Pro is the only model that reliably selects the SP-optimal policy across all games. In contrast, the other LLM agents alternate between the two competing strategies. We find that Claude Opus 4.5 is particularly biased toward semantic alignment, selecting the semantic-aligned joint policy in all six runs of the semantic selection and in three out of six runs of the semantic convention game.

Avoiding local optima Direct reasoning about the environment allows LLM agents to identify optimal strategies that are difficult for traditional RL methods to discover through Monte Carlo rollouts, which are often prone to local minima. As shown in Appendix F.4, LLM agents reliably discover *self-explaining deviations* in the trampoline–tiger coordination challenge [25], while widely used MARL algorithms such as MAPPO [59] and QMIX [49] fail to uncover these solutions.

6 Coordination Friendly Definitions

The results in Figure 4 and Section 5 show that LLM agents fail even in simple settings, highlighting the need to adapt LLM agents for zero-shot coordination. Towards this, we propose a simple approach inspired by the core principle underlying ZSC algorithms. Prior work [24] motivates the idea of a ZSC learning rule as follows: "suppose that multiple independent AI designers will construct agents that have to interact in various but ex-ante unknown Dec-POMDPs without being able to coordinate beforehand, what learning rule should these designers agree on?" In the context of LLM agents, we introduce the analogous notion of a coordination-friendly definition (CFD). A CFD comprises a set of textual instructions that encode general coordination principles used to guide agent behaviour. We assume that independent developers can agree on a shared CFD to facilitate coordination among their

agents. Given the diversity of scaffolds, including the text and code agents described in Section 2, we emphasise that CFDs should remain simple to ensure compatibility across implementations.

The CFD must influence the behaviour of the LLM agent, for example, by shaping the induced local policies of text agents or the joint policy generation process of code agents. In this work, we focus on the most straightforward application: incorporating the CFD directly into the prompt. We also briefly explore a specialised scaffold for code agents that introduces a policy critic to modify or reject generated joint policies based on compliance with the CFD. More advanced approaches, such as reinforcement learning from human feedback (RLHF) [52, 4] based on CFD compliance (similar to [5]), remain a promising direction for future work but are beyond the scope of this work.

Hand-crafted CFD We experiment with a simple hand-crafted CFD as shown in Figure 6, which is based on preventing arbitrary conventions and role assignments [24, 55]. As shown in Figure 1 and Table 1, Table 6 and Table 7, the CFD improves the performance of both agents in the Cat & Dog environment, where agents now consistently choose to remove the wall. We find that the CFD has a stronger effect on text agents and smaller models, reducing the coordination failure rate in $6/9$ environments for Grok 4.1 Fast and $5/9$ for GPT-5.2 (Table 1 in the Appendix). That said, the CFD can be too restrictive in some cases. In the ad-hoc conventions environments, the CFD biases the agents toward the safe fallback, preventing them from forming conventions at test time.

The effect of the CFD is less pronounced for code agents, where it fails to reduce the SP-XP gap beyond the Cat & Dog and the Semantic Selection Game environments. We attribute this to the generally weaker coordination abilities of code agents, as documented in Figure 4 and Section 5. Further examination of execution traces revealed that, while code agents adjust their generated policies in response to the CFD, these policies rarely account for edge cases, leading to coordination failures. Taken together, these results show that **conditioning LLM agents on general coordination principles via a CFD is a simple and effective method for improving coordination.**

Automatic discovery of CFDs The hand-crafted CFD is only an initial attempt and, as noted above, does not apply to all coordination challenges. Rather than refining it by hand, we explore whether CFDs can be discovered automatically via meta-learning ("learning to learn") [56, 22, 7], which has been shown to outperform human design choices.

To this end, we implement a meta-learning loop to autonomously discover CFDs using LLM-driven evolution. Our method is loosely based on GEPA [2], which employs Pareto-based candidate selection to improve sample efficiency in the evolutionary process. We combine this with other optimizations which increase the diversity of proposals in the evolutionary process and reduce the cost of "meta-step". We provide further implementation details in Appendix G.1. We meta-train the CFD on Cat & Dog and Brittle Bridge using Grok-4.1-Fast as the code agent, XP_{seed} as the measure of fitness and GPT 5.4 as the meta-optimisation agent. We additionally modify the code agent scaffold by introducing a policy critic that refines generated joint policies based on compliance with the CFD, while leaving the code generation process itself unconditioned on the CFD to preserve diversity. Empirically, this leads to more diverse solutions and reduced variance in the XP_{seed} score of generated policies. Further implementation details regarding the policy critic are provided in Appendix G.2.

The discovered CFD is roughly two-thirds shorter than the hand-crafted variant (Figure 6) and generalises beyond its training conditions. Although meta-training only uses Grok 4.1 Fast as a code agent (direct meta-training with text agents would have raised token cost and runtime by two orders of magnitude), the discovered CFD also improves Grok 4.1 Fast operating as a text agent (Figure 7). We further observe modest XP_{LLM} gains across other models (GPT 5.2, Claude 4.5, Gemini 3 Pro) and the generic code-agent scaffold (Figure 23), even though meta-training optimised XP_{seed} in the critic-based scaffold; we do not, however, observe XP_{seed} gains across LLM types (Figure 22). Overall, CFDs can be efficiently meta-learned with inexpensive scaffolds while generalising to more capable agents for which direct optimisation would be infeasible.

7 Conclusion and Future Work

We study the ZSC performance of LLM agents across a range of coordination tasks under the commonly used *text* and *code* scaffolds, and find that they consistently fail to coordinate effectively, even in simplified environments. Although the two scaffolds are not directly comparable, we

qualitatively observe that code agents are more prone to brittle, arbitrary conventions than text agents, an observation worth noting given their increasing deployment in real-world settings [32, 28, 27]. We identify several factors that contribute to these failures: a limited understanding of the coordination problem, a failure to recognise that no shared convention has been established, and weak *Theory of mind (ToM)* reasoning about how a partner will interpret one's actions. Our experiments do not causally isolate these factors from one another; rather, they jointly characterise the observed failures. We additionally find that LLM agents are biased toward exploiting *semantic representations* of the environment, which can both aid and impede coordination. Next, we introduce *Coordination-friendly definition (CFD)*, a simple and general mechanism for improving coordination that can be readily integrated into existing agent scaffolds, and show that CFDs can be automatically discovered through meta-learning, reducing the need for manual design.

Limitations Both the hand-crafted and meta-learned CFDs remain relatively simple and do not apply to every coordination challenge. For the meta-learned CFD, we attribute this in part to the limited number of meta-training environments, in line with prior meta-learning work showing that broader task distributions yield more robust generalisation [45, 44, 16]; the hand-crafted CFD, being designed manually, would similarly benefit from refinement on a wider range of coordination settings. In particular, CFDs can lead agents to *under-coordinate*: in the ad-hoc conventions environments, agents repeatedly choose the safe fallback instead of forming a convention at test time. This failure mode is not unique to our approach. Off-belief learning (OBL) [23], for instance, would behave similarly, since it reinterprets the interaction history as though the partner had followed a random policy and therefore discards any ad-hoc convention formed during the interaction. Avoiding under-coordination while retaining robustness remains an open problem in ZSC. Nevertheless, we view under-coordination as preferable to the prevailing failure mode of *over-coordination*, in which agents rely on brittle, arbitrary conventions that lead to catastrophic coordination failures. Finally, due to cost, our variable-LLM evaluation of text agents is less extensive than that of code agents and concentrates on Cat & Dog, Hint-Guess, and the 3×3 Hanabi variant.

Future work Alternative integration mechanisms beyond prompting, such as RLHF-based CFD compliance [5], are discussed in Section 6. In addition, principles from traditional ZSC algorithms [23, 24] may be adapted in conjunction with LLM-RL methods such as GRPO [19] to improve coordination in LLM agents. We find a mixed relationship between reasoning capacity and ZSC performance (Appendix F.6): more capable models or higher reasoning effort do not reliably yield better coordination, and understanding when they do is an important open question. Beyond our minimal environments, LLM agents should be evaluated in more complex settings such as Overcooked [15], where agents can adapt to their partner within an episode. Two related settings fall outside the standard ZSC framework but capture important real-world challenges: adaptation to a partner across repeated episodes, and coordination given a small number of demonstrations of successful coordination (e.g., autonomous vehicles learning from human drivers); our few-shot baseline (Appendix F.7) is a first step in this direction. Finally, our analysis focuses on textual descriptions; extending it to Vision–Language Models (VLMs), which can leverage visual cues for coordination, is a natural direction for future work.

Acknowledgments and Disclosure of Funding

Shashank Reddy is partially supported by the Cooperative AI PhD Fellowship from the Cooperative AI Foundation. Bidipta Sarkar is supported by the Clarendon Fund Scholarship in partnership with a Department of Engineering Science Studentship for his Oxford DPhil. Andrei Lupu was partially funded by a Fonds de recherche du Québec doctoral training scholarship. Jakob N. Foerster is partially funded and Johannes Forkel is funded by the UKRI grant EP/Y028481/1 (originally selected for funding by the ERC). The authors thank Darius Muglich, Antonio León Villares, and Michael Beukman for their thoughtful feedback on the manuscript.

References

[1] 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*. Association for Computational Linguistics, April 2025.

[2] 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, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[3] Eltayeb Ahmed, Uljad Berdica, Martha Elliott, Danijela Horak, and Jakob N. Foerster. Intent factored generation: Unleashing the diversity in your language model. *arXiv preprint arXiv:2506.09659*, 2025.

[4] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[5] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[6] 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. *Artif. Intell.*, 280(C), March 2020.

[7] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A tutorial on meta-reinforcement learning. *Foundations and Trends in Machine Learning*, 18(2–3):224–384, 2025.

[8] Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, November 2002.

[9] Allan Dafoe, Edward Hughes, Yoram Bachrach, Tantum Collins, Kevin R. McKee, Joel Z. Leibo, Kate Larson, and Thore Graepel. Open problems in cooperative AI. *arXiv preprint arXiv:2012.08630*, 2020.

[10] Tim R. Davidson, Adam Fourney, Saleema Amershi, Robert West, Eric Horvitz, and Ece Kamar. The collaboration gap: Exploration and benchmarking of open-world agentic cooperation. In *Conference on Language Modeling (COLM)*, 2026.

[11] 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*, ICML'24. JMLR.org, 2024.

[12] Jakob Foerster. Two Waymo cars fail to coordinate with each other. Video post on X (formerly Twitter), `https://x.com/j_foerst/status/1811676036418937208`, July 2024.

[13] Jakob N. Foerster, H. Francis Song, Edward Hughes, Neil Burch, Iain Dunning, Shimon Whiteson, Matthew M. Botvinick, and Michael H. Bowling. Bayesian action decoder for deep multi-agent reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1942–1951. PMLR, 2019.

[14] Johannes Forkel, Constantin Ruhdorfer, Michael Beukman, Andreas Bulling, and Jakob Foerster. High entropy leads to symmetry-equivariant policies in Dec-POMDPs. *arXiv preprint arXiv:2511.22581*, 2025.

[15] Tobias Gessler, Tin Dizdarevic, Ani Calinescu, Benjamin Ellis, Andrei Lupu, and Jakob Nicolaus Foerster. OvercookedV2: Rethinking Overcooked for zero-shot coordination. In *The Thirteenth International Conference on Learning Representations*, 2025.

[16] Alexander D. Goldie, Zilin Wang, Adrian Hayler, Deepak Nathani, Edan Toledo, Ken Thampiratwong, Aleksandra Kalisz, Michael Beukman, Alistair Letcher, Shashank Reddy Chirra, Clarisse Wibault, Theo Wolf, Charles O'Neill, Uljad Berdica, Nicholas Roberts, Saeed Rahmani, Roberta Raileanu, Shimon Whiteson, and Jakob N. Foerster. Procedural generation of algorithm discovery tasks in machine learning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 35600–35715. PMLR, 2026.

[17] H. P. Grice. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and Semantics*, pages 41–58. Academic Press, New York, 1975.

[18] David Gringras. Safety under scaffolding: How evaluation conditions shape measured safety. *arXiv preprint arXiv:2603.10044*, 2026.

[19] Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025.

[20] Xudong Guo, Kaixuan Huang, Jiale Liu, Wenhui Fan, Natália Vélez, Qingyun Wu, Huazheng Wang, Thomas L. Griffiths, and Mengdi Wang. Embodied LLM agents learn to cooperate in organized teams. *IEEE Transactions on Computational Social Systems*, 13(2):2514–2530, 2026.

[21] Horace He and Thinking Machines Lab. Defeating nondeterminism in LLM inference. *Thinking Machines Lab: Connectionism*, 2025. `https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/`.

[22] Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5149–5169, 2022.

[23] Hengyuan Hu, Adam Lerer, Brandon Cui, 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*, pages 4369–4379. PMLR, 2021.

[24] Hengyuan Hu, Adam Lerer, Alexander Peysakhovich, and Jakob Nicolaus Foerster. "Other-Play" for zero-shot coordination. In *Proceedings of the 37th International Conference on Machine Learning*, PMLR 119, Online, 2020.

[25] Hengyuan Hu, Samuel Sokota, David J Wu, Anton Bakhtin, Andrei Lupu, Brandon Cui, and Jakob Nicolaus Foerster. Self-explaining deviations for coordination. In *Advances in Neural Information Processing Systems*, 2022.

[26] Saffron Huang, Divya Siddarth, Liane Lovitt, Thomas I. Liao, Esin Durmus, Alex Tamkin, and Deep Ganguli. Collective constitutional AI: Aligning a language model with public input. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, page 1395–1417. ACM, June 2024.

[27] Shu Ishida, Gianluca Corrado, George Fedoseev, Hudson Yeo, Lloyd Russell, Jamie Shotton, Joao F. Henriques, and Anthony Hu. LangProp: A code optimization framework using language models applied to driving. *arXiv preprint arXiv:2401.10314*, 2024.

[28] W. Ji, H. Chen, M. Chen, et al. GenSwarm: Scalable multi-robot code-policy generation and deployment via language models. *npj Robotics*, 4:5, 2026.

[29] Chris Kirkham, Norihiko Shirouzu, and Rachael Levy. How Tesla and Waymo's radically different robotaxi approaches will shape the industry. *Reuters*, August 2025.

[30] Matthew Le, Y-Lan Boureau, and Maximilian Nickel. Revisiting the evaluation of theory of mind through question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, November 2019.

[31] Huao Li, Yu Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Charles Lewis, and Katia Sycara. Theory of mind for multi-agent collaboration via large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 180–192, Singapore, December 2023. Association for Computational Linguistics.

[32] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500, 2023.

[33] Wilson Lin. Scaling long-running autonomous coding. Cursor blog, `https://cursor.com/blog/scaling-agents`, January 2026.

[34] Jijia Liu, Chao Yu, Jiaxuan Gao, Yuqing Xie, Qingmin Liao, Yi Wu, and Yu Wang. LLM-powered hierarchical language agent for real-time human-AI coordination. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1219–1228, 2024.

[35] 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*. PMLR, 18–24 Jul 2021.

[36] Andrei Lupu, Timon Willi, and Jakob Foerster. The Decrypto benchmark for multi-agent reasoning and theory of mind. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 83284–83323. PMLR, 2026.

[37] Mingwei Ma, Jizhou Liu, Samuel Sokota, Max Kleiman-Weiner, and Jakob Foerster. Learning intuitive policies using action features. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[38] Ellen M Markman and Gwyn F Wachtel. Children's use of mutual exclusivity to constrain the meanings of words. *Cognitive Psychology*, 20(2):121–157, 1988.

[39] Mariana Meireles, Rupali Bhati, Niklas Lauffer, and Cameron Allen. The influence of scaffolds on coordination scaling laws in LLM agents. In *First Workshop on Multi-Turn Interactions in Large Language Models*, 2025.

[40] Darius Muglich, Johannes Forkel, Elise van der Pol, and Jakob Nicolaus Foerster. Expected return symmetries. In *The Thirteenth International Conference on Learning Representations*, 2025.

[41] Magnus Müller and Gregor Žunič. Browser Use: Enable AI to control your browser. `https://github.com/browser-use/browser-use`, 2024.

[42] Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S. Yu, and Qing Li. A survey of WebAgents: Towards next-generation AI agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25. Association for Computing Machinery, 2025.

[43] 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. *arXiv preprint arXiv:2506.13131*, 2025.

[44] Junhyuk Oh, Gregory Farquhar, Iurii Kemaev, Dan A. Calian, Matteo Hessel, Luisa Zintgraf, Satinder Singh, Hado van Hasselt, and David Silver. Discovering state-of-the-art reinforcement learning algorithms. *Nature*, 648(8093):312–319, 2025.

[45] Junhyuk Oh, Matteo Hessel, Wojciech M. Czarnecki, Zhongwen Xu, Hado van Hasselt, Satinder Singh, and David Silver. Discovering reinforcement learning algorithms. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[46] Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer, 2016.

[47] David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and Brain Sciences*, 1(4):515–526, 1978.

[48] Mahesh Ramesh, Kaousheik Jayakumar, Aswinkumar Ramkumar, Pavan Thodima, Aniket Rege, and Emmanouil-Vasileios Vlatakis-Gkaragkounis. Sparks of cooperative reasoning: LLMs as strategic Hanabi agents. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 103567–103641. PMLR, 2026.

[49] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2018.

[50] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

[51] Melanie Sclar, Jane Dwivedi-Yu, Maryam Fazel-Zarandi, Yulia Tsvetkov, Yonatan Bisk, Yejin Choi, and Asli Celikyilmaz. Explore theory of mind: program-guided adversarial data generation for theory of mind reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[52] Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20. Curran Associates Inc., 2020.

[53] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. Large language models as generalizable policies for embodied tasks. In *The Twelfth International Conference on Learning Representations*, 2024.

[54] Gerald Tesauro. TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

[55] 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*, PMLR 139, 2021.

[56] Joaquin Vanschoren. Meta-learning. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *Automated Machine Learning: Methods, Systems, Challenges*, pages 35–61. Springer, 2019.

[57] Wei Wang, Dan Zhang, Tao Feng, Boyan Wang, and Jie Tang. BattleAgentBench: A benchmark for evaluating cooperation and competition capabilities of language models in multi-agent systems. *arXiv preprint arXiv:2408.15971*, 2024.

[58] Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. Exploring large language models for communication games: An empirical study on Werewolf. *arXiv preprint arXiv:2309.04658*, 2024.

[59] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc.

