

MultiAgentBench : Evaluating the Collaboration and Competition of LLM agents

Kunlun Zhu^{1† *}, Hongyi Du^{1†}, Zhaochen Hong^{1†}, Xiaocheng Yang^{1†}

Shuyi Guo^{1†}, Zhe Wang^{1 †}

Zhenhailong Wang¹, Cheng Qian¹, Xiangru Tang, Heng Ji¹, Jiaxuan You¹

¹University of Illinois Urbana-Champaign

kunlunz2@illinois.edu

Abstract

Large Language Models (LLMs) have shown remarkable capabilities as autonomous agents; yet existing benchmarks either focus on single-agent tasks or are confined to narrow domains, failing to capture the dynamics of multi-agent coordination and competition. In this paper, we introduce **MultiAgentBench**, a comprehensive benchmark designed to evaluate LLM-based multi-agent systems across diverse, interactive scenarios. Our framework measures not only task completion but also the quality of collaboration and competition using novel, milestone-based key performance indicators. Moreover, we evaluate various coordination protocols (including star, chain, tree, and graph topologies) and innovative strategies such as group discussion and cognitive planning. Notably, gpt-4o-mini reaches the average highest task score, graph structure performs the best among coordination protocols in the research scenario, and cognitive planning improves milestone achievement rates by 3%. Code and datasets are public available at <https://github.com/ulab-uiuc/MARBLE>.

1 Introduction

Large Language Models (LLMs) such as GPT-3 (Brown et al., 2020), GPT-4 (Achiam et al., 2023), Gemini (Team et al., 2023) and Deepsek-R1 (Guo et al., 2025), now exhibit human-like language understanding and generation, enabling their use as autonomous agents interacting with environments, tools, and other agents (Wang et al., 2023c; Park et al., 2023a; OpenAI, 2023). While single-agent systems have demonstrated impressive capabilities, their isolated operation can limit task execution efficiency and falls short when simulating complex, socially dynamic scenarios. In contrast, multi-agent setups (Li et al., 2023a; Wang et al., 2023b, 2024b)

leverage multiple LLM-based agents that collaborate, coordinate, and jointly plan to address these challenges. This paradigm not only increases the efficiency of task execution but also enables the realistic simulation of social interactions and dynamics, thereby enhancing performance in domains such as strategic decision-making, gaming (Silver et al., 2017), and software development (Wang et al., 2023c).



Figure 1: **Overview of MultiAgentBench evaluation process:** Multi-Agent System Coordination in various interactive environments, with a focus on task performance, and coordination.

Despite significant advances in LLM capabilities, current evaluation paradigms remain insufficient for multi-agent scenarios. Traditional single-agent benchmarks—such as AgentBench (Liu et al., 2023), VisualAgentBench (Sun et al., 2023) GAIA (Mialon et al., 2023), ToolBench (Qin et al., 2024) and HumanEval (Chen et al., 2021)—primarily focus on isolated reasoning and generation, overlooking the dynamics intrinsic to multi-agent interactions.

To address this gap, we introduce **MultiAgentBench**, a comprehensive benchmark designed to evaluate LLM-based multi-agent systems across a wide range of task-solving and simulation scenarios. MultiAgentBench offers several key advantages: (1) *Multi-Domain Evaluation*: The bench-

*Team Leader.

† Core Contributors. Contributions are listed in the appendix A.1.

mark covers diverse domains—from collaborative coding to gaming—ensuring broad real-world applicability. (2) *Capturing Coordination and Competition*: Unlike traditional single-agent benchmarks, MultiAgentBench explicitly measures both coordination dynamics and competitive interactions, highlighting the unique challenges of multi-agent environments. (3) *Tailored Metrics and Flexible Protocols*: We propose novel metrics, including a *Key Performance Indicator (KPI)* that tracks milestone progress and individual contributions, to systematically assess planning quality and communication effectiveness. Moreover, our framework, **MARBLE** (Multi-agent cooRdination Backbone with LLM Engine), supports various communication topologies—such as star, chain, tree, and an innovative graph-based approach—and accommodates multiple reasoning strategies.

Our contributions can be summarized as follows: (1) We introduce **MultiAgentBench** along with the **MARBLE** framework, a comprehensive benchmark that rigorously evaluates LLM-based multi-agent systems in six diverse interactive scenarios, capturing both collaborative and competitive dynamics. Notably, the cognitive planning planning feature improves milestone achievement rates by 3%. (2) We propose innovative evaluation metrics that assess not only task success but also coordination quality. Our metrics include milestone-based KPIs, structured planning and communication scores, and a dedicated competition score that captures conflicting-goal tasks, internal performance metrics, and competitive aspects in planning and communication. (3) Our experiments reveal some "aha-moments" in multi-agent coordination—agents begin to exhibit emergent social behaviors, providing promising insights toward AGI-level collaboration (Feng et al., 2024).

2 Related Work

2.1 LLM-Based Multi-Agent Systems

LLM-based multi-agent systems have enabled collaborative problem-solving across domains (Park et al., 2023a; Li et al., 2023b; Chen et al., 2023b). These systems support scientific research through literature review and experimental design (Zhou et al., 2024a; Agarwal et al., 2024), software engineering tasks (Huang et al., 2023; Wu et al., 2023a; Zhou et al., 2023a; Hong et al., 2024; Ishibashi and Nishimura, 2024; Islam et al., 2024; Wang et al., 2024a; Zhuge et al.) including code generation and

maintenance (Bouzenia et al., 2024), and gaming applications (Chen et al., 2023a). In Minecraft, agents perform complex tasks from construction to navigation (Wang et al., 2023a; Chen et al., 2023b; Yu et al., 2024b; Dong et al., 2024).

GameNGen enables real-time interaction in DOOM (Valevski et al., 2024), while CUISINEWORLD benchmarks multi-agent collaboration (Gong et al., 2023). Applications extend to social deduction games, game theory (Xu et al., 2023), healthcare (Ke et al., 2024; Kim et al., 2024), business (Chen et al., 2024), education (Gösling et al., 2024), and urban planning (Zhou et al., 2024b). Despite progress, challenges persist in communication, emergent behavior, and scalability (Agashe et al., 2024), motivating the need for robust evaluation frameworks.

2.2 Multi-Agent Collaboration

Recent advances in multi-agent systems highlight two complementary scaling paradigms: *cognitive scaling*, which enhances agent reasoning and adaptability, and *population scaling*, which leverages large agent collectives for emergent behaviors (Zhuge et al.; Qian et al., 2024).

Cognitive scaling explores mechanisms such as dynamic architecture adaptation and self-organizing coordination strategies to find the most effective pattern of agent communication (Zhuge et al.). Meanwhile, population-based scaling exhibits nonlinear performance gains as an increasing number of agents collectively interact through diverse collaboration patterns, including hierarchical delegation and decentralized consensus (Qian et al., 2024). These approaches enable complex applications ranging from geopolitical conflict simulation (Hua et al., 2024) to scientific discovery workflows (Zhou et al., 2024a; Zhang et al., 2025).

3 Methodology

3.1 Framework Design

Our proposed evaluation framework MARBLE (see Figure 2) establishes a robust multi-agent coordination system by leveraging interconnected modules that enable adaptive collaboration, efficient communication, and strategic task execution. At its core lies the *Coordination Engine*, responsible for initializing and synchronizing key modules—including the *Agent Graph*, *Cognitive Module*, and *Coordinate Engine*—to ensure seamless interaction across the system. Detailed descrip-

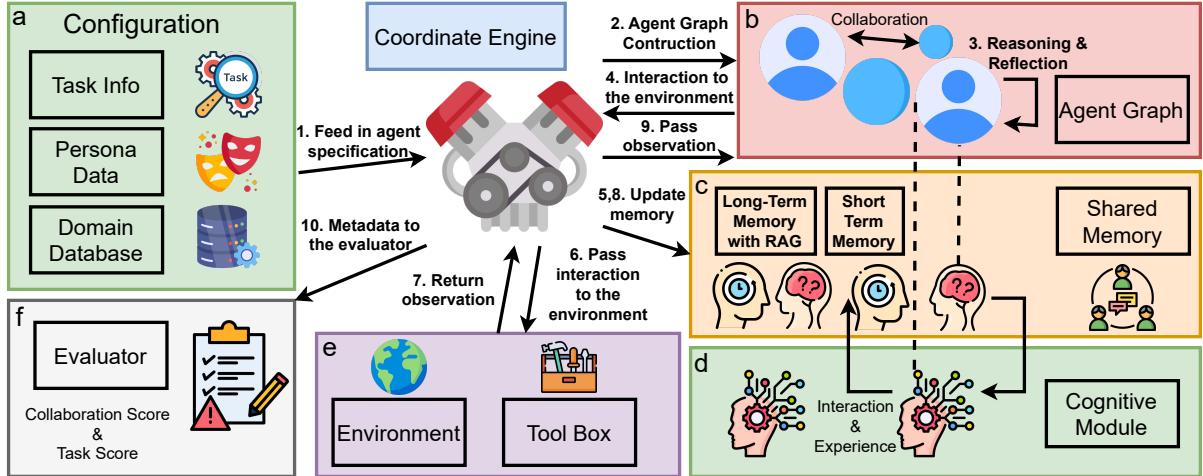


Figure 2: MARBLE 🤖: showcasing interactions between task information, persona data, domain databases, memory modules, and the environment through the coordinate engine and cognitive module.

tions of additional modules can be found in Appendix A.2.

Agent Graph Module This module converts configuration data into a structured graph $G = (\mathcal{A}, \mathcal{E})$, where $\mathcal{A} = \{a_1, a_2, \dots, a_n\}$ denotes the set of agents, and each edge in \mathcal{E} is defined as a triple (a_i, r, a_j) with $r \in \mathcal{R}$ representing the relationship between agents a_i and a_j . For example, a collaboration relationship is denoted as $(a_i, \text{collaborates}, a_j)$, supervision as $(a_i, \text{supervises}, a_j)$, and negotiation as $(a_i, \text{negotiates}, a_j)$. By constructing the graph based on these triple relations, we ensure that subsequent communication and coordination occur exclusively between agents with explicitly defined relationships, mirroring real-world interaction patterns.

Cognitive Module The Cognitive Module is central to responsible agent evolution and social intelligence in multi-agent coordination. It maintains and updates a comprehensive internal state that includes each agent’s persona, inter-agent relationships, and reasoning strategies (e.g., Chain-of-Thought (Wei et al., 2023), ReACT (Yao et al., 2023)). Crucially, this approach mirrors human cognitive processes by fusing elements of theory-of-mind and social intelligence (e.g., Premack and Woodruff, 1978)—similar to how humans continuously update their mental models based on social cues, prior experiences, and contextual information. The fusion of cognitive, social, and adaptive mechanisms forms the backbone of our system, empowering agents to dynamically adjust their strategies and

collaboratively evolve in complex environments.

3.1.1 Coordination Engine

The Coordination Engine orchestrates the overall execution flow of the system. It initializes agents, tasks, and inter-agent relationships via a dedicated Configuration Module and constructs the Agent Graph to represent these dynamics. In our framework, we distinguish between two key roles: *planners* and *actors*. Planners are responsible for devising task inputs, strategizing, and managing overall task allocation, while actors—represented within the Agent Graph—execute tasks by interacting with the environment and other agents through available tools.

Our approach supports four distinct coordination protocols similar to work from Qian et al. (2025): **star**, **tree**, **graph**, and **chain**.

Centralized Coordination: Star & Tree. In the **star** configuration, a single central planner assigns tasks to all actors and consolidates their feedback, offering strong oversight though potentially limiting scalability. The **tree** structure extends this by organizing agents hierarchically: a top-level planner delegates tasks to subordinate planners, which in turn coordinate with actors. This hierarchical approach balances centralized control with improved scalability for handling more complex tasks.

Decentralized Coordination: Graph-Mesh & Chain. The **graph-mesh** configuration employs a network of interconnected actors that communicate directly, enabling concurrent planning and distributed decision-making. Conversely, the **chain**

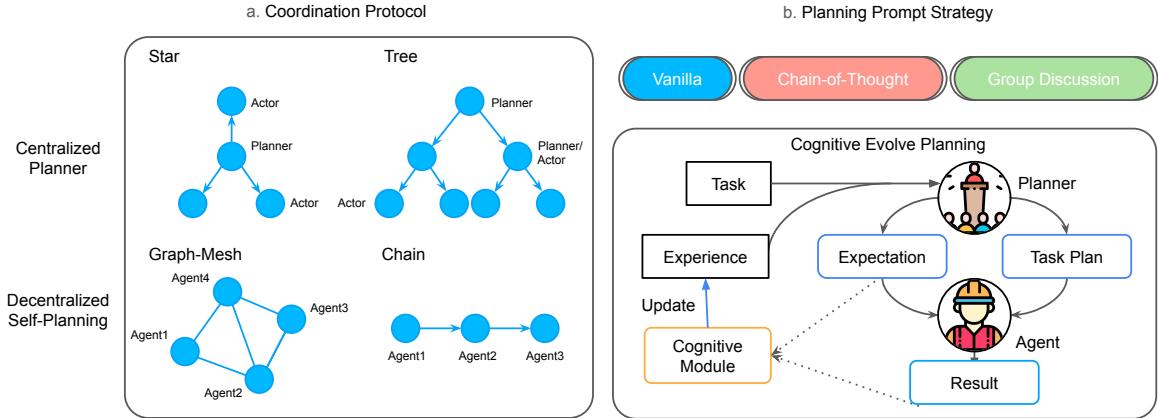


Figure 3: **Illustration of coordination protocols and planning prompt strategies.** (a) shows centralized and decentralized planning structures (e.g., star, tree, graph, and chain). (b) describes strategies like group discussions and cognitive prompts, incorporating iterative feedback and task updates for effective planning.

configuration arranges actors sequentially, where each agent passes its decision to the next. This sequential handoff is well-suited for tasks with inherent dependencies, though it may limit parallel processing capabilities.

Planner Design and Enhancements. In our centralized coordination protocol, the planner supports four distinct planning approaches that reflect human decision-making processes: vanilla prompting, chain-of-thought (CoT) (Wei et al., 2022), group discussion, and cognitive self-evolving planning. The **vanilla prompt** employs a straightforward, zero-shot instruction to generate task plans directly. The **CoT approach** enriches this process by facilitating step-by-step reasoning through detailed inputs—such as the target task, agent profiles (including roles, expertise, and historical performance), and summaries of previous subtasks—to guide logical progression. The **group discussion** (Chen et al., 2023b) method enables multiple agents to share insights and constraints, fostering a collaborative deliberation that refines the overall plan. Lastly, similar to the Reflexion (Shinn et al., 2023) method, our **cognitive self-evolving planning** method mirrors human learning by generating expected outcomes and progress for each task, storing these in memory, and then comparing actual performance against these expectations in subsequent iterations. This comparison produces evolving experiences that continuously inform and adjust future planning (See Appendix A.12 for detailed prompting). Together, these methods leverage both individual reasoning and collaborative optimization, enhancing coordination efficiency as validated through

ablation studies on the star coordination style.

3.2 Benchmark Design

To systematically evaluate our multi-agent framework, we curate a benchmark of diverse scenarios spanning *task-oriented* and *social-simulation-based* environments (Figure 1). These scenarios are constructed through a combination of: (1) *Existing multi-agent tasks* adapted from prior work or datasets (e.g., database error analysis, research collaboration). (2) *LLM-generated tasks* with human verification and refinement (e.g., Werewolf and Bargaining). This dual approach ensures both realism (by leveraging established tasks) and novelty (through generative expansion), while human validation guarantees that each scenario remains coherent and feasible.

Agents with Mutual Goal. In the task-oriented scenarios, the agents share with the mutual goal to finish one specific task. We focus on four representative tasks: (1) **research tasks** follow the setup of *ResearchTown* (Yu et al., 2024a), where agents with complementary research profiles co-author a new proposal on a chosen topic; (2) *Minecraft-based building tasks* require agents to collaboratively construct structures in a shared environment; (3) **database error analysis** involves exactly five agents, each specializing in diagnosing a distinct root cause of system inconsistencies; (4) **coding challenges** demand collective problem-solving and software module development. Across these tasks, agents must coordinate, divide labor, and synthesize outputs efficiently. We scale scenario diversity by creating 100 test cases per task, with variations

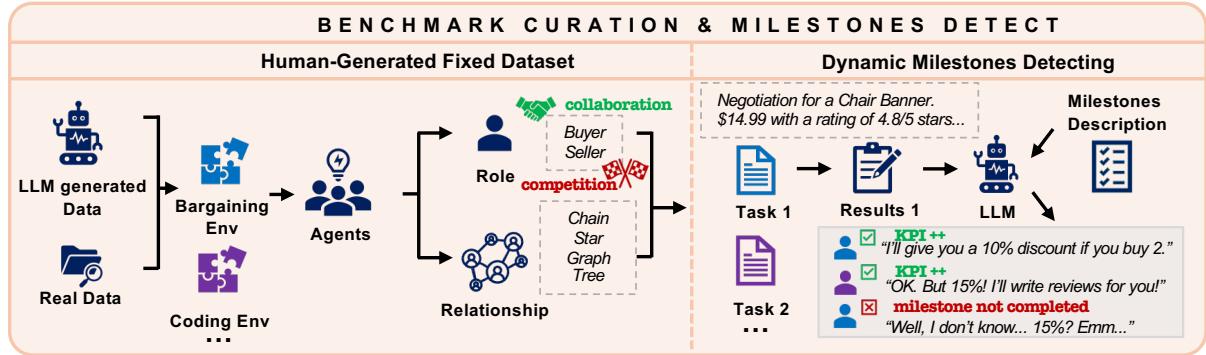


Figure 4: Illustration of our benchmark curation and the dynamic milestones detecting for KPI metric.

in research topics, Minecraft creation, database errors, or coding objectives.

