Communication and Verification in LLM Agents towards Collaboration under Information Asymmetry

Run Peng[1*]   **Ziqiao Ma**[1*]   **Amy Pang**[1]   **Sikai Li**[2†]

Zhang Xi-Jia[3†]   **Yingzhuo Yu**[4†]   **Cristian-Paul Bara**[5,6†]   **Joyce Chai**[1]

[1]University of Michigan [2]UNC, Chapel Hill [3]Georgia Tech

[4]Apple [5]Robert Bosch SRL [6]Babeş-Bolyai University

Abstract

While Large Language Model (LLM) agents are often approached from the angle of action planning/generation to accomplish a goal (e.g., given by language descriptions), their abilities to collaborate with each other to achieve a joint goal are not well explored. To address this limitation, this paper studies LLM agents in task collaboration, particularly under the condition of information asymmetry, where agents have disparities in their knowledge and skills and need to work together to complete a shared task. We extend Einstein Puzzles, a classical symbolic puzzle, to a table-top game. In this game, two LLM agents must reason, communicate, and act to satisfy spatial and relational constraints required to solve the puzzle. We apply a fine-tuning-plus-verifier framework in which LLM agents are equipped with various communication strategies and verification signals from the environment. Empirical results highlight the critical importance of aligned communication, especially when agents possess both information-seeking and -providing capabilities. Interestingly, agents without communication can still achieve high task performance; however, further analysis reveals a lack of true rule understanding and lower trust from human evaluators. Instead, by integrating an environment-based verifier, we enhance agents' ability to comprehend task rules and complete tasks, promoting both safer and more interpretable collaboration in AI systems. https://github.com/Roihn/EinsteinPuzzles

1 Introduction

In recent years, there has been growing interest in Large Language Model (LLM) agents (e.g., Web-Agents) and their diverse applications (Wang et al., 2024a). While much of the current work focuses on action planning or goal completion in LLM agents (Ahn et al., 2022; Durante et al., 2024; Song

et al., 2023) (e.g., based on natural language instructions), their ability to collaborate with one another toward a shared goal remains underexplored. This paper addresses this gap by studying LLM agents in the context of task collaboration, particularly under conditions of information asymmetry.

Information asymmetry is a fundamental and pervasive feature of human interaction. In daily life, we often possess different knowledge, perspectives, or intention. We must reason about others' knowledge and beliefs and coordinate our differences in order to collaborate (Ma et al., 2023). As LLM agents become increasingly integrated into real-world workflows — not just to perform tasks independently, but to act as human proxies and collaborators — it becomes essential to examine how well these agents can coordinate under asymmetric information, and what mechanisms might enhance their collaborative capabilities in such settings.

To this end, we adapt Einstein Puzzles (Groza, 2021), a classical logic problem, into a table-top environment where two agents must solve spatial and relational constraints, despite having partial, asymmetric information. Using a fine-tuning–plus–verifier framework, we equip LLM agents with different communicative abilities (e.g., asking, sharing, or both) and study their collaborative behaviors under various configurations.

Furthermore, we introduce an environment-based verifier to guide and evaluate agent decisions. This verifier leverages environmental signals to determine whether the proposed actions and known constraints are consistent, mirroring the implicit feedback mechanisms in interactive test-and-trials. It is training-free, lightweight, and broadly applicable across various environments. We examine whether this simple yet generalizable approach can greatly improve collaboration among LLM agents.

Our empirical results have shown that under information asymmetry, LLM agents with both information-seeking and -providing capabilities

---

collaborate most effectively. Meanwhile, mismatched agent communicative abilities leads to significant performance degradation, highlighting the importance of aligned interaction protocols. Additionally, through detailed error analysis, we show that environment-based verification, using naturally available feedback without additional training, offers a simple but powerful mechanism to improve agent performance on both task completion and understanding. Finally, our human study reveals a gap between task efficiency and human preference: participants favor agents that proactively share information, even if such agents are less optimal in task completion. These findings indicate the need for communication-aware and interpretable design in LLM-based collaborative systems.

2 Related work

Collaboration among LLM Agents The field of multi-agent coordination and communication has a long-established history (Albrecht and Stone, 2018; Gronauer and Diepold, 2022; Stone and Veloso, 2000), with applications increasingly extending to human-AI collaboration (Carroll et al., 2019; Puig et al., 2020). Recently, multi-agent systems built upon Large Language Models (LLMs) (Guo et al., 2024; Tran et al., 2025) have pushed the boundaries of collaborative intelligence in a wide range of downstream tasks, including collaborative coding (Gao et al., 2024; Hong et al., 2024; Qian et al., 2024), social simulation (Li et al., 2023; Wu et al., 2024; Yang et al., 2024; Zhou et al., 2024c), and problem-solving (Chen et al., 2023; Qian et al., 2025; Wang et al., 2024c; Zhang et al., 2024a).

However, most existing work focuses on settings with information transparency (Chen et al., 2023; Gao et al., 2024; Li et al., 2023; Wu et al., 2024; Yang et al., 2024), one-way communication (Qian et al., 2025), or information asymmetry within asymmetric role assignments (Hong et al., 2024; Qian et al., 2024; Zhou et al., 2024c). In contrast, this work centers on LLM agents operating under information asymmetry in symmetric, collaborative roles, as well as involves human studies to assess real-world human-AI interaction. Liu et al., 2024a studied autonomous agents for collaborative tasks under information asymmetry, while there were no environments involved. As echoed in recent efforts (Liu et al., 2024b; Zhou et al., 2024b), addressing collaboration under information asymmetry—especially in human-AI contexts—remains

a core challenge for LLM-based systems.

Test-time Compute in LLM Recent works have demonstrated the effectiveness of leveraging additional computational resources at inference time to improve response quality in LLMs. One line of research introduces a verification model to evaluate the correctness and utility of generated responses. Typically, this involves sampling multiple responses from an LLM and applying a best-of-n strategy (Charniak and Johnson, 2005; Cobbe et al., 2021), where a verifier model selects the most appropriate response. Such approaches have been widely adopted across domains including mathematics (Cobbe et al., 2021; Lifshitz et al., 2025; Lightman et al., 2023; Wang et al., 2024b; Yao et al., 2023), code generation (McAleese et al., 2024; Wang et al., 2025), and web navigation (Koh et al., 2024; Putta et al., 2024). We refer readers to (Guan et al., 2024; Zhang et al., 2025) for comprehensive reviews of verification-based methods.

While most existing approaches employ a separate value model, often another LLM, with (Lifshitz et al., 2025; Lightman et al., 2023; McAleese et al., 2024; Wang et al., 2025, 2024b; Zhang et al., 2024b; Zhou et al., 2024a) or without (Yao et al., 2023; Yu et al., 2023) further training, we propose an alternative verification mechanism that directly leverages environmental feedback. In interactive tasks within simulated environments, the environment itself provides fine-grained, up-to-date, and objective signals about task progression and action validity. This enables a training-free, compute-efficient form of verification that naturally integrates with agent decision-making.

3 Collaboration Under Information Asymmetry

3.1 Understanding Einstein's Puzzle on the tabletop setup

Einstein Puzzles is a logical game requiring deductive reasoning and constraint satisfaction. We modify it into a collaborative game in which two agents work together to place a set of objects into designated bins. Each object has a target bin as its destination, but the information about these destinations is split between the two agents. Importantly, agents are unaware of which pieces of information their partner possesses. To succeed, agents must engage in rich communication and apply strategic reasoning to solve the task collaboratively.
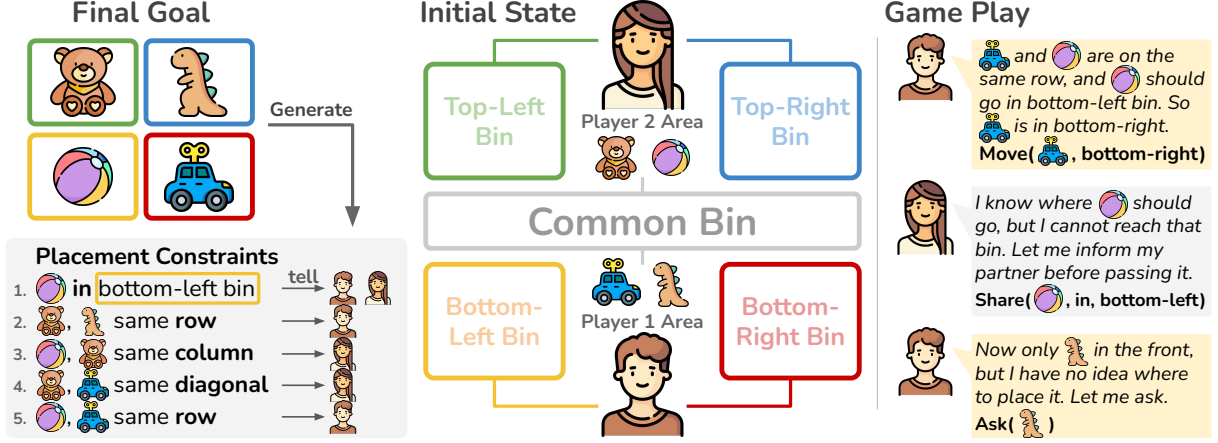
Figure 1: Illustration of our collaborative game. Each game features a final goal (top-left), where objects are assigned to specific goal bins (e.g. toy bear to top-left bin). Placement constraints are generated based on this final goal and are distributed to the two players. At the start, objects are randomly positioned in front of the players. Players must collaborate and communicate effectively to reason the final goal and accordingly complete the placement.

The setup consists of a rectangular table with two collaborators sitting on opposite sides of the table (See Figure 1). The table is split into three regions: a region immediately in front of each collaborator and a common area in the center. Each collaborator can only reach the area directly in front of them and the common area in the center. They can also each reach the two corner bins on either side of the area in front of them, totaling four destination bins. In the beginning, several objects are distributed between the two players' bins, so they are exclusively reachable by one of the collaborators. No objects start in any bins.

We describe the task as a distributed constraint satisfaction problem. A goal configuration is defined, which is where the objects should be placed, and can be described with a set of constraints. There are four types of constraints that describe the relationship of a pair of objects or between an object and a bin. Given a pair of objects, it must either be:

- **in the same bin**: both objects have the same destination.
- **in the same column**: the two objects must end up in bins on either exclusive side of the table but on the same side, left or right, from either collaborator's perspective.
- **in the same row**: both objects must end up on one of the collaborator's exclusive zones but in bins on opposite sides, left or right.
- **or on the same diagonal**: the two objects must end up in opposite areas and opposite sides.

The exhaustive set of rules between all pairs of objects is overly descriptive. We select a random, minimal subset of these rules (denoted as $\mathcal{C}$). A minimal subset $\mathcal{C}$ must satisfy the following:

- Eliminating any rule will describe more than one final configuration.
- Moving any object from the final configuration will violate at least one rule.

An additional starting constraint is given, which grounds one of the objects to a specific bin. Together, the minimal set of object pair rules and the one object-to-bin grounding uniquely describe the goal configuration.