[60] Ceyao Zhang, Kaijie Yang, Siyi Hu, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, Xiaojun Chang, Junge Zhang, Feng Yin, Yitao Liang, and Yaodong Yang. ProAgent: Building proactive cooperative agents with large language models. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024.

[61] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[62] 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 Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5348–5375, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

Appendix

Table of Contents

A Related Work **16**

B Broader Impact Statement **17**

C Licenses for Existing Assets **17**

D Environments **17**

D.1 Navigation games 17
D.2 Convention games 18
D.3 Simultaneous selection games 21
D.4 Semantic games 22

E Schematic diagram of LLM agents **23**

F Additional details **23**

F.1 Coordination Failure Rate 23
F.2 Text Agents with CFD 24
F.3 Code Agents with CFD 24
F.4 Trampoline Tiger Game 25
F.5 Local Policies of Text Agents 26
F.6 Reasoning Effort 28
F.7 Few-Shot Demonstrations 29
F.8 Hanabi 3x3 variant 29
F.9 Hint-Guess Implicatures 29
F.10 Automatic discovery of CFDs 30

G Implementation details **34**

G.1 Meta-Learning Loop 34
G.2 Policy Critic 37
G.3 Prompts 38
G.4 Experimental Resources 38

H Reasoning Traces **45**

A Related Work

Prior work has evaluated LLM agents across a variety of multi-agent settings, including embodied interaction [31, 61, 20], theory of mind (ToM) [36, 30, 51], and games [62, 58, 57]. Closest to our setting, work on fully cooperative tasks has largely focused on spatial reasoning environments such as Overcooked [34, 60] and mazes [10], often relying on specialised scaffolds to achieve performance competitive with traditional RL methods. However, in the zero-shot coordination (ZSC) setting, independently developed agents are unlikely to share identical model-scaffold pairings, an implicit assumption in much of prior work, with [10, 36] being notable exceptions. Moreover, the choice of the scaffold can substantially influence performance [39], making it difficult to disentangle inherent model capabilities from scaffold-induced effects, a challenge that has been noted in other domains [18].

In contrast, a key advantage of using simplified environments, as in this work, is that they make failure modes explicit and easier to diagnose. In complex benchmarks such as Hanabi [6], the core challenge is to communicate private information indirectly through actions and signals, which requires a combination of abilities, including theory of mind, convention formation, and long-horizon planning based on past interactions that can span dozens of timesteps. As a result, diagnosing coordination failures is challenging, as they may stem from deficiencies in any of the required abilities. Even among human players, post-game analysis involves reconstructing beliefs at each timestep of the game to identify where coordination broke down; doing so with LLM agents is even more challenging, given the difficulty of interpreting the reasons behind their actions. [1] attempt to address this by querying models at fixed game states with limited success. More recently, [48] benchmark LLM agents in

Hanabi and report XP scores between different LLMs that improve with model strength. Their strongest agents average around 15 points, but rely on scaffolding with programmatic deductions and do not assign a score of zero to games that end after all life tokens are lost, which is the convention in prior ZSC work [6]. Even so, these agents remain well below experienced human players (20+) and state-of-the-art ZSC methods (24+) [14], consistent with our findings. More broadly, complex environments require capabilities beyond coordination, for instance, strong planning in Hanabi or adequate state coverage in Overcooked [15], making it difficult to explicitly test for coordination abilities.

B Broader Impact Statement

This work studies coordination among independently developed LLM agents in a zero-shot setting, characterising both their strengths and failure modes. Although insights into multi-agent coordination are dual-use in principle, our focus on fully cooperative simplified environments substantially limits the risk of misuse, and the proposed CFDs serve as an alignment-oriented mechanism for safer, more transparent coordination among independently developed agents.

C Licenses for Existing Assets

Environments. The figures used to illustrate the Cat & Dog [23], Tiny Hanabi [13], Hint–Guess [37], Trampoline–Tiger [25] and Matrix Game [35] environments are reproduced from the corresponding arXiv submissions, which are released under the CC BY 4.0 license. We reimplement each of these environments from scratch based on the descriptions in the original papers.

LLMs accessed via API. Grok 4.1 Fast and Grok 4.5 (xAI); GPT-5.2, GPT-5 Mini, GPT-5.4, and GPT-5.6 Sol (OpenAI); Gemini 3 Pro and Gemini 3.1 Pro (Google); Claude Opus 4.5, Claude Opus 5, Claude Sonnet 4.6, and Claude Haiku 4.5 (Anthropic); DeepSeek V3.2 (DeepSeek); Qwen3-Max-Thinking (Alibaba); and Ministral-14B-2512 (Mistral) are accessed through OpenRouter and used in accordance with OpenRouter's terms of service and the underlying providers' usage policies.

Open-weight LLMs. The token-level probability experiments use Qwen3-14B, OLMo-3-32B-Think, and Ministral-14B locally, all released under the Apache 2.0 license by their respective authors.

D Environments

D.1 Navigation games

We design a set of navigation challenges intended to capture the kinds of coordination challenges that arise in robotics and autonomous driving. A real-world example, documented in this video [12], shows two Waymo vehicles hesitating at a narrow passage, each attempting to yield to the other. While such cautious behaviour is highly effective when interacting with proactive human drivers, it results in deadlock when both autonomous agents act symmetrically. More generally, navigation games highlight a central coordination problem: two symmetric players must "break symmetry" by adopting distinct, complementary roles in order to progress.

Narrow Tunnel formalises the coordination challenge illustrated in the above example. The environment consists of a tunnel, laid out left to right, with two agents positioned at opposite ends, tasked with exchanging positions in as few turns as possible. The setting is turn-based, with agents selecting actions simultaneously from the action set $\{\texttt{LEFT}, \texttt{RIGHT}, \texttt{STAY}\}$. The tunnel is composed of two types of tiles: *wide* tiles, which can accommodate both agents simultaneously, and *narrow* tiles, which can host at most a single agent.

Collision rules restrict movement into narrow tiles. Specifically, (i) if two agents attempt to swap positions within a single turn and at least one of them occupies a narrow tile, the swap is blocked, and no movement occurs; (ii) if both agents attempt to enter the same narrow tile simultaneously, neither move is executed. Given these restrictions, at narrow points in the tunnel, agents must assume

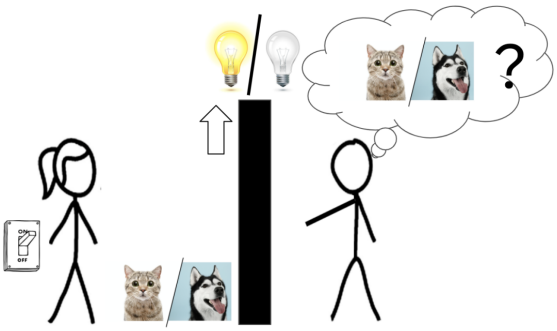


Figure 8: **An illustration of the Cat & Dog game taken from [23].**

asymmetric roles to successfully traverse it (leader and follower). To incentivise minimising the number of turns, agents incur a per-step joint penalty of -1 until both agents have successfully exchanged positions.

The environment is designed to support any custom layout via a simple hyperparameter at initialisation. The default layout, shown below, mimics the coordination challenge illustrated in the previously shown video:[2]

$$[\texttt{start_left}, \texttt{wide}, \texttt{narrow}, \texttt{wide}, \texttt{start_right}]$$

Brittle Bridge is inspired by cooperative platformer video games in which successful progression requires players to adopt complementary roles, such as one agent acting while the other temporarily waits. Although the assignment of such leader–follower roles is arbitrary, it is essential for successful task completion. Two agents are initially located on the same side of a bridge of length n. Their joint objective is to traverse the bridge such that both agents reach the opposite side in the minimal number of turns. The environment is turn-based, with agents selecting actions simultaneously from the set $\{\texttt{forward}, \texttt{backward}, \texttt{stay}\}$. No collision constraints are imposed, so multiple agents may occupy the same tile at any given time. The coordination challenge stems from the bridge's fragility. If both agents remain on the bridge, regardless of their exact positions, for more than a fixed number of consecutive turns, given by the parameter `bridge_break_threshold` (typically set to 1), the bridge collapses. When this happens, the episode ends and both agents receive a large negative reward of -1000. Importantly, the timer associated with the fragility constraint resets whenever at least one agent steps off the bridge. Consequently, when `bridge_break_threshold` equals 1, one agent may safely cross the bridge in its entirety while the other agent alternates between entering the first bridge tile and retreating to land, thereby preventing collapse. For our experiments, we fix n to 5 and the `bridge_break_threshold` to 1. To incentivise minimising the number of turns, agents incur a per-step joint penalty of -1 until both agents have crossed the bridge.

D.2 Convention games

Convention games typically assign players fixed asymmetric roles and proceed sequentially. The coordination challenge arises from partial observability: the first player has access to some information that must be conveyed to the second player, who then makes a guess. To achieve a high reward, the players must establish a convention that maps possible signals to possible guesses.

Cat & Dog is a two-player, fully cooperative coordination game introduced by [23]. The setup is straightforward: two players, *Alice* and *Bob*, are initially separated by a wall. At the beginning of the game, Alice privately observes a randomly chosen animal, either a cat or a dog, on her side of the wall. She must then select one of the following actions:

- Turn on a light bulb on Bob's side of the wall at no cost,
- Leave the light bulb off (the default; explicitly available as an action) at no cost,

[2] `start_left` and `start_right` are considered wide tiles.

Game Description $\mathcal{D}_\mathcal{M}$ — Cat & Dog (shared by both players)

You are playing a game of cat and dog with one other player. One of you is Alice and the other is Bob. Alice is separated from Bob by a wall. The goal of the game is maximize your (joint) reward. On Alice's side of the wall, there is either a cat or a dog. Additionally, there is a light switch on Alice's side of the wall, which she can turn on to illuminate a bulb on Bob's side of the wall.

Alice goes first and can do exactly one of the following:
– bail for a (joint) reward of 1 (this ends the game immediately)
– remove the wall for a (joint) reward of -5
– turn on the light bulb (no reward)
– do nothing, i.e. the light bulb stays off (no reward)

If Alice didn't bail, it is Bob's turn. Bob observes the light status and the animal on Alice's side of the wall, if the wall was removed by Alice. Bob can do exactly one of the following:
– bail for a (joint) reward of 0.5
– or guess the animal, i.e. either cat or dog
  – if Bob guesses correctly, he receives a (joint) reward of 10
  – if Bob guesses incorrectly, he receives a (joint) reward of -10

The players cannot communicate with each other except the options presented above (i.e. light status and removing the wall).

Turn 1 — Alice

`$Game History$`

You are Alice. $(\mathcal{D}_\mathcal{R})$ On your side of the wall, there is a cat. $(\mathcal{D}_\mathcal{H})$

`$Legal Actions$`

`bail`, `remove_wall`, `turn_on_light`, `do_nothing`

Response:

```
REASONING: ...
ACTION JSON: {"action":  "remove_wall"}
```

Outcome: The wall is removed. Immediate joint reward: -5.

Turn 2 — Bob

`$Game History$`

You are Bob. $(\mathcal{D}_\mathcal{R})$ The light bulb on your side of the wall is off. Furthermore, the wall has been removed. You can see a cat on the other side of the wall. $(\mathcal{D}_\mathcal{H})$

`$Legal Actions$`

`bail`, `guess_dog`, `guess_cat`

Response:

```
REASONING: ...
ACTION JSON: {"action":  "guess_cat"}
```

Outcome: Bob's guess is correct. Immediate joint reward: 10.

Figure 9: **Worked example of the natural language descriptions $\mathcal{D} = (\mathcal{D}_\mathcal{M}, \mathcal{D}_\mathcal{R}, \{\mathcal{D}_{\mathcal{H}_i}\})$ for Cat & Dog with text agents.** Both players receive the same game description $\mathcal{D}_\mathcal{M}$ (top). Each text agent additionally receives its role $\mathcal{D}_\mathcal{R}(r)$ and a description of its action-observation history $\mathcal{D}_{\mathcal{H}_i}(\tau_i)$, which together fill the `Game History` field of the text-agent prompt template (Figure 24). The game description is taken verbatim from our implementation. In this episode, Alice plays the ZSC-optimal action of removing the wall, and the players obtain a total joint reward of $-5 + 10 = 5$. Code agents receive only $\mathcal{D}_\mathcal{M}$ and generate a joint policy for both roles.

- Remove the wall at a cost of -5,
- Bail, yielding a joint reward of 1, terminating the game immediately.

Subsequently, it is Bob's turn. He observes the state of the light bulb and, if Alice has removed the wall, also observes the animal. Bob then chooses one of the following actions:

- Guess the correct animal (*Cat* or *Dog*). If his guess is correct, both players receive a joint reward of 10; otherwise, they incur a joint penalty of -10. In expectation, a random guess yields a reward of 0.
- Bail, yielding a smaller but guaranteed reward of 0.5.

The term *Cat & Dog* refers specifically to the one-shot version of the game. The iterated version is examined in the following section. Since there are two equivalent but arbitrary signalling conventions (either `light_on` $\rightarrow$ *Cat* or `light_on` $\rightarrow$ *Dog*), independent agents without prior communication cannot coordinate on a convention within a single round. Therefore, removing the wall is considered "optimal" in the ZSC setting.

Worked example of $\mathcal{D}$. Figure 9 illustrates the natural language descriptions introduced in Appendix 2 for a single episode of Cat & Dog. The game description $\mathcal{D}_\mathcal{M}$ is shared by both players and specifies the roles, actions, and rewards. A text agent is additionally told which role it plays via $\mathcal{D}_\mathcal{R}$ and what it has observed so far via $\mathcal{D}_{\mathcal{H}_i}$; for example, Bob's history description reports the status of the light and, since Alice removed the wall, the animal. A code agent only receives $\mathcal{D}_\mathcal{M}$, since it generates a joint policy that specifies the behaviour of both Alice and Bob.

Ad-Hoc Conventions. Inspired by [15], we introduce games that require protocol formation at test time. *Ad-Hoc Conventions* is an iterated version of Cat & Dog. In contrast to the one-shot version discussed in the previous section, iteration enables the formation of conventions that can lead to guaranteed high payoffs in later rounds. We replace the animals with generic levers that can be assigned labels (in our case $\{\texttt{blue}, \texttt{red}\}$) and the lightbulb is replaced by labelled signals (in our case $\{\texttt{square}, \texttt{circle}\}$). Removing the wall is now represented by revealing the current correct lever. While we do not make further use of it in our experiments, we support an arbitrary number of signals and levers to increase the difficulty of the game. Additionally, the lever labels, signal labels, and the number of rounds can be specified as a hyperparameter. In our experiments, we play ten rounds per Ad-Hoc Conventions game.

Ad-Hoc Conventions with permuted signals. We also introduce a harder variant of Ad-Hoc Conventions, called *Ad-Hoc Conventions with permuted signals* (A.C.P.S.). Here we distinguish between *buttons* and *signals*, each with its own set of labels ($\texttt{apple}, \texttt{banana}$ and $\texttt{square}, \texttt{circle}$ in our case). The only change per round is that, instead of directly revealing a signal, Alice can now press one of the buttons (or choose reveal or bail), which then displays a signal to Bob. The connection between buttons and signals is given by a random bijection unknown to both players. Importantly, Alice does not see the resulting signal, and Bob does not see the pressed button. Without prior communication, it is therefore impossible to recover the exact bijection between signals and levers. Note that the button to signal mapping is fixed within the episode. As a result, the game cannot be solved through a simple lexicographical order strategy: Alice cannot know which button will trigger which signal for Bob. Nevertheless, in both the permuted and non-permuted versions, the players retain a simple fallback strategy, continuously revealing (i.e. removing the wall), which guarantees a reward of 5 per round, for a total "safe" return of 50 across the 10 rounds we typically consider in our experiments.

Tiny Hanabi is a toy-game introduced in [13]. It consists of two players, who are each dealt a random card at the start of the game. The final joint reward depends on both the players' cards and their actions throughout the game. The second player observes the first player's actions before picking his action, which allows the first player's action to be interpreted as a signal. We show the payoff matrix of Tiny Hanabi in Figure 10. Due to the asymmetry of the payoff matrix, the original version of Tiny Hanabi has a unique SP optimal policy, yielding a joint reward of 10. This policy is given by the first player picking action A if he has card 2 and C otherwise, with the second player responding accordingly. To see why it is unique, note that a joint reward of 10 requires that neither player ever plays B, and that the second player conditions on the first player's action, since no column of the payoff matrices consists entirely of 10s. The first player must therefore play different actions for the two cards. If the first player instead plays C with card 2 while the second player holds card 1, the joint reward is 0 regardless of the second player's response. Hence, the first player must map card 2 to A and card 1 to C. In Tiny Hanabi Fixed, the second player could respond to this case with C and still obtain a reward of 10, which gives rise to two symmetric SP-optimal policies.

To create a version of Tiny Hanabi with a coordination challenge, we symmetrize the payoff matrix, also shown in Figure 10. As in the previous convention games, there is a safe fallback option, which does not require any conventions: Playing B as the first player leads to a reward of 8 as long as the second player responds rationally. Hence Tiny Hanabi Fixed is similar to Cat & Dog wherein there exists two symmetrical conventions with identical payoffs and a safe fallback option.

Hanabi In the standard game of Hanabi, each colour has a deck of 10 cards: three 1s, two 2s, two 3s, two 4s, and one 5, for a total of 50 cards across 5 colours. The game is fully cooperative and features 5 ranks and 5 colours, with each card defined by a rank–colour pair. Players share a limited pool of 3 life tokens and 8 hint tokens. At the start, each player is dealt 5 cards; crucially, players cannot see their own hands but can observe those of their teammates. The objective is to build "fireworks" by stacking cards of each colour in increasing rank order. On each turn, a player may either play a card, discard a card, or provide a hint to another player. Hints can specify either a rank or a colour, indicating all cards of that type in the recipient's hand, and each hint consumes one token. An incorrect play—i.e., one that cannot be placed on the fireworks—costs a life token. Discarding a card restores a hint token, up to the maximum. After playing or discarding, the player draws a new card from the deck, maintaining a constant hand size while cards remain. Players receive +1 for each correctly played card, and the game ends with a score of 0 if all life tokens are lost.



Figure 10: **Payoff matrices for Tiny Hanabi adapted from [13].** On the left side, we show the matrix from the original paper, and on the right side, the matrix of our version with a coordination challenge.