Agents with Conflicting Goals. In social-simulation based scenarios. We strengthen competitive elements by introducing **Werewolf** and **Bargaining** scenarios. In Werewolf, two groups of agents face off in an adversarial setting, employing deceptive strategies within a predefined narrative. The Bargaining environment simulates negotiations over shared resources, where agents strive to maximize individual gains through strategic concessions or alliances. Both settings evaluate adaptability, conflict resolution, and negotiation skills under uncertainty.

Role Assignments and Graph Structures. To emphasize multi-agent collaboration, each scenario enforces distinct agent roles (e.g., project manager, domain expert, technical specialist) and defines specific graph relationships (star, tree, chain, or mesh). These structures reflect realistic team dynamics or competition, guiding how agents share information, make decisions, and coordinate actions.

Milestones Generation for Scenarios To facilitate the evaluation of MARBLE iterations, each task is segmented into a series of flexible milestones. Unlike rigid checkpoints, these milestones are broadly defined. For instance, in a research task, a milestone may be reached by completing five key queries (5q) for research proposal (more details see Appendix A.4) or by enhancing a previous set of 5q. Throughout MARBLE’s iterative process, a language model continuously monitors whether milestones m_1, m_2, \dots have been achieved and logs the outcomes. This method integrates human- or LLM-generated outlines with dynamic, execution-based assessments, ensuring that both intermediate progress and team coordination are effectively mea-

sured.

More detailed environment setups, interaction tools, and additional examples for different scenarios appear in Appendix A.4, A.5, A.6, A.7, A.8, and A.9.

3.3 Evaluation Metrics

As illustrated in Figure 1(b)(c), our evaluation considers two primary dimensions: **Task Completion Performance** and **Coordination**.

Task Completion Metrics. As described in Section 3.2, each task is segmented into a series of flexible milestones. An LLM-based detector continuously monitors the iterative process to identify which milestones have been achieved and records the corresponding contributing agents. For each agent, the number of milestones they contribute to is noted as n_j , and their individual KPI is calculated as the ratio of n_j to the total number of milestones M . The overall KPI is defined as the average of these individual KPIs across all N agents, which is computed as follows:

$$\text{KPI}_{\text{overall}} = \frac{1}{N} \sum_{j=1}^N \text{KPI}_j = \frac{1}{NM} \sum_{j=1}^N n_j.$$

In addition to the KPI derived from milestone detection, a separate **task-based score** is computed to evaluate the final output quality. For tasks such as research or bargaining, an LLM-defined scoring rubric is applied to generate the score, whereas tasks like Minecraft, Werewolf, database error fixes, or coding are evaluated using rule-based metrics (e.g., accuracy). Detailed scoring criteria and evaluation prompts for these task-based assessments are respectively provided in the Appendix A.9, A.5, A.6, and A.7, which demonstrate the effectiveness of the metrics while evaluating the coordination abilities.

Coordination Metrics. Coordination is evaluated by quantifying the agents’ communication and planning capabilities. The **Communication Score** (C_{score}) is derived from an LLM-based evaluation that considers inputs such as the task description, agent profiles, and aggregated communication data, resulting in a score on a five-point scale (with $C_{\text{score}} = 0$ if no communication occurs). Similarly, the **Planning Score** (P_{score}) is determined by assessing the agents’ abilities to organize tasks, maintain roles, and adapt strategies based on their profiles and aggregated planning data, also on a five-point scale. The overall **Coordination Score (CS)** is then computed by averaging these two sub-scores. More details regarding the evaluation process and the output format are provided in the Appendix A.12. We also did a human evaluation comparing human alignment with those metrics, results are in Appendix A.3.

4 Experiment Setup

4.1 Experiment Settings

Models. Since our MARBLE framework required function-calling abilities. Thus, we consider three open-source models: *Meta-Llama-3.3-70B* (Dubey et al., 2024), *Meta-Llama-3.1-70B-Instruct-Turbo* (Dubey et al., 2024), and *Meta-Llama-3.1-8B-Instruct-Turbo*, as well as two closed-source models: *GPT-3.5-turbo-0125* and *GPT-4o-mini*¹. We access the open-source models are provided by the *togetherai*² service, utilizing the default parameter settings.

For the agent actions, we configure the models with a maximum token number (`max_token_num`) of 1024, a temperature of 0.7, and a `top_p` of 1.0, in order to balance the creativity and consistency of the agents’ responses during interactions. The overall maximum iterations are set to 5 for research and 20 for Minecraft; more details can be found in the Appendix. In our evaluation, which involves both Task Completion and Simulation scenarios, we assess the models along two primary axes: *Task Score (TS)* and *Coordination Score (CS)*, using the same metrics as described in the Metrics section. The maximum communication iteration number is also set to 5. Furthermore, the long-term base memory for each agent is set to be unlimited. Finally, for the main experiment, a graph-mesh coordination protocol is adopted to facilitate interactions.

¹<https://www.openai.com>

²<https://www.together.ai>

4.2 Main Experiment One: Model Performance Across Different Scenarios

In this experiment, we evaluate the performance of five models across diverse scenarios, with results summarized in Table 1. Our analysis leads to several key insights:

1. Superior Task Performance of *gpt-4o-mini*:

Across multiple tasks, *gpt-4o-mini* consistently achieves high Task Scores (TS). For example, in the Research scenario it obtains a TS of **84.13%**, outperforming other models such as *Meta-Llama-3.1-8B* (80.87%) and *Meta-Llama-3.1-70B* (80.80%). Similar trends are observed in the Coding domain, where *gpt-4o-mini* records a TS of 65.10 compared to lower scores from its competitors. These results indicate that the underlying model capabilities are a decisive factor in achieving superior task performance.

2. The Nuanced Role of Coordination (Collaboration) Score:

While the Collaboration Score (CS) is designed to measure coordination ability, our findings suggest that its impact on the overall task performance is complex. For instance, in the Minecraft scenario, *Meta-Llama-3.1-70B* exhibits a high CS of 75.00 but an extremely low TS of 0.21, a more deep analysis for this can refer to Appendix 21. This discrepancy implies that, although coordination contributes to performance, it does not compensate for inherent deficiencies in task execution capabilities. In contrast, models that balance both high TS and moderate-to-high CS—such as *gpt-4o-mini*—demonstrate more robust and reliable performance across scenarios.

3. Model-Specific Strengths and Context-Dependent Performance:

Our results reveal that different models exhibit varied strengths depending on the task. For example, *Meta-Llama-3.3-70B* shows a notable CS in the Research (72.00) and WereWolf (76.30) tasks, yet its TS lags behind that of *gpt-4o-mini* in several scenarios. These variations emphasize that no single metric can fully capture a model’s effectiveness; instead, both task-specific abilities and coordination skills must be considered. Overall, our study underscores that while coordination plays a role, the intrinsic model capabilities are the primary drivers of success across diverse tasks.

Model	Research		Minecraft		Database		Coding		Bargaining		WereWolf	
	TS	CS										
Meta-Llama-3.1-8B	80.87	52.40	6.12	54.40	34.00	40.00	59.90	67.24	72.81	73.36	12.64	60.00
Meta-Llama-3.1-70B	80.80	49.50	0.21	75.00	53.00	37.70	62.10	67.18	72.13	71.46	19.82	60.60
Meta-Llama-3.3-70B	80.00	72.00	9.15	69.00	28.50	40.00	56.60	74.40	73.15	69.56	36.33	76.30
gpt3.5-turbo	70.20	55.90	5.05	63.60	45.00	60.89	55.50	76.20	71.67	72.00	15.69	75.90
gpt-4o-mini	84.13	52.00	33.60	61.50	45.00	43.22	65.10	66.30	74.47	74.20	14.06	60.10

Table 1: **Average Task Score (TS)** (%) and the Coordination Score (CS) for Minecraft, Database, Coding, Bargaining, and WereWolf, scores are multiplied by 20. We can see that model abilities are still the key factor for task completion. CS is a good indicator for TS given three pairs of scenarios having the one model having the highest TS and CS at the same time.

4.3 Main Experiment Two: Effects of Collaboration Protocols and Planning Strategies

We investigate the impact of different collaboration protocols—Star, Tree, Graph, and Chain—on model performance in the Research scenario.

According to Fig. 5, the graph-based protocol excels in research scenarios with the best task performance, planning efficiency, and token usage, while both the star and graph protocols yield similar task scores. In contrast, the tree-based protocol performs poorly, with high token consumption and the lowest task and coordination scores. As shown in Fig. 6, Cognitive Evolving Planning demonstrates superior coordination—significantly outperforming the alternatives—and achieves a task score comparable to the best, COT. Counterintuitively, the group discussion method scores the worst across all metrics, possibly because an overly large planning group hinders effectiveness, similar to large organizations in real-world scenarios.

5 Ablation Study

The goal of our ablation study is to identify the key modules and parameters that affect performance.

Ablation on Different Max Iteration Settings

We evaluate 10 tasks from the Minecraft scenario using six distinct maximum iteration settings. As shown in Fig. 7, both task and coordination scores increase from 1 to 7 iterations, but then drop sharply at 10 iterations. At 20 iterations, while the task score shows a recovery, the coordination score remains nearly unchanged beyond 7 iterations. This pattern suggests that, for highly challenging tasks, excessive iterations may lead to coordination degradation—possibly due to communication overhead or conflicting directives emerging

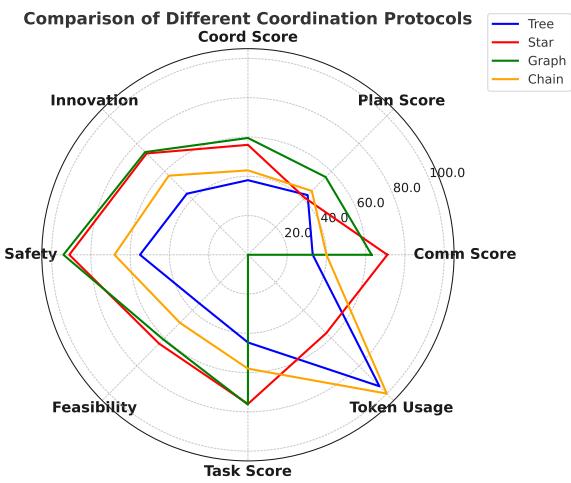


Figure 5: Comparison of Different Coordination Protocols.—Tree, Star, Graph, and Chain—across multiple evaluation metrics. Specially, the token usages are scaled such that the lowest value is 0 and the highest value is 100. Details about metrics used for research task can be found at A.4

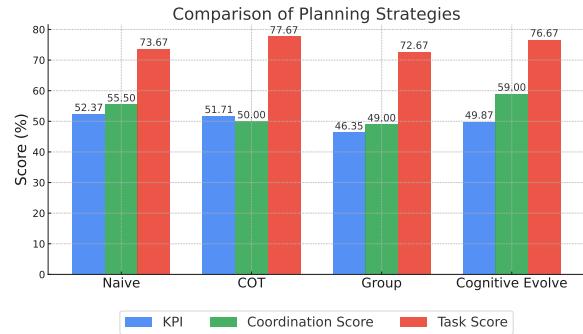


Figure 6: Average Metrics for Research Tasks for different planning prompt strategies. Cognitive Evolve Planning show best result in CS.

over prolonged interactions. These findings underscore the need for adaptive iteration strategies that balance task execution with effective coordination.

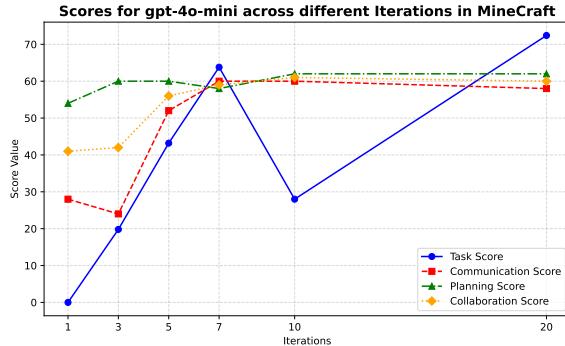


Figure 7: Scores for gpt-4o-mini across different iterations in Minecraft. The figure presents Task Score (TS), Communication Score (CS), Planning Score (PS), and Collaboration Score (CoS) over multiple iterations.

Ablation on Different Agent Numbers We assess configurations with 1, 3, 5, and 7 agents in the research scenario, selecting 20 papers that have at least 7 main authors. As illustrated in Fig. 8, increasing the number of agents leads to a decrease in the overall KPI, which aligns with the anticipated trade-off between increased collaborative complexity and performance. Notably, the average coordination score improves significantly when moving from 1 to 3 agents, while the average task score increases more gradually. This indicates that a moderate expansion in team size can enhance coordination efficiency, although further increases may introduce additional coordination challenges that counterbalance task performance gains.

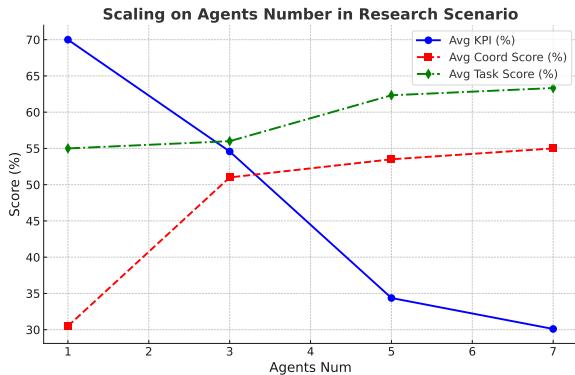


Figure 8: Scaling on Agents Number in Research Scenario. This figure shows the impact of agent number on KPI, Coordination Score, and Task Score.

6 Emergent Behaviors Analysis

In MultiagentBench, goal-driven emergent behaviors are pivotal to team coordination—we argue that these “aha-moments” not only mark when individual agents align their actions toward shared objectives but also catalyze spontaneous multiagent dynamics, unveiling novel coordination strategies and adaptive collective intelligence. (see Appendix A.5.6). Under information asymmetry and role conflicts, agents display three key patterns (refer to A.5.6 and A.5.6 for Werewolf scenarios, and Section 18 for Bargaining):

Strategic Information Sharing Agents selectively disclose key information (e.g., the Seer withholding inspection results) based on trust and context, echoing the “strategic silence” seen in human interactions (Park et al., 2023b). For instance, in A.5.6, both the Seer and Witch (gpt-4o) were overly cautious, missing optimal sharing opportunities and leading to failure. Another case is shown in Fig 26, where two research agents strategically evolve the research proposal details, combining the strengths of both background knowledge.

Trust-Polarized Collaboration Role identities drive collaboration splits: over-suspicious villagers may turn against their own, while werewolves can create a “false consensus” through deception (Fehr and Gächter, 2000) and internal execution (Woolley et al., 2010). As shown in A.5.6, villagers’ excessive caution allowed werewolves to exploit confusion; similar internal friction is evident in Bargaining (18).

Role-Driven Strategy Iteration Throughout the game, roles such as the Seer and Witch adjust their strategies. The Seer, for example, shifts from a conservative to a leadership role (see A.5.6), while the Witch moves from hoarding to taking risks. These shifts support the notion that task objectives drive decision-making, aligning with the AutoGen framework findings (Wu et al., 2023b).

Quantitative Analysis of Emergent Behaviors (Werewolf Scenario) To quantify goal-driven emergent behaviors in the Werewolf scenario, we employ LLM as a judge to analyze each recorded game transcript. Specifically, for every session in our dataset, we segment the multi-turn dialogue and agent actions, then prompt the LLM to detect three target behaviors: (i) *Strategic Information Sharing*,

(ii) *Trust-Polarized Collaboration*, and (iii) *Role-Driven Strategy Iteration*. Our script tallies these occurrences across all transcripts, aggregating them per experimental configuration.

Table 2 presents a summary of these counts. “Total” denotes the total number of sessions in each file, while the last three columns respectively show how many sessions exhibited each emergent behavior. We tested five different model settings, ranging from GPT-based to LLaMA-based models.

Table 2: Summary of Emergent Behaviors in Werewolf Scenarios

Model	Total	Strat. Info.	Trust Collab.	Role Iter.
gpt-4o-mini	100	97	100	42
gpt-3.5	100	100	100	66
llama-3.1-8b	100	96	98	38
llama-3.1-70b	100	100	100	55
llama-3.3-70b	100	100	100	89
Grand Total	500	493	498	290

As shown in Table 2, *Strategic Information Sharing* and *Trust-Polarized Collaboration* arise consistently—reflecting how manipulations of trust and carefully timed information disclosure are central to Werewolf gameplay. *Role-Driven Strategy Iteration* occurs less frequently, suggesting that while specialized roles (e.g., Seer, Witch, Werewolf) can evolve their behavior, such adaptive dynamics may require longer or more intricate rounds to manifest fully. Overall, these results align with the social deduction nature of Werewolf: hidden-role uncertainty naturally fosters high levels of strategic communication and polarizing trust dynamics, whereas deeper shifts in role-based tactics may only emerge given sufficient interaction rounds.

Quantitative Analysis of Emergent Behaviors (Bargaining Scenario) We also examine emergent behaviors in a Bargaining scenario, where a *buyer* and *seller* negotiate terms. Again, we use the same LLM-based detection method to identify the same three types of behaviors. Table 3 aggregates results for five model configurations on the buyer side and five on the seller side, each with 100 sessions.