The set of constraints that uniquely determines the final placement of the objects is then divided between the two players (denoted as $\mathcal{C}_1$ and $\mathcal{C}_2$), creating **information asymmetry**. This distribution ensures that neither player can complete the task independently, but together, they possess all the necessary information to succeed, i.e., $\mathcal{C}_1 \cup \mathcal{C}_2 = \mathcal{C}$. Coupled with the fact that some objects must end on the opposite side of the table, this constitutes the disparity in knowledge and skills as described in (Bara et al., 2021).

The two players take turns, each consisting of a single move. A move can be placing an object into an available destination bin or the center of the table, sharing a piece of information, or requesting information from their partner. Task performance is measured by the number of moves taken to complete the task. Since both object placement and information exchange count as a move, this scoring system encourages players to communicate concisely. The players' objective is to achieve the final configuration using the least number of moves

within the set of given constraints.

3.2 Information Providing & Seeking

We study two communicative actions in collaborative contexts with information asymmetry: information seeking and information providing. We want to investigate whether LLM agents are able to exchange necessary information with their partners, as well as perform collaborative reasoning for task completion. To systematically analyze the role of communication actions in this process, we design four action space configurations for communication as below:

1. *Information Providing Only:* agents are allowed to proactively share their knowledge of constraints with their partners, but they cannot ask for this information.

2. *Information Seeking Only:* agents can ask about constraints of specific objects with their partners. They are allowed to share a constraint only when their partner first asks about a object involved in it. In other words, agents cannot initiate the information providing.

3. *Information Providing & Seeking:* agents can both share and ask freely.

4. *No Information Exchange:* no communicative actions are enabled. Agents can only choose to move an object or to pass the turn. Since neither player has complete knowledge about the placement constraints, this configuration inevitably involves random guessing for objects that may have goal bins that are not deducible.

We enabled LLMs to complete the games under these different action space configurations. To be specific, we applied supervised fine-tuning (SFT) on several well-known open-source models, including Meta-Llama-3-8B-Instruct (AI@Meta, 2024), Llama3.1-8B-Instruct (Grattafiori et al., 2024) and Qwen2.5-7B-Instruct (Team, 2024) models. To prepare data for fine-tuning, we designed a planner to generate solutions under different configurations of action space. Please refer to the appendix for the details of planner design. We varied the number of objects per game and collected trajectories for games with 4, 5, and 6 objects, totaling 250, 500, and 500 games, respectively. For each game, we generated five distinct solution trajectories (both optimal and near-optimal solutions), creating a large pool from which we sampled 1,000 trajectories for fine-tuning. Each trajectory averages around 10 steps, resulting in approximately 10,000 training

samples in the chat form. We evaluated the models on 300 unseen games (100 games for each object count) and reported average performance across them. We also evaluated models with and without chain-of-thought (CoT) reasoning (Wei et al., 2022). The reasoning traces for the training data are generated using GPT-4o. Examples of the reasoning traces can be found in the game play section of Figure 1. A full example of a game play can be found in Appendix A.

4 Environment-based Verifier

Inspired by the trial-and-error paradigm, we draw an analogy to how agents refine decisions through iterative interaction with their environment. Rather than training a dedicated verifier, we directly leverage environment-provided feedback as a source of verification during inference. We evaluate whether this lightweight, training-free approach is sufficient to support LLM agents in collaborative reasoning tasks under information asymmetry.

We design a **reasoning verifier** based on environment feedback. Firstly, It is capable to examine the generated action with the game rules (e.g. physical affordance) and previous communication (e.g. redundant information sharing). Secondly, as a strategy-driven game, we implement a graph expansion algorithm to enhance agents' reasoning ability. It treats objects and bins as nodes and constraints as edges, and infers new constraints by combining existing ones. It is expanded iteratively using the transitivity of adjacent edges until no new constraints emerge. For example, objects A and C can be inferred to be on the same row if both A and B, and B and C, are known to be on the same row. This enables the verifier to assess whether a proposed action aligns well with the current knowledge, and avoids unnecessary trials.

Further, we tested several alternatives of environment-based verifiers, and discussed their effectiveness and potential generality in Appendix C.

5 Experiments and Results

We conduct extensive experiments to evaluate the performance of language models in reasoning and communication under information asymmetry. Our experiments address four key research questions:

- **RQ1: Collaboration under Varying Communicative Action Space.** How do agents collaborate under information asymmetry when equipped with different communicative actions?

- **RQ2: Effectiveness of Environment-Based Verification.** To what extent can environment-based verification enhance the performance?

- **RQ3: Collaboration under Mismatched Communicative Action Space.** What happens when agents with mismatched communication capabilities are paired together?

- **RQ4: Human Preferences Toward Different Communication Behaviors.** While agents may perform well in self-play, do these behaviors align with human preferences in collaboration?

5.1 Experiment Setup

To systematically investigate our research questions, we design a series of experiments to investigate specific aspects of the collaboration:

- **Exp1: Collaboration with Different Communicative Action Spaces (RQ1)** We evaluated both closed-source language models via API calls and open-source models with supervised fine-tuning (SFT). We deploy the same model (with the same action space) to perform self-play in the game. This setting allows direct comparison of collaboration with different communicative action space.

- **Exp2: Environment-Based Verifier (RQ2)** We augment the base models with an environment-based verifier, which provides binary feedback indicating whether each sampled action is valid. At each decision step, we sample 4 candidate responses using `temperature=0.2` and `top-p=0.9`, and apply the verifier to filter out invalid actions, selecting the first valid one as the final output. This setup enables a direct comparison of model performance with and without the verifier.

- **Exp3: Collaboration with Mismatched Action Spaces (RQ3)** We select the well-performing models from those action spaces (Llama3.1-8B without CoT for *no information exchange*, and Llama3.1-8B with CoT for the rest), equipped with reasoning verifier, and let them play with each other. This setting further reveals the behaviors of collaboration between agents with different communication capabilities.

- **Exp4: Human Performance (RQ4)** Once we identified the most effective configuration of communicative action space for collaborative reasoning among LLM agents, we further examined whether such models are also preferred by human users. We recruited 12 college students as human participants to interact with the best-performing models, each equipped with different communicative action spaces and supported by the reasoning verifier. We sampled 27 unseen games (9 games for each object count) and measured both task completion rate and efficiency. Each participant was assigned a sequence of 9 games—3 games each with 4, 5, and 6 objects—while being given the complete communicative action space. In each game, they interacted with a model configured with a specific communicative action space, which remained unknown to the participant. To complement quantitative metrics, we also collected qualitative feedback from participants after each game using the following three questions:

1. *Did you find the information communicated by the bot useful?*
2. *Did the bot make effective use of the information you shared?*
3. *Were you ever confused by the bot's behavior?*

These questions allowed us to assess the perceived helpfulness, responsiveness, and clarity of the model's behavior from a human-centered perspective.

Evaluation Metrics We employ complementary metrics to assess both effectiveness and efficiency. For effectiveness, we report the Success Rate (`SR`) and subgoal success rate (`Sub.R`) at the first attempt (Pass@1). `SR` measures the percentage of games successfully completed within a limited number of steps (30 for all the experiments), while `Sub.R` reflects partial progress by capturing the proportion of objects correctly placed, even when the full game is not successfully completed. For efficiency, we track the number of steps taken to complete each game and compare it to the optimal solution calculated by our planner. We then compute the Step Ratio (`StepR`), defined as the ratio of the number of executed steps to the number of steps in the optimal solution. When a verifier is applied, we additionally report the correction rate (`Corr.R`), indicating the proportion of responses corrected by the verifier. We report standard error for each metrics.

5.2 Result Analysis

Effects of Communicative Action Space (RQ1) Across all model families, we observe a generally clear hierarchy in task performance: *Information Seeking & Providing > Seeking Only > No Infor-*