In this work, we consider a simplified 3×3 variant with 3 colours and 3 ranks, where each colour has three 1s, two 2s, and one 3, for a total maximum score of 9. The game includes a single life token and a limit of 3 hint tokens. The central challenge of Hanabi arises from the limited and costly communication channel provided by hints. Effective play therefore requires agents to use hints efficiently, often relying on implicit conventions to convey information about hidden states to their partners.

D.3 Simultaneous selection games

In *simultaneous selection games*, symmetric players have to coordinate on one of many similar, often symmetric, choices to achieve a high joint reward. We study both one-shot versions of these games (*Matrix Game*) and iterated versions (*Permuted Levers*).

Permuted levers Lever Games are common in ZSC literature [24, 55]. For our purposes, we focus on iterated versions of lever games with n levers all with identical payoff. In each round, both players simultaneously chose one of the levers. If players manage to pick the same lever, they receive a joint reward of 1. To prevent arbitrary focal points introduced by numbering the levers (which is needed for the LLM agents to interact with the environment), we introduce a further complication inspired by the other-play objective [23]: Each player observes and acts on the levers through a random permutation of the lever numbers, which stays fixed across the rounds of the game. While players are aware of the fact, they do not know their own or others' permutations.

Both the number of levers k and the number of rounds n are hyperparameters, which can be set at initialisation. For our experiments, we fix $k = 5$ and $n = 10$.

Matrix game Our version of the *Matrix Game* was introduced by [35]. In this game, two players simultaneously choose a row/column and receive the reward in the selected cell. The matrix used to play this game is shown in Figure 11. For humans, there is an easy-to-identify focal point given by the first column and first/second row, which can be exploited. Our version of the matrix game uses a 11×11 matrix extending the diagonal pattern by 1. This yields another possible focal point at the middle of the diagonal pattern (see also Figure 39). Matrix game differs from the other coordination challenges presented in this section in that there is no salient policy that could be considered "ZSC-optimal". Successful coordination in the matrix game largely depends on players forming the same arbitrary conventions to form a focal point in the matrix, exactly the inverse of what ZSC is trying to achieve. Therefore, we have chosen to exclude Matrix game from the main experiments of the paper, but report results for completeness in Appendix F.2 and Appendix F.3.

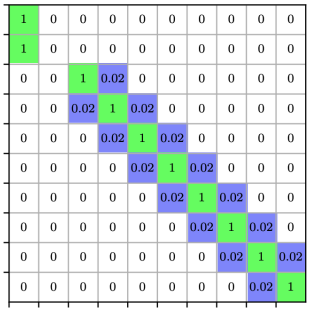


Figure 11: **Payoff matrices for the Matrix Game** adapted from [35]. Players simultaneously pick a row and column, respectively, and receive the reward at the selected cell.

	1	2	3	4
A	1	0	1.01	0
B	1.01	1	0	0
C	0	0	1	1.01
D	0	1.01	0	1

Figure 12: **Semantic Convention Game.** Alice and Bob chose a row and column in the matrix sequentially. Additionally, only Alice is informed of the currently active column, which is chosen at random. Only if the agents end up selecting a cell in the active column do they receive the reward in that cell. Agents must choose between the lexicographical mapping convention (green) and the SP-optimal mapping convention (blue).

D.4 Semantic games

Semantic selection and convention games demonstrate that the addition of semantics can create coordination challenges in games that previously did not exhibit them, as they have a unique SP optimal policy. In particular, the semantic games presented in this paper, the *semantic selection game* (Figure 5) and the *semantic convention game* (Figure 12), force players to choose between two mutually incompatible meta-strategies: either playing the SP optimal strategy or following a strategy that is aligned with the game semantics. Figure 4 has demonstrated that LLM agents often struggle to consistently interpret semantics and fail to properly reason about these difficulties. Semantic games, on the other hand, show that even when the semantic interpretation is clear across agents, semantics can still hinder the coordination abilities of agents, just by the fact that they introduce another possible meta-strategy.

Figure 5 demonstrates that LLM agents are generally not able to consistently choose among these competing meta-strategies, leading to coordination failures. Interestingly, even though the CFD does not specify how to decide among the competing meta-strategies, it does lead agents to consistently opt for the SP optimal policy in the semantic selection game. In the semantic convention game, the CFD does not yield significant improvement in the agents' coordination as shown in Figure 13 (Left).

Hint-Guess The *hint-guess* game was introduced in [37] to study whether agents can exploit shared semantic structure between actions and observations for zero-shot coordination. The game involves two players, a *hinter* and a *guesser*, each dealt a hand of N cards, where every card is described by two discrete features. One card from the guesser's hand is secretly designated the target, and its identity is revealed only to the hinter. The hinter must select a card from their *own* hand to

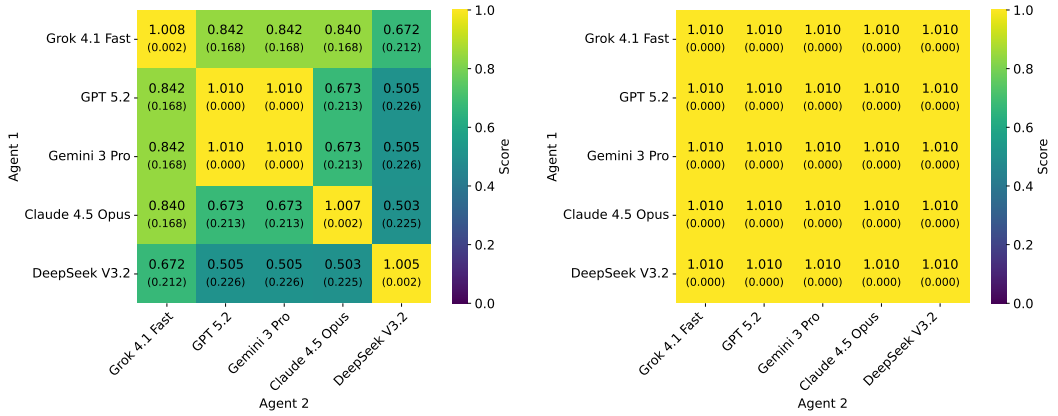


Figure 13: **XP scores of code agents in the semantic games using the hand-crafted CFD.** (Left) Semantic convention game. (Right) Semantic selection game. Scores are averaged across six i.i.d. XP matrices; where each cell contained the mean return over 100 episodes.

show the guesser; the guesser then picks which card in their hand corresponds to the target. Both players receive a reward of 1 if the guess is correct and 0 otherwise. The environment is visualised in Figure 14. Interestingly, implicatures [38, 17] represent sophisticated patterns of human reasoning, making them a compelling test of whether LLM agents can exhibit similar capabilities. As shown in Table 8, text agents consistently exhibit human-like reasoning and, notably, demonstrate the ability to coordinate via implicatures.

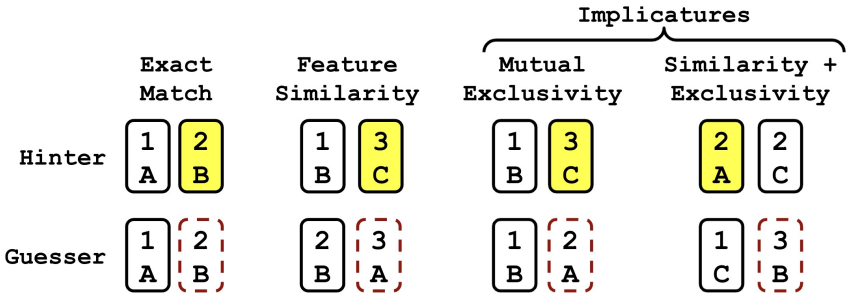


Figure 14: **Example hint-guess scenarios, adapted from Figure 2 of [37].** Each panel shows a distinct coordination challenge: exact feature match, partial feature similarity, mutual exclusivity, and a combination of the latter two, which represents implicatures [38, 17]. The card outlined in red is the target; the card highlighted in yellow is the human-compatible hint choice. The two rightmost scenarios require reasoning about implicature, where the intuitive hint shares no direct feature with the target.

E Schematic diagram of LLM agents

We present a schematic diagram of the two agents scaffolds in Figure 15 and the different evaluation schemes in Figure 16.

F Additional details

F.1 Coordination Failure Rate

In addition to reporting joint rewards, we measure the coordination failure rate, which captures the frequency of catastrophic coordination failures arising from paired local policies. For example, in Tiny Hanabi, a grounded joint policy that consistently reaches (B, B) achieves a guaranteed reward

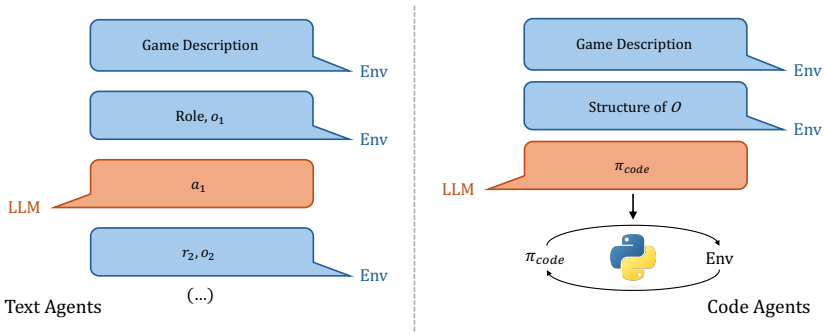


Figure 15: **Illustration of text and code agents.** While text agents only act out the local policy associated with their assigned roles, each code agent produces a joint policy in Python code. Code agents receive additional information on the structure of the observation space.

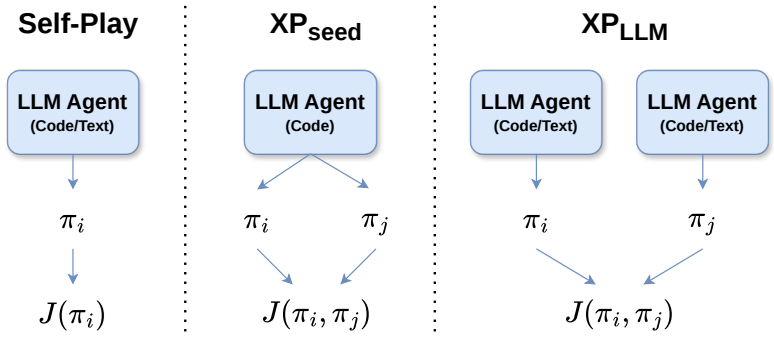


Figure 16: **Illustration of the evaluation protocols for text and code agents.** Note that XP_{seed} is defined only for code agents, as they generate joint policies, whereas text agents induce only local policies. While text agents induce a joint policy which can be obtained by varying the role assignment (see Figure 15), they tend to significantly underperform code agents in self-play as the implicit policies are often not self-compatible (see Figure 18).

of 8. In contrast, we observe paired policies that attain the maximum reward of 10 in 80% of episodes but receive a reward of 0 in the remaining 20%, resulting in the same XP score. Despite the same average score, the grounded policy is clearly more robust. To capture this, we report coordination failure rates. An episode is classified as a failure if the agents' joint reward is at or below the threshold specified in Table 2.

F.2 Text Agents with CFD

We report the SP scores of the text agents with and without the hand-crafted CFD in the fixed LLM setting for Grok 4.1 Fast and GPT-5.2 in Table 1. We find that the agents exhibit high coordination failure rates and low SP scores across multiple environments, indicating poor coordination. Introducing the hand-crafted CFD improves coordination in several environments, with a larger effect on Grok 4.1 Fast than on GPT-5.2. However, as discussed in Appendix F.3, the CFD can be overly restrictive. This is also evident in Brittle Bridge, where agents enter a deadlock, each waiting for the other to step onto the bridge to avoid collapse.

F.3 Code Agents with CFD

We report the SP and XP scores of the code agents under both the *fixed* LLM setting (denoted XP_{seed}) and the *variable* LLM setting (denoted XP_{LLM}) in Table 6. The scores are calculated as follows. First, we generate six independent cross-play matrices, with each LLM contributing one joint policy to each matrix. Every joint-policy pairing is evaluated over 100 episodes. We then average scores across the six XP matrices to obtain a single $(N_{\text{LLM}} \times N_{\text{LLM}})$ matrix. For SP, we report the mean and standard

Table 1: SP performance comparison of text agents (mean ± SEM) across environments, evaluated with and without the CFD. Results are shown for Grok 4.1 Fast and GPT-5.2. The coordination failure rate captures the frequency of catastrophic coordination breakdowns. Given the simplicity of these environments, we require coordination failure rates to be near zero in order to consider coordination robust. We report the mean and standard error over 100 episodes for single-turn games and 50 episodes for multi-turn games. We find that, with the exception of Narrow Tunnel and Permuted Levers, both Grok 4.1 Fast and GPT-5.2 exhibit high coordination failure rates.

		w/o CFD		w/ CFD	
Env	Model	SP Score	Coord. failure rate	SP Score	Coord. failure rate
Cat and Dog	Grok 4.1 Fast	$-0.75_{\pm 0.99}$	$0.53_{\pm 0.05}$	$5.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	GPT 5.2	$4.50_{\pm 0.49}$	$0.09_{\pm 0.03}$	$5.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
Ad Hoc Conv.	Grok 4.1 Fast	$95.80_{\pm 2.65}$	$0.04_{\pm 0.03}$	$50.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	GPT 5.2	$82.71_{\pm 4.01}$	$0.16_{\pm 0.05}$	$50.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
Ad Hoc Conv. PS	Grok 4.1 Fast	$82.01_{\pm 3.30}$	$0.08_{\pm 0.04}$	$49.64_{\pm 0.28}$	$0.04_{\pm 0.03}$
	GPT-5.2	$76.95_{\pm 3.81}$	$0.14_{\pm 0.05}$	$49.91_{\pm 0.09}$	$0.02_{\pm 0.02}$
Tiny Hanabi	Grok 4.1 Fast	$7.38_{\pm 0.41}$	$0.23_{\pm 0.04}$	$8.22_{\pm 0.21}$	$0.05_{\pm 0.02}$
	GPT-5.2	$8.92_{\pm 0.30}$	$0.10_{\pm 0.03}$	$7.64_{\pm 0.37}$	$0.18_{\pm 0.04}$
Tiny Hanabi Fixed	Grok 4.1 Fast	$7.80_{\pm 0.40}$	$0.21_{\pm 0.04}$	$7.78_{\pm 0.14}$	$0.03_{\pm 0.02}$
	GPT-5.2	$7.88_{\pm 0.40}$	$0.20_{\pm 0.04}$	$7.20_{\pm 0.29}$	$0.13_{\pm 0.03}$
Brittle Bridge	Grok 4.1 Fast	$-149.69_{\pm 48.5}$	$0.14_{\pm 0.05}$	$-15.86_{\pm 0.49}$	$0.30_{\pm 0.06}$
	GPT-5.2	$-13.00_{\pm 0.48}$	$0.10_{\pm 0.04}$	$-13.72_{\pm 0.50}$	$0.10_{\pm 0.04}$
Narrow Tunnel	Grok 4.1 Fast	$-5.58_{\pm 0.17}$	$0.00_{\pm 0.00}$	$-5.94_{\pm 0.19}$	$0.00_{\pm 0.00}$
	GPT-5.2	$-8.92_{\pm 0.48}$	$0.02_{\pm 0.02}$	$-8.38_{\pm 0.45}$	$0.02_{\pm 0.02}$
Matrix Game	Grok 4.1 Fast	$0.62_{\pm 0.05}$	$0.34_{\pm 0.05}$	$0.97_{\pm 0.02}$	$0.03_{\pm 0.02}$
	GPT-5.2	$0.78_{\pm 0.04}$	$0.22_{\pm 0.04}$	$0.84_{\pm 0.04}$	$0.16_{\pm 0.04}$
Permuted Levers	Grok 4.1 Fast	$6.08_{\pm 0.36}$	$0.02_{\pm 0.02}$	$6.28_{\pm 0.41}$	$0.02_{\pm 0.02}$
	GPT-5.2	$7.24_{\pm 0.38}$	$0.02_{\pm 0.02}$	$6.88_{\pm 0.39}$	$0.02_{\pm 0.02}$

Table 2: Thresholds used to define coordination failures for each environment. An episode counts as a failure if its joint reward is at or below the threshold.

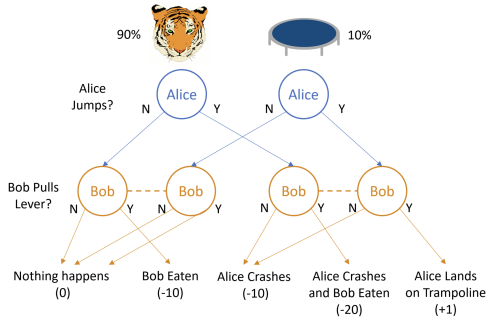
Environment	**Threshold**
Permuted Levers	0.0
Cat and Dog	0.0
Brittle Bridge	-20.0
Narrow Tunnel	-20.0
Ad-hoc Conventions	49.5
Ad-hoc Conventions (Permuted Signals)	49.5
Tiny Hanabi Fixed	0.0
Tiny Hanabi	0.0
Matrix Game	0.0

error over the diagonal entries. For variable-LLM XP_{LLM}, we compute the mean and standard error over the off-diagonal entries in each LLM′s corresponding row and column. For fixed-LLM XP_{seed}, we pair joint policies produced by the same LLM across different XP matrices into disjoint pairs (e.g., $(1, 2)$, $(3, 4)$, ...), evaluate each pair jointly, and report the resulting mean and standard error. We do the same when reporting the coordination failure rates in Table 7.

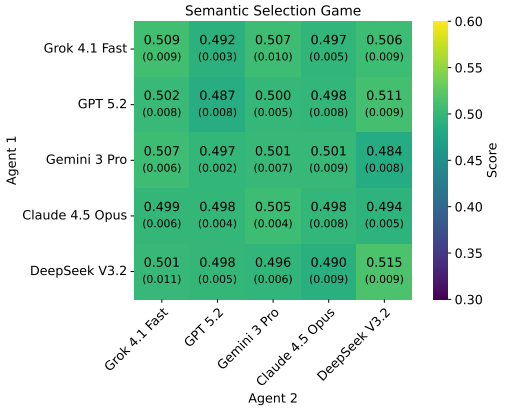
We observe a substantial SP–XP gap across most environments, and in some cases (e.g., Brittle Bridge), agents fail to generate even high-performing self-play policies, indicating a limited understanding of the underlying coordination challenge. Introducing the hand-crafted CFDs yields only modest gains, improving performance primarily in Cat & Dog and the semantic selection game.