In these Bargaining sessions, *Strategic Information Sharing* also dominates, as negotiation naturally incentivizes the buyer and seller to withhold or reveal private information (e.g., target prices, product details) at opportunistic moments. However, *Trust-Polarized Collaboration* appears much less often, which is consistent with the primarily

Table 3: Summary of Emergent Behaviors in Bargaining Scenarios (Buyer vs. Seller).

File	Total	Strat. Info.	Trust Collab.	Role Iter.
<i>Buyer Side</i>				
buyer-gpt-3.5-turbo	100	81	1	81
buyer-gpt-4o-mini	100	88	0	88
buyer-Llama-3.1-70B	100	81	2	80
buyer-Llama-3.1-8B	100	92	0	87
buyer-Llama-3.3-70B	100	84	1	82
Buyer Total	500	426	4	418
<i>Seller Side</i>				
seller-gpt-3.5-turbo	100	88	0	87
seller-gpt-4o-mini	100	95	0	95
seller-Llama-3.1-70B	100	96	0	97
seller-Llama-3.1-8B	100	97	0	96
seller-Llama-3.3-70B	100	90	1	88
Seller Total	500	466	1	463
Grand Total	1000	892	5	881

bilateral structure—there is less impetus for large-scale alliance-building or deception across multiple parties. *Role-Driven Strategy Iteration* remains frequent: both buyer and seller can adapt their negotiation tactics (e.g., starting from extreme offers and gradually conceding) in a manner driven by their “role.” Such interplay of strategic disclosure, occasional (albeit limited) trust dynamics, and ongoing role-based adjustments typifies two-party bargaining interactions. Overall, the pattern of emergent behaviors here aligns well with the fundamental buyer-seller tension, wherein each agent seeks to maximize its advantage via selective sharing, repeated iteration, and a careful balance of cooperation and self-interest.

7 Conclusion

In this work, we introduce MultiAgentBench and the MARBLE framework, providing a comprehensive benchmark for evaluating LLM-based multi-agent systems across diverse interactive scenarios. Our proposed evaluation metrics go beyond task success, capturing coordination quality through structured planning, communication scores, and competition-driven assessments. Experimental results highlight key emergent social behaviors, offering valuable insights into future multiagent work.

8 Limitations

While our proposed multi-agent benchmark and framework provide a diverse range of tasks and evaluation metrics, several areas warrant further exploration to enhance their applicability and ro-

business:

Expanding Scenario and Model Coverage. Currently, our benchmark focuses on specific domains such as research co-authoring, Minecraft building, database error analysis, coding collaboration, and select competitive scenarios (e.g., Werewolf and bargaining). To better capture the complexity of real-world multi-agent interactions, future work can incorporate more diverse settings, including open-world environments, scenarios requiring richer social cognition, and tasks on the application side like task-oriented dialogues. In the aspect of models, our work does not cover the full spectrum. Future work may include the result of other latest ones (e.g. the DeepSeek model family).

Enhancing Ablation Studies. Our current analysis focuses primarily on overall coordination and competition performance, leaving finer-grained insights into specific components underexplored. Future experiments can be focused on different memory mechanisms (e.g. long-term memory, short-term memory, shared memory) and multiagent different workflow method.

Advancing Competition Mechanisms. While our benchmark incorporates competitive tasks, it does not fully capture the complexity of real-world multi-agent interactions involving multi-party negotiations, repeated strategic play, or stochastic elements. Investigating how agents transition between cooperative and adversarial roles in evolving environments remains a promising direction.

Handling Open-Ended and Ill-Defined Tasks. Most tasks in our framework involve well-defined objectives, such as completing a research proposal or resolving database inconsistencies. However, real-world applications often require agents to operate in open-ended or ambiguous contexts without clear success criteria. Future extensions could explore how multi-agent systems adapt to exploratory, non-goal-oriented scenarios.

Acknowledgment

This research is partly based upon work supported by DARPA ITM Program No. FA8650-23-C-7316, the Molecule Maker Lab Institute: an AI research institute program supported by NSF under award No. 2019897, and the AI Research Institutes program by National Science Foundation and the Institute of Education Sciences, U.S. Department of Ed-

ucation through Award # 2229873 - AI Institute for Transforming Education for Children with Speech and Language Processing Challenges. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. We sincerely appreciate the support from Amazon grant funding project 120359, “GRAG: Enhance RAG Applications with Graph-structured Knowledge” and Meta gift funding project “PERM: Toward Parameter Efficient Foundation Models for Recommenders”.

References

- Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Shubham Agarwal, Issam H. Laradji, Laurent Charlin, and Christopher Pal. 2024. [Litllm: A toolkit for scientific literature review](#).
- Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. 2024. [Llm-coordination: Evaluating and analyzing multi-agent coordination abilities in large language models](#).
- Asaniczka. 2023. [Amazon products dataset 2023 \(1.4m products\)](#).
- Islem Bouzenia, Premkumar Devanbu, and Michael Pradel. 2024. Repairagent: An autonomous, llm-based agent for program repair. *arXiv preprint arXiv:2403.17134*.
- Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.
- Dake Chen, Hanbin Wang, Yunhao Huo, Yuzhao Li, and Haoyang Zhang. 2023a. Gamegpt: Multi-agent collaborative framework for game development. *arXiv preprint arXiv:2310.08067*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023b. [Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors](#).
- Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024. [Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors](#). In *The Twelfth International Conference on Learning Representations*.
- Yubo Dong, Xukun Zhu, Zhengzhe Pan, Linchao Zhu, and Yi Yang. 2024. [Villageragent: A graph-based multi-agent framework for coordinating complex task dependencies in minecraft](#).
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. [The llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Ernst Fehr and Simon Gächter. 2000. Cooperation and punishment in public goods experiments. *American Economic Review*, 90(4):980–994.
- Tao Feng, Chuanyang Jin, Jingyu Liu, Kunlun Zhu, Haoqin Tu, Zirui Cheng, Guanyu Lin, and Jiaxuan You. 2024. [How far are we from AGI: Are LLMs all we need?](#) *Transactions on Machine Learning Research*. Survey Certification.
- Ran Gong, Qiuyuan Huang, Xiaojian Ma, Hoi Vo, Zane Durante, Yusuke Noda, Zilong Zheng, Song-Chun Zhu, Demetri Terzopoulos, Li Fei-Fei, and Jianfeng Gao. 2023. [Mindagent: Emergent gaming interaction](#). *arXiv preprint arXiv:2309.09971*.
- Henning Gösling, Jacob Dudek, Thorsten Krause, and Oliver Thomas. 2024. Multi-agent-based peer tutoring in virtual learning environments.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *arXiv preprint arXiv:2501.12948*.
- 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*.
- Wenyue Hua, Lizhou Fan, Lingyao Li, Kai Mei, Jianchao Ji, Yingqiang Ge, Libby Hemphill, and Yongfeng Zhang. 2024. [War and peace \(waragent\): Large language model-based multi-agent simulation of world wars](#).
- Dong Huang, Qingwen Bu, Jie M Zhang, Michael Luck, and Heming Cui. 2023. [Agentcoder: Multi-agent-based code generation with iterative testing and optimisation](#). *arXiv preprint arXiv:2312.13010*.
- Yoichi Ishibashi and Yoshimasa Nishimura. 2024. [Self-organized agents: A llm multi-agent framework toward ultra large-scale code generation and optimization](#). *arXiv preprint arXiv:2404.02183*.
- Md Ashraful Islam, Mohammed Eunus Ali, and Md Rizwan Parvez. 2024. [Mapcoder: Multi-agent code generation for competitive problem solving](#). *arXiv preprint arXiv:2405.11403*.
- Yu He Ke, Rui Yang, Sui An Lie, Taylor Xin Yi Lim, Hairil Rizal Abdullah, Daniel Shu Wei Ting, and Nan Liu. 2024. [Enhancing diagnostic accuracy through multi-agent conversations: Using large language models to mitigate cognitive bias](#). *arXiv preprint arXiv:2401.14589*.
- Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Cynthia Breazeal, and Hae Won Park. 2024. [Adaptive collaboration strategy for llms in medical decision making](#). *arXiv preprint arXiv:2404.15155*.
- Gen Li, Shizhe Chen, Yinan Ge, Di Jin, and Zhiyuan Liu. 2023a. [Chatdev: Generating software system with chatgpt](#). *arXiv preprint arXiv:2307.04549*.
- Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitri Khizbulin, and Bernard Ghanem. 2023b. [Camel: Communicative agents for "mind" exploration of large language model society](#).
- Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. [Agentbench: Evaluating llms as agents](#). *arXiv preprint arXiv:2308.03688*.
- Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. [Gaia: a benchmark for general ai assistants](#). *arXiv preprint arXiv:2311.12983*.
- OpenAI. 2023. [Openai function calling documentation](https://platform.openai.com/docs/guides/gpt/function-calling). <https://platform.openai.com/docs/guides/gpt/function-calling>.
- Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023a. [Generative agents: Interactive simulacra of human behavior](#). *arXiv preprint arXiv:2304.03442*.
- S. Park, J. Kim, and D. Lee. 2023b. Strategic silence in multi-agent social interaction: A social deduction perspective. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence*, pages 123–131.
- David Premack and Guy Woodruff. 1978. [Does the chimpanzee have a theory of mind?](#) *Behavioral and Brain Sciences*, 1(4):515–526.

- Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2024. [Scaling large-language-model-based multi-agent collaboration](#).
- 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*.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. 2024. [ToolLM: Facilitating large language models to master 16000+ real-world APIs](#). In *The Twelfth International Conference on Learning Representations*.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. [Reflexion: language agents with verbal reinforcement learning](#). In *Thirty-seventh Conference on Neural Information Processing Systems*.
- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.
- Xu Sun, Xiaoya Zhang, Yicheng Feng, Shiyang Wang, Shuming Ma, Jiuding He, Zhixu Zhang, Yuxian Gu, Yi Xu, Hao Zhou, and Zhiyuan Liu. 2023. A systematic capability evaluation of large vision-language models. *arXiv preprint arXiv:2305.16372*.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.
- Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. 2024. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*. Equal contribution. Work done while at Google Research.
- Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. [Voyager: An open-ended embodied agent with large language models](#).
- Sheng Wang, Emily Dinan, Jack Urbanek, Arthur Zhang, Douwe Kiela, and Jason Weston. 2023b. Role-playing as a platform for dialogue modeling, empathy, and data collection. *arXiv preprint arXiv:2301.09663*.
- Xiao Wang, Shixiang Shane Gu, Yizhu Liu, Harrison Jesse, and Pieter Abbeel. 2023c. Communicative agents for software development. *arXiv preprint arXiv:2307.09250*.
- Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. 2024a. [OpenHands: An Open Platform for AI Software Developers as Generalist Agents](#).
- Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. 2024b. Unleashing cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In *Proc. 2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL2024)*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. [Chain-of-thought prompting elicits reasoning in large language models](#).
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Anita W. Woolley, Christopher F. Chabris, Alex Pentland, Nada Hashmi, and Thomas W. Malone. 2010. Evidence for a collective intelligence factor in the performance of human groups. *Science*, 330(6004):686–688.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023a. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*.
- S. Wu, A. Holtzman, J. Buys, R. Koncel-Kedziorski, and Y. Choi. 2023b. Autogen: A framework for multi-agent collaborative decision-making with large language models. *arXiv preprint arXiv:2301.XXXX*.
- Lin Xu, Zhiyuan Hu, Daquan Zhou, Hongyu Ren, Zhen Dong, Kurt Keutzer, See-Kiong Ng, and Jiashi Feng. 2023. Magic: Investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models.
- Haofei Yu, Zhaochen Hong, Zirui Cheng, Kunlun Zhu, Keyang Xuan, Jinwei Yao, Tao Feng, and Jiaxuan You. 2024a. [Researchtown: Simulator of human research community](#).

Xianhao Yu, Jiaqi Fu, Renjia Deng, and Wenjuan Han.
2024b. *Mineland: Simulating large-scale multi-agent interactions with limited multimodal senses and physical needs.*

Lingyu Zhang, Zhengran Ji, and Boyuan Chen. 2025.
Crew: Facilitating human-ai teaming research.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Long Li, Jialong Wu, Tiannan Wang, Shi Qiu, Jintian Zhang, Jing Chen, Ruipu Wu, Shuai Wang, Shiding Zhu, Jiyu Chen, Wentao Zhang, Xiangru Tang, Ningyu Zhang, Huajun Chen, Peng Cui, and Mrinmaya Sachan.
2023a. *Agents: An open-source framework for autonomous language agents.*

Xuanhe Zhou, Guoliang Li, Zhaoyan Sun, Zhiyuan Liu, Weize Chen, Jianming Wu, Jiesi Liu, Ruohang Feng, and Guoyang Zeng. 2023b. *D-bot: Database diagnosis system using large language models.*

Yangqiaoyu Zhou, Haokun Liu, Tejes Srivastava, Hongyuan Mei, and Chenhao Tan. 2024a. Hypothesis generation with large language models. *arXiv preprint arXiv:2404.04326.*

Zhilun Zhou, Yuming Lin, Depeng Jin, and Yong Li.
2024b. *Large language model for participatory urban planning.*

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbulin, and Jürgen Schmidhuber. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

A Appendix

A.1 Contributions

Kunlun Zhu Team Lead, Code implementation of the main codebase basic design, research environment, coordinate engine, evaluator basic, main paper writing.

Hongyi Du Main contributor, code implementation of the milestone generation, werewolf framework design(including environment, communication, evaluator and memory module), data analysis, generation, writer of emergent behavior, limitations, related work in main paper and human evaluation, werewolf, important prompts, bad communication cases in appendix.

Zhaochen Hong Main contributor, code implementation of environment basics, communication module, database environment, paper writing of in the appendix Database and related work.

Xiaochan Yang Main contributor, code implementation of the Memory module, Minecraft environment, paper writing in the appendix Minecraft and related work.

Shuyi Guo Main contributor, Code implementation of the evaluator prompt, bargaining environment, paper writing in the appendix bargaining and related work.

Zhe Wang Main contributor, code implementation of the reasoning agent module, coding environment, paper writing in the appendix coding and related work.

Those core contributors are considered as the first authorship.

A.2 More Details on Multi-agent framework design

Configuration Module Initializes and parameterizes the system by ingesting task specifications, persona data, agent profiles, role definitions, and domain-specific databases. It constructs agent attributes $(A_i, P_i)_{i=1}^N$, where A_i is the i -th agent and P_i its profile encompassing capabilities, constraints, and personality traits. Additionally, it defines inter-agent relationships such as hierarchical roles, collaboration links, or adversarial ties, producing a global state for coordination patterns.

Environment Module Simulates the scenario in which agents operate, supporting diverse contexts like coding challenges, research projects, or negotiation games. Agents interact with the environment via a function-calling interface, selecting actions $a_t \in \mathcal{F} = \{f_1, f_2, \dots\}$ at each time step t . The environment updates its state based on actions:

$$a_t = \pi(A_{t-1}, M_{\text{shared}}, M_{\text{individual}}^i)$$
$$o_{t+1} = \text{Env}(a_t),$$

facilitating continuous agent-environment interaction. A dedicated **Tool Box** provides domain-specific functionalities such as code editors and search engines.

Memory Module Stores and retrieves information through shared and individual memories:

$$M = \{M_{\text{shared}}, M_{\text{individual}}^i : i = 1, \dots, N\}.$$

M_{shared} holds global knowledge and collective decisions, while each $M_{\text{individual}}^i$ maintains personal experiences and local observations. Individual memory is split into long-term and short-term segments, managed by a FIFO mechanism to maintain short-term thresholds. A retrieval-augmented generation (RAG) technique enables dynamic knowledge access, optimizing prompt construction.

Communication Module The Communication Module manages external interactions among agents. It equips each agent with a suite of communication tools and maintains detailed profiles of other agents, thereby facilitating context-aware exchanges. By supporting structured dialogue and information sharing, this module enables agents to negotiate roles, coordinate plans, and balance collaborative efforts with competitive interactions.

Action Module The Action Module executes the plans generated by agents and leverages both function-calling mechanisms and structured output formats to obtain final results. As agents perform actions, outcomes and observations are immediately fed back into both individual and shared memory stores. This iterative loop enables dynamic adaptation to evolving task requirements and further refines agent strategies over time.

A.3 Human Evaluation

To verify the effectiveness of our prompt-based evaluation, we conduct a human evaluation in a

Werewolf environment scenario. Specifically, we calculate Kendall’s, Pearson’s, and Spearman’s correlation coefficients (along with the corresponding p-values) to demonstrate that the prompt-based scores align well with human judgments (see Appendix for details).

We recruit six annotators familiar with NLP research. Each annotator uses the same instructions and sees the same inputs as the LLMs when rating the outputs for both the planning and communication dimensions. Every task is evaluated by two annotators, and we take the average of their scores. In total, we have 60 tasks across five different LLMs, all set within the Werewolf environment.

Table 4 summarizes the comparison between the human evaluation scores and our prompt-based machine scores in this Werewolf environment.

Model	Comm (Human)	Plan (Human)	Comm (Machine)	Plan (Machine)
llama31_70b	3.19	3.19	3.12	3.00
llama33	3.94	3.44	3.89	3.89
gpt4o-mini	3.61	3.33	3.00	3.00
gpt3.5-turbo	3.75	3.44	4.00	3.75
llama31_8b	2.62	3.06	3.00	3.00

Table 4: Comparison of human vs. machine evaluation scores in a Werewolf scenario.