	No Verifier			**With Reasoning Verifier**			
Model	SR↑(%)	Sub.R↑(%)	StepR↓	SR↑(%)	Sub.R↑(%)	StepR↓	Corr.R(%)
Information Providing & Seeking							
GPT4o CoT	$51.00_{\pm2.89}$	$76.73_{\pm1.64}$	$1.92\text{x}_{\pm0.04\text{x}}$	80.00 (+29.00)$_{\pm2.31}$	$91.34_{\pm1.17}$	$1.65\text{x}_{\pm0.03\text{x}}$	$17.42_{\pm0.57}$
Llama3-8B	$13.67_{\pm1.98}$	$60.04_{\pm1.37}$	$\mathbf{1.47x}_{\pm0.08x}$	32.33 (+18.66)$_{\pm2.70}$	$73.28_{\pm1.36}$	$1.40\text{x}_{\pm0.05\text{x}}$	$6.36_{\pm0.43}$
Llama3.1-8B	$27.33_{\pm2.57}$	$68.74_{\pm1.44}$	$1.50\text{x}_{\pm0.05\text{x}}$	53.00 (+12.67)$_{\pm2.88}$	$81.05_{\pm1.40}$	$\mathbf{1.36x}_{\pm0.03x}$	$7.19_{\pm0.44}$
Qwen2.5-7B	$27.00_{\pm2.56}$	$72.31_{\pm1.26}$	$1.56\text{x}_{\pm0.06\text{x}}$	47.00 (+20.00)$_{\pm2.88}$	$83.96_{\pm1.05}$	$1.45\text{x}_{\pm0.04\text{x}}$	$8.34_{\pm0.42}$
Llama3-8B CoT	$29.67_{\pm2.64}$	$62.51_{\pm1.75}$	$1.94\text{x}_{\pm0.09\text{x}}$	70.00 (+40.33)$_{\pm2.65}$	$87.26_{\pm1.34}$	$1.61\text{x}_{\pm0.04\text{x}}$	$14.28_{\pm0.58}$
Llama3.1-8B CoT	$\mathbf{58.67}_{\pm2.84}$	$\mathbf{79.39}_{\pm1.64}$	$1.87\text{x}_{\pm0.05\text{x}}$	$\mathbf{89.33}$ (+30.66)$_{\pm1.78}$	$\mathbf{95.64}_{\pm0.81}$	$1.52\text{x}_{\pm0.03\text{x}}$	$14.38_{\pm0.59}$
Qwen2.5-7B CoT	$44.67_{\pm2.87}$	$74.31_{\pm1.66}$	$1.91\text{x}_{\pm0.07\text{x}}$	81.00 (+36.33)$_{\pm2.26}$	$92.53_{\pm1.03}$	$1.61\text{x}_{\pm0.03\text{x}}$	$14.59_{\pm0.54}$
Information Providing Only							
Llama3-8B	$9.33_{\pm1.68}$	$62.46_{\pm1.19}$	$1.26\text{x}_{\pm0.08\text{x}}$	27.00 (+17.67)$_{\pm2.56}$	$74.81_{\pm1.23}$	$1.45\text{x}_{\pm0.08\text{x}}$	$6.43_{\pm0.38}$
Llama3.1-8B	$26.00_{\pm2.53}$	$\mathbf{69.71}_{\pm1.40}$	$1.56\text{x}_{\pm0.06\text{x}}$	40.33 (+14.33)$_{\pm2.83}$	$80.43_{\pm1.19}$	$1.43\text{x}_{\pm0.05\text{x}}$	$6.38_{\pm0.39}$
Qwen2.5-7B	$8.00_{\pm1.57}$	$51.96_{\pm1.30}$	$\mathbf{1.12x}_{\pm0.02x}$	24.67 (+16.67)$_{\pm2.49}$	$67.53_{\pm1.43}$	$\mathbf{1.17x}_{\pm0.04x}$	$6.00_{\pm0.40}$
Llama3-8B CoT	$26.00_{\pm2.53}$	$60.06_{\pm1.76}$	$1.62\text{x}_{\pm0.10\text{x}}$	56.00 (+30.00)$_{\pm2.87}$	$80.27_{\pm1.54}$	$1.56\text{x}_{\pm0.05\text{x}}$	$10.70_{\pm0.56}$
Llama3.1-8B CoT	$\mathbf{37.00}_{\pm2.79}$	$68.34_{\pm1.72}$	$1.70\text{x}_{\pm0.07\text{x}}$	65.33 (+28.33)$_{\pm2.75}$	$\mathbf{84.25}_{\pm1.44}$	$1.52\text{x}_{\pm0.04\text{x}}$	$11.54_{\pm0.56}$
Qwen2.5-7B CoT	$17.00_{\pm2.17}$	$52.61_{\pm1.69}$	$1.81\text{x}_{\pm0.10\text{x}}$	38.00 (+21.00)$_{\pm2.80}$	$70.40_{\pm1.69}$	$1.59\text{x}_{\pm0.06\text{x}}$	$11.01_{\pm0.53}$
Information Seeking Only							
Llama3-8B	$17.67_{\pm2.20}$	$63.70_{\pm1.39}$	$1.21\text{x}_{\pm0.05\text{x}}$	37.33 (+19.66)$_{\pm2.79}$	$76.87_{\pm1.30}$	$1.28\text{x}_{\pm0.04\text{x}}$	$7.61_{\pm0.47}$
Llama3.1-8B	$33.67_{\pm2.73}$	$74.76_{\pm1.39}$	$1.50\text{x}_{\pm0.05\text{x}}$	52.00 (+18.33)$_{\pm2.88}$	$84.07_{\pm1.19}$	$1.37\text{x}_{\pm0.03\text{x}}$	$6.41_{\pm0.36}$
Qwen2.5-7B	$14.00_{\pm2.00}$	$60.06_{\pm1.29}$	$\mathbf{1.06x}_{\pm0.03x}$	24.33 (+10.33)$_{\pm2.48}$	$70.18_{\pm1.29}$	$\mathbf{1.14x}_{\pm0.04x}$	$3.68_{\pm0.28}$
Llama3-8B CoT	$52.33_{\pm2.88}$	$77.91_{\pm1.64}$	$1.35\text{x}_{\pm0.03\text{x}}$	$\mathbf{82.33}$ (+30.00)$_{\pm2.20}$	$91.84_{\pm1.15}$	$1.28\text{x}_{\pm0.03\text{x}}$	$10.15_{\pm0.47}$
Llama3.1-8B CoT	$\mathbf{56.67}_{\pm2.86}$	$\mathbf{81.14}_{\pm1.45}$	$1.42\text{x}_{\pm0.04\text{x}}$	82.67 (+24.00)$_{\pm2.19}$	$\mathbf{93.42}_{\pm0.93}$	$1.25\text{x}_{\pm0.02\text{x}}$	$10.59_{\pm0.47}$
Qwen2.5-7B CoT	$39.33_{\pm2.82}$	$69.57_{\pm1.77}$	$1.59\text{x}_{\pm0.05\text{x}}$	76.67 (+37.34)$_{\pm2.44}$	$89.90_{\pm1.27}$	$1.35\text{x}_{\pm0.03\text{x}}$	$14.44_{\pm0.53}$
No Information Exchange							
Llama3-8B	$81.33_{\pm2.25}$	$92.79_{\pm0.98}$	$\mathbf{1.88x}_{\pm0.04x}$	$\mathbf{97.67}$ (+16.34)$_{\pm0.87}$	$\mathbf{99.39}_{\pm0.25}$	$\mathbf{1.61x}_{\pm0.03x}$	$11.66_{\pm0.57}$
Llama3.1-8B	$\mathbf{84.67}_{\pm2.08}$	$\mathbf{96.13}_{\pm0.62}$	$2.08\text{x}_{\pm0.04\text{x}}$	94.33 (+9.66)$_{\pm1.33}$	$98.50_{\pm0.42}$	$1.68\text{x}_{\pm0.03\text{x}}$	$11.09_{\pm0.60}$
Qwen2.5-7B	$34.67_{\pm2.75}$	$76.90_{\pm1.24}$	$1.91\text{x}_{\pm0.07\text{x}}$	61.33 (+26.66)$_{\pm2.81}$	$88.21_{\pm1.03}$	$1.79\text{x}_{\pm0.05\text{x}}$	$10.28_{\pm0.49}$
Llama3-8B CoT	$33.33_{\pm2.72}$	$66.78_{\pm1.79}$	$2.21\text{x}_{\pm0.10\text{x}}$	73.00 (+39.67)$_{\pm2.56}$	$90.51_{\pm1.06}$	$1.91\text{x}_{\pm0.06\text{x}}$	$17.14_{\pm0.62}$
Llama3.1-8B CoT	$38.33_{\pm2.81}$	$73.83_{\pm1.53}$	$2.16\text{x}_{\pm0.08\text{x}}$	54.33 (+16.00)$_{\pm2.88}$	$83.37_{\pm1.28}$	$1.82\text{x}_{\pm0.06\text{x}}$	$14.35_{\pm0.58}$
Qwen2.5-7B CoT	$39.67_{\pm2.82}$	$71.02_{\pm1.73}$	$2.43\text{x}_{\pm0.10\text{x}}$	65.33 (+25.66)$_{\pm2.75}$	$85.99_{\pm1.31}$	$1.96\text{x}_{\pm0.06\text{x}}$	$15.79_{\pm0.58}$

Table 1: Performance comparison of models with and without verifier assistance across four communicative action space configurations. In each game, both agents are assigned the same action space.

mation Exchange > Providing Only (See Table 1). Enabling both seeking and providing actions consistently yields the highest Success Rate (SR) and the lower Step Ratio (StepR). confirming that bidirectional exchange is both expressive and effective for coordinating constraints, especially when enhanced with chain-of-though reasoning.

Permitting only *information seeking* is better than permitting only *information providing*. Targeted queries minimize redundant traffic and let agents actively access the missing piece of information, whereas blind sharing often floods the channel with constraints that are irrelevant or already known by their partners. Therefore, the result suggests that *information seeking* is a more task-efficient option than unprompted *information providing*.

Surprisingly, disabling communication altogether (*"No Information Exchange"*) ranks high for some variants. Without communication and reasoning process through CoT, the task reduces to pure object manipulation and random guessing; therefore, the model's entire capacity is devoted to mastering game rules and action affordance. Llama variants exploit this by memorizing high-probability transition patterns, achieving

a success rate greater than 81%. Qwen, which is pre-trained on a different instruction distribution, does not show the same effect, suggesting the phenomenon is model–specific rather than an intrinsic property of the environment.

Verifier Assistance (RQ2) As shown in Table 1, adding the reasoning verifier (our best verifier) to the base models leads to large absolute SR gains (labeled in red), ranging from 10 to 40% increment. This reflects the high potential capabilities of the base models, as well as the effectiveness of the assistance from our verifier.

To better understand the underlying behaviors, we further conduct the error analysis across four different action configurations on variants with the best performance (LLaMA-3.1-8B without CoT for no information exchange, and with CoT for the rest, the same for later experiments). We provide detailed explanations on the error taxonomy in Appendix E. Each cell in the table presents error rates in two forms:

- (ERRORS/TOTAL STEPS)%: The proportion of steps affected by this error type.
- (ERRORS/RELEVANT ACTION TYPE)%: The

Category	No Verifier				With Reasoning Verifier			
	Provide & Seek	Provide Only	Seek Only	None	Provide & Seek	Provide Only	Seek Only	None
Format Following	0.19%	0.59%	0.00%	0.00%	0.08%	0.48%	0.02%	0.00%
Physical Understanding								
Object not in source bin	8.34%/11.27%	4.17%/7.75%	10.81%/15.16%	1.47%/1.60%	2.92%/4.06%	1.94%/3.39%	4.89%/7.05%	1.11%/1.19%
Source bin not reachable	1.43%/1.94%	0.77%/1.43%	0.84%/1.18%	0.02%/0.02%	0.97%/1.34%	1.00%/1.75%	0.24%/0.35%	0.02%/0.03%
Dest. bin not reachable	4.27%/5.77%	3.83%/7.11%	3.67%/5.15%	0.00%/0.00%	2.36%/3.28%	2.50%/4.37%	2.06%/2.98%	0.00%/0.00%
Source & destination same	0.19%/0.26%	0.70%/1.30%	0.81%/1.13%	0.02%/0.02%	0.18%/0.26%	0.60%/1.04%	0.24%/0.35%	0.02%/0.03%
Communication								
Redundant knowl. sharing	14.04%/59.86%	33.20%/79.35%	0.08%/0.89%	–	9.60%/39.44%	27.61%/69.37%	0.10%/0.83%	–
No share after seek	0.30%/12.99%	–	0.93%/9.65%	–	0.14%/4.00%	–	0.16%/1.31%	–
Wrong share after seek	0.04%/0.19%	–	0.00%/0.00%	–	0.04%/0.17%	–	0.00%/0.00%	–
Seek known object	0.34%/14.94%	–	1.21%/12.54%	–	0.02%/0.57%	–	0.24%/1.96%	–
Task Reasoning								
Wrong rule understanding	28.22%/38.15%	19.07%/35.46%	27.40%/38.41%	27.35%/29.79%	14.12%/19.62%	13.13%/22.94%	17.53%/25.27%	19.31%/20.67%
Wrong random guessing	8.64%/11.64%	4.25%/7.90%	5.76%/8.07%	16.11%/17.54%	6.12%/8.51%	4.90%/8.56%	3.57%/5.14%	14.77%/15.81%
No Error	44.29%	38.80%	59.61%	55.92%	67.19%	51.37%	75.68%	65.60%
Total Actions	6704	7293	6427	5308	4867	5881	4991	4143

Table 2: Error analysis across different communicative action space with/without verifier.

proportion of a specific action type affected by this error (e.g., move, share, seek).

For example, in the "Wrong Rule Understanding" row under the "Provide & Seek" column without verifier, the value 28.22% indicates that 28.22% of all 6704 total steps contain this error. The accompanying 38.15% denotes that this error occurred in 38.15% of the 4681[1] total "move" actions. One wrong action may fall into multiple error types.