F.4 Trampoline Tiger Game

The tiger trampoline game, shown in Figure 17a, is a coordination game introduced in [25] to illustrate the concept of *self-explaining deviations (SED)*. A self-explaining deviation is an action by a teammate in a fully cooperative setting that, at first, appears to hurt the team's performance.



(a) An illustration of the trampoline tiger environment [25]. Alice observes whether Bob's lever would deploy a tiger or a trampoline and then chooses whether to jump. Bob observes only Alice's action and subsequently decides whether to pull the lever. Dotted lines between Bob's decision nodes indicate information sets that Bob cannot distinguish.



(b) XP scores of code agents in the trampoline tiger game [25]. Scores are averaged across six different XP matrices; each cell reports the mean return over 100 episodes. For our experiments, we use $p_{\text{tiger}} = 0.5$ for equal coverage of tiger and non-tiger states. The LLM agents are not aware of p_{tiger}.

Figure 17: **The trampoline tiger environment.**

But if players act under the belief that their teammates' actions are rational, those actions can signal hidden information to them, ultimately aiding the team. In the trampoline tiger game, Bob controls a lever that can release a hidden object: either a tiger or a trampoline. Alice stands above and can observe the hidden state, but cannot communicate further with Bob. She has the option to jump, which yields only a small positive joint reward if the hidden state is the trampoline and Bob pulls the lever. Bob can observe whether Alice jumps before pulling the lever. If Alice acts rationally, she will only decide to jump if the hidden state is a trampoline, thereby signalling it to Bob. Notably, never jumping and never pulling the lever is a pure Nash equilibrium. Therefore, algorithms that independently learn Alice's and Bob's policies risk getting stuck in this local maximum. **(author?)** [25] showed that popular MARL algorithms like MAPPO [59] and QMIX [49] converge to this Nash Equilibrium in most cases. LLM agents naturally consider bilateral deviations of Alice's and Bob's actions by directly reasoning about the environment dynamics. As shown in Figure 17b, this allows a variety of LLM agents to determine the optimal policy in this game.

F.5 Local Policies of Text Agents

Since text agents interact with the environment by producing actions sequentially, rather than explicitly specifying joint policies as code, their decision-making is less interpretable. To better understand their behaviour, we empirically estimate their induced local policies.

We estimate the local policies $p_{\text{alice}}(\cdot \mid \text{animal})$ and $p_{\text{bob}}(\cdot \mid a_{\text{alice}})$ by fixing the environment state and repeatedly prompting the agent 100 times as shown in Figure 18. Next, we study how these local policies change under variations in the game's semantics $\mathcal{D}$. To this end, we consider three variants: (1) replacing the binary light-bulb signal ("on/off") with a colour choice ("red/blue"), (2) translating the natural-language descriptions into German, (3) the agent is informed that it is being evaluated against an identical instance of itself.

Note that the agent is given complete information about the Dec-POMDP structure, including the payoffs. Consequently, the natural-language description of the MDP should not affect the induced local policy as long as the underlying Dec-POMDP structure is preserved, with the exception of the final setting, in which the agent has the option to use a convention to maximise reward. Despite this, we observe substantial differences in the induced local policies across semantic variants, as shown in Figure 18. In particular, the SP variant leads to a drastic reduction in XP scores, as discussed in Figure 4.

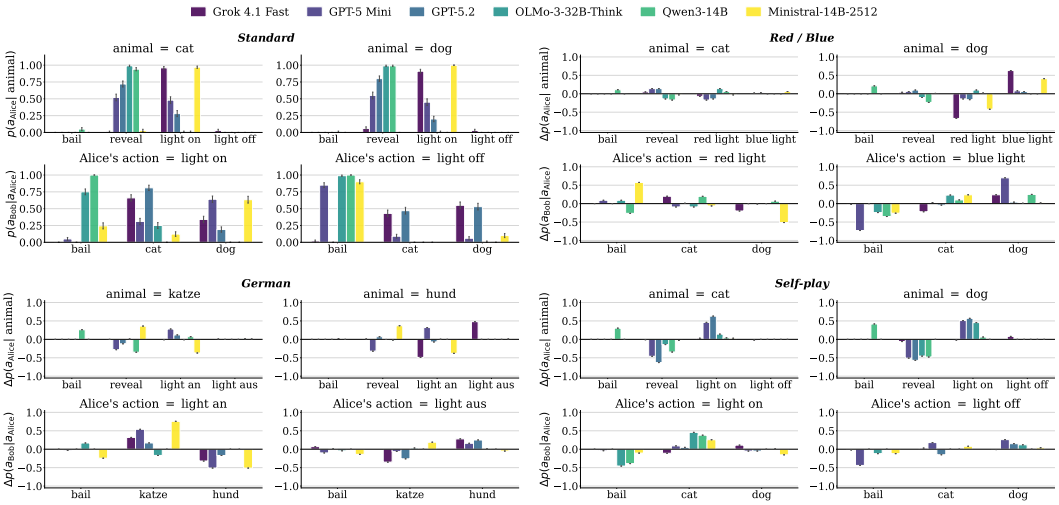


Figure 18: **Local policies of text agents in Cat & Dog and its variants.** The top-left plot shows action probabilities for the standard Cat & Dog environment; each conditional distribution $p(\cdot \mid c)$ is estimated from 100 samples obtained by repeatedly prompting the model from a fixed game state. The remaining plots show the delta, i.e., the change in action probabilities relative to the standard condition, when the environment semantics are varied (changed action labels, German translation, self-play).

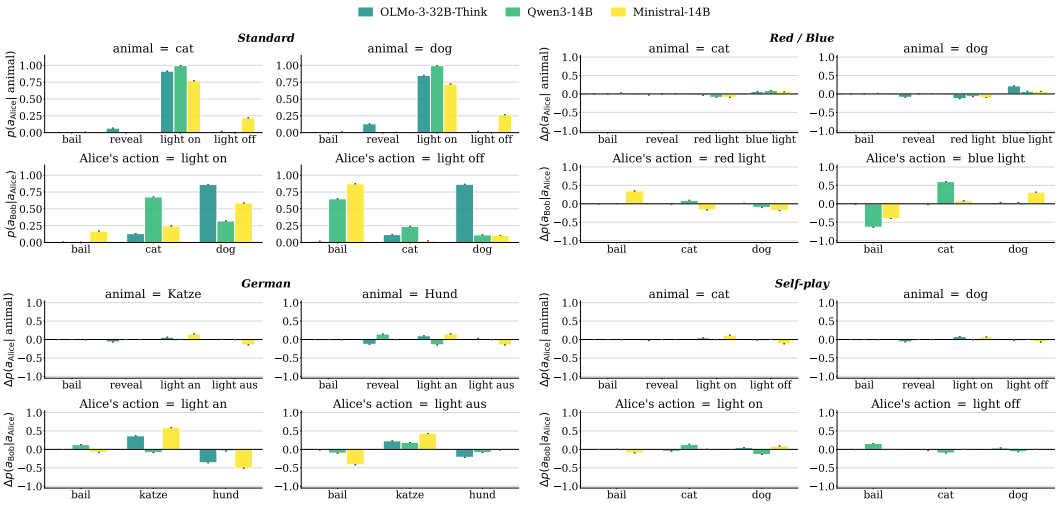


Figure 19: **Token-probability local policies of text agents in Cat & Dog and its variants.** The top-left plot shows action probabilities for the standard Cat & Dog environment, computed from token-level log-probabilities rather than sampling. The remaining plots show the delta, i.e., the change in action probabilities relative to the standard condition, when the environment semantics are varied (changed action labels, German translation, self-play).

Table 3: **XP scores of text agents under varying levels of reasoning effort on Cat & Dog.** We evaluate GPT-5 mini and Grok 4.1 Fast under varying levels of reasoning effort, as well as with and without the use of our CFD. XP scores are reported as mean ± standard error over 100 games.

		Mean return	
Model	Reasoning effort	w/o CFD	w/ CFD
GPT-5 mini	Low	3.23 ± 0.56	4.47 ± 0.14
	Medium	0.97 ± 0.81	4.88 ± 0.07
	High	0.86 ± 0.91	4.92 ± 0.06
Grok 4.1 Fast	Low	1.20 ± 0.98	5.00 ± 0.00
	Medium	-0.75 ± 0.99	5.00 ± 0.00
	High	0.41 ± 0.99	5.00 ± 0.00

Table 4: **Action frequencies of frontier text agents as Alice in Cat & Dog under varying reasoning effort.** We report how often Alice bails, reveals the animal by removing the wall (the ZSC-optimal action), or uses the light as a signal. As in Figure 3, frequencies are estimated by repeatedly prompting each agent 100 times from a fixed game state. Values are reported as mean ± standard error. Higher reasoning effort increases the reveal rate for all three models, most strongly for Claude Opus 5.

Model	Reasoning effort	Bail	Reveal	Signal
GPT-5.6 Sol	Low	0.00 ± 0.00	0.06 ± 0.02	0.94 ± 0.02
	Medium	0.00 ± 0.00	0.06 ± 0.02	0.95 ± 0.02
	High	0.00 ± 0.00	0.11 ± 0.02	0.89 ± 0.02
Claude Opus 5	Low	0.00 ± 0.00	0.00 ± 0.00	1.00 ± 0.00
	Medium	0.00 ± 0.00	0.13 ± 0.02	0.87 ± 0.02
	High	0.00 ± 0.00	0.74 ± 0.03	0.26 ± 0.03
Grok 4.5	Low	0.00 ± 0.00	0.26 ± 0.03	0.75 ± 0.03
	Medium	0.00 ± 0.00	0.46 ± 0.04	0.54 ± 0.04
	High	0.00 ± 0.00	0.49 ± 0.04	0.51 ± 0.04

We also probe whether model internals, specifically token-level action probabilities and chain-of-thought reasoning tokens, are consistent with the empirical local policies and the explicit reasoning outputs elicited before each action. We focus on open-weight models that expose these internals: OLMo-3-32B-Think, Qwen3-14B, and Ministral-14B.

To estimate token-level action probabilities, we compute the log-probability of each legal action completion under a fixed prompt and normalise across legal actions using a softmax. This requires disabling chain-of-thought reasoning, since reasoning tokens precede the action token and would otherwise confound the estimate. We find that disabling reasoning substantially degrades policy quality: models such as OLMo-3-32B-Think and Qwen3-14B, which exhibit strong empirical local policies when reasoning is enabled, produce markedly worse token-level policies in this setting (Figure 19), an effect we attribute to the removal of the reasoning step.

When reasoning is re-enabled, the generated chain-of-thought tokens closely match the content of the explicit reasoning outputs, indicating that the reasoning is faithful to the model's decision process. This supports the validity of our reasoning-based analysis for closed-weight models such as GPT-5.2, for which token-level internals are inaccessible.

F.6 Reasoning Effort

As noted earlier, coordination requires reasoning about a partner's behaviour (ToM). In formal reasoning tasks, increasing reasoning effort, i.e the budget of reasoning tokens available, often yields substantial performance gains. We therefore examine how varying reasoning effort affects the performance of text agents, as reported in Table 3. Surprisingly, for the smaller models (Grok 4.1 Fast and GPT-5 Mini), reasoning effort is negatively correlated with XP score. Further analysis shows that higher reasoning effort increases the likelihood that these models hallucinate a shared convention and assume that their partner will adopt the same convention. That said, we find that the hand-crafted CFD facilitates robust coordination even under increased reasoning effort.

Table 5: **XP scores of text agents in Cat & Dog without a CFD, with few-shot demonstrations, and with the hand-crafted CFD.** Rows denote the LLM playing Alice and columns the LLM playing Bob. The few-shot prompts contain the environment description and a hand-crafted rollout of the ZSC-optimal strategy for Ad-Hoc Conventions, Tiny Hanabi Fixed, and Brittle Bridge. Scores are mean $\pm$ standard error over 5 games per cell. The safe fallback of removing the wall yields a score of 5.

Alice / Bob	Grok 4.5	Gemini 3.1 Pro	DeepSeek V3.2	GPT-5.2
		No CFD		
Grok 4.5	$0.00_{\pm 4.18}$	$4.00_{\pm 3.67}$	$6.00_{\pm 1.00}$	$3.00_{\pm 3.39}$
Gemini 3.1 Pro	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$
DeepSeek V3.2	$-1.00_{\pm 3.67}$	$3.00_{\pm 3.39}$	$7.00_{\pm 1.22}$	$-1.00_{\pm 3.67}$
GPT-5.2	$5.00_{\pm 0.00}$	$2.00_{\pm 3.00}$	$-1.00_{\pm 3.67}$	$6.00_{\pm 1.00}$
		Few-shot demonstrations		
Grok 4.5	$5.00_{\pm 0.00}$	$-1.00_{\pm 3.67}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$
Gemini 3.1 Pro	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$
DeepSeek V3.2	$7.00_{\pm 1.22}$	$7.00_{\pm 1.22}$	$-1.00_{\pm 3.67}$	$6.00_{\pm 1.00}$
GPT-5.2	$2.00_{\pm 3.00}$	$6.00_{\pm 1.00}$	$6.00_{\pm 1.00}$	$2.00_{\pm 3.00}$
		Hand-crafted CFD		
Grok 4.5	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$
Gemini 3.1 Pro	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$
DeepSeek V3.2	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$
GPT-5.2	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$	$5.00_{\pm 0.00}$

We additionally evaluate frontier models (GPT-5.6 Sol, Claude Opus 5, and Grok 4.5) as text agents in Cat & Dog (Table 4). In contrast to the smaller models, increasing reasoning effort *improves* the reveal rate for all three, most strongly for Claude Opus 5 (from 0% to 74%). At the same time, we observe degradation within model families: Qwen3-Max-Thinking performs worse than Qwen3-14B, and GPT-5.6 Sol reveals far less often than GPT-5.2 (Table 4 and Figure 3), despite being positioned as a substantial advance in overall capability. Taken together, these results suggest a **mixed relationship between reasoning capacity and ZSC performance**: more capable models or higher reasoning effort do not reliably translate into stronger zero-shot coordination.

F.7 Few-Shot Demonstrations

A simple alternative to CFDs is to provide agents with in-context demonstrations of successful coordination. Whereas a CFD states the principles underlying successful coordination explicitly, an agent learning from demonstrations must first infer these principles implicitly before transferring them to an unseen environment. To compare the two, we rerun the Cat & Dog XP experiment from Figure 1 with prompts augmented by examples of successful coordination from three other environments. For each, we include the environment description and a hand-crafted rollout of the ZSC-optimal strategy: establishing and maintaining a stable convention across rounds in Ad-Hoc Conventions; selecting the safe fallback (B, B) in Tiny Hanabi Fixed, whose payoff structure closely resembles Cat & Dog; and avoiding bridge collapse by retreating when both players enter the bridge, while establishing an ad-hoc leader–follower convention, in Brittle Bridge. As Grok 4.1 Fast had been deprecated by the time of this experiment, we replace it with Grok 4.5 and regenerate the full XP matrices for all three settings. As shown in Table 5, only the CFD enables reliable coordination across all pairs of agents. While the small sample size makes it difficult to determine whether few-shot demonstrations provide any marginal benefit, they perform substantially worse than the CFD.

F.8 Hanabi 3x3 variant

From Figure 21, we observe that smaller models perform worse on Hanabi 3×3, and that neither CFD leads to score improvements for text agents.

F.9 Hint-Guess Implicatures

We evaluate whether text agents can match human actions, which requires human-like reasoning patterns such as implicatures [38, 17]. To this end, we pair each LLM (Grok-4.1-Fast, Gemini-3.1-Pro,

Table 6: **SP and XP scores of code agents across environments, under fixed (XP_{seed}) and variable (XP_{LLM}) LLM settings, with and without CFD.** Agents tested: GPT-5.2, Grok 4.1 Fast, Claude Opus 4.5, Gemini 3 Pro, and Deepseek V3.2 Terminus.