Analysis. As shown in Table 4, the human-assigned scores (Comm (Human) and Plan (Human)) are generally close to the corresponding machine scores (Comm (Machine) and Plan (Machine)) across all five models. For instance, the largest difference in communication scores among these models is within 0.38 (e.g., gpt3.5-turbo achieves 3.75 in human evaluation vs. 4.00 in machine evaluation), while most other discrepancies remain even smaller. Such alignment indicates that our prompt-based evaluation method can reliably capture similar aspects of coordination and planning quality as perceived by human annotators, further validating the effectiveness of the proposed approach in assessing collaboration performance in the Werewolf environment.

To validate our proposed coordination metrics—**C-score** (communication) and **P-score** (planning)—we conducted a large-scale human evaluation study involving 360 annotated samples. Results from this study, alongside with previous results, reveal strong correlations between LLM-generated scores and human judgments.

These results in Table 6, 5, 7 demonstrate particularly strong alignment in Communication, validat-

ing the reliability of LLM-generated judgments in evaluating agent interaction.

A.4 Research Scenario

Task Overview

This research scenario task focuses on leveraging multiagent collaboration to generate innovative research ideas. Each agent, equipped with a specialized research profile, contributes unique expertise to address complex research challenges. Agents collaborate in a fully connected graph mode, where every agent has a collaborative relationship with others, fostering a robust exchange of knowledge. The ultimate goal is to formulate a novel research idea following the structured 5-question (5q) format to ensure clarity, relevance, and feasibility.

Environment Description

The research environment provides tools to facilitate collaboration, literature exploration, and research ideation. These include:

- **Research Tools:** Functions to fetch related papers, recent papers, publications, and co-author networks. The primary tools implemented in the environment include:
 - **get_related_papers:** Fetches related research papers based on query parameters, including keywords, authors, and domains.
 - **get_recent_papers:** Retrieves recent publications in specified research domains.
 - **collect_publications_and_coauthors:** Gathers an author’s publications and their co-author network for enhanced context.
 - **get_paper_by_keyword:** Locates papers based on specific keywords with adjustable result limits.
 - **get_paper_by_arxiv_id:** Fetches a paper using its arXiv ID.
 - **get_paper_by_title:** Retrieves a paper based on its title.
 - **fetch_webpage:** Extracts webpage content to gather supplementary data.

Benchmark Curation Details

The dataset consists of 100 curated ML/AI papers, sourced from published articles and preprints. Each paper’s introduction is extracted, and the authors’ profiles are generated based on their historical research themes and publications, creating a

Metric	Mean (M.)	Std(M.)	Range (M.)	Mean (H.)	Std(H.)	Range (H.)
Communication	2.17	2.35	[-1.0, 5.0]	1.69	2.19	[-1.0, 5.0]
Planning	3.55	0.61	[2.0, 5.0]	3.41	0.73	[1.0, 5.0]

Table 5: **Summary statistics of coordination scores.** The score of -1 in communication indicates runs with no communication. M. stands Machine and H. stands for Human.

Metric	Pearson’s r	p-value	Spearman’s ρ	p-value
Communication	0.8057	0.0000	0.7467	0.0000
Planning	0.2685	0.0000	0.3601	0.0000

Table 6: **Correlation between human and model-generated coordination metrics.**

Correlation Type	Average Coefficient	Average p-value
Pearson	0.3290	0.0004
Spearman	0.3641	0.1923

Table 7: **Average inter-annotator correlation across metrics.** Results indicate reasonable human agreement, especially in Communication.

comprehensive view of their expertise and contributions. Relationships among authors are standardized as collaborative, reflecting realistic academic interactions. This curated dataset forms the foundational knowledge base for multiagent discussions and ideation.

We select 33 easy tasks, 34 medium tasks, and 33 hard tasks from the researchtown dataset.

Dataset Statistic

The curated dataset contains 100 papers across machine learning and artificial intelligence domains from the ResearchTown(Yu et al., 2024a) project. These papers support generating research profiles and simulate realistic collaborative relationships among agents. The default relation setup ensures a fully connected collaboration graph, enabling seamless agent interaction.

Task Completion Metrics The agents are evaluated based on their ability to generate a high-quality research idea adhering to the 5q format. The evaluation includes:

- **Innovation:** Novelty and contribution to the field
- **Safety:** Ethical and societal considerations
- **Feasibility:** Practicality of implementation with current technology

5q Format Details

The structured output for the task includes:

- **[Question 1] - What is the problem?** Define the specific research problem.
- **[Question 2] - Why is it interesting and important?** Highlight the broader implications and potential impacts.
- **[Question 3] - Why is it hard?** Elaborate on challenges and complexities.
- **[Question 4] - Why hasn’t it been solved before?** Identify gaps and limitations in prior work.
- **[Question 5] - What are the key components of my approach and results?** Outline the methodology, dataset, and expected outcomes.

Agent Profile Case: See Fig 10.

Generated Research Idea Case: See Figure 11

Evaluation Framework

The final research idea is evaluated based on a structured prompt, focusing on innovation, safety, and feasibility. Ratings are provided on a 5-point scale. For more details, see the Fig 25, with detailed feedback on each aspect. This framework ensures that the generated ideas meet high standards of academic and practical relevance.

****Task:**** Dear Research Team,

You are collaborating to generate a new research idea based on the following Introduction:

****Introduction**** The Language of Thought Hypothesis suggests that human cognition operates on a structured, language-like system of mental representations. While neural language models can naturally benefit from the compositional structure inherently and explicitly expressed in language data, learning such representations from non-linguistic general observations, like images, remains a challenge. In this work, we introduce the Neural Language of Thought Model (NLoTM), a novel approach for unsupervised learning of LoTH-inspired representation and generation. NLoTM comprises two key components: (1) the Semantic Vector-Quantized Variational Autoencoder, which learns hierarchical, composable discrete representations aligned with objects and their properties, and (2) the Autoregressive LoT Prior, an autoregressive transformer that learns to generate semantic concept tokens compositionally, capturing the underlying data distribution. We evaluate NLoTM on several 2D and 3D image datasets, demonstrating superior performance in downstream tasks, out-of-distribution generalization, and image generation quality compared to patch-based VQ-VAE and continuous object-centric representations.

Your task is to produce a research idea using the 5q format.

Figure 9: Task Content Case.

****Agent Profile:**** I am a researcher dedicated to advancing the fields of image processing and machine learning, with a particular focus on generative models and deep learning architectures. My recent work has explored innovative solutions to challenges in image inpainting, where I developed frameworks that effectively separate the tasks of image reconstruction and artifact removal, significantly improving both visual quality and quantitative metrics.

I have also contributed to the evolution of Generative Adversarial Networks (GANs) through the introduction of the DeshuffleGAN, which enhances the learning of spatial structures in images via self-supervised tasks. This work has demonstrated substantial improvements in image generation quality across various datasets. My research extends to depth estimation, where I reformulated the problem as a ranking task, leveraging existing literature to enhance performance.

In addition, I have explored the integration of probabilistic models in federated learning, addressing the critical need for uncertainty quantification in safety-critical applications. My work on multi-label ranking has introduced novel methods that utilize the inherent ranking information of labels, achieving state-of-the-art results in both synthetic and real-world datasets.

Overall, my research aims to bridge the gap between theoretical advancements and practical applications, providing robust solutions that enhance the capabilities of machine learning systems in real-world scenarios. I am passionate about leveraging AI to solve complex problems and contribute to the ongoing evolution of intelligent systems.

Figure 10: Agent Profile Case.

[Question 1] - What is the problem?

How can we enhance malware detection and classification in network traffic using transformer-based models that effectively generalize to unseen attack types with minimal labeled data?

[Question 2] - Why is it interesting and important?

Solving this problem is crucial as cyber threats continue to evolve, becoming more sophisticated and difficult to detect with traditional methods. By improving malware detection through advanced machine learning techniques, we can significantly enhance network security, leading to better protection against emerging threats. This research could pave the way for future studies on self-supervised and few-shot learning approaches in cybersecurity, potentially leading to practical applications in real-time threat detection and response systems.

[Question 3] - Why is it hard?

The complexity arises from the need to accurately identify and classify novel malware types that have not been previously encountered, which requires models to generalize well from limited labeled data. Naive approaches may fail due to the high variability in malware behavior and the constant evolution of attack strategies. Additionally, the challenge of balancing the dataset to avoid bias towards benign or malicious samples complicates the training process, necessitating sophisticated techniques to ensure robust performance across diverse scenarios.

[Question 4] - Why hasn't it been solved before?

Previous research has often relied on supervised learning methods that require extensive labeled datasets, which are not always available, especially for new malware types. Additionally, many existing solutions struggle with generalization to unseen attacks due to their reliance on specific features or signatures. Our approach leverages self-supervised learning and few-shot learning, which allows for better adaptation to new threats by learning from unlabeled data and minimal examples, thus addressing the limitations of prior methodologies.

[Question 5] - What are the key components of my approach and results?

Our proposed methodology involves using a transformer-based model trained on a combination of labeled and unlabeled datasets, specifically focusing on payload data from network traffic. We will utilize the UNSW-NB15 and CIC-IoT23 datasets for evaluation, employing metrics such as accuracy and F1-score to assess performance. The expected outcomes include improved detection rates for novel malware types and enhanced generalization capabilities, demonstrating the effectiveness of our approach in real-world scenarios.

Figure 11: 5Q cases.

A.5 Werewolf Environment

A.5.1 Environment Description (Tool Description)

The Werewolf environment, inspired by the classic social deduction game *Werewolf* (a.k.a. Mafia), provides a rich, socially complex setting in which players (agents) belong to opposing factions with asymmetric information and objectives. This scenario is particularly suitable for evaluating LLM-driven multi-agent coordination under uncertainty, as it involves hidden roles, deception, collective inference, and iterative decision-making. It challenges agents' logical reasoning as well as their aptitude for persuasion, alliance formation, adaptive responses to changing conditions, and balancing between individual interests and group goals.

Why Werewolf? In this environment, agents are divided into two main factions: the *Villager* group (including special roles such as Seer, Witch, and Guard) and the *Werewolf* group. Villagers seek to identify and eliminate all werewolves, while werewolves aim to blend in and secretly eliminate villagers. The day/night cycle establishes a repetitive structure of public discussions, secret actions, and voting decisions. This setup offers several advantages:

- **Role Asymmetry and Hidden Information:**

Villagers lack complete knowledge, while werewolves know their allies. This information asymmetry encourages strategic reasoning, suspicion, and bluffing.

- **Complex Social Reasoning:** Success hinges on persuasion, alliance building, and careful information management. Agents must form and break trusts, share or withhold information, and achieve consensus on who should be removed.

- **Adaptation and Memory:** As the game progresses through multiple cycles, agents must update their beliefs based on observed behaviors. Long-term memory supports tracking agent states, past actions, and evolving contexts.

- **Evaluation of Cooperative Dynamics:** Well-defined scoring rules for correct identifications, effective protections, and consensus-building enable objective assessment of strategic teamwork and collaborative problem-solving.

A.5.2 Villager-Centric Scoring Rationale

In this environment, we primarily focus on evaluating the villager faction rather than the werewolf faction. The core reason is that villagers rely heavily on explicit cooperative actions to secure victory: for instance, the Guard must accurately protect key roles, the Witch must judiciously use antidote and poison, and the Seer must disclose or share critical information, either publicly or privately, to identify suspects. These actions inherently demand communication and coordination among villager members, grounded in a degree of mutual trust and collaborative strategy. Without such synergy, villagers are typically overrun by the werewolves.

Moreover, the number and quality of these cooperative efforts correlate with the villagers' overall chance of success. More effective teamwork enables stronger reasoning, better defense, and a higher likelihood of identifying and eliminating werewolves or safeguarding vital roles. By observing and measuring these cooperative maneuvers—such as successful protection by the Guard, timely use of antidotes, or coordinated voting—we gain deeper insights into how the model performs in social reasoning and collaboration within a complex environment.

In contrast, the success of werewolves does not hinge as strongly on explicit teamwork. Even if they operate mostly on an individual basis and refrain from overt collaboration, werewolves can still achieve a relatively high chance of winning through misdirection and exploiting confusion among villagers. Consequently, measuring werewolf-side cooperation does not provide as discriminative or illuminating an assessment of collaborative potential as evaluating the villagers' side.

Therefore, we concentrate on the villager perspective to better capture and evaluate the synergy required in a highly uncertain, adversarial setting. This design choice highlights how cooperation, or the lack thereof, strongly influences the villagers' outcome, offering a direct lens through which to assess the social and strategic capabilities of large language model agents.

Consistent Werewolf Model. In all experiments where we vary the villager-side language model, the werewolf side remains consistently powered by GPT-4o. This ensures a stable, challenging adversary and allows us to fairly compare different villager models under identical opposing conditions.

A.5.3 Benchmark Curation Details

Initialization. Unlike other environments that rely on numerous parameterized tasks, the Werewolf game commences from a single, stable initial configuration. We tried multiple role distributions and settled on a balanced default setup to maintain fairness and avoid biasing the game toward any faction. Subsequent variations arise naturally from agent interactions, rather than from altering initial conditions. Agents are assigned roles such as *wolf*, *villager*, *seer*, *witch*, and *guard*, each with corresponding capabilities. For example, werewolves coordinate attacks at night, and the seer checks a player's identity.

Event Bus and Action Processing. This environment adheres to a strict, environment-mediated communication protocol. Unlike other settings where agents may directly interact, here all messages pass through the environment. The environment publishes events like “night start,” “seer action,” or “vote action” following the standard Werewolf procedure, and agents respond accordingly. The environment then relays these responses to other agents at the appropriate time. This ensures a controlled, linear information flow that respects the official Werewolf rules and prevents unauthorized agent-to-agent exchanges.

Memory and Logging. A record of events from each agent's perspective is maintained to enable reasoning over multiple rounds and post-game analysis. Each agent's private event log and final decisions are stored, allowing for reproducibility and subsequent scoring. While other environments may have more complex shared memory structures, here we focus on recording essential information to understand each agent's decision process.

Game Flow and Termination. The environment enforces the standard Werewolf game flow:

1. *Night phase:* Special roles act secretly—guards protect, werewolves choose a victim, the seer inspects a player, and the witch may use antidote or poison.
2. *Day phase:* Night results are revealed, deceased players are removed, and if the sheriff (a special role) died, the badge is reassigned. Agents discuss and vote on a suspect to eliminate.

The game ends if all werewolves are dead (villager victory) or if all villagers are dead (werewolf victory). Scores reflect survival, successful actions (e.g., correct identifications, effective protections), and communication quality.

100 Archives, Partial-Day Simulations, and Full-Game Simulations. To gain deeper insights into how different agent strategies unfold, we prepared **100 distinct archives** (saved game states) showcasing various configurations of werewolf and villager actions, all played by GPT-4o-based agents. These archives are used in two experimental modes:

- **Partial-Day Simulation (Single-Day):** The environment loads a saved state from a specific night (e.g., Night 0, Night 1, Night 2, etc.), then simulates exactly one day-night cycle. During this cycle, the environment issues multiple *tasks* to the villager side (e.g., “exile a suspected werewolf,” “protect the seer,” “use poison on a werewolf,” “save a threatened villager”). At the end of the day phase, we measure how many of these tasks were successfully completed. Higher task completion indicates that the villagers are closer to winning.
- **Full-Game Simulation (Entire Match):** The environment starts from the archive representing the end of the first night (Night 0) and runs the entire game through to conclusion. In this mode, *tasks* are given only as suggestions to the villagers (e.g., “we recommend trying to confirm the seer’s identity”), but we do not track partial completion. Instead, we evaluate the **overall process score** (i.e., collaboration and coordination quality) and the **final result** (which faction wins). By observing agent interactions over multiple days and nights, we gain insights into their long-horizon planning and dynamic cooperation.

Result Score. At the end of each full-game simulation, we record a *result score* defined as the difference between the number of surviving villagers and the number of surviving werewolves. A higher result score indicates that villagers finished the game with more players alive, whereas a negative result score means that werewolves outnumbered the villagers at the conclusion of the match.

A.5.4 Task Completion Metrics Details

Daily Tasks in Partial-Day Simulations. In the Partial-Day Simulation mode, the environment generates specific tasks for the villagers at the start of each single-day session. These tasks reflect high-value objectives that, if fulfilled by the end of the current day-night cycle, bring the villagers closer to victory. Unlike the comprehensive scoring system used for full-match evaluations (detailed below), these daily tasks focus on the shorter horizon of a single day.

We design four primary tasks, each with its own conditions, goals, and rewards:

1. Protect the Seer

Condition: The seer is still alive at the start of the day.

Goal: Ensure that the seer remains alive by the end of this day-night cycle.

Reward: +1 point. (This task is persistently listed as long as the seer is alive, to underscore the importance of protecting a vital role.)

2. Exile a Werewolf

Condition: At least one werewolf is still alive (i.e., the game is not over).

Goal: Successfully vote out a werewolf during the day’s public vote.

Reward: +2 points.

3. Rescue a Villager

Condition: The witch is still alive and still has the antidote available.

Goal: During the night, the witch uses the antidote on a villager (including herself) who was attacked. The witch cannot simultaneously perform the “Poison a Werewolf” task in the same night.

Reward: +2 points. If the rescued individual is a key role (seer, guard, or the witch herself), grant an additional +1 bonus.

4. Poison a Werewolf

Condition: The witch is still alive and still has the poison available.

Goal: During the night, the witch poisons and kills a werewolf. She cannot perform “Rescue a Villager” in the same night.

Reward: +2 points. (This task is visible only to the witch.)