Without using verifier, agents across all action configurations demonstrate a similar level of skill in task reasoning. The relatively better performance of the "no information exchange" might be primarily due to fewer distractions from communication, allowing agents to focus more on enhancing physical understanding and trying all possible solutions. Also, agents struggle with efficient communication. When they are allowed to proactively share information, a large portion of their sharing actions are labeled as redundant (59.86% and 79.35%, respectively). Moreover, responding to partners' requests and actively seeking necessary information are challenging. Interestingly, when agents are restricted to only share information upon request, the redundancy issue is reduced. However, their overall performance still lags behind agents with full abilities, indicating that balanced, bidirectional communication remains crucial for effective collaboration.

With the integration of the reasoning verifier, we observe a consistent reduction in error rates and a decrease in the total number of actions taken by the agents. Notably, there is **a significant improvement in rule understanding**, which aligns well with the design objective of the reasoning verifier. In addition, several issues in communication are

---
[1] The total number of "move" action, 4681, is not presented on the table.

Act. Mode 1	Act. Mode 2	SR↑(%)	StepR↓
Provide & Seek	Provide Only	67.33$_{\pm2.71}$	1.57x$_{\pm0.04x}$
Provide & Seek	Seek Only	78.67$_{\pm2.37}$	1.33x$_{\pm0.02x}$
Provide & Seek	None	78.67$_{\pm2.37}$	1.68x$_{\pm0.04x}$
Provide Only	Seek Only	41.00$_{\pm2.84}$	1.63x$_{\pm0.07x}$

Table 3: Comparison across different action space combinations. All of them are using reasoning verifier.

also well improved with the involvement of the reasoning verifier, indicating its effectiveness in guiding more efficient and purposeful communication behaviors.

These findings support our central claim: although agents may achieve good performance without information exchange, they do so without truly learning the underlying rules, posing severe safety risks. In contrast, **with the support of an environment-based verifier, agents with full communication capabilities exhibit gains both in task performance and rule comprehension**. This combination offers a promising path toward developing safer, more interpretable AI systems.

Mismatched Action Spaces (RQ3) We further examine model behavior in scenarios where two agents are assigned different communicative action spaces. Specifically, we evaluate four asymmetric pairings of action spaces, as shown in Table 3. We exclude the pair Provide only vs. None, as it is functionally equivalent to *Provide & Seek* vs. *None*. In both cases, one agent is unable to respond to queries. Similarly, *Seek only* vs. *None* is the same as *None* vs. *None*, since the *None* agent cannot initiate or respond to communication.

Across the evaluated pairings, we observe a consistent drop in performance when agents have mismatched communicative abilities. This finding highlights the importance of aligning communication protocols in collaborative tasks and provides

Action Mode	SR↑(%)	StepR↓
Provide & Seek	$100.00_{\pm 0.00}$	$1.39x_{\pm 0.10x}$
Provide Only	$100.00_{\pm 0.00}$	$1.41x_{\pm 0.08x}$
Seek Only	$100.00_{\pm 0.00}$	$1.33x_{\pm 0.06x}$
None	$100.00_{\pm 0.00}$	$1.86x_{\pm 0.11x}$

Table 4: Performance of collaboration between best-performing agents and 12 human participants.

insights into designing agent communication strategies in multi-agent systems. Notably, in the last two pairings listed in Table 3, agents are sometimes forced to rely on random guessing. For instance, a *Seek only* agent cannot proactively share unless asked—yet its *Provide only* partner lacks the ability to initiate queries. These structural mismatches lead to particularly sharp declines in task success, strengthening the need for matched-communication agent design.

Human Preference (RQ4) We follow the model selection in Table 3 from **RQ3** and recruit human participants to play games with them. As shown in Table 4, human participants, with complete communicative action space, achieve 100% success rate across all action space configurations within 30 steps for each game. Agents with communicative actions perform efficient collaboration with human participants, while agents that cannot communicate spend far more rounds completing tasks.

In addition to qualitative metrics, we also conduct quantitative analyses to better understand the perceived helpfulness, responsiveness, and clarity of the models from a human-centered perspective. Figure 2 presents the distributions of participant responses across these three dimensions. We find that most participants agree the agents generally communicate useful information and make effective use of the information shared by the human. Notably, agents with the Seek only configuration are rated as especially helpful—likely because they have learned to selectively share the most relevant information when prompted. Interestingly, in the None condition for Q1 (usefulness of communicated information), participants often interpret the agents' physical actions as implicit communication from which they base their evaluations.

However, we observe **strong disagreement regarding the clarity** of agent behavior. Agents with Seek only or None configurations tend to cause more confusion. Specifically, while Seek only agents are seen as helpful and responsive, they still leave users uncertain—possibly because they never proactively offer information before making
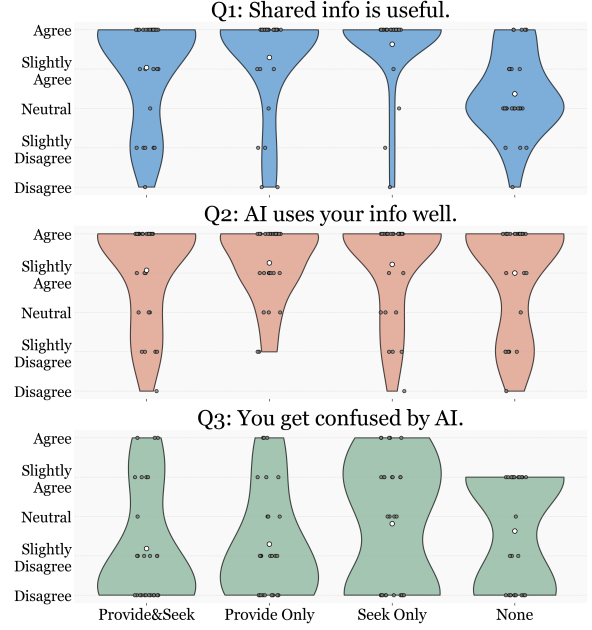

Figure 2: Distributions of answers with 108 data points from 12 human participants for each question. Each participant plays 9 games with one model. The white dots represent the average scores.

a move. This suggests that even if an agent is effective and efficient in completing the task, a lack of initiative in communication can reduce the perceived clarity of its behavior. Similarly, models without any communicative capabilities achieve near perfect performance in self-play (94.33% SR), yet they are perceived as unclear in human-AI collaboration. Instead, although models that can initiate *information providing* take slightly more steps, they offer better clarity in human-AI collaboration. These results highlight that beyond task success, **proactive and transparent communication plays a critical role in fostering human trust and understanding** in collaborative settings.

6 Conclusion

We adapted Einstein's Puzzle to a tabletop environment to study collaboration under information asymmetry between LLM agents. Our empirical results show the critical role of aligned communication, especially information seeking and providing abilities in the success of collaboration. Through detailed error analysis, we identify general limitations in task understanding, which are effectively mitigated by incorporating environment-based verification. Furthermore, a human study highlights the importance of proactive and transparent communication in fostering trust and interpretability. These findings point to a pressing need for reliable,

communication-aware, and interpretable design in future LLM-based collaborative systems.

Limitations

Still our study has known limitations. First, a ready-to-use environment-based verifier relies on the assumption that invalid actions are always recoverable—a prerequisite for trial-and-error-style interaction. While this holds in many simulated environments, extending the approach to real-world settings remains challenging. Doing so would require agents to possess richer perceptual capabilities and a deeper understanding of the environment's dynamics. Nevertheless, the verifier is readily applicable and easily deployable in a wide range of simulated environments, where structured feedback is available.

Second, our human evaluation was conducted on a relatively small scale. Due to constraints in time and computational resources, we were unable to deploy multiple models simultaneously, limiting the ability to compare different configurations within a single user study. In future work, we plan to expand the study by recruiting more participants and enabling broader comparisons across models. This will allow us to conduct more robust analyses of both human preferences and model behaviors in collaborative settings.

Acknowledgments

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, and 26 others. 2022. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*.

AI@Meta. 2024. Llama 3 model card.

Stefano V Albrecht and Peter Stone. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258:66–95.

Cristian-Paul Bara, Sky CH-Wang, and Joyce Chai. 2021. MindCraft: Theory of mind modeling for situated dialogue in collaborative tasks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1112–1125, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. 2019. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems*, 32.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan. Association for Computational Linguistics.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. 2023. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. *arXiv preprint arXiv:2309.13007*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Zane Durante, Qiuyuan Huang, Naoki Wake, Ran Gong, Jae Sung Park, Bidipta Sarkar, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Yejin Choi, and 1 others. 2024. Agent ai: Surveying the horizons of multimodal interaction. *arXiv preprint arXiv:2401.03568*.

Jie Gao, Yuchen Guo, Gionnieve Lim, Tianqin Zhang, Zheng Zhang, Toby Jia-Jun Li, and Simon Tangi Perrault. 2024. Collabcoder: a lower-barrier, rigorous workflow for inductive collaborative qualitative analysis with large language models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–29.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Sven Gronauer and Klaus Diepold. 2022. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2):895–943.

Adrian Groza. 2021. *Einstein Puzzles*, pages 103–130. Springer International Publishing, Cham.

Xinyan Guan, Yanjiang Liu, Xinyu Lu, Boxi Cao, Ben He, Xianpei Han, Le Sun, Jie Lou, Bowen Yu, Yaojie Lu, and 1 others. 2024. Search, verify and feedback: Towards next generation post-training paradigm of foundation models via verifier engineering. *arXiv preprint arXiv:2411.11504*.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*.

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

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. 2024. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*.

Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. 2024. Tree search for language model agents. *arXiv preprint arXiv:2407.01476*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Shalev Lifshitz, Sheila A McIlraith, and Yilun Du. 2025. Multi-agent verification: Scaling test-time compute with multiple verifiers. *arXiv preprint arXiv:2502.20379*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Wei Liu, Chenxi Wang, YiFei Wang, Zihao Xie, Rennai Qiu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, and Chen Qian. 2024a. Autonomous agents for collaborative task under information asymmetry. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Wei Liu, Chenxi Wang, Yifei Wang, Zihao Xie, Rennai Qiu, Yufan Dnag, Zhuoyun Du, Weize Chen, Cheng Yang, and Chen Qian. 2024b. Autonomous agents for collaborative task under information asymmetry. *arXiv preprint arXiv:2406.14928*.

Ziqiao Ma, Jacob Sansom, Run Peng, and Joyce Chai. 2023. Towards a holistic landscape of situated theory of mind in large language models. *Findings of Empirical Methods in Natural Language Processing*.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz, and Jan Leike. 2024. Llm critics help catch llm bugs. *arXiv preprint arXiv:2407.00215*.

Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Yuan-Hong Liao, Joshua B Tenenbaum, Sanja Fidler, and Antonio Torralba. 2020. Watch-and-help: A challenge for social perception and human-ai collaboration. *arXiv preprint arXiv:2010.09890*.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, and 1 others. 2024. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186.

Chen Qian, Zihao Xie, YiFei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2025. Scaling large language model-based multi-agent collaboration. In *The Thirteenth International Conference on Learning Representations*.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2998–3009.

Peter Stone and Manuela Veloso. 2000. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8:345–383.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*.