		w/o CFD			w/ CFD		
Env	LLM	SP	XP_{seed}	XP_{LLM}	SP	XP_{seed}	XP_{LLM}
C.D.	Grok	10.00±0.00	3.33±5.44	0.68±1.75	6.67±0.96	3.50±0.61	3.94±0.37
	GPT	9.58±0.38	9.54±0.37	−3.03±1.88	5.00±0.00	5.00±0.00	4.50±0.38
	Gemini	10.00±0.00	10.00±0.00	−1.63±2.05	5.00±0.00	5.00±0.00	4.72±0.18
	Claude	9.17±0.76	−7.50±2.04	1.20±0.55	5.00±0.00	5.00±0.00	4.72±0.18
	DeepSeek	10.00±0.00	−3.33±5.44	0.26±0.86	5.83±0.76	4.25±0.61	4.44±0.18
	Mean	9.75±0.15	2.41±2.40	−0.51±1.04	5.50±0.30	4.55±0.23	4.46±0.19
A.C.	Grok	100.00±0.00	100.00±0.00	9.64±28.74	54.12±3.76	46.18±3.12	39.53±3.53
	GPT	83.33±9.62	83.33±6.80	62.20±9.21	58.33±7.61	42.50±6.12	38.55±3.36
	Gemini	100.00±0.00	100.00±0.00	−5.92±23.55	63.50±7.79	35.00±6.12	42.98±4.41
	Claude	100.00±0.00	33.33±54.43	23.65±16.36	75.00±10.21	59.17±17.49	36.15±7.07
	DeepSeek	98.33±0.96	64.27±28.16	37.90±18.95	64.66±11.10	18.23±13.02	23.92±4.08
	Mean	96.33±2.92	76.19±13.94	25.49±13.96	63.12±3.15	40.22±5.88	36.23±3.31
A.C.P.S.	Grok	69.73±15.05	36.13±3.16	36.05±7.47	84.12±2.87	19.33±9.85	29.52±6.05
	GPT	72.59±7.33	59.19±11.77	64.24±8.79	57.44±6.67	57.87±3.42	46.69±4.13
	Gemini	90.50±0.00	90.50±0.00	65.43±9.06	90.73±0.16	90.17±0.15	58.62±7.08
	Claude	73.08±15.17	79.05±9.41	59.42±10.39	56.05±19.35	38.35±1.86	47.85±7.97
	DeepSeek	61.08±13.91	8.90±4.12	30.89±7.40	38.02±9.49	−8.14±9.84	35.99±6.39
	Mean	73.40±4.28	54.76±8.25	51.21±6.41	65.27±8.70	39.51±9.10	43.73±4.66
T. H.	Grok	10.00±0.00	10.00±0.00	7.35±1.00	9.53±0.31	8.56±1.18	7.08±0.46
	GPT	9.67±0.30	8.72±1.05	7.27±0.83	9.67±0.30	8.95±0.86	6.70±0.92
	Gemini	10.00±0.00	10.00±0.00	7.32±0.99	10.00±0.00	10.00±0.00	7.18±0.85
	Claude	9.32±0.31	2.64±0.56	6.03±0.41	9.00±0.33	4.01±0.95	6.08±0.49
	DeepSeek	8.42±0.33	5.63±1.53	4.41±0.66	7.80±0.62	6.87±0.93	5.70±0.85
	Mean	9.48±0.26	7.40±0.84	6.48±0.57	9.20±0.34	7.68±0.67	6.55±0.49
T.H.F	Grok	10.00±0.00	10.00±0.00	9.71±0.19	9.67±0.30	8.00±1.63	7.05±0.91
	GPT	9.67±0.30	9.67±0.27	9.58±0.22	8.67±0.38	5.33±1.09	6.57±0.53
	Gemini	10.00±0.00	10.00±0.00	9.71±0.19	8.00±0.00	8.00±0.00	8.42±0.28
	Claude	10.00±0.00	10.00±0.00	9.71±0.19	9.67±0.30	8.00±1.63	7.36±0.89
	DeepSeek	9.67±0.30	8.00±1.63	8.96±0.25	8.00±0.00	8.00±0.00	6.10±0.87
	Mean	9.87±0.07	9.53±0.39	9.53±0.15	8.80±0.33	7.47±0.58	7.10±0.50
B.B	Grok	−9.00±0.00	−9.00±0.00	−172.53±72.54	−14.99±4.88	−28.40±15.10	−123.77±57.98
	GPT	−12.84±0.40	−12.76±0.33	−24.48±5.35	−13.29±0.70	−12.23±0.11	−16.68±1.67
	Gemini	−10.00±0.00	−10.00±0.00	−120.70±69.36	−12.49±0.09	−12.51±0.07	−17.44±2.11
	Claude	−292.74±102.20	−11.91±0.09	−146.69±36.19	−197.78±55.21	−127.05±31.52	−122.91±50.84
	DeepSeek	−598.45±180.85	−388.02±153.55	−269.83±76.67	−867.50±121.87	−671.00±269.44	−186.36±68.69
	Mean	−184.61±104.65	−86.34±49.60	−146.85±41.20	−221.21±147.99	−170.24±85.17	−93.43±32.78
N.T	Grok	−5.17±0.15	−5.17±0.14	−17.06±5.40	−5.00±0.00	−36.67±25.86	−19.36±6.98
	GPT	−5.33±0.30	−5.67±0.27	−13.33±3.96	−5.72±0.34	−6.03±0.08	−7.88±1.82
	Gemini	−5.00±0.00	−5.00±0.00	−17.00±5.40	−5.92±0.03	−5.96±0.04	−6.03±0.02
	Claude	−5.00±0.00	−68.33±25.86	−34.83±3.29	−5.00±0.00	−68.33±25.86	−29.19±7.25
	DeepSeek	−36.83±18.24	−36.83±12.79	−26.94±6.20	−20.83±14.45	−20.83±12.93	−17.31±5.99
	Mean	−11.47±5.67	−24.20±8.70	−21.83±3.61	−8.49±2.76	−27.57±9.82	−15.96±3.84
M.G	Grok	1.00±0.00	1.00±0.00	0.96±0.03	1.00±0.00	1.00±0.00	0.83±0.10
	GPT	1.00±0.00	1.00±0.00	0.96±0.03	1.00±0.00	1.00±0.00	0.83±0.10
	Gemini	1.00±0.00	1.00±0.00	0.96±0.03	1.00±0.00	1.00±0.00	0.83±0.10
	Claude	1.00±0.00	1.00±0.00	0.96±0.03	1.00±0.00	1.00±0.00	0.83±0.10
	DeepSeek	1.00±0.00	0.67±0.27	0.83±0.00	1.00±0.00	0.01±0.01	0.33±0.00
	Mean	1.00±0.00	0.93±0.06	0.93±0.02	1.00±0.00	0.80±0.10	0.73±0.07
P.L	Grok	5.11±0.20	3.00±0.25	4.29±0.15	5.30±0.21	4.67±0.62	4.55±0.24
	GPT	5.44±1.01	7.87±0.44	6.05±0.46	7.29±0.49	6.90±0.17	6.09±0.40
	Gemini	6.54±0.10	6.44±0.03	5.78±0.31	8.36±0.06	8.37±0.07	6.40±0.49
	Claude	2.27±0.22	9.18±0.01	6.20±0.42	3.57±1.02	6.56±1.87	6.28±0.51
	DeepSeek	3.53±0.67	4.00±0.81	5.15±0.31	4.33±0.65	5.65±0.36	6.27±0.20
	Mean	4.58±0.67	6.10±0.63	5.50±0.27	5.77±0.80	6.43±0.51	5.92±0.29

GPT-5.4, and Claude-Sonnet-4.6) with every other model, sampling each game type in Figure 14 three times per pairing. The results, reported in Table 8, show that LLM agents are able to align with human-like reasoning, achieving performance comparable to state-of-the-art methods [37] (Table 2).

F.10 Automatic discovery of CFDs

We report the cross-LLM and scaffold generalization scores of the meta-learned CFD in Figure 22 and Figure 23.

Table 7: **SP and XP failure rates of code agents across environments, under fixed- and variable-LLM settings, with and without CFD.** Agents tested: GPT-5.2, Grok 4.1 Fast, Claude Opus 4.5, Gemini 3 Pro, and Deepseek V3.2 Terminus.

		w/o CFD			w/ CFD		
Env	LLM	SP	XP_{seed}	XP_{LLM}	SP	XP_{seed}	XP_{LLM}
C.D.	Grok	0.00 ± 0.00	0.33 ± 0.27	0.46 ± 0.09	0.00 ± 0.00	0.33 ± 0.27	0.02 ± 0.02
	GPT	0.00 ± 0.00	0.00 ± 0.00	0.63 ± 0.10	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.02
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.57 ± 0.10	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
	Claude	0.00 ± 0.00	0.83 ± 0.14	0.42 ± 0.03	0.00 ± 0.00	0.83 ± 0.14	0.00 ± 0.00
	DeepSeek	0.00 ± 0.00	0.67 ± 0.27	0.48 ± 0.05	0.00 ± 0.00	0.67 ± 0.27	0.00 ± 0.00
	Mean	0.00 ± 0.00	0.37 ± 0.12	0.51 ± 0.05	0.00 ± 0.00	0.37 ± 0.12	0.01 ± 0.01
A.C.	Grok	0.00 ± 0.00	0.00 ± 0.00	0.46 ± 0.14	0.00 ± 0.00	0.00 ± 0.00	0.29 ± 0.07
	GPT	0.00 ± 0.00	0.00 ± 0.00	0.16 ± 0.04	0.00 ± 0.00	0.00 ± 0.00	0.29 ± 0.07
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.54 ± 0.13	0.00 ± 0.00	0.00 ± 0.00	0.26 ± 0.05
	Claude	0.00 ± 0.00	0.33 ± 0.27	0.39 ± 0.09	0.00 ± 0.00	0.33 ± 0.27	0.33 ± 0.08
	DeepSeek	0.00 ± 0.00	0.33 ± 0.27	0.39 ± 0.11	0.14 ± 0.13	0.33 ± 0.27	0.52 ± 0.04
	Mean	0.00 ± 0.00	0.13 ± 0.09	0.39 ± 0.07	0.03 ± 0.03	0.13 ± 0.09	0.34 ± 0.05
A.C.P.S.	Grok	0.16 ± 0.14	0.60 ± 0.05	0.42 ± 0.05	0.00 ± 0.00	0.55 ± 0.06	0.47 ± 0.07
	GPT	0.08 ± 0.08	0.28 ± 0.17	0.20 ± 0.09	0.12 ± 0.11	0.28 ± 0.17	0.36 ± 0.06
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.17 ± 0.07	0.00 ± 0.00	0.00 ± 0.00	0.23 ± 0.08
	Claude	0.08 ± 0.08	0.08 ± 0.07	0.19 ± 0.07	0.14 ± 0.08	0.09 ± 0.07	0.30 ± 0.07
	DeepSeek	0.12 ± 0.07	0.62 ± 0.06	0.43 ± 0.07	0.33 ± 0.12	0.65 ± 0.04	0.45 ± 0.08
	Mean	0.09 ± 0.02	0.32 ± 0.08	0.28 ± 0.05	0.12 ± 0.05	0.31 ± 0.08	0.36 ± 0.05
T.H.	Grok	0.00 ± 0.00	0.00 ± 0.00	0.24 ± 0.10	0.00 ± 0.00	0.00 ± 0.00	0.23 ± 0.05
	GPT	0.00 ± 0.00	0.08 ± 0.06	0.23 ± 0.08	0.00 ± 0.00	0.07 ± 0.06	0.27 ± 0.09
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.24 ± 0.10	0.00 ± 0.00	0.00 ± 0.00	0.25 ± 0.09
	Claude	0.00 ± 0.00	0.66 ± 0.06	0.34 ± 0.05	0.00 ± 0.00	0.65 ± 0.06	0.31 ± 0.06
	DeepSeek	0.05 ± 0.04	0.36 ± 0.18	0.50 ± 0.08	0.07 ± 0.07	0.34 ± 0.19	0.33 ± 0.10
	Mean	0.01 ± 0.01	0.22 ± 0.08	0.31 ± 0.06	0.01 ± 0.01	0.21 ± 0.08	0.28 ± 0.05
T.H.F	Grok	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.18 ± 0.10
	GPT	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.21 ± 0.07
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
	Claude	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.02	0.00 ± 0.00	0.00 ± 0.00	0.18 ± 0.10
	DeepSeek	0.00 ± 0.00	0.17 ± 0.14	0.08 ± 0.03	0.00 ± 0.00	0.17 ± 0.14	0.23 ± 0.11
	Mean	0.00 ± 0.00	0.03 ± 0.03	0.03 ± 0.01	0.00 ± 0.00	0.03 ± 0.03	0.16 ± 0.06
B.B	Grok	0.00 ± 0.00	0.00 ± 0.00	0.19 ± 0.08	0.01 ± 0.01	0.00 ± 0.00	0.15 ± 0.05
	GPT	0.01 ± 0.00	0.01 ± 0.01	0.08 ± 0.03	0.04 ± 0.02	0.02 ± 0.02	0.04 ± 0.01
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.13 ± 0.07	0.02 ± 0.00	0.00 ± 0.00	0.04 ± 0.01
	Claude	0.62 ± 0.14	0.00 ± 0.00	0.18 ± 0.03	0.31 ± 0.14	0.00 ± 0.00	0.16 ± 0.06
	DeepSeek	0.58 ± 0.18	0.49 ± 0.24	0.34 ± 0.06	1.00 ± 0.00	0.50 ± 0.24	0.20 ± 0.08
	Mean	0.24 ± 0.13	0.10 ± 0.07	0.18 ± 0.04	0.28 ± 0.17	0.10 ± 0.07	0.12 ± 0.03
N.T	Grok	0.00 ± 0.00	0.00 ± 0.00	0.12 ± 0.06	0.00 ± 0.00	0.00 ± 0.00	0.15 ± 0.07
	GPT	0.00 ± 0.00	0.00 ± 0.00	0.08 ± 0.04	0.00 ± 0.00	0.00 ± 0.00	0.02 ± 0.02
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.12 ± 0.06	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
	Claude	0.00 ± 0.00	0.67 ± 0.27	0.31 ± 0.04	0.00 ± 0.00	0.67 ± 0.27	0.25 ± 0.08
	DeepSeek	0.33 ± 0.19	0.33 ± 0.14	0.23 ± 0.07	0.17 ± 0.15	0.33 ± 0.14	0.13 ± 0.06
	Mean	0.07 ± 0.06	0.20 ± 0.09	0.17 ± 0.04	0.03 ± 0.03	0.20 ± 0.09	0.11 ± 0.04
M.G	Grok	0.00 ± 0.00	0.00 ± 0.00	0.04 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.17 ± 0.10
	GPT	0.00 ± 0.00	0.00 ± 0.00	0.04 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.17 ± 0.10
	Gemini	0.00 ± 0.00	0.00 ± 0.00	0.04 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.17 ± 0.10
	Claude	0.00 ± 0.00	0.00 ± 0.00	0.04 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.17 ± 0.10
	DeepSeek	0.00 ± 0.00	0.33 ± 0.27	0.17 ± 0.00	0.00 ± 0.00	0.33 ± 0.27	0.67 ± 0.00
	Mean	0.00 ± 0.00	0.07 ± 0.06	0.07 ± 0.02	0.00 ± 0.00	0.07 ± 0.06	0.27 ± 0.07
P.L	Grok	0.20 ± 0.07	0.02 ± 0.01	0.15 ± 0.02	0.31 ± 0.05	0.02 ± 0.02	0.21 ± 0.04
	GPT	0.30 ± 0.15	0.02 ± 0.01	0.11 ± 0.03	0.07 ± 0.03	0.01 ± 0.01	0.13 ± 0.03
	Gemini	0.12 ± 0.01	0.09 ± 0.01	0.10 ± 0.00	0.00 ± 0.00	0.11 ± 0.01	0.12 ± 0.03
	Claude	0.79 ± 0.02	0.00 ± 0.00	0.11 ± 0.03	0.61 ± 0.13	0.00 ± 0.00	0.15 ± 0.04
	DeepSeek	0.49 ± 0.08	0.37 ± 0.18	0.15 ± 0.01	0.39 ± 0.09	0.40 ± 0.19	0.06 ± 0.02
	Mean	0.38 ± 0.11	0.10 ± 0.05	0.12 ± 0.01	0.28 ± 0.10	0.11 ± 0.05	0.14 ± 0.02

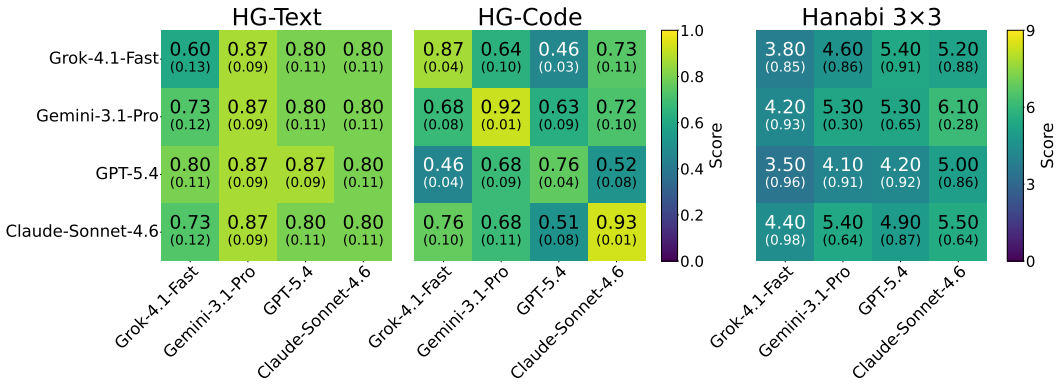


Figure 20: **XP matrices of LLM text and code agents.** (Left and Middle) XP matrix in the Hint-Guess game; text agents coordinate well, whereas code agents do not. Human players score 0.75 on average [37]. (Right) XP matrix of text agents on the 3×3 Hanabi variant; text agents fall well short of the 7.7 achieved by IPPO with high entropy regularization [14]. For text agents, each cell reports the mean and s.e.m. over 10 episodes; for code agents, each cell is averaged over 100 episodes across 6 independent XP matrices.

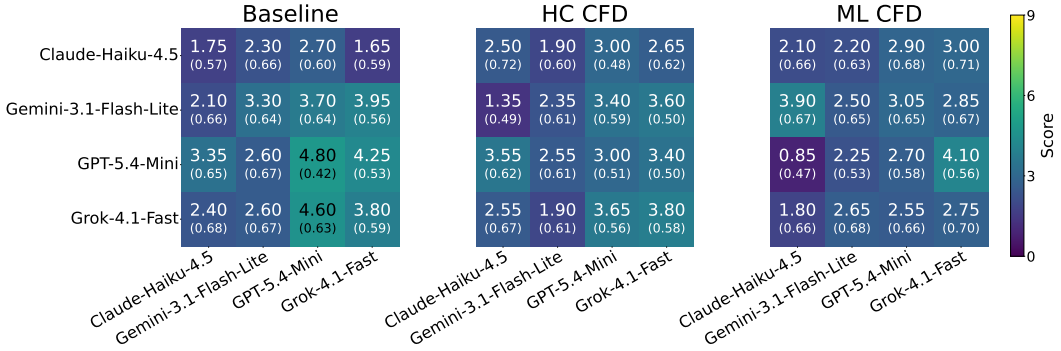


Figure 21: **Hanabi 3×3 results of text agents with and without a CFD.** We find that the LLM agents achieve low scores in all 3 cases.

Table 8: **Hint-Guess Implicatures: Human-compatible hint rate and win rate (%) by scenario.** Scenarios correspond exactly to those in Figure 14. Each cell shows Human % / Win % (n=12 games per hinter–scenario cell). *Exact*: target card available in hinter's hand. *Similarity (S)*: feature similarity required, no exact match. *Exclusivity (E)*: mutual exclusivity (Gricean implicature) required. *S+E*: combined similarity + exclusivity scenario. All models give the human-compatible hint perfectly on Exact, Sim, and Excl; failures concentrate in the S+E condition.