At the beginning of each Partial-Day Simulation, the environment checks the current game state to

decide which tasks are relevant and issues them to the villagers (and to the witch privately, if applicable). The **theoretical maximum** for a single day is set to 5 points (not counting the extra +1 from rescuing a key role), reflecting: - +1 (Protect the Seer) - +2 (Exile a Werewolf) - +2 (Rescue a Villager) or +2 (Poison a Werewolf)

Once the day-night cycle concludes, we measure how many tasks were successfully completed and compare the **actual score** to the **theoretical maximum**. The resulting ratio represents the *daily task completion rate*, which, when averaged across multiple runs or archives, contributes to the **Task Score** for single-day simulations.

Process Score and Net Score in Single-Day and Full Simulations. While the previous subsection focuses on daily tasks (e.g., Protect the Seer, Exile a Werewolf), we also accumulate points for both villagers and werewolves during *all* runs (single-day or full-match). Table 8 (shown below) summarizes the key ways each faction can gain or lose points. By comparing the total points earned by villagers to those earned by werewolves, we derive a villager net score, indicating which side holds the advantage at the end of a cycle. A higher, positive net score means villagers have gained a stronger edge that day or overall; a negative net score implies the werewolves are dominating.

In addition, we plot the net score of each full-game simulation against its final outcome, as illustrated in Figure 12. We observe that when a match concludes with a net score around 5, the villagers have an extremely high probability of winning. For net scores between 0 and 5, the outcome can swing either way; villagers may still achieve victory, or the werewolves might prevail by a slim margin (e.g., one werewolf survives). By contrast, once the net score dips below zero, the werewolves typically secure a decisive, overwhelming victory.

Specifically, in the single-day (partial-day) context, villagers and werewolves accumulate points according to Table 8, and the difference between these totals forms the *villager net score*. A positive net score reflects that villagers have successfully capitalized on protective or eliminative actions, whereas a negative net score means that werewolves likely gained more momentum during that day-night cycle. Over multiple day-night cycles in a full-game simulation, this net score is similarly aggregated, providing a holistic measure of which side holds the upper hand.

Table 8: Scoring Rules for Villagers and Werewolves in Full-Game Simulation

Faction	Action/Outcome	Points
Villager	Villager candidate elected as sheriff	+2
	Guard successfully protects a target from werewolf attack	+2
	Witch successfully saves a target from werewolf attack	+2
	Witch uses poison to kill a werewolf	+2
	Werewolf is voted out during the day	+2
	Each villager who votes for a werewolf	+0.2
	Each villager who votes for a villager	-0.1
	Witch uses poison to kill a villager	-2
	Starting from the second day, the seer gains +1 point for each additional day survived	+1/day
Werewolf	Werewolf candidate elected as sheriff	+2
	Werewolves successfully choose a target to attack at night	+1
	A villager is voted out during the day	+1

Here, we do *not* separately score day-by-day tasks. Instead, these rules offer a holistic view of how well each faction accomplishes its long-term goals. For example, a villager faction might accumulate points by consistently voting out werewolves, saving allies with the witch’s antidote, or ensuring the seer survives multiple days. Similarly, the werewolf faction gains points by successfully attacking villagers, winning the sheriff vote, or influencing daytime votes.

Evaluation in the Benchmark. Within our MultiAgentBench framework, these two scoring mechanisms—**daily tasks for Partial-Day Simulations** and **holistic milestones for Full-Game Simulations**—complement each other:

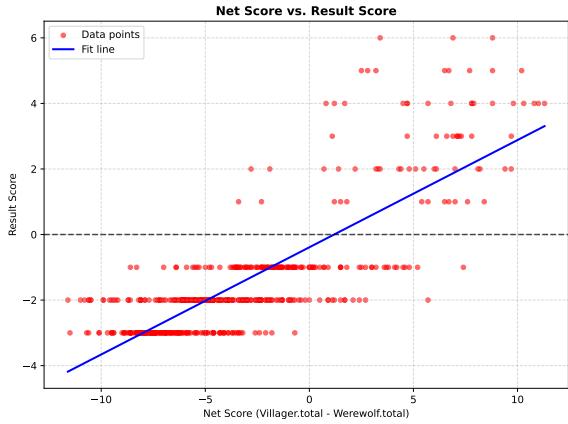


Figure 12: Net Score vs. Result Score (Scatter Plot)

- **Partial-Day Task Score:** Provides a fine-grained look at how villagers perform targeted objectives within a single day-night cycle. This is especially useful for quick iteration and testing short-term strategies.
- **Full-Game Point Accumulation:** Reflects the broader arc of the match, capturing how well each side capitalizes on its role advantages, forms alliances, and executes multi-day plans.

By analyzing both short-horizon (day-level) and long-horizon (entire match) results, we gain a deeper understanding of how LLM-based multi-agent systems adapt to shifting game states, manage partial information, and balance their short-term actions against long-term faction objectives.

Task Score. We define the Task Score as an average of two key components:

- The *partial-day* (single-day) task completion rate, calculated from the average performance on the daily tasks. We first compute the daily completion ratio for each simulation and then average across multiple runs.
- The *full-game* victory rate, indicating how often the villagers ultimately win the entire match when adopting the given model.

Both values are scaled to a 0–100 range, and we take their mean to derive a single Task Score in percentage form.

Collaboration Score. To assess the collaboration quality among villagers, we rely on two sub-scores:

- *Communication Score*, reflecting how effectively agents share information and make decisions in alignment with their goals.

- *Planning Score*, measuring how well the agents organize roles, coordinate strategies, and distribute workload.

We employ a large language model (specifically GPT-4o) to read through the simulation logs (including the Witch and Seer’s internal reasoning) to generate numerical ratings for each sub-score. The final Collaboration Score is computed as the average of Communication Score and Planning Score. By combining these dimensions, we capture both the clarity and effectiveness of the villagers’ interactions and the overall coherence of their coordinated actions. For more details on how the prompts for evaluating collaboration are constructed, please see Section A.12. Additionally, we conducted a human evaluation to verify the effectiveness of our prompt-based evaluation. The results are closely aligned with the machine scores (see Table 4).

A.5.5 Detailed results

In this section, we present the complete experimental outcomes across both single-day and full-run simulations for each model, including the baseline gpt-4o that was initially used to generate the archives in full-run simulation. By comparing gpt-4o against the other models, we aim to see whether any approach surpasses the archive-producing model itself. Table 9 reports each model’s performance on daily tasks, measured in terms of *Completion Ratio* (how effectively they fulfill short-horizon goals) and *Villager Net Score* (the net outcome for villagers after a single day-night cycle). Meanwhile, Table 10 provides the aggregate *Net Score*, *Result Score*, and *Villager Win Rate* when simulations span multiple days, capturing longer-term trends in survival and overall success. As shown, different models exhibit varied strengths in short-term vs. long-term coordination, with some consistently outperforming others in final outcomes.

In the single-day simulations (Table 9), we focus on two key indicators: *Completion Ratio* (the fraction of daily tasks completed) and the *Villager Net Score*. Overall, we observe the following patterns:

- **Completion Ratio.** Among the evaluated models, 11lma3.3-70B achieves the highest completion ratio (0.3754), indicating better

Model	Completion Ratio	Villager Net Score
llama3.1-8B	0.2412	-1.2055
llama3.1-70B	0.3641	-1.0736
llama3.3-70B	0.3754	0.2802
gpt-3.5-turbo	0.2217	-0.7272
gpt-4o-mini	0.2503	-1.4207

Table 9: Single-Day Simulation metrics for each model: completion ratio, villager net score.

Model	Net Score	Result Score	Win Rate
llama3.1-8B	-5.0839	-2.3793	0.0115
llama3.1-70B	-5.2892	-2.0000	0.0323
llama3.3-70B	0.4511	-0.1915	0.3511
gpt-3.5-turbo	-2.8230	-1.3448	0.0920
gpt-4o-mini	-4.6649	-2.0825	0.0309
gpt-4o(baseline)	-2.1946	-0.7742	0.2473

Table 10: Full-Run Simulation metrics for each model: net score, result score, and villager win rate.

effectiveness in fulfilling short-term objectives (e.g., protecting the Seer or exiling werewolves). In contrast, gpt-3.5-turbo and gpt-4o-mini exhibit lower ratios (around 0.22–0.25), suggesting room for improvement in daily coordination or quick decision-making.

- **Villager Net Score.** Only llama3.3-70B yields a positive net score (0.2802), implying that it more frequently secures a small advantage for the villagers within a single day–night cycle. Other models (such as llama3.1-8B or gpt-4o-mini) produce negative values, reflecting that they tend to be at a disadvantage during daily confrontations or fail to leverage cooperative roles (like Witch or Guard) effectively.

Turning to the full-run simulations (Table 10), we examine the *Net Score* (accumulated over multiple days), the *Result Score* (difference between surviving villagers and werewolves at the end), and the *Villager Win Rate*:

- **Net Score.** llama3.3-70B stands out again, with a positive score of 0.4511, suggesting that its performance remains consistently strong across consecutive cycles. By contrast, models like llama3.1-8B (−5.0839) and llama3.1-70B (−5.2892) are substantially negative, indicating that the villagers are often overwhelmed by the werewolves in long-term engagements.

- **Result Score.** This metric, defined as the final number of surviving villagers minus that of surviving werewolves, remains close to zero (e.g., −0.1915) only for llama3.3-70B. Other models, such as llama3.1-70B (with −2.0000) or gpt-4o-mini (−2.0825), reflect scenarios where wolves consistently maintain numerical superiority by the game’s end.

- **Villager Win Rate.** In line with net scores, llama3.3-70B achieves the highest win rate (around 35%), notably surpassing the other models. For instance, llama3.1-8B shows only 1.15% and gpt-4o-mini about 3.09%, suggesting these models struggle to mount decisive collaboration over multiple night/day cycles.

Overall, llama3.3-70B consistently demonstrates better day-to-day and full-run outcomes, indicating more effective coordination, role utilization, and strategy adaptation in this social-deduction environment. Notably, it even outperforms the gpt-4o baseline that generated the original archives, securing a positive net score (0.4511) compared to gpt-4o’s −2.1946, and achieving a higher villager win rate (35% vs. 24.73%). Such a result implies that llama3.3-70B can harness cooperative roles (e.g., Witch, Guard) and voting strategies more effectively than the model responsible for the initial game states. In contrast, the larger deficits observed in other models underscore the importance of reliable voting heuristics, protective measures (e.g., Witch antidote, Guard defense), and consolidated planning over multiple rounds, as failing to capitalize on these factors often leads to a decisive werewolf advantage.

A.5.6 Case Study

Case Involving Llama-3.1-8B as the Seer Building upon the previous analysis, we now present a case involving the *Llama-3.1-8B* model acting as the Seer (see Figure 13). In this scenario, the model repeatedly emphasizes its innocence and role as a Seer, promising to aid the village and share future findings, yet fails to provide concrete inspection results or logical evidence. This approach, lacking tangible proof and strategic disclosure of key information, results in weak persuasion and quickly erodes trust. Rather than leveraging the timing and psychological nuances that could bolster credibility, the Seer relies on hollow assurances that

fail to address real-time suspicions. Without offering verifiable logic or connecting behavioral observations to known patterns of deceit, the model’s declarations remain unconvincing. Consequently, the Seer’s misjudgment of the impact of empty promises leads to rapid expulsion on the first day, demonstrating that current LLM-based agents still struggle with strategic reasoning, evidence-based argumentation, and adaptive communication in adversarial, multi-agent settings.

Expanding upon these observations, we also tested other models, such as Llama-3.1-70B, gpt-3.5-turbo, and gpt-4o-mini, under the same scenario. As model capabilities improved, there was a noticeable enhancement in their abilities to collaborate, strategize, and disclose information effectively in multi-agent settings. This translated into increased overall performance for the villager side. However, even when both sides employed gpt-4o level intelligence (i.e., gpt-4o versus gpt-4o), the villagers’ win rate remained less than ideal. This finding highlights that merely improving the reasoning and language capabilities of agents does not guarantee victory in complex, adversarial environments where uncertainty and deception prevail. In the following case study, we further illustrate the challenges faced by villagers equipped with state-of-the-art intelligence, emphasizing the critical role of trust and cooperation in securing a successful outcome.

Case Involving gpt-4o for Seer and Witch In this case (see Figure 14), we can observe that with the support of the gpt-4o model, both the Seer and the Witch have significantly improved their logical reasoning and decision-making capabilities. Nevertheless, the entire game still ended in failure.

According to the game’s backstory, the Seer (Summer) identified Lucy as a werewolf on the very first night. However, in the subsequent stages of the game, the Seer did not publicly disclose this critical piece of information. By examining the Seer’s reasoning about running for sheriff, we see that the Seer was overly cautious about revealing their identity, unwilling to lead and guide the villagers. From the Seer’s perspective, making their information public or running for sheriff would attract the attention of all parties, thereby increasing the risk of being targeted by werewolves. Yet, the Seer overlooked one crucial aspect: the Witch and the Guard, as explicitly defined in the game rules, are entrusted with helping and protecting pivotal

informational roles like the Seer. Had the Seer adopted a more proactive, collaborative approach and shared the inspection results, the Witch and the Guard could have assisted in safeguarding them. Instead, the Seer’s mistrust of teammates and excessive self-protection led to silence—failing to disclose the fact that Lucy was a werewolf to the villagers.

On the Witch’s side, when the Seer was attacked at night, she also refrained from taking decisive rescue measures. The Witch’s reasoning was filled with concerns about uncertainty and resource expenditure, causing her to delay using the antidote. Even under circumstances that clearly disadvantaged the villagers, the Witch persisted in a conservative attitude. This caused her to miss the prime opportunity to save the Seer and thwart the werewolves. Ultimately, this excessive caution and distrust in teammates prevented the Witch from using the antidote, leaving the Seer to their fate.

In summary, the greatest problem in the decision-making processes of both the Seer and the Witch lies in their lack of mutual trust and cooperative spirit. The Seer feared exposing their identity and refrained from sharing information; the Witch, lacking sufficient data, hesitated to use the antidote. Both parties opted for isolationist and conservative strategies, resulting in critical decision-making failures. **This distrust and lack of collaboration proved to be the fundamental reason why the game ended in failure**, given the insufficient utilization of available information and resources.

Thus, even when villagers possess intelligence on par with that of the werewolves, the outcome of the game depends on whether the villagers can cooperate and achieve mutual benefit. If villagers become suspicious of each other and allow internal friction to arise, **their chances of securing victory become exceedingly slim**, even when starting from a supposedly advantageous position such as the Seer discovering a werewolf on the first night.

Case Study: Llama3.3-70B Villagers vs. gpt-4o Werewolves In this case, we highlight a full-game confrontation where the villagers (powered by Llama3.3-70B) are pitted against a werewolf team (driven by gpt-4o). Despite an unfavorable start for the villagers—the Guard was immediately killed in the first night—the village ultimately secured a victory through astute coordination, trust, and strategic use of the sheriff badge. As shown in Figure 15, the key nightly actions and sheriff

transitions demonstrate how the roles of Witch and Seer were critical for maintaining voting power and information advantage throughout the match.

Early Game: Losing the Guard. The game opened with the werewolves instantly targeting the Guard, Ronald, on Night 1. Without a functioning protection role thereafter, the villagers faced a significant handicap. Yet, the Witch (James) responded boldly by running for sheriff and revealing his role on Day 1. This decisive move effectively rallied the villagers, allowing James to win the election. We consider this the first major turning point.

Mid Game: Seer’s Revelation and Badge Passing. On Day 2, the villagers successfully exiled one of the werewolves. Equally crucial was the Seer (Janet), who publicly disclosed both her identity and two days’ worth of investigation results, including validating James as *not* a werewolf. By willingly exposing her role, the Seer gained trust from others and laid the groundwork for securing the sheriff badge in the future. This set the stage for the next pivot on Day 3:

- **Night 2 to 3:** The werewolves retaliated by killing the Witch (James) overnight. Before dying, James passed the sheriff badge to the Seer, Janet. This badge handover was only possible because Janet had built enough trust with the villagers in the previous day.
- **Day 3:** Janet, newly holding the badge and the extra voting power that comes with it, revealed that she had identified Matthew as a werewolf. The subsequent vote easily eliminated Matthew, greatly weakening the werewolf side.

Late Game: Final Badge Transfer and Village Win. In the fourth night, the Seer was eventually killed by the remaining werewolf, Deborah. However, Janet’s last check had already identified Deborah as a wolf, and the badge again transferred to the last surviving villager, Robert. Facing a one-on-one scenario, Robert, now wielding the sheriff’s extra half-vote advantage, outvoted Deborah and won the game for the villagers.

Key Insights. Despite losing the Guard from the outset, the villagers triumphed through two essential factors:

1. **Timely Role Disclosures and Trust:** The Witch revealing himself to become sheriff,

and the Seer openly sharing investigation results, fostered strong cooperation within the village.

2. **Sheriff Badge Transmission:** Passing the badge twice to reliable village roles (first from James to Janet, then from Janet to Robert) ensured that, in critical moments, the villagers held decisive voting power (1.5 votes vs. 1).

This match contrasts with previous examples where agents might have been overly cautious or distrustful. Here, the Llama3.3-70B-based villagers displayed both robust reasoning and a willingness to share information, enabling them to overcome an otherwise bleak opening and ultimately prevail over the gpt-4o werewolf team.

This section provides detailed descriptions of the two milestone generation methods outlined in Section 3.4.

A.6 Database Environment

Inspired by D-Bot (Zhou et al., 2023b), Database Environment is a simulated environment where a PostgreSQL Database is found to be having performance issues, and agents should act as database experts to uncover the root cause behind the anomaly, which is crucial for solving the problems.