Jian Wang, Yinpei Dai, Yichi Zhang, Ziqiao Ma, Wenjie Li, and Joyce Chai. 2025. Training turn-by-turn verifiers for dialogue tutoring agents: The curious case of llms as your coding tutors. *arXiv preprint arXiv:2502.13311*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. Association for Computational Linguistics.

Yulong Wang, Tianhao Shen, Lifeng Liu, and Jian Xie. 2024c. Sibyl: Simple yet effective agent framework for complex real-world reasoning. *arXiv preprint arXiv:2407.10718*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zengqing Wu, Run Peng, Shuyuan Zheng, Qianying Liu, Xu Han, Brian Kwon, Makoto Onizuka, Shaojie Tang, and Chuan Xiao. 2024. Shall we team up: Exploring spontaneous cooperation of competing LLM agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, and 1 others. 2024. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Xiao Yu, Maximillian Chen, and Zhou Yu. 2023. Prompt-based monte-carlo tree search for goal-oriented dialogue policy planning. *arXiv preprint arXiv:2305.13660*.

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

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. 2024b. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin King, Xue Liu, and Chen Ma. 2025. What, how, where, and how well? a survey on test-time scaling in large language models. *arXiv preprint arXiv:2503.24235*.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024a. Language agent tree search unifies reasoning acting and planning in language models. *URL https://arxiv.org/abs/2310.04406*.

Xuhui Zhou, Zhe Su, Tiwalayo Eisape, Hyunwoo Kim, and Maarten Sap. 2024b. Is this the real life? is this just fantasy? the misleading success of simulating social interactions with llms. *arXiv preprint arXiv:2403.05020*.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. 2024c. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *The Twelfth International Conference on Learning Representations*.

Appendix

We include the following contents as our supplementary materials.

- **A. Example of Game Play** — An illustrative walkthrough of a full game session demonstrating player turns, actions, and reasoning strategies.
- **B. Data Generation** — A detailed description of how we generate trajectories using a planner with perspective-taking, inference, and communication modeling.
- **C. Ablation on Verifiers** - A detailed explanation and experiment on different design of verifier, which is potentially generalizable to other simulated environments.
- **D. Error Taxonomy** - A detailed explanation on the error types we analyze in **Exp2** in section 4.
- **E. Experiment Details** — Technical and procedural specifications of model training, deployment, and human evaluation setup.
- **F. Code of Ethics** — Statement of ethical compliance, consent protocol, and risk mitigation measures approved by the IRB.
- **G. Human Study Interface** — Screenshot and description of the web interface and tutorial used for guiding participants in the human study.
- **H. Prompts Used** — A comprehensive collection of system and user prompts used across training, evaluation, and reasoning trace generation for different agent configurations.

In terms of generative AI usage, we use it for purely improving the language of the paper.

A Example of Game Play

To provide a clear understanding of the Einstein Puzzle gameplay, we illustrate a complete game session in Figure 3, building on the initial setup introduced in Figure 1. This example demonstrates how players are expected to take turns, make decisions, and communicate throughout the game. We hope this serves as a helpful reference for understanding the nature of the task, as well as the types of reasoning and interaction involved.

It is worth noting that the ask action—used to seek information—is not utilized in this example. This is primarily due to the relative simplicity of the case. The ask action tends to be used more frequently in scenarios involving a larger number of objects, or when agents need to inquire about specific object-related information.

B Data Generation

B.1 Perspective Taking

To prepare data for fine-tuning, we designed a planner to generate solutions under different configurations of action space.

The planner operates from the perspective of a single player in each turn, selecting moves based on the player's limited knowledge of the constraints and their specific communicative action space. It performs breadth-first search to explore possible action sequences until a valid solution is found. Throughout the search process, the planner maintains an up-to-date representation of the player's knowledge, incorporating both the communication history and the current positions of objects on the board. Based on this evolving knowledge state, it selects valid next actions—such as sharing constraints that have not yet been communicated, or asking about objects that remain unknown from the player's perspective.

To improve search efficiency, we incorporate inferred knowledge into action selection, which is also used by the reasoning verifier introduced in Section 4. This inferred knowledge enables the planner to determine whether:

1. The goal of an object is already **known**, in which case redundant information-seeking is avoided, and a valid move (placing the object into its goal bin, if reachable) is added;
2. The goal of an object is still **unknown**, in which case the agent may ask for information about that object.

By tracking communication history, the planner can also avoid redundant sharing by identifying which constraints have already been communicated.

If no valid move is available in a given search step, we manually add a pass action to allow the player to skip their turn. This mechanism is important when the two players take different numbers of actions to complete a task. Any trajectory ending with two consecutive passes is pruned to avoid unnecessary stalling.

In scenarios where no communication is allowed, or where information flow is unidirectional, the agent may have to randomly guess the goals of some objects. This is triggered only after all valid actions have been exhausted, before skipping their
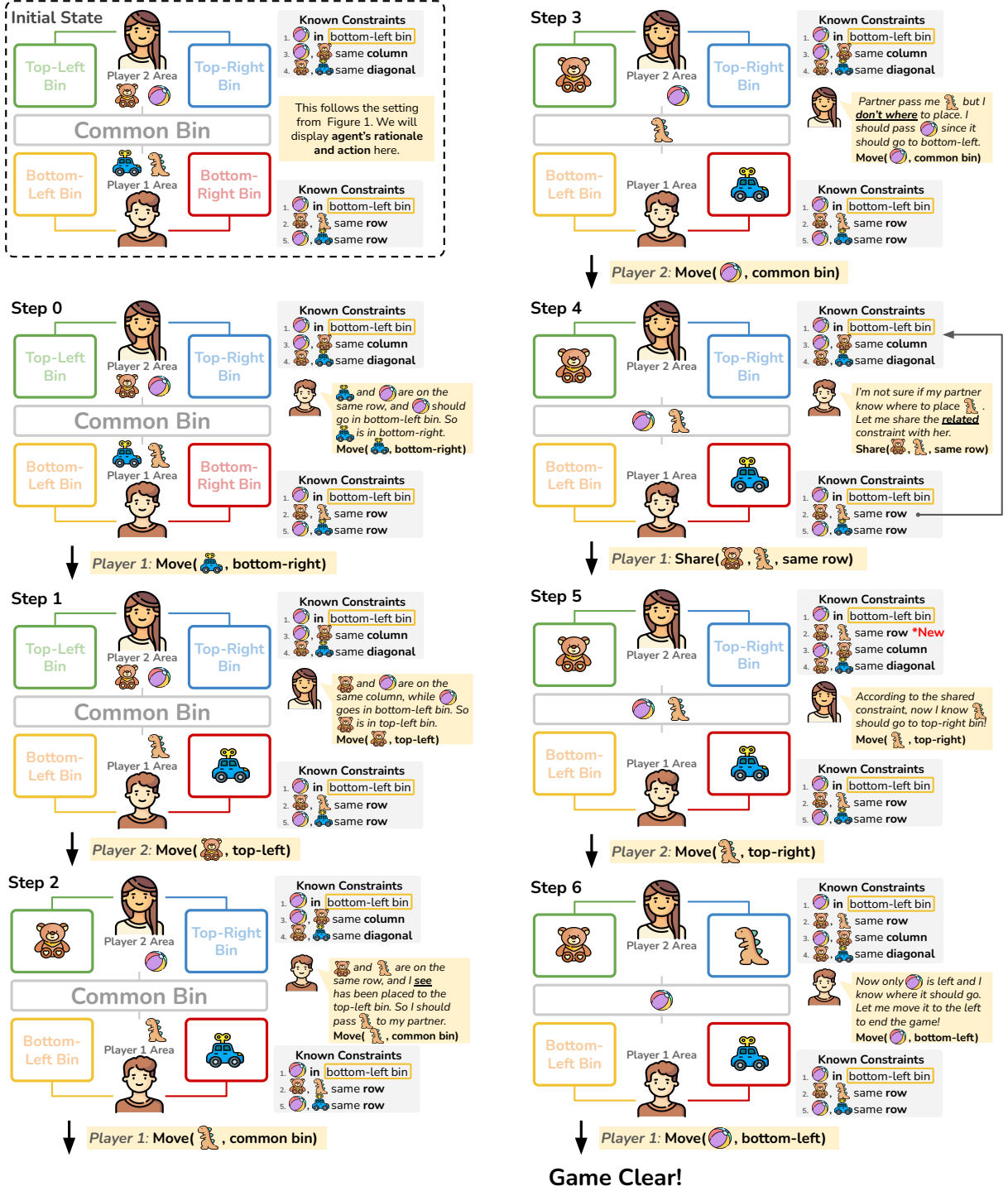
Figure 3: Full game playthrough with actions and rationales. The game begins with Player 1, and the two players take turns performing physical moves, sharing information, or asking questions until all objects are correctly placed.

turn. The agent will then attempt to place reachable, unplaced objects into all reachable bins (including the common bin), one by one, until the correct placement is accepted by the environment. Since the environment prevents invalid placements, this process can still lead to a valid solution. However, agents are discouraged from guessing prematurely and are designed to prioritize valid actions before

resorting to this strategy.

B.2 Optimal And Near-Optimal Trajectories

We collect both optimal and near-optimal trajectories—those that deviate from the optimal solution by only one or two steps but exhibit diverse strategies. This diversity ensures that both types of communicative actions—information providing

and information seeking—are well represented in the demonstrations. Without this balance, solutions dominated by unprompted sharing would disproportionately appear, as they often require fewer steps to complete.

In scenarios involving random guessing, the optimal trajectories are those in which agents correctly guess the target location on the first attempt. However, such demonstrations provide little guidance for learning robust guessing strategies. To address this, we also include trajectories where agents try multiple possible bins—an approach we observe being learned by several fine-tuned models in Table 1.

C Ablation on Environment-Based Verifiers

C.1 Verifier Design

In addition to the reasoning verifier introduced in Section 4, we design two more types of verifiers as is illustrated in Figure 4.

1. **Affordance Verifier**: The environment inherently enforces physical rules that govern the actions an agent can take under different conditions. In our game, this refers to whether the selected action is executable, such as whether a object or bin is reachable, or whether the chosen object is in the correct source bin. This helps validate the agent's decision from the perspective of physical feasibility.

2. **Communication Verifier**: In multi-agent environments, other agents can be viewed as part of the environment, and their interactions can provide additional feedback. In our game, the communication verifier assesses whether the communicative action selected by the agent is meaningful. For example, it identifies when an agent shares already-known knowledge, repeats sharing an existing constraint, or asks about a object that has already been placed.

These three verifiers (including reasoning verifier) are hierarchically related in our design. The affordance and communication verifiers address physical actions and communication, respectively, while the reasoning verifier builds upon both and extends coverage through inferred knowledge.

The affordance verifier is widely applicable and available in most environments. The communication verifier is similarly accessible whenever the task involves communication. In contrast, the reasoning verifier is more environment-

and task-specific, although it can often be enabled through custom algorithm design. Used together, the environment-based verifier framework can be viewed as a general and flexible approach that can be adapted to a wide range of simulated environments.