Hinter	Exact Match	Similarity (S)	Exclusivity (E)	S+E
Claude Sonnet 4.6	100 / 100	100 / 100	100 / 100	100 / 100
Gemini 3.1 Pro	100 / 100	100 / 100	100 / 100	100 / 92
GPT-5.4	100 / 100	100 / 100	100 / 100	100 / 100
Grok 4.1 Fast	100 / 100	100 / 100	100 / 100	75 / 75
All	100 / 100	100 / 100	100 / 100	94 / 92

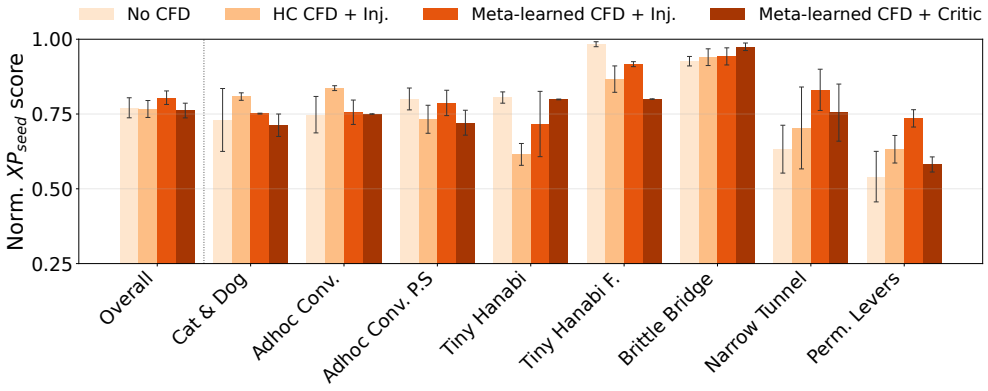


Figure 22: **XP_{seed} scores of code agents aggregated across different LLM types and scaffolds.** Results are shown for Grok 4.1 Fast, GPT-5.2, Gemini 3 Pro, and Opus 4.5. Since the CFD was meta-trained with Grok 4.1 Fast on Cat & Dog and Brittle Bridge, Grok 4.1 Fast is excluded from these two environments in all conditions, including the overall score. Scores are min–max normalised per environment using fixed reward ranges, and error bars denote the standard error over seed pairs.

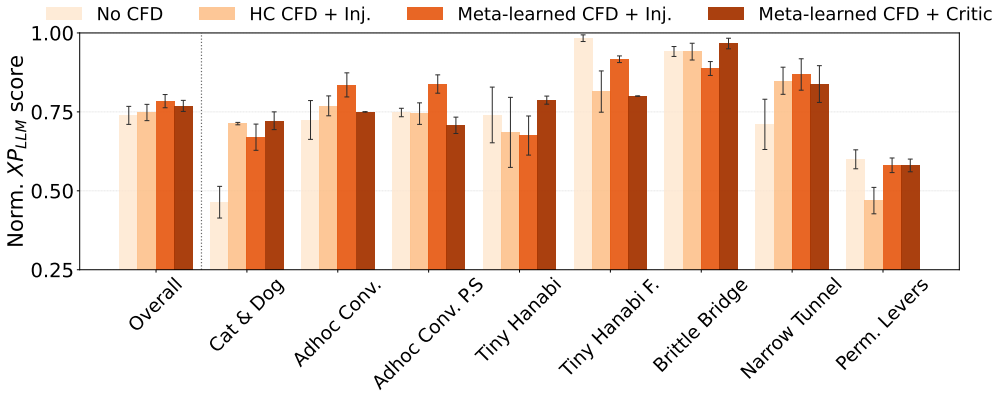


Figure 23: **XP_{LLM} scores of code agents aggregated across different LLM types and scaffolds.** Results are shown for Grok 4.1 Fast, GPT-5.2, Gemini 3 Pro, and Opus 4.5.

G Implementation details

G.1 Meta-Learning Loop

We outline the algorithm used in our pareto-evolution meta-learning loop in Algorithms 1, 2, and 3. The algorithm is roughly based on GEPA [2], implementing both pareto-based candidate selection and the cheap pre-filtering of new CFDs. Still, there are notable differences to GEPA. In particular, GEPA cannot be applied directly to the ZSC setting as the number of environments is generally far smaller than the number of "tasks" assumed in GEPA. Compared to GEPA, we therefore compute the actual pareto frontier in Algorithm 2 across our meta-train set, without any previous task-wise elite filtering. We also adopt a caching mechanism for the set of initial policies used in each scoring step. This significantly reduces the costs of running the meta-loop. Additionally, by using the same set of initial policies when comparing CFDs, we significantly reduce the variance in XP-scores (as we eliminate the source of variance from the initial generation step), which generally leads to a more stable meta-loop.
At last, the meta-loop differs in the used mutation operators. Similar to GEPA, we tried to include environment feedback into the mutation operators used in the meta-loop, both through direct or obfuscated conditioning on observed coordination failures. In general, we found that the inclusion of such operators unhelpful, often leading to either overfitting or low-quality uni-modal proposals. Instead, we found that ensuring the creation of diverse proposals in the mutation operators was of key importance to the success of the meta-loop. To facilitate such proposals, we apply the following modifications: Mutation operators generally prompt an LLM for a specific kind of modification of the CFD instead of leaving the modification to the LLM. This is similar to the mutation prompts used in PromptBreeder [11]. In our case, the LLM is prompted to either add, remove, reformulate, or freely modify a bullet point in the CFD. Orthogonal to this, we found that prompting an LLM for a set of diverse proposals and then sampling the final proposal uniformly at random from this set can significantly increase diversity, as it allows us to sample proposals from the long tail, which would normally not make it into the LLM's first response. These two axes yield a total of eight mutation operators. We also add the re-initialization of CFDs as an additional mutation operator.

Hyperparameter choices. We outline in this paragraph the hyperparameter choices used in Algorithm 3 to obtain the CFD described in Appendix 6. The distribution over the mutation operators ν assigns each a sample weight of 1, except the re-initialization operator, which receives a weight of 2. To increase the diversity of proposals and generated policies, the meta-optimisation agent and the code agents generating policies during meta-training use a temperature of 1.0, while the policy critic uses a temperature of 0. The remaining hyperparameter choices are outlined in Table 9.

Symbol	Value	Quick explanation
$\mathcal{E}$	{CATANDDOG, BRITTLEBRIDGE}	Meta-train coordination environments
K	5	Number of seed CFDs sampled at initialisation
T_{evo}	20	Outer-loop evolution steps
n_c	10	Size of the XP-matrix in the cheap single-env gate
n_f	15	Size of the XP-matrix in the full all-envs evaluation
L	3	Max retries to draw a non-duplicate CFD
T_{crit}	4	Critic iteration budget per starting policy

Table 9: Inputs to Algorithm 3 used in our experiments.

Algorithm 1 SCORE: evaluate a CFD on one environment via $n \times n$ cross-play

Require: CFD c; Dec-POMDP $\mathcal{M}$ with natural-text descriptions $(\mathcal{D}_{\mathcal{M}}, \mathcal{O}_{\mathcal{M}}, \mathcal{A}_{\mathcal{M}})$ of the game, observations, and actions; preset starting policies $\Pi = (\pi_0^{(1)}, \ldots, \pi_0^{(n)})$, one per seed; critic iteration budget T_{crit}
Ensure: mean off-diagonal cross-play score $\sigma \in \mathbb{R}$

procedure SCORE($c, \mathcal{M}, \Pi, T_{\text{crit}}$)
  $n \leftarrow |\Pi|$
  for $i = 1$ **to** n **do**
    $(\pi^{(i)}, \cdot) \leftarrow \text{REFINE}(\pi_0^{(i)}, \mathcal{M}, c, T_{\text{crit}})$ ▷ see Algorithm 5
  end for
  $X \in \mathbb{R}^{n \times n}$ with $X[i,j] = \mathbb{E}[J(\pi^{(i)}, \pi^{(j)})]$ ▷ cross-play return matrix
  return $\dfrac{1}{n^2 - n} \sum_{i \neq j} X[i,j]$
end procedure

Algorithm 2 PARETOFRONTIER: non-dominated subset of a candidate set

Definition (Pareto dominance). Given candidates c, c' with per-environment scores $\sigma_c, \sigma_{c'} \in \mathbb{R}^{|\mathcal{E}|}$, c' *dominates* c iff $\sigma_{c'}(\mathcal{M}) \geq \sigma_c(\mathcal{M})$ for all $\mathcal{M} \in \mathcal{E}$ and $\sigma_{c'}(\mathcal{M}) > \sigma_c(\mathcal{M})$ for at least one $\mathcal{M} \in \mathcal{E}$.

Require: candidate set P; environment set $\mathcal{E}$
Ensure: non-dominated subset $F \subseteq P$

procedure PARETOFRONTIER($P, \mathcal{E}$)
  $F \leftarrow \varnothing$
  for $c \in P$ **do**
    if no $c' \in P \setminus \{c\}$ dominates c **then**
      $F \leftarrow F \cup \{c\}$
    end if
  end for
  return F
end procedure

Algorithm 3 PARETOEVOLUTION: meta-learning over CFDs across environments

Require: environment set $\mathcal{E}$ (Dec-POMDPs); seed population size K; evolution steps T_{evo}; cheap-eval seeds n_c; full-eval seeds n_f; mutation operator distribution ν; max mutation retries L; critic iteration budget T_{crit}

Ensure: selected CFD $c^\star$

procedure PARETOEVOLUTION($\mathcal{E}, K, T_{\text{evo}}, n_c, n_f, \nu, L, T_{\text{crit}}$)

Phase 1: Initialization.

pre-sample fixed starting policies $\hat{\Pi}_{\mathcal{M}}$ for each $\mathcal{M} \in \mathcal{E}$ ▷ $|\hat{\Pi}_{\mathcal{M}}| = n_f$ (one per seed)

sample K diverse seed CFDs $\{c_1, \ldots, c_K\}$

$P \leftarrow \varnothing$

for $c \in \{c_1, \ldots, c_K\}$ **do**

$\sigma_c(\mathcal{M}) \leftarrow \text{SCORE}(c, \mathcal{M}, \hat{\Pi}_{\mathcal{M}}, T_{\text{crit}})$ for all $\mathcal{M} \in \mathcal{E}$

$P \leftarrow P \cup \{c\}$

end for

$P_{\text{alive}} \leftarrow P$ ▷ all seeds initially alive

Phase 2: Evolution.

for $t = 1$ **to** T_{evo} **do**

$F \leftarrow \text{PARETOFRONTIER}(P_{\text{alive}}, \mathcal{E})$

$c_p \leftarrow$ uniform sample from F ▷ parent

Mutation with duplicate retry:

$c_x \leftarrow$ `nil`

for $\ell = 1$ **to** L **do**

$m \sim \nu$; $c \leftarrow m(c_p)$

if $c \notin \{\text{CFDs in } P\}$ **then**

$c_x \leftarrow c$

break

end if

end for

if $c_x =$ `nil` **then**

continue ▷ no novel child after L tries

end if

Cheap gate: same fresh starting policies for child and parent.

$\mathcal{M} \leftarrow$ uniform sample from $\mathcal{E}$

$\tilde{\Pi} \leftarrow$ fresh sample of n_c starting policies for $\mathcal{M}$

if $\text{SCORE}(c_x, \mathcal{M}, \tilde{\Pi}, T_{\text{crit}}) \leq \text{SCORE}(c_p, \mathcal{M}, \tilde{\Pi}, T_{\text{crit}})$ **then**

continue ▷ discard child

end if

Full evaluation on all environments:

$\sigma_{c_x}(\mathcal{M}) \leftarrow \text{SCORE}(c_x, \mathcal{M}, \hat{\Pi}_{\mathcal{M}}, T_{\text{crit}})$ for all $\mathcal{M} \in \mathcal{E}$

Frontier admission:

$F' \leftarrow \text{PARETOFRONTIER}(P_{\text{alive}} \cup \{c_x\}, \mathcal{E})$

if $c_x \in F'$ **then**

$P \leftarrow P \cup \{c_x\}$

$P_{\text{alive}} \leftarrow F'$ ▷ dominated candidates removed

end if

end for

Phase 3: Final Selection.

From the final frontier, return the candidate with the highest mean score after per-environment min–max normalisation. This balances performance across environments without privileging those with larger absolute score ranges.

$F^\star \leftarrow \text{PARETOFRONTIER}(P_{\text{alive}}, \mathcal{E})$

for $\mathcal{M} \in \mathcal{E}$ **do**

$lo_{\mathcal{M}} \leftarrow \min_{c \in F^\star} \sigma_c(\mathcal{M})$; $\quad hi_{\mathcal{M}} \leftarrow \max_{c \in F^\star} \sigma_c(\mathcal{M})$

end for

return $\arg\max_{c \in F^\star} \dfrac{1}{|\mathcal{E}|} \displaystyle\sum_{\mathcal{M} \in \mathcal{E}} \dfrac{\sigma_c(\mathcal{M}) - lo_{\mathcal{M}}}{hi_{\mathcal{M}} - lo_{\mathcal{M}}}$

end procedure

G.2 Policy Critic

We outline in Algorithm 4 and Algorithm 5 how the policy critic can be used to iteratively refine starting policies π_0. Starting policies are generated using the code-agent generation process as outlined in Figure 15, but without conditioning on any CFD. We provide the prompt used for the policy critic in Figure 26. Separating generation from verfication can increase diversity [3], as well as reduce the occurrence of false positives in the policy creation pipeline.

Algorithm 4 CRITICSTEP: single critic pass over a policy

Require: candidate policy π; Dec-POMDP $\mathcal{M}$ with natural-text descriptions $(\mathcal{D}_{\mathcal{M}}, \mathcal{O}_{\mathcal{M}}, \mathcal{A}_{\mathcal{M}})$ of the game, observations, and actions; coordination-friendly definition c; critic language model LLM
Ensure: verdict $cf \in \{\text{true}, \text{false}\}$; revised policy π' (with $\pi' = \pi$ when $cf = \text{true}$)
procedure CRITICSTEP($\pi, \mathcal{M}, c$)
  $p \leftarrow$ BUILDCRITICPROMPT($\pi, \mathcal{M}, c$)
  resp $\leftarrow \text{LLM}(p,\ \text{temperature} = 0)$
  parse resp as JSON with fields `coordination_friendly`, `modified_policy`
  if resp.`coordination_friendly` **then**
    $\pi' \leftarrow \pi$
  else
    $\pi' \leftarrow$ PARSE(resp.`modified_policy`)
  end if
  return (resp.`coordination_friendly`, π')
end procedure

Algorithm 5 REFINE: iterative policy refinement via the critic

Require: starting policy π_0; Dec-POMDP $\mathcal{M}$; coordination-friendly definition c; iteration budget T
Ensure: final policy $\pi^\star$; flag *approved* indicating whether $\pi^\star$ was confirmed coordination-friendly
procedure REFINE($\pi_0, \mathcal{M}, c, T$)
  $\pi \leftarrow \pi_0$
  for $t = 1$ **to** T **do**
    $(cf,\ \pi') \leftarrow$ CRITICSTEP($\pi, \mathcal{M}, c$)
    if cf **then**
      return $(\pi,\ \text{true})$
    end if
    $\pi \leftarrow \pi'$ ▷ adopt the proposed revision
  end for
  return $(\pi,\ \text{false})$ ▷ budget exhausted; return last revision
end procedure

G.3 Prompts

The prompt template for text agents is shown in Figure 24, for code agents in Figure 25, and for the policy critic in Figure 26. To ensure consistent policies and reasonable actions, we use a temperature of 0 for text and code agents unless otherwise specified. We note that even at these settings, most frontier models are non-deterministic [21].

Prompt Template — Text Agents

You are playing a cooperative game with another player. The game is generally turn-based. While you cannot communicate with the other player, you can observe their actions and the joint rewards from previous turns. Your objective is to maximise the joint reward.

You will play only a single round of the game (a round may consist of multiple turns).

Below is a detailed description of the game and, if applicable, the history of previous turns.

`$Game Description$` `$Game History$`

You must now select an action for the current turn from the following set of legal actions:

`$Legal Actions$` `$CFD Instructions (if applicable)$`

Respond using the following format:

````
REASONING:
<REASONING>
ACTION JSON:
```json
<JSON>
```
````

Figure 24: Template of the prompt used for text agents. Blue-shaded fields are filled in at runtime.

The prompt templates used by the mutation operators in our meta-learning loop are shown in Figure 27, Figure 28, Figure 29, Figure 30, and Figure 31. All mutation prompts begin with the shared background block in Figure 27. The four `unary_*` operators (`add`, `remove`, `reformulate`, `atomic`) share the template in Figure 28; the thinking-style block from Figure 29 specifies which atomic edit the LLM is asked to perform. The four `diverse_*` operators reuse the same thinking-style blocks but ask for $n_{\text{proposals}}$ diverse candidates per call (Figure 30); we then sample one proposal uniformly at random. The `reinit` operator generates fresh single-bullet CFDs from scratch using the seed template in Figure 31.

G.4 Experimental Resources

Running all experiments reported in this paper incurred approximately $1,000 in API costs across the evaluated LLM agents. The token-level probability experiments (Appendix F.5) for the open-weight models were run locally on a single H100 GPU.

Prompt Template — Code Agents

General Instructions

You will need to generate policies for a collaborative two-player game. The game is generally turn-based. Agents cannot communicate with each other. Agents can see their actions and the (joint) reward from previous turns. Generally, agents play each game only once; if the game is iterated this will be stated explicitly. **Ensure the policy is coordination-friendly.**

Zero-Shot Coordination (ZSC) Setting

The policy will be evaluated against independently generated policies (potentially from different LLMs) with similar instructions. Poor coordination often arises when policies adopt incompatible conventions or break symmetry arbitrarily. **Ultimate goal:** maximise coordination in the ZSC setting.

`$CFD Instructions (if applicable)$`

Policy Specification

Each player has a local policy: an `act` function with the following signature:

```
def act(observation_dict: dict, memory: dict) -> tuple[str, dict]
```

The first return value is the action (string); the second is the memory dictionary (initialised to `{}` at game start, not shared between players). You may use Python standard libraries `random` and `math`.

Observation dict — index values using the *exact* keys shown below (case-sensitive):

`$Sample Observation Dict$`

Legal actions — the returned action must exactly match one of:

`$Legal Actions Dict$`

Policies that return an invalid action end the game immediately with a penalty.

Game description:

`$Game Description$`

Output Specification

Propose 1 joint policy (one `act` function per player). Format the output as JSON under the key `"joint_policies"`:

```
{
"joint_policies": [{
"name": <name shared by all players>,
"reason": <why this design>,
"player_0": { "code": <escaped Python string for act> },
"player_1": { "code": <escaped Python string for act> }
}]
}
```

```
REASONING:
<REASONING>
Policy JSON:
```json
<JSON>
```
```

Figure 25: Template of the prompt used for code agents. Blue-shaded fields are filled in at runtime.

Prompt Template — Policy Critic

General Instructions

You are a policy critic. Your goal is to judge whether a given policy is *coordination-friendly* according to the definition below. "Coordination-friendly" is a defined term: do not add your own interpretation, even if the definition feels short or incomplete — reason strictly from it. It is critical that you never classify a coordination-unfriendly policy as coordination-friendly.