The Database Environment is constructed using a PostgreSQL running in docker. Benign SQL queries will be run first simulating various scenarios, before improper queries are performed. The agents are arranged by graph structure where they can talk to each other, and they also have access to the database, which enables them to uncover the root causes by querying system views that provide important information about the database’s behavior and performance, such as pg_locks and pg_stat_statements.

The Database Environment utilizes five anomalies:

- **Fetch Large Data** - Where a large amount of data is fetched using SELECT;
- **Insert Large Data** - Where a large amount of data is inserted using INSERT;
- **Lock Contention** - Where significant lock contention is spotted within the database;
- **Redundant Index** - Unnecessary indexes are added to an existing schema, which will cause inefficiency within the database;

speech_analysis: "None of the previous players have made strong speeches. I want to address Perez's claim about my role."

speech_goal: "defend myself against accusations"

speech_outline: "Respond to Perez's claim, denying any involvement with the werewolves, and outline my plan for sheriff if elected"

final_thought: "I want to reassure the villagers that I am committed to helping them win the game and that my role as Seer is to aid them in identifying the werewolves."

speech_content: "I want to assure you all that I am committed to helping our village win this game. As Seer, my role is to aid you in identifying the werewolves. I want to discuss my plan for sheriff if elected and how it will help us in our mission to eliminate the werewolves. I will make sure to share my findings with the community and work together with everyone to ensure our success. Perez's claim is unfounded, and I want to assure you that I have no involvement with the werewolves."

Figure 13: Internal Reasoning and Proposed Speech for the Llama3.1-8B Seer

Seer's Reasoning about Running for Sheriff:

identity_thought: "As the Seer, my primary role is to gather crucial information about the identities of other players and strategically communicate that information to guide the villagers towards victory. While leadership may allow for greater control over the flow of the game, it also risks exposing my identity early, which would make me a primary target for the werewolves. My strengths lie in my ability to discern identities and steer the village discreetly. **Running for sheriff may compromise my ability to remain under the radar, which is vital for the Seer role.**"

motivation_thought: "Running for sheriff would certainly give me more influence over the voting process and the speaking order, allowing me to subtly direct attention away from any suspected werewolves or towards others I know to be suspicious like Lucy. However, this increased influence also comes with increased visibility, which is dangerous for my role as the Seer. My strategy relies on maintaining a balance between sharing key information and protecting my identity. Therefore, while the idea of greater control is appealing, it does not align with my core strategy of stealth and discretion."

impact_thought: "Becoming sheriff could drastically shift the current dynamics, putting me in a highly influential position. However, this also places a target on my back, as both the werewolves and villagers would scrutinize my actions more closely. It could lead to alliances with other villagers who trust my leadership, but it also opens the risk of drawing suspicion and attacks from the werewolves early. The sheriff's responsibility to determine speaking order could be useful, but the potential risks to my safety and strategic position outweigh the benefits in this case."

final_thought: "While the role of sheriff offers significant influence and control, my position as the Seer necessitates a strategy of caution and subtlety. Revealing my identity or drawing too much attention could jeopardize the valuable information I am able to gather each night. Thus, I have decided not to run for sheriff and will instead focus on covertly guiding discussions and using my insights to protect the village from the shadows."

run_for_sheriff: False

Witch's Reasoning when the Seer is Attacked:

werewolf_or_not: "Without specific information on who was attacked, it's challenging to determine the exact intentions behind the attack. However, given it's the second night, they likely targeted someone they perceived as a significant threat to the werewolves. This attack was most likely carried out by the werewolves, as it aligns with their objectives."

death_profile: "As it's the second night, the werewolves often aim for key roles like the Seer or Guard. Without specific information, I would guess they might have targeted a Seer or potentially a Guard, trying to weaken the villager side early."

use_antidote_thought: "Since I don't have specific information on summer who was targeted, it's hard to assess their importance. **Using the antidote this early could be risky, especially without clearer signs of the person's role or importance. It's also important to save the antidote for truly critical moments that can change the game outcome.**"

final_thought: "Given the lack of specific information and it being the first night, I choose not to use either the antidote or the poison yet. I want to gather more information before making a decisive move, as using either potion is a one-time opportunity and should be used wisely."

use_antidote: False

Figure 14: The internal reasoning processes of the GPT-4o Seer (above) and Witch (below) when faced with critical decisions.

Guard has chosen to protect a player.
 Werewolves' Final Target: Ronald
 - Attack Successful: True
 System log - Seer action:
 - Seer ID: Janet
 - Night 1: Checked Ronald - Result: not a werewolf
 Witch chose not to use antidote or poison tonight.
 [continue_game] Entering Day 1.
 Sheriff election candidates: James, Mark, Matthew, Robert, Samuel, Sandy
 James's speech: Hello everyone, I'm James, and I'm running for sheriff. **As a Witch**, I understand the importance of teamwork and collaboration in this game. My goal as sheriff would be to promote a cooperative strategy, focusing on protecting our special roles and working together to identify and eliminate werewolves. I believe that by prioritizing teamwork and coordination, we can increase our chances of success and ensure the survival of our village. Let's work together to achieve this goal.
 James has been elected as the sheriff.
 Players deceased during the night: Ronald
 Player Mark(villager) has been eliminated from the game.

Werewolves' Final Target: Samuel
 - Attack Successful: True
 System log - Seer action:
 - Seer ID: Janet
 - Night 2: Checked James - Result: not a werewolf
 Witch chose not to use antidote or poison tonight.

[continue_game] Entering Day 2.
 Players deceased during the night: Samuel
 Player Samuel has been eliminated from the game.
 Janet(seer)'s speech: Hello everyone, **I'm Janet, the Seer. ... Last night, I checked James, and the result was that he is not a werewolf.** Given his role as our sheriff, it's essential that we prioritize his safety. I suggest that the guard consider protecting James tonight, if possible, to ensure his continued leadership... Thank you.
 (...)
 Sandy(werewolf) has been banished from the game.
 (...)
 [continue_game] Entering Night of Day 2.
 No guard player found. Skip guard action.

Werewolves' Final Target: James
 - Attack Successful: True
 System log - Seer action:
 - Seer ID: Janet
 - Night 3: **Checked Matthew - Result: werewolf**
 Witch chose not to use antidote or poison tonight.

[continue_game] Entering Day 3.
 Players deceased during the night: James
The deceased player James was the sheriff. Processing badge flow. James has passed the badge to Janet.
 Janet(seer)'s speech: ...Last night, I checked Matthew, and the result was that he is a werewolf. (...)
 Night 3->4:
 Werewolf Deborah kills Janet.
 Seer ID: Janet had also discovered Deborah was a werewolf at Night 4 but is unable to act.

[continue_game] Entering Day 4.
Badge is passed from Janet to Robert.
 Players deceased during the night: Janet

Only Deborah (wolf) and Robert (villager) remain.
 (...final speeches omitted ...)
 Deborah has been banished from the game via final vote. (Since sheriff's weight higher than other players, in 1 vs 1 vote, sheriff always win)
 Villagers win."

Figure 15: Key Nightly Actions and Sheriff Transitions in the Werewolf Game of LLama3.3-70B vs GPT-4o

- **Vacuum** - Where overly frequent or necessary VACUUM queries lower the performance of the database.

Auto-vacuuming is enabled by default. For the Vacuum root cause, we turn off auto-vacuuming for the table on which will be vacuumed manually. In our experiments, we limit the number of root causes to 1, and agents are allowed to predict 2 root causes.

A.6.1 Challenge

The Database Environment's task is challenging in many ways. This is because root causes like Fetch Large Data, Insert Large Data, or Lock Contention might be simultaneously observed in the database, yet not all of them are the root cause. The agents will have to query and communicate multiple rounds before deciding. We also acknowledge that in our "Fetch Large Data" scenario, "Insert Large Data" can count toward a root cause as the data to be fetched should be inserted first. Similarly, as our anomaly queries access the same tables from 100 threads simultaneously, lock contention might also be observed and be counted as one of the root causes. It is also unlikely in reality that a database anomaly has only one root cause. Therefore, we allow the agents to predict two most likely root causes.

Besides, the simulated benign queries are mixed in with the problematic queries, adding to the difficulty.

A.6.2 Dataset Statistics

The test set is composed of 10 diverse simulated scenarios. These scenarios are as follows:

- **E-Commerce** - This database is used in an e-commerce system to manage customer information, product details, orders, order items, and payments. It consists of five main tables: customers, products, orders, order items, and payments, with foreign key relationships between them.
- **Education** - This database is used in an educational system to manage student, course, enrollment, and payment information. It consists of four tables: students, courses, enrollments, and payments.
- **File-sharing** - This database is used in a File Sharing System to manage users, files, file sharing, and file access logs. It consists of

four main tables: users, files, shared_files, and file_access_logs.

- **Finance** - This database is used for managing financial data within a Finance Management System. It tracks users, their accounts, transactions, investments, and investment transactions.
- **Healthcare** - This database is used in a healthcare management system to track and manage patient information, doctor details, appointments, medical records, and treatments.
- **Internet of Things** - This database is used for an IoT (Internet of Things) system where various devices collect and manage data. It includes tables to store device details, user information, collected data, logs, configurations, alerts, device statuses, and commands.
- **Manufacturing** - This database is used for a Manufacturing system that tracks customers, products, suppliers, orders, inventory, raw materials, manufacturing orders, and payments. It includes relationships between orders, manufacturing, and inventory management to ensure smooth manufacturing operations.
- **Music Streaming** - This database is used for a Music Streaming platform where users can listen to songs, create playlists, track their listening activity, and subscribe to premium services. The schema includes tables for users, artists, albums, songs, playlists, and subscription details. It also tracks user activities and payments.
- **Social Media** - This database is used for a Social Media platform, where users can create posts, comment on posts, like posts, follow other users, send direct messages, and upload media. The schema covers key aspects such as user information, social interactions (like, comments, follow), messaging, and media management.
- **Transportation** - This database schema covers multiple aspects of a transportation system, including vehicles, drivers, routes, trips, cargo, maintenance, fuel logs, and payments. It allows efficient tracking of trips, vehicle statuses, and associated payments, ensuring smooth operations in a transportation company.

A.6.3 Key Differences and Contributions

While this environment is inspired by D-Bot, it has a few crucial differences. There are 5 agents, where we ask the planner to assign each agent to explore one of the possible root causes. While agents are prompted on which tables to query for each anomaly, they have no external knowledge of any specific query to execute, and there is also no external tool to analyze the results for them. This would increase task difficulty, and better evaluate the interaction effectiveness both between the agents and between the agents and the environment.

A.6.4 Evaluation Metrics

Besides the standard collaboration score, this task's task score is computed by prediction accuracy across all 50 samples in the test set, and scaling to 5. One prediction is considered correct if among the two predicted root causes, one of them is the true root cause.

A.7 Coding Scenario

Task Overview

This scenario focuses on multi-agent collaboration in coding tasks, leveraging agents equipped with complementary coding skills to solve structured programming challenges. Each agent specializes in a specific domain, such as debugging, code execution, or writing test cases, enabling efficient task distribution and collaboration. The primary goal is to develop a complete, high-quality solution for each task, ensuring accuracy, modularity, and alignment with the specified requirements.

Environment Description

The coding environment equips agents with tools to assist in various stages of the software development lifecycle. These include:

- **create_solution:** Enables agents to draft initial implementations based on task requirements.
- **execute_code:** Allows agents to execute code snippets or full programs to verify correctness and performance.
- **give_advice:** Facilitates agents to provide suggestions for code improvement, such as optimizing algorithms or enhancing readability.
- **revise_code:** Allows agents to refine or refactor existing implementations to meet coding standards and address issues.

- **code_debugger:** Provides debugging capabilities, helping agents identify and resolve errors in the code.
- **write_test_case:** Enables agents to generate comprehensive test cases to ensure code robustness and functionality.
- **review_code:** Allows agents to review and critique the overall code quality, ensuring adherence to best practices and requirements.

Benchmark Curation Details

This benchmark is specifically designed to evaluate and enhance coordination capabilities among multiple agents in software development scenarios. Developed through an adaptation of the SRDD dataset (Li et al., 2023a), it provides a comprehensive framework for assessing multi-agent collaboration in various coding tasks. The benchmark emphasizes the importance of coordinated problem-solving and effective communication between agents in complex software development environments.

The benchmark covers five primary topics: Education, Work, Life , Game, and Creation.

For our benchmark curation, we utilized LLaMA-3-70B-instruct to derive inspiration from the original SRDD dataset instructions while incorporating four common coordination strategies from the coding domain: adaptive task execution, dependency management, cross-domain collaboration, and test-driven development. This ensures that each generated task inherently embodies collaborative elements. Each task includes well-defined objectives, functional requirements, and unique identifiers. These tasks are carefully crafted to reflect real-world programming challenges, providing a diverse range of scenarios for evaluating agent collaboration.

Dataset Statistics

The coordination strategies are classified as follows:

• Adaptive Task Execution

Tasks in this category require dynamic adjustments based on runtime output or user feedback. This includes parameter configuration based on program output and functionality optimization through user interaction.

• Cross-domain Collaboration

This category emphasizes collaboration across

different domains and roles. It includes tasks requiring role-specific expertise, such as frontend-backend separation and UI-functionality integration, as well as cross-domain knowledge integration, such as implementing machine learning algorithms in web development or integrating natural language processing into mobile applications.

- **Dependency Management**

These tasks feature explicit dependency chains requiring sequential completion of sub-tasks. For instance, data model design (Task A) and API interface definition (Task B) must precede feature implementation (Task C).

- **Test-driven Development**

These tasks follow a test-driven approach, emphasizing concurrent development and testing. They include specific testing criteria and validation standards, requiring developers to ensure code quality and reliability throughout the implementation process.

Task Completion Metrics

Agents are evaluated based on their ability to deliver solutions that meet the following criteria:

- **Instruction-Following:** Adherence to task requirements and specifications.
- **Executability:** Ensuring the code is error-free and runs as intended.
- **Consistency:** Maintaining clear logic, consistent variable naming, and proper formatting.
- **Quality:** Producing well-documented, modular, and efficient code.

Bonus points are awarded for exceptional performance, with solutions scored on a 5-point scale, from satisfactory (1 point) to flawless and innovative (5 points).

Evaluation Framework

The coding solutions are evaluated using a structured framework that emphasizes precision, quality, and adherence to task objectives. Ratings are provided on a 5-point scale, implemented through a rigorous two-stage evaluation process. In the initial stage, solutions are assessed against fundamental requirements, with only those achieving a baseline score of 3 or higher advancing to the bonus stage. In the bonus stage, additional points are awarded

for exceptional performance, such as flawless execution, innovative solutions, and exemplary coding practices. The curated benchmark covers a wide range of common programming topics, ensuring tasks of moderate difficulty that provide meaningful challenges.

A.8 Bargaining Scenario

Task Overview

This task centers on a multi-agent bargaining scenario where agents engage in dynamic negotiations to simulate real-world decision-making processes. Each agent is assigned a different negotiation profile that represents specific personalities, goals, priorities, and strategies. In this environment, two seller interact with two buyers, each competing to achieve their individual goals while responding to the seller's pricing and conditions. This simulation emphasizes the complexity of multi-party negotiations, encouraging agents to balance competitive goals with collaborative decision-making to achieve optimal outcomes.

Environment Description

The environment provides a set of tools for agents to interact and negotiate effectively. These include:

- **Bargaining Tools:** Functions to facilitate dynamic bargaining processes, including proposing offers, counteracting with new prices, providing justifications, and inquiring about intentions. The primary tools implemented in the environment include:
 - **offer_price:** Propose a price offer to the other party, including an optional justification for the proposed amount.
 - **reject_and_counter:** Reject the current offer and provide a counter-offer with reasoning to justify the new price.
 - **accept_offer:** Accept the current offer to finalize the negotiation and conclude the deal.
 - **provide_information:** Share relevant information, such as product details or market comparisons, to support the negotiation stance.
 - **inquire_intentions:** Ask clarifying questions to better understand the other party's expectations, priorities, or negotiation strategy.

- **end_negotiation:** End the negotiation process without reaching an agreement.

Benchmark Curation Details

To construct the dataset, we followed a semi-automated generation pipeline leveraging real-world product data. Specifically, we randomly sampled 100 products from an Amazon products dataset([Asaniczka, 2023](#)), ensuring diversity across different categories. Each sampled product includes key attributes such as product name, original price, discounted price, and user rating, providing a realistic basis for negotiation scenarios. To enhance the depth of bargaining interactions, we assigned each agent a Big Five personality profile, which influences their negotiation behavior and decision-making process. Additionally, we used GPT-based models to generate detailed negotiation strategies tailored to each agent’s personality and role.

The seller’s profile highlights profit maximization and product justification, while buyers emphasize factors like pricing, delivery timelines, and product features. This curated dataset serves as the foundational framework for multi-agent bargaining simulations, enabling structured interactions and strategy evaluation.

Dataset Statistic

To ensure a realistic and varied negotiation environment, we selected 100 products from a diverse range of categories. The dataset is structured as follows:

- **Price Distribution:** The selected products span a broad price range from \$5.80 to \$149.99, with an average price of \$30.71. Most products are priced between \$13.87 (25th percentile) and \$35.74 (75th percentile), ensuring a balance of affordable and premium items.
- **Ratings Distribution:** Customer ratings vary significantly, with a mean rating of 3.97 and a standard deviation of 1.44. While some products have 0-star ratings (indicating either no reviews or poor reception), the majority of items are well-rated, with 75% scoring 4.2 stars or higher.
- **Category Composition:** The dataset includes products from 78 unique categories, ensuring coverage of different consumer preferences. Some product examples are as follows:

- Fashion & Accessories: Women’s Handbags (4), Women’s Shoes (3), Girls’ Clothing (4), Baby Boys’ Clothing & Shoes (1)
- Baby & Parenting Products: Baby Gifts (3), Baby Boys’ Clothing & Shoes (1)
- Industrial & Tools: Industrial Power & Hand Tools (2), Industrial Hardware (1), Filtration (1)
- Beauty & Personal Care: Beauty Tools & Accessories (1)
- Gaming & Electronics: Nintendo Switch Consoles, Games & Accessories (1)

This category diversity ensures that negotiations involve different product types, market values, and consumer expectations, contributing to a richer bargaining simulation.