C.2 Ablation Study

We follow the same setting of Exp2 (see Section 4 with two additional verifiers that separately target action affordance and communication. We compare performance across different verifier settings with Llama3.1-8B CoT model as base, and assess whether the environment-based verifier is potentially generalizable to other environments.

As is shown in Table 5, the affordance verifier evaluates physical preconditions of actions and improves success rates (SR) by 2–7%. The communication verifier filters out redundant or uninformative exchanges, contributing up to a 9% SR increase in task completion. Importantly, these verifiers operate solely on feedback from the environment and interaction history, requiring no additional training or computational overhead.

We argue that such a mechanism **offers a promising alternative** to recent agent modeling approaches, especially in simulated environments where rich, structured feedback is readily available. These results invite a broader reconsideration of the environment's role—not merely as a testing ground, but as an active, model-free verifier that can guide agent behavior in a lightweight manner.

D Experiment Details

D.1 Model Configuration, Fine-tuning and Deployment

We utilize the Azure OpenAI services for our GPT models. For GPT-4o, we employ the GPT-4o-20241120 version, and in all experiments, the temperature is set to 0.2 and the top-p value to 0.9. For all the model fine-tuning, we employ LoRA (Hu et al., 2022) with a rank of 32, training with a global batch size of 128 and a learning rate of 2e-4 using a cosine decay schedule for 1 epoch. Fine-tuning is conducted using OpenRLHF (Hu et al., 2024), while FlashAttention-2 (Dao, 2024) is used to speed up training. The process takes approximately 30 minutes on 4 A40 GPUs with 48GB RAM each. For evaluation, we deploy the model using PeFT (Mangrulkar et al., 2022). For inference in the human study, we deploy the model
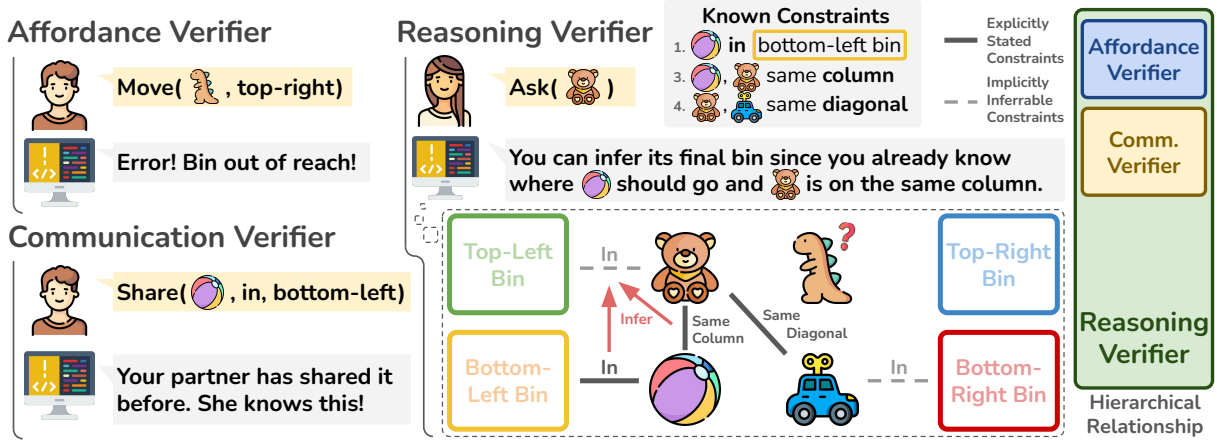
Figure 4: Illustration of three types of verifications we consider. Following the same setup as we showed in Figure 1, the game environment supports providing feedbacks related to action affordance, communication and strategies, which can be directly used as verifiers for agents' decisions.

Action Mode	No Verifier (Pass@1)		Affordance Verifier		Communication Verifier		Reasoning Verifier	
	SR↑(%)	StepR↓	SR↑(%)	StepR↓	SR↑(%)	StepR↓	SR↑(%)	StepR↓
Provide & Seek	$58.67_{\pm2.84}$	$1.87x_{\pm0.05x}$	$65.67_{\pm2.74}$	$1.83x_{\pm0.05x}$	$65.00_{\pm2.75}$	$1.74x_{\pm0.05x}$	$89.33_{\pm1.78}$	$1.52x_{\pm0.03x}$
Provide Only	$37.00_{\pm2.79}$	$1.70x_{\pm0.07x}$	$38.67_{\pm2.81}$	$1.72x_{\pm0.06x}$	$46.00_{\pm2.88}$	$1.65x_{\pm0.05x}$	$65.33_{\pm2.75}$	$1.52x_{\pm0.04x}$
Seek Only	$56.67_{\pm2.86}$	$1.42x_{\pm0.04x}$	$62.33_{\pm2.80}$	$1.43x_{\pm0.04x}$	$55.33_{\pm2.87}$	$1.40x_{\pm0.04x}$	$82.67_{\pm2.19}$	$1.25x_{\pm0.02x}$
None	$38.33_{\pm2.81}$	$2.16x_{\pm0.08x}$	$42.67_{\pm2.86}$	$2.18x_{\pm0.09x}$	-	-	$54.33_{\pm2.88}$	$1.82x_{\pm0.06x}$

Table 5: Performance under different verification settings tested on Llama3.1-8B model with CoT.

using vLLM (Kwon et al., 2023).

We follow the license requirement of Llama3.1, Qwen2.5, and GPT-4o model when using these artifacts, and our implementation is licensed under the MIT License.

D.2 Human Evaluation Setup

We recruited 12 human subjects with no prior experience in Einstein Puzzles on Tabletop to evaluate the models under four different action space configurations. Before the experiment began, each participant signed a consent form. We prepared 27 unseen game scenarios across the four configurations and divided them into three groups, each containing 9 distinct games (3 games each with 4, 5, and 6 objects). Each participant was assigned to one group and paired with a model using one of the action space configurations, without being informed of which model they were interacting with. As a result, each group was tested by 4 participants—one per configuration. Each session lasted approximately 30 minutes, and participants received a $20 Amazon gift card as compensation.

At the start of the study, participants were introduced to the task environment via a detailed tutorial that explained the environment, task setup, and interface (see Appendix G). After the tutorial, participants completed 10 sessions sequentially, be-

ginning with a practice session which is not taken into the result. In each session, they were presented with an initial game board layout and explicit constraints, and were required to communicate with the model with communicative actions to solve the task. Upon task completion, a feedback form with three questions was shown. Once the form was submitted, the interface advanced to the next session, continuing until all games in the assigned group were completed. Participants were allowed to give up at any point if they felt stuck or not comfortable. Additionally, a maximum step limit of 30 was imposed to prevent excessive task duration. This same constraint was applied in all the evaluations (see Table 1) to ensure a fair comparison.

E Error Taxonomy

To better understand LLM agents' behaviors, we define several error types that LLM agents may encounter during interaction. Broadly, these fall into four categories: format following, physical understanding, communication, and task reasoning.

- **LLM's format following**

  - **Invalid Action**: The LLM fails to follow the required output format or exceeds the token limit.

- **Physical Understanding**

  – **Object not in source bin**: The agent specifies a move involving an incorrect source location for the object.

  – **Source bin not reachable**: The agent attempts to move an object from a bin that is not reachable (only bins at the front and the common bin are reachable).

  – **Destination bin not reachable**: The agent attempts to place an object into a non-reachable bin.

  – **Source and destination bin are same**: The agent mistakenly assigns the same bin as both the source and destination.

- **Communication**

  – **Redundant knowledge sharing**: The agent redundantly shares knowledge already communicated by itself or its partner.

  – **No share after seek**: The agent fails to respond to its partner's information-seeking request.

  – **Wrong share after seek**: The agent provides incorrect or irrelevant information in response to a request.

  – **Seek known object**: The agent asks for the location of an object whose location it already knows, indicating inefficient behavior.

- **Task Reasoning**

  – **Wrong rule understanding**: The agent failed to interpret or infer the right location, leading to incorrect moves when it should be able to do so.

  – **Wrong random guessing**: The agent, lacking sufficient information, guesses randomly and places the object incorrectly.

F   Code of Ethics

The institution's Institutional Review Board (IRB) considered this project exempt from ongoing review. The data collection process among researchers and participants is in line with standard ethical practice.

Consent Statement.   You are invited to participate in a research study that intends to evaluate generative AI agents that can communicate and collaborate with their human partners to complete tasks. If you agree to be part of the research study,

you will be asked to interact with the AI agents to accomplish a set of tasks. The tasks include: (1). completing a logical board game with AI agents; (2). sharing necessary information with AI agents to help them complete the tasks; (3). asking AI agents for necessary information that will help you to complete the tasks. The study will last approximately an hour. The interaction history, i.e., only the text generated by AI models and the subjects' symbolic inputs, and numerical evaluations, will be recorded in a datafile. The data collected in this study will be analyzed and used for research purposes. No personally identifiable information will be stored in the datafile.

Potential Harm.   The game setting and the tasks assigned to participants were designed and strictly controlled by the research team. This ensured that the potential for safety concerns was minimized, allowing participants to engage with the study with minimal risk. Data collection involved only non-personal information, adhering to standard ethical practices and was used exclusively for research purposes. We ensured confidentiality and privacy, and the data will not be published publicly. Please refer to Appendix D.2 for implementation details of our human study.

G   Human Study Interface

We deploy a web-based interface to facilitate our human study. To ensure that all participants understand the task, interface elements, and available actions, we provide a detailed tutorial at the beginning of the study. This tutorial is displayed before the first game session and serves as a self-contained guide covering the game objective, interaction mechanics, and platform layout. For completeness and transparency, we include the full tutorial content below, as it was shown to participants, without modification. This also naturally serves as the introduction of the interface we design.

G.1   Overview

In this study, you will play a logic-based tabletop game in collaboration with an AI agent. The goal of the game is for you and the AI agent to work together to place objects into designated bins according to a given set of constraints.

Each constraint defines either a relationship between two objects or between an object and a bin. The bins are the two player bins, the four destination bins, and the common area bin. The types of
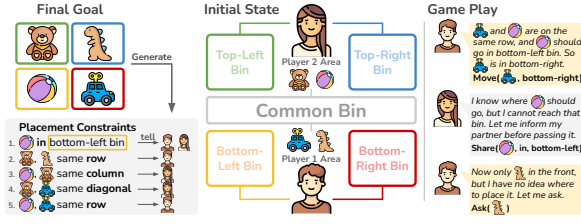
Figure 5: A visual overview of the collaborative game setting. Two players (you and the AI agent) work together to place objects into goal bins based on a set of relational constraints.

Figure 6: Overview of the user interface layout.

constraints that you may receive include the following:

1. Object1 and Object2 must be in the same **row**
2. Object1 and Object2 must be in the same **column**
3. Object1 and Object2 must be on the same **diagonal**
4. Object1 and Object2 must be in the same **bin**
5. Object1 must be placed in **binA**

To avoid ambiguity, each pair of objects has only one constraint type describing their relationship. For example, if Object1 and Object2 are said to be in the same row, it implies that they are not in the same bin. Below is a visualization of our collaborative game.