Judge each local policy independently. Evaluate `player_0` and `player_1` as if each existed in isolation, without knowledge of the other. At deployment, each local policy is paired with an independently generated partner — not with the partner shown here. A joint policy is coordination-friendly *only if* both local policies are individually compliant.

Also return a modified policy.

- If the policy is not coordination-friendly, propose a modification that is, ensuring each local policy independently satisfies the CFD.
- Among compliant modifications, prefer ones that maximise expected joint reward.
- If the policy is already coordination-friendly, return it unchanged as the modified policy.

Coordination-Friendly Definition (CFD)

`$Coordination-Friendly Definition$`

Policy under Review

`$Policy (player_0 and player_1 code)$`

Game Description

The policy was generated for the game described below.

`$Game Description$`

Policy Specification

Each player has a local policy: an `act` function with the following signature:

```
def act(observation_dict: dict, memory: dict) -> tuple[str, dict]
```

The first return value is the action (string); the second is the memory dictionary (initialised to `{}` at game start, not shared between players). Policies may use Python standard libraries `random` and `math`.

Observation dict — index values using the *exact* keys shown below (case-sensitive):

`$Sample Observation Dict$`

Legal actions — the returned action must exactly match one of:

`$Legal Actions Dict$`

Policies that return an invalid action end the game immediately with a penalty.

Output Specification

Return a single JSON object under the structure below:

```
{
"coordination_friendly": <true or false -- verdict on the original policy>,
"reasoning": <why the original policy is or is not coordination-friendly>,
"modified_policy": {
"name": <name shared by both players>,
"reasoning": <why the modification is designed this way>,
"player_0": { "code": <escaped Python string for act> },
"player_1": { "code": <escaped Python string for act> }
}
}
```

```
REASONING:
<REASONING>
JSON:
```

````
```json
<JSON>
```
````

Figure 26: Template of the prompt used for the policy critic. Blue-shaded fields are filled in at runtime.

Prompt Template: Shared Mutation Background

Background

In zero-shot coordination (ZSC), independently trained policies must coordinate successfully without any shared training or communication.

Game Structure:

- Fully cooperative two-player games
- Each player's behavior is defined by a Python function (local policy)
- Two local policies together form a joint policy

Training Process:

- N independent training runs (seeds) are conducted
- In each seed, an LLM generates a policy candidate
- A **policy critic** then reviews each candidate against the CFD. The critic judges each local policy (`player_0`, `player_1`) **independently** — as if it exists in isolation, without knowledge of the other player's policy. A joint policy is only approved if both local policies individually satisfy the CFD. The critic also tries to maximize the **self-play (SP) reward** — but it has no knowledge of cross-play or other seeds.
- The CFD is the **only mechanism** through which you can influence compatibility across seeds. The critic will enforce whatever the CFD says on each local policy independently, so the CFD must be specific and unambiguous enough that independent critics produce compatible local policies.
- There is no shared training or communication between the seeds/policies.

Evaluation - Cross-Play (XP) Score:

- After N independent runs, we have N final policies
- Each policy is paired with every other policy ($N \times N$ matchups)
- XP score = average joint reward across all pairwise combinations

Goal: Evolve CFDs that maximize XP scores by encouraging policies to adopt compatible, coordination-friendly behaviors. Priortize avoiding total coordination failures, then think about raising the XP scores in general.

Figure 27: Shared background block prepended to every mutation prompt.

Prompt Template: Unary Mutation Operators

You are helping to evolve a "coordination-friendly definition" (CFD) for zero-shot coordination (ZSC).

`$Shared Background (Figure 27)$`

Current CFD

`$CFD$`

Your Task

`$Task Instructions (see Figure 29)$`

Constraints

- The CFD must have at most `$max_bullet_points$` bullet points
- Each bullet point should be a single, concise sentence
- `$Extra Constraints (see Figure 29)$`

Response Format

Respond with valid JSON:

```
{
"reasoning": <reasoning prompt, varies by thinking style>,
"new_cfd": "- First bullet point\n- Second bullet point\n..."
}
```

Figure 28: Combined template for the four `unary_*` mutation operators (`unary_add`, `unary_remove`, `unary_reformulate`, `unary_atomic`). The thinking-style blocks substituted into the Task Instructions, Extra Constraints, and Reasoning Prompt placeholders are listed in Figure 29.

Thinking-Style Blocks for Mutation Operators

The blocks below are substituted into the Task Instructions, Extra Constraints, and Reasoning Prompt placeholders in the unary template (Figure 28) and the diverse template (Figure 30) to specialise each prompt for one of the four thinking styles.

ADD (`unary_add`, `diverse_add`)

Task Instructions: **ADD** a new bullet point to the CFD. The new bullet point should:

- Address an aspect of coordination not yet covered by existing points
- Be complementary to (not redundant with) existing points
- Help policies coordinate better across different game scenarios

Extra Constraints: (none)

Reasoning Prompt: Explain why you're adding this bullet point and what coordination aspect it addresses

REMOVE (`unary_remove`, `diverse_remove`)

Task Instructions: **REMOVE** one bullet point from the CFD. Choose to remove a bullet point that:

- Is redundant or overlaps significantly with other points
- Is too restrictive and may hurt coordination flexibility
- Is less essential than other points for achieving high XP scores

Extra Constraints:

- The resulting CFD must have at least 1 bullet point. If the CFD has only 1 bullet point, reformulate it instead of removing it.

Reasoning Prompt: Explain which bullet point you removed and why removing it improves the CFD

REFORMULATE (`unary_reformulate`, `diverse_reformulate`)

Task Instructions: **REFORMULATE or REPLACE** one bullet point in the CFD. You may either:

- Reformulate: refine the wording to be clearer, more precise, or more actionable
- Replace: swap the bullet point for an entirely different principle that better encourages coordination
- Adapt: Change some part of the bullet point while keepint the rest the same

Choose whichever approach you believe will most improve the CFD's effectiveness.

Extra Constraints: (none)

Reasoning Prompt: Explain what you changed and why it improves the CFD

ATOMIC (`unary_atomic`, `diverse_atomic`)

Task Instructions: Make exactly **ONE atomic change** to the CFD. Choose ONE of the following operations:

1. **ADD**: Add exactly one new bullet point that addresses a missing coordination aspect
2. **REMOVE**: Remove exactly one bullet point that is redundant, too restrictive, or less essential
3. **CHANGE**: Modify exactly one existing bullet point (rewording, refining, or replacing it)

Pick the operation that you believe will most improve the CFD's effectiveness for zero-shot coordination.

Extra Constraints:

- You must make exactly ONE change (add one, remove one, or change one bullet point)
- The resulting CFD must have at least 1 bullet point

Reasoning Prompt: Explain which atomic operation you chose (add/remove/change), what specific change you made, and why

Figure 29: Thinking-style blocks substituted into the unary (Figure 28) and diverse (Figure 30) mutation prompt templates.

Prompt Template: Diverse Mutation Operators

You are helping to evolve a "coordination-friendly definition" (CFD) for zero-shot coordination (ZSC).

`$Shared Background (Figure 27)$`

Current CFD

`$CFD$`

Your Task

`$Task Instructions (see Figure 29)$`

IMPORTANT: Provide `$n_proposals$` DIVERSE proposals for the modified CFD. Each proposal should take a meaningfully different approach to the mutation. Maximize diversity across your proposals.

Note: For some operations (e.g., removing a bullet point from a short CFD), fewer than `$n_proposals$` meaningfully different options may exist. In that case, return as many distinct proposals as possible — do not pad with near-duplicates.

Constraints

- Each CFD must have at most `$max_bullet_points$` bullet points
- Each bullet point should be a single, concise sentence
- `$Extra Constraints (see Figure 29)$`

Response Format

Respond with valid JSON:

```
{
"reasoning": <reasoning prompt, varies by thinking style>,
"proposals": [
"- First bullet point\n- Second bullet point\n...",
"- First bullet point\n- Second bullet point\n...",
...
]
}
```

Figure 30: Combined template for the four `diverse_*` mutation operators (`diverse_add`, `diverse_remove`, `diverse_reformulate`, `diverse_atomic`). The differences from the unary template (Figure 28) are that the prompt asks for $n_{\text{proposals}}$ diverse candidates and that the singleton `new_cfd` field is replaced with a `proposals` list. After parsing, we sample one proposal uniformly at random as the child CFD. Thinking-style blocks (Figure 29) are reused unchanged.

Prompt Template: Reinitialization Operator (`reinit`)

You are helping to create diverse "coordination-friendly definitions" (CFDs) for zero-shot coordination (ZSC).

`$Shared Background (Figure 27)$`

Your Task

Generate `$n$` DIVERSE seed CFDs. Each CFD should:

- Be a single bullet point (one concise sentence)
- Encourage coordination in a unique way
- Be simple and general enough to apply to various multi-agent games
- NOT overlap significantly with other CFDs you generate

Constraints

- Generate exactly `$n$` CFDs
- Each CFD is a single bullet point (one sentence)
- Maximize diversity across the CFDs

Response Format

Respond with valid JSON:

```
{
"reasoning": <Explain your strategy for creating diverse CFDs>,
"cfds": [
"- First CFD",
"- Second CFD",
...
]
}
```

Figure 31: Template used by the `reinit` operator. The operator calls this prompt with n candidates per invocation and samples one uniformly at random as the child CFD. The same template is also used to create the initial seed population at the start of the meta-learning loop.

Gemini 3 Pro (Turn 2) — giving H1

```
Fireworks:  Y=1   Info:  2/3   Lives:  1/1
Own hand (hidden):          Partner's hand:
pos 0:  [RYG123]            pos 0:  Y2
pos 1:  [RYG123]            pos 1:  Y3
pos 2:  [RYG123]            pos 2:  G2
pos 3:  [RYG123]            pos 3:  Y1  hint:1    (duplicate of placed Y1)
pos 4:  [RYG123]            pos 4:  R1  no hint   (playable)

The other player has a playable R1 at position 4.  By giving a rank hint of '1', they will know that
the card at position 4 is a 1 (since position 3 was already hinted as a 1 previously).  This will
allow them to safely play the R1 to start the Red fireworks.
Action:  H1 → touches positions 3 and 4.
```

GPT 5.2 (Turn 3) — responding to H1

```
Fireworks:  Y=1   Info:  1/3   Lives:  1/1
Own hand (hidden):                                      Partner's hand:
pos 0:  [RYG23]                                         pos 0:  R1
pos 1:  [RYG23]                                         pos 1:  G3
pos 2:  [RYG23]                                         pos 2:  G1
pos 3:  hint:1  [RYG1]   (actually Y1, duplicate)       pos 3:  R1
pos 4:  hint:1  [RYG1]   (actually R1, playable)        pos 4:  G1

My cards at positions 3 and 4 were just hinted as 1s after Y1 was already played, so this strongly
suggests they are the other playable 1s.  Playing the older hinted card is the best safe progress.
Action:  play position 3.  → Card was Y1 (duplicate of placed Y1).  Life lost.  Final score:  0.
```

Figure 32: **Coordination failure between Gemini 3 Pro and GPT 5.2 in Hanabi 3×3 due to mismatched assumptions about the "Good Touch" principle (i.e, all clued cards will be played in the future), a convention widely adopted by human Hanabi players.** Gemini provides a rank-1 hint to signal a newly drawn R1 at position 4, but the hint also re-touches an already-played duplicate Y1 at position 3, violating the "Good Touch" principle. GPT, assuming that a cooperative partner "would not hint cards that are already played," infers that both touched cards must be playable 1s and plays the older hinted card (the Y1 duplicate), costing the team its only life. This failure is compounded by an implicit convention, playing the oldest hinted card, that was not mutually established; had the clue giver anticipated this assumption, they would have avoided giving the hint.

H Reasoning Traces

This section contains reasoning traces of various LLM agents across the environments studied in this paper, illustrating both successful coordination and common failure modes.

GPT 5.2

To maximize zero-shot coordination, use an (essentially forced) perfect-signaling convention: when player_0 has Card 2, they must play "A" to enable a payoff-10 outcome regardless of player_1's card (since Case 3 only allows 10 via (A,C)). When player_0 has Card 1, they can safely play "C" because it still allows payoff-10 outcomes for both possible player_1 cards ((C,A) in Case 1 and (C,C) in Case 2). Player_1 then best-responds deterministically based on (player_0_action, own card) to always reach payoff 10. Additionally, if player_1 ever observes "B", responding with "B" is a robust fallback since (B,B)=8 in all cases.

Opus 4.5

Looking at this game, I need to analyze the payoff structure and find a coordination-friendly strategy.
Analysis:
Key observations from the payoff tables:

1. (B, B) always gives 8 regardless of cards - this is a "safe" play
2. 10-point payoffs exist but require knowing which case we're in
3. Player 0 can signal their card through their action choice, enabling Player 1 to know the exact case

Signaling strategy:

- If Player 0 plays A $\rightarrow$ signals Card 1
- If Player 0 plays B $\rightarrow$ signals Card 2

Player 1's optimal responses:

- When P0 plays A (P0 has Card 1):
  - P1 Card 1 (Case 1): Play A $\rightarrow$ 10
  - P1 Card 2 (Case 2): Play C $\rightarrow$ 10
- When P0 plays B (P0 has Card 2):
  - P1 Card 1 (Case 3): Play B $\rightarrow$ 8 (best available)
  - P1 Card 2 (Case 4): Play B $\rightarrow$ 8 (best available)

Expected value: $(10 + 10 + 8 + 8)/4 = 9$
This is better than always playing (B, B), which gives 8, and the convention "A for 1, B for 2" is natural and coordination-friendly.
REASONING:
The optimal ZSC strategy uses Player 0's action to signal their card. Using "A" for Card 1 and "B" for Card 2 is a natural, memorable convention. Player 1 can then deduce the exact game case and respond optimally. When P0 has Card 1 (plays A), P1 can always achieve 10 points. When P0 has Card 2 (plays B), P1's best option is B, giving 8 points. This yields an expected value of 9.

Figure 33: **Reasoning traces of GPT 5.2 and Opus 4.5 agents in Tiny Hanabi.** GPT 5.2 chooses a joint policy based on SP optimality wheres Opus picks a joint policy based on *lexicographical* ordering.

Grok 4.1 Fast

```
The optimal coordination strategy leverages the asymmetry:  player_0 signals its card (A for 1, C
for 2).  Player_1 interprets this signal to infer player_0's card, determines if cards match (both 1
or both 2) or mismatch, and responds to achieve the 10 payoff:
Match (signaled same as own):  Play A (yields A,A or C,C = 10).
Mismatch (signaled different):  Play C (yields A,C or C,C = 10).  This works in all cases and is
robust for ZSC as it uses a natural, symmetric signaling convention (A≈1, C≈2, B safe fallback).
B,B=8 always as fallback if signal invalid.  Expected payoff 10 if both use this policy.  Memory
unused (single turn).
```

GPT 5.2

```
This game has a ‘‘safe’’ coordination point:  (B,B) yields 8 regardless of both cards, so it is highly
compatible with many independently-designed policies and avoids catastrophic 0s.  However, if
player_0 uses A/C as a signal for their card (a natural convention:  A→Card1, C→Card2), player_1
can often achieve 10 by responding conditionally on their own card.  The joint design below
therefore uses a risk-dominant anchor (player_0 always plays B), while player_1 is compatible with B
and also opportunistically decodes A/C in the most likely signaling scheme.
```

Figure 34: **Reasoning traces of Grok 4.1 Fast and GPT 5.2 as code agents in Tiny Hanabi Fixed.** Both agents independently converge on the lexicographic mapping ($A \approx 1$, $C \approx 2$): Grok uses it as an active signalling convention, while GPT uses it to decode the partner's signal. This shared semantic bias enables agents to coordinate via a convention rather than defaulting to the safe fallback.

Grok 4.1 Fast

The optimal strategy for zero-shot coordination is a fixed, symmetric signaling convention that both policies can independently derive from the shared prompt. The levers are listed as ['blue', 'red'] and legal actions list 'blue' before 'red'; signals as ['square', 'circle'] with 'square' before 'circle' in legal actions. This suggests a canonical 1:1 mapping: 'square'↔'blue', 'circle'↔'red'. Alice always encodes the correct lever into the matching signal (free), expecting +10 joint reward per round if Bob decodes correctly. Bob decodes the signal to the matching lever or follows a reveal if present (-5 +10 = +5). No bailing (low reward: +1 or +0.5) or revealing (costly unless needed). This yields +100 expected if coordinating, robust as Bob safely follows reveals/signals. Policies are simple, use no arbitrary choices/memory, ignore history (no need if convention holds), and always valid. Excellent ZSC as different LLMs likely infer the same prompt-order mapping.

Gemini 3 Pro

The game is a pure coordination game where the agents must agree on a mapping between signals ('circle', 'square') and levers ('blue', 'red'). Since the agents cannot communicate and are evaluated in a Zero-Shot Coordination (ZSC) setting, they must rely on a "Schelling point" - a solution that is naturally salient to both parties without prior agreement.
The most robust Schelling point for mapping distinct sets of strings is Lexicographical Ordering (Alphabetical Sort). This method is external, objective, and independent of arbitrary presentation order in prompts.
Sorting Levers: 'blue' comes before 'red' alphabetically. Index 0: 'blue' Index 1: 'red' Sorting Signals: 'circle' comes before 'square' alphabetically. Index 0: 'circle' Index 1: 'square'
Therefore, the proposed policy adopts the convention:
'circle' maps to 'blue' 'square' maps to 'red' Alice's policy simply observes the correct lever and sends the corresponding alphabetic signal. Bob's policy checks for a revealed lever first (taking the guaranteed points), and if none, interprets the signal using the same alphabetic sorting convention. This maximizes the joint reward (10 per round) and minimizes the risk of miscoordination.

Deepseek V3.2

We design a policy that establishes a clear convention for signaling the correct lever without direct communication. The key idea is to use the previous known correct lever as a reference point. Alice reveals the correct lever in the first round to establish a baseline. In subsequent rounds, Alice signals "square" if the current correct lever is the same as the last known correct lever, and "circle" if it changes. Bob uses the same rule to interpret signals and picks the lever accordingly. This approach minimizes ambiguity and ensures coordinated guesses, maximizing the chance of +10 rewards while avoiding costly mistakes. Both policies maintain a memory of the last known correct lever, updated based on revealed levers and outcomes, ensuring robustness across rounds.