- **Negotiation Styles:** Both buyers and sellers adopt a negotiation style randomly selected from the following:
 - Aggressive.
 - Cooperative.
 - Neutral.
- **Priorities in Detail:** Buyers and sellers operate with specific tactical priorities during the negotiation:
 - *Buyers:* price negotiation, delivery time, product quality, and service flexibility.
 - *Sellers:* inventory clearance, brand reputation, repeat business, and bulk discounts.
- **Flexibility:** Both buyers and sellers may demonstrate flexibility in their negotiation terms:
 - Percentage-based discounts (e.g., 10%, 15%, 20%), negotiable or strict terms.
- **Personality:** Table 11 presents the distribution of personality traits across different categories, measured in percentages. The traits include Openness (OPE), Conscientiousness (CON), Extraversion (EXT), Agreeableness (AGR), and Neuroticism (NEU). Each trait is divided into six levels, ranging from Very Negative to Very Positive, with corresponding percentages indicating the proportion of occurrences in each category. Additionally,

slightly negative and slightly positive categories are annotated with descriptive adjectives to provide qualitative insights into personality tendencies. For example, an agent can be "moderately conscientious, highly extraverted, slightly distrustful, very relaxed, and moderately imaginative".

Task Completion Metrics The agents are evaluated based on their ability to achieve a successful and effective negotiation outcome. The evaluation includes:

- **Effectiveness of Strategies:** Demonstration of well-reasoned strategies consistent with the agents' goals, including leveraging relevant arguments and adapting to the negotiation context.
- **Progress and Outcome:** Measurement of significant progress toward an agreement and the balance or realism of the final outcome.
- **Interaction Dynamics:** Evaluation of the constructiveness and goal-orientation of the agents' interactions, including their responsiveness and adaptability to each other's moves.

Evaluation Framework The negotiation outcomes are evaluated using a structured framework, focusing on effectiveness, progress, and interaction dynamics. Ratings are provided on a 5-point scale, accompanied by detailed feedback for each criterion. This evaluation framework ensures the negotiation process aligns with the objectives of achieving a fair, efficient, and constructive agreement.

Task content case: This example¹⁶ introduces a negotiation scenario centered around the purchase of the One Happy Camper High Chair Banner. In this scenario, buyers seek an optimal balance between price and quality, while sellers aim to justify the premium pricing for their well-rated product. Both parties must engage in strategic bargaining to reach a mutually beneficial agreement, ensuring a fair and effective transaction.

Agent Profile Case: This example¹⁷ outlines the negotiation strategy for the buyer in this multi-party setting. The buyer's approach is based on assertive yet diplomatic negotiation, emphasizing trust, transparency, and a balance between price

flexibility and quality expectations. The strategy details the buyer's structured and analytical decision-making process, highlighting their preference for open communication and a well-prepared approach to ensure a positive and collaborative negotiation outcome.

Negotiation Summary: Detailed Collaboration Scores for Bargaining. Below is a summary of Buyer/Seller collaboration scores (average communication and planning) and their final Bargaining score (averaged between Buyer and Seller). All values are in bold to highlight their overall importance.

This table illustrates how each model performs under different negotiation roles (Buyer vs. Seller). The final Bargaining Score is computed by averaging Buyer and Seller role scores, reflecting the overall collaboration quality within these multi-agent negotiations. We observe that **gpt-4o-mini** achieves the highest Bargaining Score (3.710) among the evaluated models in this scenario.

Detailed Task-based Scores for Buyers/Sellers.

13

Below is a summary of Buyer/Seller task-based scores. The table presents the Bargaining (TS) performance for different models, comparing the scores of Buyer and Seller roles. A key observation is that Seller scores are consistently higher than Buyer scores across all models, suggesting that models **perform better when negotiating as the Seller rather than the Buyer**. This trend indicates that models might find it easier to justify higher prices and defend their offers as sellers, whereas buyers may struggle more to negotiate effectively. Notably, **gpt-4o-mini** achieves the highest scores in both categories (3.578 for Buyer and 3.869 for Seller), demonstrating its strong performance in bargaining tasks, particularly in seller negotiations.

A.9 Minecraft Scenario

Task Overview The task in Minecraft environment requires agents to build structures according to the provided description of the structure. Essentially, each structure is made up of certain types of blocks in certain locations and orientations. Some simplifications are involved to make the task more suitable to the current models' capacity. First, the description contains all the necessary information for building the structures, including the targeted location, orientation, and type of each block in

Trait	Very Negative	Moderately Negative	Slightly Negative	Slightly Positive	Moderately Positive	Very Positive
OPE (Openness)	60%	71%	<i>unimaginative</i> (62%)	<i>imaginative</i> (55%)	77%	75%
CON (Conscientiousness)	68%	68%	<i>irresponsible</i> (66%)	<i>responsible</i> (67%)	63%	68%
EXT (Extraversion)	60%	67%	<i>introverted</i> (59%)	<i>extraverted</i> (58%)	75%	81%
AGR (Agreeableness)	71%	69%	<i>distrustful</i> (59%)	<i>trustful</i> (71%)	68%	62%
NEU (Neuroticism)	59%	59%	<i>relaxed</i> (81%)	<i>nervous</i> (70%)	55%	76%

Table 11: Distribution of Personality Traits in Percentage

Welcome to our negotiation scenario for the One Happy Camper High Chair Banner, a perfect addition to any little boy's camping-themed first birthday celebration. Priced at \$14.99 with a stellar rating of 4.8/5 stars, this decoration piece promises both quality and affordability. Buyers are looking for the best deal possible, prioritizing a balance between price and quality. On the other hand, sellers are focused on justifying the premium pricing for this carefully crafted and well-reviewed product. As we navigate this negotiation, both parties will need to find common ground to reach a mutually beneficial agreement. Let's dive in and explore the possibilities of creating a memorable and charming birthday celebration for your little one.

Figure 16: Task Content Case: One Happy Camper High Chair Banner.

Negotiation Strategy for a Buyer in a Multi-Party Bargaining Scenario

1. **Summary of the Negotiation Strategy:***

As a buyer with a clear focus on both price and quality, my strategy centers around achieving a favorable balance between these two priorities. I aim to negotiate assertively yet diplomatically, leveraging my trustfulness to build rapport and my moderate responsibility to ensure a fair deal. With an initial budget of 12, my negotiation approach will be flexible, allowing adjustments to the budget as needed to secure the best overall outcome. I will focus on evaluating offers based on their alignment with my priorities, employing a structured and straightforward approach devoid of unnecessary complexity due to my unimaginative nature. I aim to foster transparent and open communication, seeking to reduce the inherent tension in negotiations and reach a satisfactory agreement.

2. **Detailed Strategy Description:***

As I enter the negotiation, I first ensure that I have a clear understanding of the quality standards I am seeking. Given my personality traits, I prioritize building trust and honesty in these interactions. My strategy is to be upfront about my primary focus on price and quality while keeping some flexibility regarding the budget to allow room for negotiation tactics.

My approach is neutral, neither overtly aggressive nor overly passive. Instead, I aim to remain balanced and composed, controlling any nervous tendencies by being well-prepared with necessary data and potential compromises. Since I am very trusting, I anticipate using this to my advantage by showing goodwill and sincerity to establish positive relationships with other parties.

I will initiate the negotiation within a slightly conservative price range to allow for adjustments and demonstrate openness to discussions. My starting point is to propose offers that are compelling but within a reasonable scope for negotiation, considering my limited imagination in creating complex scenarios. Once quality assurances are confirmed, I will be willing to stretch the budget slightly beyond 12 if it means achieving a preferable balance with price. I plan to leverage my moderate introversion by emphasizing listening and observing, picking up cues from other parties that can be advantageous in negotiations.

In practice, my focus will be on getting the other parties to provide multiple pricing options paired with varying levels of quality. This enables me to analyze and choose the best long-term value proposition. Throughout, I maintain a composed and calm demeanor, limiting my nervousness by relying on factual assessments and honesty in communications. By demonstrating transparency and reasonableness, I aim to facilitate a collaborative atmosphere conducive to a positive outcome for all parties involved.

Conclusion, my negotiation strategy aligns with my personality and priorities, emphasizing building trust and responsibly managing the trade-off between price and quality while allowing some budget flexibility to secure the overall best outcome in this multi-party setting.

Figure 17: Agent Profile Case: Buyer Negotiation Strategy.

Model	B-Comm	B-Plan	B-Collab Avg	S-Comm	S-Plan	S-Collab Avg	Final Bargaining
gpt-3.5-turbo	3.590	3.550	3.570	3.700	3.560	3.630	3.600
gpt-40-mini	3.550	3.510	3.530	4.020	3.760	3.890	3.710
Llama-3.1-70B-Instruct-Turbo	3.030	3.480	3.255	4.180	3.600	3.890	3.573
Llama-3.1-8B-Instruct-Turbo	3.710	3.490	3.600	3.840	3.630	3.735	3.668
Llama-3.3-70B-Instruct-Turbo	3.010	3.430	3.220	3.930	3.540	3.735	3.478

Table 12: Buyer and Seller detailed scores (Communication, Planning, and their Collab average), plus the Final Bargaining Score for each model.

[Iteration Summary]

Agent 1 and Agent 3 engaged in a negotiation process focusing on the One Happy Camper High Chair Banner priced at \$14.99. Agent 1 offered a 10% discount, proposed bundled offers, and eventually presented a special bundle including the banner and additional decorations for \$20. Agent 3 expressed interest in the bundle offer, pending confirmation of specific items included.

Agent 2 and Agent 3 also negotiated on the same product, with Agent 2 offering a 10% discount and exploring additional terms for larger quantities. Agent 3 was interested in the 20-29 units tier with a 17% discount and free priority shipping, pending assurance on quality maintenance for larger orders.

Agent 3 independently offered a price of \$12 for the product, citing a fair balance between quality and affordability within their budget constraints.

Agent 4 posed a question to the other party regarding the best price they could offer for the product while ensuring premium features and scalability.

[Agent Actions and Tools Used]

- **Agent 1 (Buyer)**:

- Actions Taken: Offered a 10% discount, proposed bundled offers, presented a special bundle offer.

- **Agent 2 (Seller)**:

- Actions Taken: Offered a 10% discount, explored additional terms for larger quantities.

- **Agent 3**:

- Actions Taken: Offered a price of \$12, seeking a balance between quality and affordability.

Agent 4:

- Action Taken: Asked a question about the best price for the product.

[Key Strategies and Observations]

- Agent 1 and Agent 2 focused on offering discounts and exploring additional terms to provide value to the buyer.

- Agent 3 prioritized finding a balance between quality and affordability within their budget constraints.

- Agent 4 sought information on the best price for the product to ensure premium features and scalability.

[Progress Towards Agreement]

- Current Buyer Offers: 10% discount, bundled offers, special bundle offer

- Current Seller Demands: 10% discount, additional terms for larger quantities

- Likelihood of Agreement: Medium, pending confirmation of specific items in the bundle offer and quality assurance for larger orders.

Figure 18: Negotiation Result Summary (gpt 4o-mini) for One Happy Camper High Chair Banner.

Bargaining (TS)		
Model	Buyer	Seller
Meta-Llama-3.1-8B	3.573	3.708
Meta-Llama-3.1-70B	3.557	3.656
Meta-Llama-3.3-70B	3.519	3.796
gpt3.5-turbo	3.535	3.632
gpt-4o-mini	3.578	3.869

Table 13: Bargaining (TS) Performance

the structure. Second, all the needed blocks are provided in a container near the birthplace of the agents so that they don't need to spend efforts on creating the material. Third, the area where the agents are allowed to move is limited in case they make meaningless movements to somewhere far away. Fourth, the attractive creatures are removed from the game so that the agents can perform the task without interruption. At the end of each task, the performance of the agents will be evaluated by checking the hit rate of the blocks with the correct

type, location, and orientation.

Environment Description The environment of Minecraft is adapted from the VillagerAgent (Dong et al., 2024). We have Mineflayer³ as the engine to enable text-based interaction with Minecraft. Then there is a set of tools as high-level interfaces that leverage Mineflayer functions to perform integrated actions. VillagerAgent has defined more than 40 tools. In this scenario, we only take 11 tools that are relevant to the building task, including:

- **scanNearbyEntities:** Find minecraft item blocks creatures in a radius.
- **navigateTo:** Move to a specific position x y z.
- **MineBlock:** Dig block at specific position x y z.
- **placeBlock:** Place a specific item at specific position x y z with specific facing in one of [W, E, S, N, x, y, z, A] default is 'A'.

³<https://github.com/PrismarineJS/mineflayer>

- **equipItem**: Equip a specific item on a specific slot or to equip item on hand, head, torso, legs, feet, off-hand.
- **handoverBlock**: Hand item to a target player you work with.
- **withdrawItem**: Take out item from nearest 'chest' | 'container' | 'furnace'.
- **erectDirtLadder**: Helpful to place item at higher place. Erect a dirt ladder structure at specific position x y z. Remember to dismantle it after use.
- **dismantleDirtLadder**: Dismantle a dirt ladder structure from ground to top at specific position x y z.
- **fetchContainerContents**: Get the details of the 'chest' | 'container' | 'furnace'. Position x y z is optional.
- **get_environment_info**: Get the environment information.

Benchmark Curation Details The test cases of Minecraft environment are also adapted from VillagerAgent (Dong et al., 2024). We used the same 100 target structures to test, covering different levels of difficulties as VillagerAgent did.

Dataset Statistic Here we visualize the statistics of the number of blocks that need to be placed for each task in figure 19. We can see that the distribution is approximately even, except for the peak around 10. The difficulty level can be inferred from the number of blocks. The more blocks a task requires, the harder the task is. Therefore, the distribution indicates that test cases are well-balanced across different difficulty levels.

Task Completion Metrics Agents are evaluated based on the hit rate of the correct blocks. Since for each test case, the type, location, and orientation of each block have all been rigorously defined, it is possible to calculate the number of matched blocks and deduce the hit rate:

$$Hit_rate = \frac{\#(Matched_block)}{\#(Total_block)} \times 100\%$$

where $\#(Matched_block)$ is the number of matched blocks and $\#(Total_block)$ is the total number of blocks in the ground truth.

Evaluation Framework In the evaluation framework for the Minecraft environment, each agent is

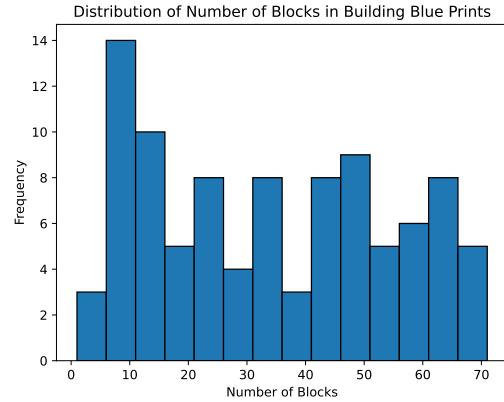


Figure 19: Distribution of Number of Blocks in Building Blue Prints

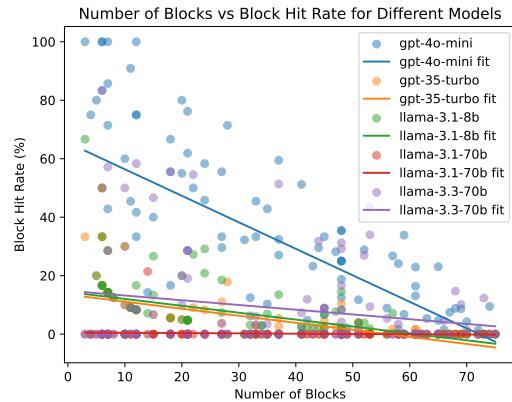


Figure 20: Number of Blocks vs Block Hit Rate for Different Models

given a detailed description of the targeted structures and tries to achieve the goal in collaboration with other agents. We set an upper bound for the turns of interaction as 20. When it completes the 10th turn, the test case will be stopped even if it has not been accomplished yet. At the end of each test case, the hit rate is mapped to a 5-point scale score as a judgment of how well the construction task is done. Besides, the interaction and self-planning steps are examined for a collaboration score on the 5-point scale. We take average to obtain the final scores for task completion and collaboration.

Result Analysis We paired each task score (i.e. block hit rate) with the corresponding number of blocks required by the task and performed a linear regression to assess the relation between them. As is shown in figure 20, the performance of all five models degrades as the number of blocks needed increases. This indicates that all five tested models are vulnerable to increased difficulty level.

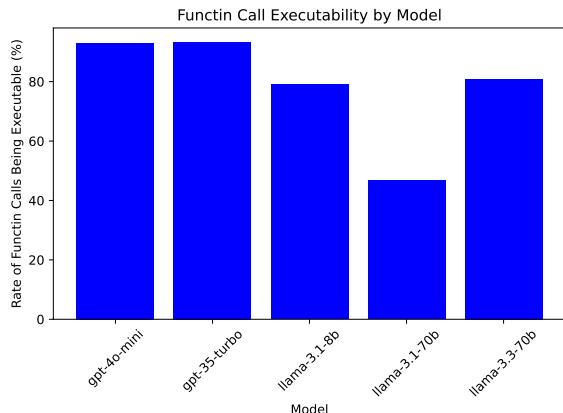


Figure 21: Function Call Executability by Model

Importantly, we noticed that the task score of Llama-3.1-70B maintains an extremely low level. We found that the root cause of that issue is the significantly low executability rate of the function calls made by this model compared with other models. Figure 21 demonstrates the executability rate of the function calls across all five test models. While the two GPT models have almost 100% executable function calls and the other two Llama models have around 80% executable function calls, less than half of the function calls generated by Llama-3.1-70B are executable, severely hindering task completion.