G.2 Actions

You can choose from four possible actions during your turn:

1. **Move** – Move a block from one bin to another.
2. **Share** – Share one of your constraints with the AI partner.
3. **Ask** – Ask your AI partner about the placement of an object.
4. **Skip** – Pass your turn without taking any action.

You and the AI agent will take turns performing actions. The objective is to complete the task using the fewest possible steps. Note that the AI agent is not perfect and may make suboptimal decisions. Your collaboration and guidance are key to success.

G.3 Platform Introduction

Once the game begins, you will see the following components:

1. The **game board**
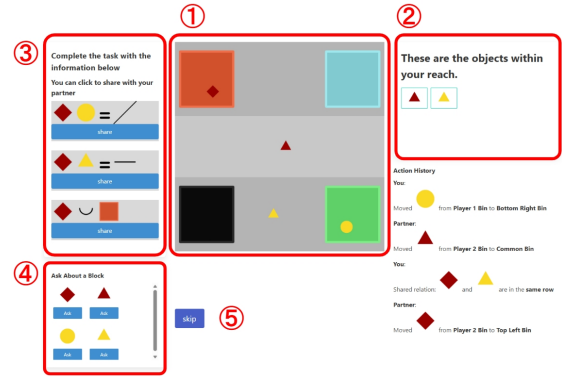2. The **objects** within your reach (which you can move)

3. The **constraints** available to you (which you can share)
4. The **objects** you can ask about
5. The **Skip** button, if you wish to pass your turn
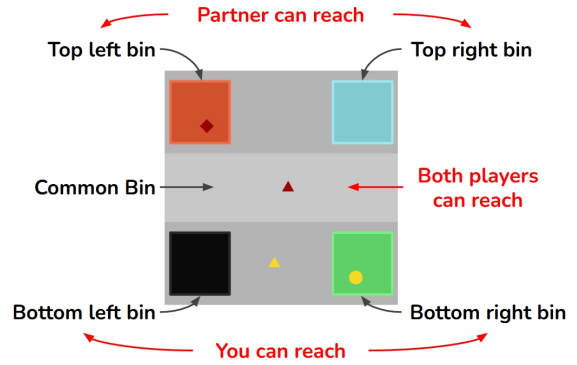
G.4 Game Board Description

Figure 7: Layout and bin positions on the game board.

The board includes four colored bins (top-left, top-right, bottom-left, bottom-right) and one common bin in the center. You (Player 1) are positioned at the bottom, and the AI agent (Player 2) is at the top.

You can only move objects to the bins in front of you and the common bin. To move an object to a bin that is out of your reach, you must place it in the **common bin** so your partner can complete the move.
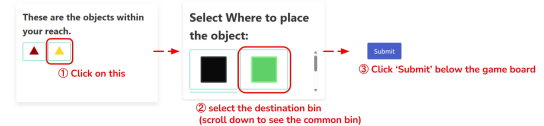
G.5 How to Move

Figure 8: Step-by-step instructions for moving objects.

17

To move an object:

1. Click on the object you want to move.

2. Select the destination bin (scroll down if needed to see all bins).

3. Click **'Submit'** below the game board to confirm the move.

G.6 Share and Ask Actions

To share a constraint or ask about an object:

1. Click the blue button under the relevant constraint or object.

2. Click **'Submit'** to confirm your action.
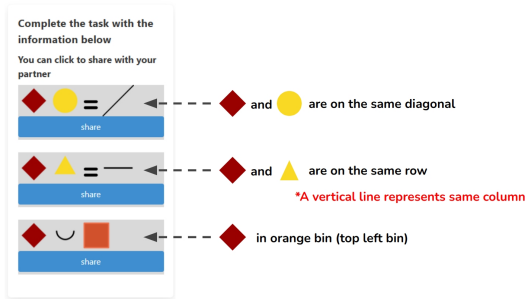
G.7 Understanding Constraints

Figure 9: Example of constraint-sharing interface.

Constraints specify how objects should be arranged. You may share **one constraint per turn**, and you may repeat the same constraint if needed for clarification.

G.8 Action History

Figure 10: The action history log at the bottom-right corner.

In the bottom-right corner of the screen, you'll find the **action history**. This log shows all actions taken by both you and the AI since the beginning of the game. Use this to:

1. Review your partner's most recent action

2. Check whether each action was successfully executed

Mistakes made by either player will also appear in this log, helping you keep track of progress and errors.

G.9 At the End of the Game

The game ends when:

1. All objects are correctly placed, or

2. The maximum number of turns (30 in this game) is reached

At the end of each game, you will be prompted to complete a short survey with three questions:
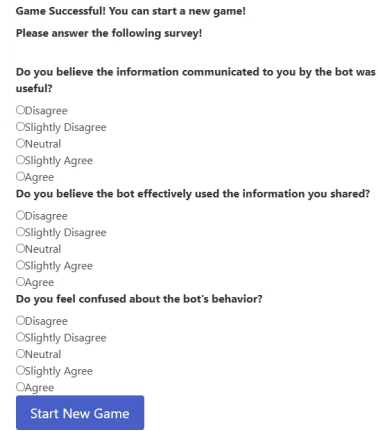

Figure 11: End-of-game feedback form.

Please answer based on your experience in this particular game and click **'Start New Game'** to proceed. You will play **10 games in total** (the first game is mainly for familiarization, and the performance will not be taken into account), with each game taking approximately **3–8 minutes**.

G.10 Final Notes

Once you start the game, you will **not** be able to return to this tutorial. Please read all instructions carefully before beginning. If you feel uncomfortable at any point or wish to exit the study, you may simply close the browser window.

If you're ready to begin, click **'Go To Game'** below.

H Prompts Used

H.1 Prompts for Model Training & Evaluation

We prepare prompts for four distinct action space configurations, each with and without chain-of-thought (CoT) reasoning. While the system prompt

is tailored to each configuration, the user prompt remains consistent across all four. For configurations with CoT, the system prompt includes several illustrative examples to demonstrate the expected reasoning process.

To enhance readability, we provide the full system prompt for the *Providing & Seeking* configuration with CoT. For the remaining configurations, we highlight only the differences relative to this version. The primary distinctions among the four configurations lie in their permitted action spaces and the corresponding reasoning examples. Although the reasoning examples are largely shared across configurations, minor variations are introduced to reflect the specific situations each agent may encounter.

For the output format, models with CoT reasoning are expected to output their reasoning traces and actions in the format of: <THINK><your reasoning></THINK><ACTION><your action></ACTION>, while the one with no CoT reasoning capability needs to follow the format of: <ACTION><your action></ACTION>.

```
You are playing a cooperative game where you and another
player must sort blocks into the correct bins as quickly as
possible. Each player has knowledge about the expected
placement of the blocks, such as whether blocks should be
aligned in the same row, column, or diagonal. You can only
move blocks into bins near you or into a shared bin
accessible to both players. You cannot access the other
player's bins.

The game concludes when all blocks are correctly placed in
the bins. During your turn, you have several options:
- Move a block from a bin to another bin. Both bins must be
accessible to you.
- Share a piece of knowledge with the other player. The
shared knowledge should be selected from the knowledge you
have. You cannot share knowledge you do not have.
- Request knowledge from the other player about a specific
block.
- Pass your turn.

## General Guidance
- You can only move blocks to bins accessible to you.
- You can only share knowledge you have. DO NOT share
knowledge you do not have, nor request knowledge you already
have.
- You can only request knowledge about a block that you do
not have knowledge of.
- You can only move one block at a time.
- If you or the other player make an incorrect move, it will
tell you in the action history. DO NOT make the same
incorrect move again.

## Action Format
Your actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Share knowledge: "share <knowledge>"
- Request knowledge: "ask <block>"
- Pass your turn: "pass"

where we have the following bin names:
- player1_bin
- player2_bin
- commonbin
- top_left_bin
- top_right_bin
- bottom_left_bin
- bottom_right_bin
```

```
and the following knowledge types:
- (<block1>, <block2>, same, row)
- (<block1>, <block2>, same, column)
- (<block1>, <block2>, same, diagonal)
- (<block1>, <block2>, same, bin)
- (<block1>, in, <bin>)

and the following block names:
- block0
- block1
- block2
...

Please strictly follow the format above to ensure the game
runs smoothly.

## Reasoning Phase

Before you take an action, you should reason about the
current state of the game.

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should share my knowledge
about block1 with my partner so that it can move it to the
correct position."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try one of the block in
front of me."
- "I don't know where the block3 should go in. I should ask
my partner about the knowledge of block3."


## Output Format
Please provide your output in this format:
<THINK><your reasoning></THINK><ACTION><your action></ACTION>
```

Listing 1: System prompt for fine-tuned *Providing & Seeking* agents with chain-of-thought reasoning.

```
You are playing a cooperative game ...

The game concludes when all blocks are correctly placed in
the bins. During your turn, you have several options:
- Move a block from a bin to another bin. Both bins must be
accessible to you.
- Share a piece of knowledge with the other player. The
shared knowledge should be selected from the knowledge you
have. You cannot share knowledge you do not have.
- Pass your turn.

## General Guidance
- You can only move blocks to bins accessible to you.
- You can only share knowledge you have. DO NOT share
knowledge you do not have.
- You can only move one block at a time.
- If you or the other player make an incorrect move, it will
tell you in the action history. DO NOT make the same
incorrect move again.

## Action Format
Your actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Share knowledge: "share <knowledge>"
- Pass your turn: "pass"


...

## Reasoning Phase

Before you take an action, you should reason about the
current state of the game.

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
```

```
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should share my knowledge
about block1 with my partner so that it can move it to the
correct position."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try one of the block in
front of me."
- "I don't know where the block3 should go in. I should wait
for my partner to inform me about where to place block3."

...
```

Listing 2: System prompt for fine-tuned *Seeking-Only* agents with chain-of-thought reasoning. Redundant part is omitted.

```
You are playing a cooperative game ...

## General Guidance
- You can only move blocks to bins accessible to you.
- You can only share knowledge you have and share it when you
are asked. DO NOT share knowledge you do not have, nor
request knowledge you already have.
- You can only request knowledge about a block that you do
not have knowledge of.

...

- Pass your turn: "pass"

Notice that you cannot initiate sharing the knowledge. Only
when you are asked by your partner can you share the
knowledge.

...

## Reasoning Phase

Before you take an action, you should reason about the
current state of the game.

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should wait for my partner
to ask about block1 so that I can share the related
knowledge."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try one of the block in
front of me."
- "I don't know where the block3 should go in. I should ask
my partner about the knowledge of block3."

...
```

Listing 3: System prompt for fine-tuned *Provide-Only* agents with chain-of-thought reasoning. Redundant part is omitted.

```
You are playing a cooperative game ...

The game concludes when all blocks are correctly placed in
the bins. During your turn, you have several options:
- Move a block from a bin to another bin. Both bins must be
accessible to you.
- Pass your turn.

## General Guidance
- You can only move blocks to bins accessible to you.
- You can only move one block at a time.
```

```
- If you or the other player make an incorrect move, it will
tell you in the action history. DO NOT make the same
incorrect move again.

## Action Format
Your actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Pass your turn: "pass"

...

## Reasoning Phase

Before you take an action, you should reason about the
current state of the game.

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should wait for my partner
to put it into the common bin so that I can reach it."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try to place it to one of
the bins in front of me."
- "I can reach Block1 and Block3 since they are in front of
me. I know they should be in the same row, but I do not know
either of their exact expected locations. I will try with
moving Block1 to the top-left bin as a start, supposing they
are both on the top row."

...
```

Listing 4: System prompt for fine-tuned *No-Information-Exchange* agents with chain-of-thought reasoning. Redundant part is omitted.

```
You are Player{player_id} on the {side} side of the game
board. You have the following knowledge for your goal:
{knowledge}
Currently, the blocks are located as follows:
{blocks}
You can access these bins:
{bins}

The history of the game till now:
{move_history}

What action would you like to take? Please provide your
reasoning before your action.
```

Listing 5: User prompt for fine-tuned agents with chain-of-thought reasoning.

```
You are Player{player_id} on the {side} side of the game
board. You have the following knowledge for your goal:
{knowledge}
Currently, the blocks are located as follows:
{blocks}
You can access these bins:
{bins}

The history of the game till now:
{move_history}

What action would you like to take?
```

Listing 6: User prompt for fine-tuned agents without chain-of-thought reasoning.