Figure 35: **Reasoning traces of Grok 4.1 Fast, Gemini 3 Pro and Deepseek V3.2 when generating the most common observed strategies in Adhoc conventions.** Note that the three strategies are mutually incompatible.

Gemini 3 Pro

So, the challenge is to establish a communication protocol over a noisy channel (unknown button -> signal mapping) without talking. The key is to realize that while Bob cannot interpret the signal in Round 1, he receives the "ground truth" (the correct lever) at the end of the round. This allows him to perform "supervised learning" on Alice's signaling strategy.
Strategy:
Exploration (Round 1): Alice sends a signal corresponding to the correct lever using a fixed convention (Alphabetical: Apple=Blue, Orange=Red). Since Bob does not know the mapping yet, he cannot safely guess a lever (risk of -10 is too high compared to guessing reward). Instead, Bob plays "bail" to secure a small positive reward (0.5).
Learning: At the start of subsequent rounds, Bob looks at the history. He sees what signal Alice sent in previous rounds and what the actual correct lever was. He deduces the mapping (e.g., "Ah, last round I saw a Square and the lever was Red, so Square means Red").
Exploitation (Round 2-10): Once the mapping is learned, Bob can perfectly decode Alice's signals and pick the correct lever for +10 reward.
This yields an expected score of 0.5 + 9 * 10 = 90.5, which is far superior to blind guessing or bailing every round.
Policy Details:
Alice: Simply maps levers to buttons Alphabetically (Blue->Apple, Red->Orange). This is a strong focal point.
Bob: Checks if he has historical data. If yes, he deduces the mapping and acts on the current signal. If no (Round 1), he bails. He also handles the edge case where Alice "reveals" the lever (though she won't in this policy, he should handle it for robustness).

Grok 4.1 Fast

The optimal strategy in ZSC requires a canonical, symmetry-breaking protocol to learn the fixed button-signal bijection using the first two rounds for calibration.
Calibration (Rounds 1-2): Alice presses buttons in lexical order ('apple' then 'orange'), ignoring the levers. Bob observes the signals in these rounds (synchronized via global round count), stores the mapping from signal to button, and bails safely (+0.5 each round).
Exploitation (Round 3+): Alice encodes the current lever lexically ('blue' -> 'apple', 'red' -> 'orange'). Bob decodes the signal to the intended lever and selects it (+10).
Synchronization: Both agents use the global round number (len(correct_levers) for Alice, +1 for Bob) to remain perfectly synchronized, regardless of bails.
Robustness: The strategy includes handling for reveals/None cases.
Convention: Canonical lexical ordering (apple < orange, blue < red) ensures coordination across LLMs.
Expected reward: 0.5 * 2 + 10 * 8 = 81

Figure 36: **Reasoning Traces of Gemini 3 Pro and Grok 4.1 Fast in Ad-hoc conventions permuted signals**. Although both models converge on a scheme where Alice remains fixed and Bob adapts, they adopt different calibration strategies for Bob, rendering them incompatible.

Grok 4.1 Fast

```
The game requires crossing positions 0 to 6 while ensuring both players are not on the bridge
(positions 1-5) for more than 1 consecutive turn.  Identical policies fail to coordinate due to
symmetry (both would forward on turn 2, breaking the bridge).  Thus, use asymmetric roles with a
clear convention:  player_0 (leader) always "forward" to cross first in 6 turns.  player_1
(follower) stays at 0 until observing leader at position 4 (after turn 4), then "forward".  This
results in both on bridge only on turn 5 (leader at 5, follower at 1; first instance), turn 6 has
leader off at 6.  Follower reaches 6 on turn 10 (-10 reward).  Safe, minimal backs, leverages ZSC by
converging on player_0=leader, player_1=follower across LLMs.  Robust as actions always legal.
```

Deepseek V3.2 (poor SP)

```
The game requires coordination to maximize reward and avoid the catastrophic bridge collapse.  Key
insights:
The bridge can only support both players for 1 consecutive turn.  This implies we must not have both
players on the bridge (positions 1-5) for two turns in a row.  To minimize turns, we need to cross
as quickly as possible, but safety is paramount due to the large penalty (-1000).  Without
communication, players must rely on observing past actions to coordinate.  A simple and robust
strategy is to have one player wait at the start while the other crosses completely, then the
waiting player crosses.  This avoids simultaneous bridge occupancy entirely and is
coordination-friendly in ZSC. To break symmetry (both players might otherwise adopt the same policy
and both wait or both go), we use a deterministic rule based on comparing paths:  the player whose
historical action sequence is lexicographically smaller becomes the "leader" who goes first.  This
is a common symmetry-breaking technique in ZSC that doesn't require randomness (which could lead to
miscoordination) and ensures both players compute the same leader/follower roles.  Memory is used to
store the computed role after the first turn to avoid recomputation.  The policy ensures valid
actions at all times and handles edge cases (e.g., already at goal).
```

GPT 5.2

```
To maximize zero-shot coordination, the policy uses a widely compatible "single-file bridge"
convention:  never enter the bridge if the other player is currently on it, and if you are on the
bridge while the other is not, keep moving forward to clear it quickly.  The only hard case is
simultaneous entry (both move from start to position 1); to resolve this without communication, the
policy uses randomized backoff at the start so that eventually only one player enters first while
the other waits.  If an unexpected overlap on the bridge occurs, an emergency rule tries to separate
the players by sending the one closer to the start backward and the one closer to the goal forward
(a symmetric, observation-based convention).
```

Figure 37: **Reasoning Traces of Grok 4.1 Fast, Deepseek V3.2 and GPT 5.2 in Brittle Bridge.** DeepSeek produces a complex role-assignment strategy that performs poorly even in self-play. In contrast, Grok 4.1 Fast and GPT 5.2 each learn role-assignment strategies that are mutually incompatible.

Opus 4.5

Looking at this game, I need to design a policy for a narrow tunnel coordination problem.
The key challenge is:
- Player 0 starts at position 0, needs to reach position 4 - Player 1 starts at position 4, needs to reach position 0 - There is a narrow tile at position 2 that creates a bottleneck - Both players cannot be on the narrow tile simultaneously - Players cannot swap positions when one is on the narrow tile
For ZSC, I need a deterministic convention that independently generated policies would converge on. A natural symmetry-breaking approach is to use the goal direction (which is observable) to determine who yields.
Chosen convention: The player going RIGHT (goal = "right") yields by waiting at position 1 until the player going LEFT has passed through the narrow section. This is a natural, deterministic convention based on observable information.
Optimal sequence with this convention:
Turn 1: P0->1, P1->3 Turn 2: P0 stays at 1, P1->2 Turn 3: P0 stays at 1, P1->1 (both on wide tile 1) Turn 4: P0->2, P1->0 Turn 5: P0->3 Turn 6: P0->4
This achieves the goal in 6 turns, which is optimal given the bottleneck constraint.

GPT 5.2

The key coordination problem is the single narrow tile (index 2): if both try to enter it from opposite sides on the same turn, nobody moves; and if they try to "swap" across it, the swap is blocked. A ZSC-friendly solution is to use a simple, role-inferable convention that both agents will independently follow: the agent whose goal is "right" (i.e., the left-starting agent) has priority to enter the narrow tile first when both are poised to enter from opposite sides (positions 1 and 3). The other agent yields by waiting. This convention is symmetric from the game description (it depends only on the publicly inferable goal direction), avoids deadlock, and achieves the standard efficient 6-turn crossing.

Figure 38: **Reasoning Traces of Opus 4.5 and GPT 5.2 in Narrow Tunnel.** These traces illustrate the two most common yet incompatible role-assignment strategies: "left goes first" versus "right goes first."

Opus 4.5

Looking at the 11x11 matrix game, I need to find a focal point that both players would independently converge on for zero-shot coordination.
Key observations about the matrix:
- The entry (0,0) = 1.0 (maximum reward) - Entry (1,0) = 1.0 (also maximum reward) - notably, rows 0 and 1 both have their maximum at column 0 - The diagonal entries (i,i) for i >= 2 all equal 1.0, with small 0.02 values on adjacent off-diagonal entries - Most other entries are 0
For zero-shot coordination, I need the most natural focal point. The (0,0) cell is the strongest candidate because:
- Smallest index convention: "Pick the first/smallest option" is a universal coordination heuristic
- Symmetry: both players choosing 0 is symmetric - Column 0 is special: both rows 0 and 1 give reward 1.0 only at column 0, making column 0 feel more "focal" than other columns - Maximum reward: the payoff of 1.0 is the best achievable
Other focal points like "meet in the middle" (5,5) or "go to the end" (10,10) are possible but less universally natural than "start at the beginning."
Therefore, the optimal coordination strategy is for both players to always choose action "0".

Deepseek V3.2

The game is a one-turn matrix game with a fixed payoff matrix. The optimal joint action is to coordinate on a cell with the highest possible reward (1.0). However, there are multiple such cells: (0,0), (1,0), and the main diagonal from (2,2) to (10,10). Since agents cannot communicate and the game is played only once, the key challenge is symmetry-breaking: both players must independently choose the same convention to coordinate on one of these high-payoff cells.
Given the matrix structure, the diagonal cells (i,i) for i=2..10 are particularly salient because they form a clear, symmetric pattern. Among these, the most natural focal point is the center of the diagonal-index 6-as it is the midpoint of the range 2-10. This is a common human convention (e.g., "meet in the middle") and likely to be chosen independently by other LLM-generated policies, maximizing zero-shot coordination.
Choosing 0 or 1 carries risk: if one player picks 0 and the other picks any index except 0, the payoff is 0. The diagonal strategy is more robust because even if coordination fails slightly (e.g., one picks 5 and the other 6), the payoff is still 0.02, which is better than 0. Thus, the policy for both players should be to choose action 6.
Since the game is one-turn, memory is unused but must be returned unchanged.

Figure 39: **Reasoning Traces of Opus 4.5 and Deepseek V3.2 in Matrix Game.** Opus 4.5 successfully identify different focal points from the payoff matrix.

Opus 4.5

This is a coordination game with hidden permutations - each player sees levers numbered 1-5, but the underlying mapping is different for each player.
Key insight:
When I observe the other player's action, I see it in MY coordinate system. If they picked their lever X and I observe it as lever Y (in my view), then if I pick lever Y, I will be selecting the same underlying lever they chose.
Naively, if both players "follow" (i.e., switch to what they observed), they will keep swapping and never converge. We therefore need exactly ONE player to follow and ONE to stay.
Symmetry-breaking rule:
- If you observe a HIGHER number than your own action, you follow - If you observe LOWER or EQUAL, you stay
Why this works:
- If I see higher (X > my_action): I follow → pick X - If they see lower (my_action in their view < their action): they stay - Result: I move to their action, they stay → convergence
Initialization:
Start with lever 1 (a Schelling focal point - the lowest number), then apply the rule above. Once both players match (reward = 1), repeat the successful action.
Edge case:
If both players observe the same number as their own action (X = 1, Y = 1), both will stay. This case has low probability under random permutations, and the system will eventually resolve once observations differ.

Deepseek V3.2

This is a zero-shot coordination problem where two players must choose the same lever under a random, fixed permutation unknown to them. The key challenge is establishing a robust convention without communication. Observing each other's actions (though permuted) and joint rewards provides limited feedback.
Since both players see the same structure, we can adopt a symmetry-breaking strategy that uses the first few turns to infer a common reference point. A reliable approach is to treat the first turn as a "coordination attempt": both players choose lever 1 initially. If they receive a reward of 1, they continue choosing lever 1. If not, they switch to "mirroring" the other player's last action, which accounts for the unknown permutation by aligning on the lever the other player just chose in their own perspective. This adaptive rule maximizes the chance of converging quickly, as mirroring is a natural coordination heuristic when permutations are unknown. Storing the initial lever in memory ensures consistency.
The policy uses a simple state machine:
Start by choosing lever "1". If the last reward was 1, stick with the last successful lever.
Otherwise, mirror the other player's last action. This balances exploration and exploitation over the 10 turns and is likely to be independently chosen by another LLM, promoting coordination.

Figure 40: **Reasoning Traces of Opus 4.5 and Deepseek V3.2 in Permuted Levers.** Opus 4.5 achieves a low self-play score but exhibits strong cross-play performance when paired with DeepSeek V3.2.

Grok 4.1 Fast (Hinter) — hand: 1B, 3A, 1B, 3C, 2B target: 1B hint: 1B

The target is 1B, an exact match in my hand. Hinting 1B maximizes similarity (score 2). Using a strategy of max similarity (exact=2 > feature match=1), with ties broken by preferring suit match over number match, this hint is unique to the target 1B: other targets prefer different hints (1A->3A, 1C->3C, 3A->3A, 3C->3C). The guesser, assuming the same strategy, will identify 1B uniquely.

GPT-5.4 (Guesser) — hand: 3C, 3A, 1A, 1B, 1C hint received: 1B guess: 1A → reward 0

A cooperative optimal code can distinguish 4 of the 5 possible targets. The most natural high-value mapping is:
- 3A hint -> 3A
- 3C hint -> 3C
- 2B hint -> 1B
- 1B hint -> one of the remaining 1X cards

So seeing 1B makes 1A/1C the most likely targets, not 1B. With no further distinction, I use the natural tie-break toward 1A.

Figure 41: **Convention mismatch in Hint-Guess.** The hinter gives the trivially correct exact-match hint 1B for target 1B. The guesser, however, applies an over-engineered bijective code in which 1B is re-mapped to a residual slot (1A).

Claude Sonnet 4.6 (Hinter) — hand: 1B, 1C, 1C, 2A, 2C target: 1C hint: 1C

```
The target card is 1C, and I need the Guesser to pick 1C from their hand.
The Guesser's hand contains:  1C, 1C, 2B, 2A, 3A.
If I hint 1C (which is in my hand), the Guesser will see that my hint card exactly matches 1C, which
is also in their hand.  The most natural cooperative strategy is:  if the hint card appears in the
Guesser's hand, pick that card.  Since 1C is in both my hand and the Guesser's hand, hinting 1C
directly and unambiguously signals that the target is 1C.
Other hints would be less direct:
- hint_1B: shares "1" with 1C but also "B" with 2B -- ambiguous
- hint_2A: shares features with 2A, 3A, or 2B -- points away from 1C
- hint_2C: shares "C" with 1C but "2" with 2B/2A -- ambiguous
Hinting 1C is the clearest possible signal.
```

Grok 4.1 Fast (Guesser) — hand: 1C, 1C, 2B, 2A, 3A hint received: 1C guess: 2A → reward 0

```
Unique card types in my hand, sorted lexicographically:  1C, 2A, 2B, 3A.
Unique card types in Hinter's hand, sorted lexicographically:  1B, 1C, 2A, 2C.
Use position-based mapping:  1st hint signals 1st guess, 2nd hint signals 2nd guess, etc.
Hinter hinted 1C (2nd in sorted list), so guess 2A (2nd in my sorted list).
This bijection allows perfect signaling in general.
```

Figure 42: **Convention mismatch in Hint-Guess.** The hinter uses a simple exact-match strategy: hint `1C` because `1C` is in both hands. The guesser ignores the direct exact-match signal and instead applies a lexicographic position-based bijection, mapping the 2nd hint in the sorted hinter hand (`1C`) to the 2nd card in the sorted guesser hand (`2A`).

GPT-5.4

```
Use a deterministic canonical ordering over cards (1A < 1B < ...  < 3C) and ignore hand permutation
by deduplicating and sorting both public hands.
Then coordinate via aligned indices:
Hinter:  find the target's index in the sorted unique Guesser hand, and hint the Hinter card at the
same index modulo the number of unique Hinter cards.
Guesser:  recover the hinted card's index in the sorted unique Hinter hand, then guess the Guesser
card(s) whose index has the same modulo class; choose the first such card.
Why this is coordination-friendly:
```

- Fully deterministic.
- Uses only public information plus the hidden target.
- Independent of arbitrary hand order.
- Very simple and likely to be rediscovered by another policy.
- Perfectly identifies the target whenever the Hinter has at least as many unique hint cards as the Guesser has unique candidate cards.

Figure 43: **Reasoning trace of GPT-5.4 as a code agent in Hint Guess.** The policy sorts both hands canonically ($1A < 1B < \ldots < 3C$), deduplicates, and coordinates by mapping target indices modulo the number of unique hinter cards. Although internally consistent, the convention is essentially arbitrary — another code agent is unlikely to independently arrive at the same convention, hence cross-play coordination breaks down.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Empirical claims are supported by experiments in Figure 4, Appendix 5, and Appendix 6, with additional results in Appendix F.3. Non-empirical claims, such as the relation to prior work, are addressed in Appendix A; out-of-scope items and limitations are discussed in Appendix 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We discuss limitations in Appendix 7.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not contain any theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Environments are described in Appendix D, prompts are given in Appendix G.3, and the remaining implementation details required to reproduce our results are provided in Appendix G.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Anonymised code with run instructions to reproduce all experiments is available at `https://anonymous.4open.science/r/zsc_llm-EAE1/README.md`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental settings and hyperparameters are described in Figure 4, and full implementation details for the meta-learning optimisation are provided in Appendix G.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the mean and standard error over multiple random seeds for all results; the protocol used to compute these statistics is detailed in Appendix F.3.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: API costs and the GPU used for the token-level probability experiments are reported in Appendix G.4.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts are discussed in Appendix B.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This paper poses no such risks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Licenses for the figures and environments reused from prior work are listed in Appendix C.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We introduce seven new coordination environments (Narrow Tunnel, Brittle Bridge, Ad-Hoc Conventions, Ad-Hoc Conventions with Permuted Signals, Permuted Levers, Semantic Selection Game, and Semantic Convention Game), all provided with documentation in the anonymised code repository (`https://anonymous.4open.science/r/zsc_llm-EAE1/README.md`). The same repository also includes our re-implementations of the reused environments (Cat & Dog, Tiny Hanabi, Hanabi 3×3, Matrix Game, Hint–Guess, and Trampoline–Tiger), each augmented with natural-language descriptions for interaction with LLM agents.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are central to this work in two ways: (i) they are the subject of study, evaluated in the sections *ZSC performance of LLM agents*, *Semantics and Coordination*, and *Coordination Friendly Definitions*; and (ii) an LLM (GPT-5.4) serves as the meta-optimisation agent in the GEPA-based loop that discovers CFDs (*Coordination Friendly Definitions*, with implementation details in Appendix G). The full list of LLMs accessed in this work and their licenses is provided in Appendix C.

    Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.