H.2 Prompts for Evaluation with GPT4o

The prompts used for GPT4o evaluation is slightly different than the ones we use for fine-tuned model

training and evaluation. The prompts designed for GPT4o involves more detailed explanations and proper guidance to make sure the comparison is relatively fair. We have also tried using the same prompts for evaluation, while the preliminary result shows that GPT4o is hard to understand the game setting. This drives us to add extra guidance for a better comparison.

```
You are playing a cooperative game where you and another
player must sort blocks into the correct bins as quickly as
possible. Each player has knowledge about the expected
placement of the blocks, such as whether blocks should be
aligned in the same row, column, or diagonal. You can only
move blocks into bins near you or into a shared bin
accessible to both players. You cannot access the other
player's bins.

The game concludes when all blocks are correctly placed in
the bins. During your turn, you have several options:
- Move a block from a bin to another bin. Both bins must be
accessible to you.
- Share a piece of knowledge with the other player. The
shared knowledge should be selected from the knowledge you
have. You cannot share knowledge you do not have.
- Request knowledge from the other player about a specific
block.
- Pass your turn.

## General Guidance
- You can only move blocks to bins accessible to you.
- You can only share knowledge you have. DO NOT share
knowledge you do not have, nor request knowledge you already
have.
- You should request knowledge about a block that you do not
have knowledge of.
- You can only move one block at a time.
- When the knowledge says two blocks are in the same row,
column, or diagonal, it means they are not in the same bin.
- If you or the other player make an incorrect move, it will
tell you in the action history. You MUST NOT make the same
incorrect move again. You MUST carefully check the action
history before you make a move.
- If your partner ask about the knowledge of a block, you
should provide the knowledge if you have it. Carefully think
about which piece of knowledge is the most helpful to share.
- The game will stop players from placing the blocks into the
wrong bins. So if you see a block placed in a bin, that means
it is the correct bin for that block.


## Reasoning Format
You need to reason about which action to take before you make
the decision. Some examples of reasoning:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is on the same row with block2. Block2 is on the
same column with block3. So I can deduce that block1 and
block3 should be on the same diagonal."
- "Block1 is still not moved but I cannot reach it. I should
share my knowledge about block1 with my partner so that it
can move it to the correct position."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may ask my partner about the knowledge
of it."
- "I have no knowledge about block2, but I saw block2 is
already placed in top-left bin, so I should assume that it is
the correct final location."

## Action Format
Your actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Share knowledge: "share <knowledge>"
- Request knowledge: "ask <block>"
- Pass your turn: "pass"

where we have the following bin names:
```

```
- player1_bin
- player2_bin
- commonbin
- top_left_bin
- top_right_bin
- bottom_left_bin
- bottom_right_bin

and the following knowledge types:
- (<block1>, <block2>, same, row)
- (<block1>, <block2>, same, column)
- (<block1>, <block2>, same, diagonal)
- (<block1>, <block2>, same, bin)
- (<block1>, in, <bin>)

and the following block names:
- block0
- block1
- block2
...

Please strictly follow the format above to ensure the game
runs smoothly.

## Output Format
Please provide your output in this format:
<THINK><your reasoning></THINK><ACTION><your action></ACTION>
```

Listing 7: System prompt for GPT4o agents with chain-of-thought reasoning.

```
You are Player{player_id} on the {side} side of the game
board. You have the following knowledge for your goal:
{knowledge}
Currently, the blocks are located as follows:
{blocks}
You can access these bins:
{bins}

The history of the game till now:
{move_history}

What action would you like to take? You need to be fully
convinced of your action before you make a move. Please
provide your reasoning before your action. Your reasoning
should be concise enough within 3 sentences.
```

Listing 8: User prompt for GPT4o agents without chain-of-thought reasoning.

H.3 Prompts for Generating Reasoning Traces with GPT4o for Model Fine-tuning

Using a large, well-trained language model to generate reasoning traces as supervision for smaller models has been widely recognized as an effective strategy to enhance reasoning capabilities. In our setup, we leverage GPT-4o to generate such reasoning traces, following the pipeline outlined below:

1. We first use a planner to generate a good solution for a given game instance. The generation process can be found in Appendix B.

2. At each turn, we present GPT-4o with both the current game state and the corresponding action suggested by the planner.

3. GPT-4o is then prompted to assume it is the agent taking the given action, and to generate a rationale for this decision from a first-person perspective.

The prompts used for this process are provided below. As with the training setup, we employ distinct system prompts for each of the four action space configurations, while keeping the user prompt consistent across all settings.

```
You are the assistant that provides the reasoning process for
the given plays of one player in the game.

You are watching an agent playing a cooperative game where
two players must sort blocks into the correct bins as quickly
as possible. Each player has knowledge about the expected
placement of the blocks, such as whether blocks should be
aligned in the same row, column, or diagonal. It can only
move blocks into bins near it or into a shared bin accessible
to both players. It cannot access the other player's bins.

The game finishes when all blocks are correctly placed in the
required bins. During the agent's turn, it has several
options:
- Move a block to a nearby bin or the shared bin.
- Share a piece of knowledge with the other player.
- Request knowledge from the other player about a specific
block.
- Pass its turn.

Its actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Share knowledge: "share <knowledge>"
- Request knowledge: "ask <block>"
- Pass its turn: "pass"

## Your Task
You are given the agent's action and you need to provide the
reasoning behind the action. Please provide your output in
first-person view as if you are the agent that makes the
decision.

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should share my knowledge
about block1 with my partner so that it can move it to the
correct position."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try one of the block in
front of me."
- "I don't know where the block3 should go in. I should ask
my partner about the knowledge of block3."

## General Guidelines

- Provide a clear and concise explanation for the agent's
action. No more than 2-3 sentences are needed.
- The given action may not be the best move, but you should
explain the reasoning behind it.
- You can refer to the agent as "I" or "me" in your response.
```

Listing 9: System prompt for GPT4o generating reasoning traces for *Providing & Seeking* agents.

```
You are the assistant that provides the reasoning process for
the given plays of one player in the game.

...

The game finishes when all blocks are correctly placed in the
required bins. During the agent's turn, it has several
options:
- Move a block to a nearby bin or the shared bin.
- Share a piece of knowledge with the other player.
- Pass its turn.

Its actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Share knowledge: "share <knowledge>"
```

```
- Pass its turn: "pass"

...

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should share my knowledge
about block1 with my partner so that it can move it to the
correct position."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try one of the block in
front of me."
- "I don't know where the block3 should go in. I should wait
for my partner to inform me about where to place block3."

...
```

Listing 10: System prompt for GPT4o generating reasoning traces for *Seeking-Only* agents. Redundant part is omitted.

```
You are the assistant that provides the reasoning process for
the given plays of one player in the game.

...

The game finishes when all blocks are correctly placed in the
required bins. During the agent's turn, it has several
options:
- Move a block to a nearby bin or the shared bin.
- Share a piece of knowledge with the other player.
- Request knowledge from the other player about a specific
block.
- Pass its turn.

Its actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Share knowledge: "share <knowledge>"
- Request knowledge: "ask <block>"
- Pass its turn: "pass"

Notice that the agent cannot initiate sharing the knowledge.
Only when the agent is asked, it can share the knowledge.

...

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should wait for my partner
to ask about block1 so that I can share the related
knowledge."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try one of the block in
front of me."
- "I don't know where the block3 should go in. I should ask
my partner about the knowledge of block3."

...
```

Listing 11: System prompt for GPT4o generating reasoning traces for *Provide-Only* agents. Redundant part is omitted.

```
You are the assistant that provides the reasoning process for
the given plays of one player in the game.

...
```

```
The game finishes when all blocks are correctly placed in the
required bins. During the agent's turn, it has several
options:
- Move a block to a nearby bin or the shared bin.
- Pass its turn.

Its actions must be formatted as follows:
- Move block: "move <block> from <bin> to <bin>"
- Pass its turn: "pass"

...

Some examples:
- "According to my knowledge, block0 should be in top-right
bin. Since I cannot reach the top-right bin, I should pass it
to the common bin so that my partner can take it."
- "I know block1 and block2 should be in the same row. I also
know block1 should be placed in the top-right bin. So I
should move block2 to the top-left bin, which is also in the
same row with the top-right bin."
- "Block1 is not in the correct position according to my
knowledge but I cannot reach it. I should wait for my partner
to put it into the common bin so that I can reach it."
- "All the blocks are in the correct position except block2.
However, according to my knowledge I don't know where the
block2 should go in. I may randomly try to place it to one of
the bins in front of me."
- "I can reach Block1 and Block3 since they are in front of
me. I know they should be in the same row, but I do not know
either of their exact expected locations. I will try with
moving Block1 to the top-left bin as a start, supposing they
are both on the top row."


## General Guidelines

- Provide a clear and concise explanation for the agent's
action. No more than 2-3 sentences are needed.
- If you are not sure about the reasoning, this may because
the given action is a random guess, and it is correct by
luck. In this case, you should reason about what you know
(briefly) and don't know (important), and clarify that the
action is a guess.
- The given action may not be the best move, but you should
explain the reasoning behind it.
- You can refer to the agent as "I" or "me" in your response.
```

Listing 12: System prompt for GPT4o generating reasoning traces for *No-Information-Exchange* agents. Redundant part is omitted.

```
Player{player_id} is on the {side} side of the game board. It
has the following information for its goal:
{knowledge}
Currently, the blocks are located as follows:
{blocks}
It can access these bins:
{bins}

The history of the game till now:
{move_history}

It takes the action: {cur_move}
What is the reasoning behind this action? Please provide your
thoughts as if you are the player.
```

Listing 13: User prompt for GPT4o generating reasoning traces for agents.