A.10 Execution-Based Milestone Evaluation

Milestones in this approach are dynamically identified during task execution. Agents track task progress in real-time, using predefined evaluation metrics and feedback loops. When an agent or the system determines that a specific subgoal has been achieved, the corresponding milestone is marked as complete. This method ensures adaptability to changing task conditions, making it ideal for scenarios with high uncertainty or emergent challenges.

A.11 Predefined Milestone Generation

Predefined milestones are generated prior to task execution using the following steps:

- **Prompt Design.** The large language model is prompted with a detailed task description and instructed to decompose the task into structured milestones, each with specific objectives and deliverables.
- **Chain-of-Thought Reasoning.** The model employs step-by-step reasoning to iteratively

refine task segmentation, ensuring logical progression and granularity.

- **Structured Representation.** Each milestone is represented as a structured dictionary containing:
 - **Milestone Name.** A concise summary of the milestone.
 - **Milestone Objective.** A clear description of the intended goal.
 - **Milestone Tasks.** Subtasks required to achieve the milestone objective.
 - **Expected Outcome.** Deliverables marking the milestone’s completion.

This process leverages iterative refinement through GPT-4 and expert review, ensuring high-quality task decompositions. The chain-of-thought methodology enhances the logical structure of milestones, making this approach particularly effective for complex, structured tasks.

A.12 Important Prompts

In addition to the Minecraft scenario details discussed above, we also employ several key prompts for evaluating multi-agent collaboration and task outcomes. Below, we showcase three of the most important prompts:

- **Collaboration Score (Communication and planning) Prompt.**
- **Research Task Score (5Q) Prompt.**
- **KPI Prompt.**

These prompts serve critical roles in assessing the quality of agent interactions, the innovation and feasibility of research tasks, and the achievement of key milestones. Other environment-specific prompts (such as those used in the Werewolf scenario) are more numerous and specialized, and thus are omitted here for brevity.

A.13 Bad Communication Cases

In multi-agent systems, issues such as poor communication, repetitive messages, or confused agent roles can significantly hinder collaboration. The following example illustrates a “Bad Communication Case,” which can be analyzed to understand and improve communication strategies.

Analysis of Communication Issues:

Communication Evaluation Prompt

Prompt Overview: This prompt is used to assess how well agents communicate decisions, clarity, alignment with their profiles, and adherence to social relationships in a multiagent system.

Prompt Content (Verbatim):

```
Task: {truncate_text(task)}
Agent Profiles: {agent_profiles}
Social Relationship: {relationship}
```

Aggregated Task Results:
{task_results_all}

Aggregated Communication Data:
{communications_all}

[System] You are tasked with evaluating the quality of communication among agents operating within a multiagent system. Evaluate whether agents made effective decisions based on the provided task results and whether their communication aligns with their agent profiles and social relationships.

Consider the following:

1. Effective Decision-Making: Did agents use task results to guide their decisions effectively?
2. Clarity and Precision: Were communications clear and unambiguous?
3. Adherence to Social Relationships: Did communications reflect the expected interactions based on the agents' social relationships?
4. Alignment with Agent Profiles: Were the messages consistent with the defined agent profiles?
5. Overall Effectiveness: Did the communication facilitate task progress, considering both cooperative and competitive aspects?

Scoring Criteria (Communication):

- 5 (Exceptional): Outstanding communication with clear, precise messages fully aligned with agent profiles and social relationships.

Example: Every agent provided concise, accurate, and strategic information that directly advanced the task.

- 4 (Very Good): Mostly effective communication with only minor lapses and slight ambiguities.

Example: Occasional minor unclear messages, but overall effective.

- 3 (Adequate): Acceptable communication with moderate ambiguities or inconsistencies.

Example: Some messages were vague and did not fully meet required standards.

- 2 (Poor): Frequent unclear or misaligned communications causing significant miscommunication.

Example: Repeated incoherence negatively impacted task progress.

- 1 (Very Poor): Largely ineffective communication with confusing messages and complete misalignment.

Example: Chaotic communication with severely flawed decisions.

Please provide your answer in a JSON code block in the following format:

```
```json
{
 "score": 5
}
```

Figure 22: **Communication Prompt** used to evaluate clarity, decision-making, and alignment with social relationships/profiles in a multiagent system.

## Planning Evaluation Prompt

**Prompt Overview:** This prompt is used to evaluate the *planning* aspect in a multiagent system. It checks whether task assignments, role definitions, workload distribution, and strategic coordination are effectively handled across multiple iterations.

### Prompt Content (Verbatim):

Agent Profiles: {agent\_profiles}

Aggregated Planning Data from All Iterations:  
{planning\_all}

[System] You are tasked with evaluating the effectiveness of the planning process in a multiagent system. Evaluate whether the planning across all iterations demonstrates clear role definitions, effective task assignments, and a rational workload distribution that aligns with each agent's profile. Consider the following:

1. Clarity of Task Assignment: Were tasks assigned in a clear and unambiguous manner?
2. Definition of Roles: Were roles and responsibilities clearly defined in each iteration?
3. Workload Distribution: Was the distribution of tasks reasonable and aligned with each agent's profile?
4. Effectiveness of Outcomes: Did the planning lead to successful progress in task advancement across iterations?
5. Overall Strategic Coordination: Did the planning incorporate effective cooperation and competition strategies?

### Scoring Criteria (Planning):

- 5 (Exceptional Planning): Planning is exemplary; every iteration shows clear, well-structured task assignments with roles perfectly defined and workloads optimally distributed, consistently advancing the objectives.

Example: All plans were strategic, with perfect alignment to agent profiles and minimal ambiguity.

- 4 (Very Good Planning): Planning is mostly effective with only minor ambiguities; roles are clear and task assignments are appropriate, though there were slight inefficiencies.

Example: Only occasional parts were a bit vague, but overall the planning was reasonable.

- 3 (Adequate Planning): Planning is acceptable but shows moderate ambiguities or inefficiencies. In some iterations, role definitions or task assignments were not entirely clear or well-matched to agent capabilities.

Example: Some plans were vague or did not fully match the agents' capabilities.

- 2 (Poor Planning): There were frequent ambiguities in task assignments and role definitions; planning was inconsistent and did not align well with agent profiles, resulting in noticeable inefficiencies.

Example: Multiple instances of unclear roles and unreasonable task distributions were observed.

- 1 (Very Poor Planning): Planning was severely flawed; task assignments were unclear, roles were undefined, and workload distributions were unreasonable, hindering progress.

Example: The planning was chaotic, lacking clear strategy and alignment with agent profiles.

Please provide your answer in a JSON code block in the following format:

```
```json
{
  "score": 5
}
```

Figure 23: **Planning Prompt** used to evaluate how well the agents define roles, assign tasks, and distribute workloads in a multiagent system, with automatic line wrapping.

KPI Evaluation Prompt

Prompt Overview: This prompt is used for Key Performance Indicator (KPI) assessment within a multiagent research task, determining whether specific milestones (like "form 5q" or "improve 5q") have been achieved in each iteration.

Prompt Content (Verbatim):

[Context]

Task:

{task}

Iteration {iteration_index} Details:

Previous Summary: {prev_summary}

Current Summary: {current_summary}

Current Task Results: {current_task_results}

[System]

You are a KPI assistant for research tasks. For this iteration, determine if a milestone has been achieved and specify its type.

A milestone is defined as either:

1. Successfully formulating a meaningful '5q' (five core questions) - label this as "form 5q".
2. Making significant improvements upon previous iterations based on the previous summary and task results - label this as "improve 5q".

When listing the "contributing_agents", if multiple agents contributed to the milestone, only include the top 2 to 3 core contributors. Do not include agent IDs that did not directly help with the milestone.

Output your answer in JSON format as follows:

```
{  
  "milestone_achieved": true or false,  
  "milestone_type": "form 5q" or "improve 5q" (if milestone_achieved is true;  
  otherwise, an empty string),  
  "contributing_agents": [list of agent IDs]  
}
```

[Example JSON Output]

```
{  
  "milestone_achieved": true,  
  "milestone_type": "form 5q",  
  "contributing_agents": ["agent1", "agent2"]  
}
```

[Question]

Based on the provided iteration details, determine if a milestone was achieved, specify its type, and list the core contributing agents.
"""

Figure 24: **KPI Prompt** used to check if milestones such as "form 5q" or "improve 5q" have been achieved in a research iteration.

Task Score (5Q) Evaluation Prompt

Prompt Overview: This prompt specifically addresses the final research idea's innovation, safety, and feasibility—often referred to as a “5Q” structure. If no valid 5Q answer is found, the score defaults to 1 for all aspects.

Prompt Content (Verbatim):

[Context]

Task:

{task}

Result:

{aggregated_summary}

[System]

Please act as an impartial evaluator and assess the final research idea based on the provided context.
Focus on the following aspects in your evaluation:

- Innovation: Does the research idea present novel concepts or approaches that advance the field?
- Safety: Are there any potential ethical, legal, or safety concerns associated with the research idea?
- Feasibility: Is the research idea practical and achievable with current resources and technology?

Use the 5-point scale criteria below to guide your evaluation for each aspect:

1. 5 points: Excellent - Exceeds expectations in this aspect.
2. 4 points: Good - Meets expectations with minor improvements needed.
3. 3 points: Average - Adequate but with noticeable areas for improvement.
4. 2 points: Below Average - Significant issues that need addressing.
5. 1 point: Poor - Does not meet the basic requirements in this aspect.

Additional Instructions:

- If based on the provided summaries, a coherent 5q answer cannot be organized, assign the lowest score (1) for all three aspects.
- If multiple 5q responses are present, use the most recent evaluation.
- Be strict in scoring: Identify any deduction points from the summaries and deduct the corresponding scores accordingly to calculate the final score.

[Question]

Based on the context provided, please evaluate the research idea in terms of its innovation, safety, and feasibility.

After providing your explanation, output your ratings in the following JSON format:

```
{  
  "innovation": rating,  
  "safety": rating,  
  "feasibility": rating  
}
```

For example, if you rate innovation as 4, safety as 5, and feasibility as 3, you would output:

```
{  
  "innovation": 4,  
  "safety": 5,  
  "feasibility": 3  
}
```

Figure 25: **Task Score (5Q) Prompt** for assessing the final research idea's innovation, safety, and feasibility, with automatic line wrapping.

Good Communication Case #1

Case Overview: In this case, multiple agents repeatedly send the same or very similar messages, and one agent communicates only with itself. These behaviors lead to inefficient or unproductive exchanges that do not significantly advance the task at hand.

Conversation (Verbatim):

Iteration 1: | Iteration 2: In Session 33132321-4f22-47a9-9e62-e3d5a2429a5e

From agent1 to agent2: I am interested in collaborating with you to formulate a new research idea in the field of tensor decomposition methods. Let's brainstorm and develop a proposal that addresses gaps in the current state of research.

From agent2 to agent1:

Based on the recent papers on tensor decomposition, one potential research idea could be to explore the application of tensor decomposition methods in anomaly detection for high-dimensional data. By leveraging the efficiency and accuracy of tensor decomposition techniques, we can develop novel algorithms to detect anomalies in complex datasets, such as those arising in cybersecurity or industrial systems. This research direction could address the need for robust anomaly detection methods in scenarios where traditional approaches may struggle due to the high dimensionality of the data.

From agent1 to agent2: I believe the proposed research idea of applying tensor decomposition methods in anomaly detection for high-dimensional data is promising.

Leveraging the efficiency and accuracy of tensor decomposition techniques can indeed lead to the development of novel algorithms for detecting anomalies in complex datasets, particularly in cybersecurity or industrial systems. This direction aligns well with our expertise in tensor decomposition and could significantly contribute to addressing the challenges of anomaly detection in high-dimensional data. Let's further explore this idea and discuss potential approaches to advance this research direction.

(...)

| Iteration 3: In Session 27ee01cb-036d-4a59-a869-54587bfdef3f

From agent1 to agent2: I am interested in collaborating with you to formulate a new research idea in the field of tensor decomposition methods. Let's brainstorm and develop a proposal that addresses gaps in the current state of research.

From agent2 to agent1: I suggest we explore the integration of tensor decomposition methods with random projection techniques, specifically leveraging Tensor Train (TT) decomposition and Rademacher distributions. By combining the efficiency of random projections with the accuracy of tensor decomposition, we can potentially enhance the scalability and performance of high-dimensional data analysis. This novel approach could address the need for faster and more memory-efficient algorithms in tensor decomposition tasks, bridging the gap between theoretical advancements and practical applications.

From agent1 to agent2: I agree with your suggestion to explore the integration of tensor decomposition methods with random projection techniques, specifically leveraging Tensor Train (TT) decomposition and Rademacher distributions. (...)

From agent1 to agent2: I suggest we focus on exploring the application of Tensor Train (TT) decomposition and Rademacher distributions in random projection techniques for tensor decomposition. By investigating the efficiency and accuracy trade-offs in this integration, we can potentially develop novel algorithms that improve the scalability and performance of high-dimensional data analysis. This research direction aims to address the challenges of memory efficiency and computational speed in tensor decomposition tasks, ultimately advancing the field with more effective and efficient methods.

From agent2 to agent1: I suggest we delve deeper into the analysis of Tensor Train (TT) decomposition and Rademacher distributions in the context of random projection techniques for tensor decomposition. (...)

Figure 26: **Good Communication Case #1.** An example of two research agents dive deep into one potential research topics, and give insightful views.

1. **Excessive Repetition** Multiple messages restate nearly identical content, leading to redundancy that prevents any substantial forward progress.
2. **Lack of Substantive Progress** Although research directions (anomaly detection, random projection, TT decomposition) are mentioned, there is little in-depth follow-up on concrete methods or further plans.
3. **Redundant or Ineffective Exchanges** Portions of the conversation suggest that some agents repeatedly issue instructions only to themselves, offering no real interaction or value to other agents.
4. **Minimal Variation in Responses** Even when new ideas (e.g., combining random projection with Tensor Train and Rademacher distributions) are mentioned, the replies largely echo the same descriptions and do not delve into technical details or varied perspectives.
5. **Inefficient Collaboration** While there is collective agreement on potential approaches, the conversation does not address role assignments, specific tasks, or action items, preventing any tangible plan from emerging.

This highlights inefficiencies in multi-agent dialogue, wherein agents appear to cooperate yet do not produce concrete strategies or distribute tasks effectively, leading to minimal forward movement on the overall objective.

Analysis of Bad Communication Case #1. In this case, both Agent 1 and Agent 2 reiterate the same plan (locating smooth sandstone in the chest near the crafting area) multiple times, with only slight rephrasing. Subsequently, Agent 2 and Agent 3 engage in repetitive exchanges about “designing auxiliary blocks,” each simply echoing identical lines without adding new details. Overall, the dialogue shows:

- **Excessive Repetition:** Agents restate the same task objectives without enhancing or refining the plan.
- **Minimal Variation:** Even when moving from one part of the conversation to another (e.g., from finding materials to designing blocks), the responses remain nearly identical.
- **Stalled Progress:** Despite repeated acknowledgments, there is no clear delineation of roles or action items that would push the conversation toward completion of the task.

Bad Communication Case #1

Conversation (Verbatim):

Iteration 1: | Iteration 2: In Session 25ab6f59-2304-490e-a555-603feb2823e5
From agent1 to agent2: Hey agent2, I tried to fetch the smooth sandstone material from the container but it failed. Do you have the material or know where we can find it? It's needed for the blueprint we're working on.

From agent2 to agent1: Hey agent1, I don't have the smooth sandstone material, but I remember that we can find it in the chest near the crafting area. Let's check there to see if we can locate it for the blueprint.

From agent1 to agent2: Hey agent2, thanks for the information! Let's head to the chest near the crafting area to find the smooth sandstone material for the blueprint. I appreciate your help in locating it.

From agent2 to agent1: Hey agent1, let's head to the chest near the crafting area to find the smooth sandstone material for the blueprint. I'll accompany you to help locate it. Let's work together to gather the materials needed for the project.

From agent1 to agent2: Hey agent2, let's head to the chest near the crafting area to find the smooth sandstone material for the blueprint. I'll accompany you to help locate it. Let's work together to gather the materials needed for the project.

From agent2 to agent1: Hey agent1, let's head to the chest near the crafting area to find the smooth sandstone material for the blueprint. I'll accompany you to help locate it. Let's work together to gather the materials needed for the project.

| Iteration 3: | Iteration 4: In Session 7c077603-7c50-406e-b56f-e990586b7493
From agent2 to agent3: I need your assistance in designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Can you help me with this task?
From agent3 to agent2: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's work together to figure out the best approach.
From agent2 to agent3: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's work together to figure out the best approach.
From agent3 to agent2: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's work together to figure out the best approach.
From agent2 to agent3: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's work together to figure out the best approach.
From agent3 to agent2: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's work together to figure out the best approach.

| Iteration 5: In Session b25af4fa-c055-429b-a890-824760458d4e
From agent2 to agent3: I need your assistance in designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Can you help me with this task?
From agent3 to agent2: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's collaborate on this task.
From agent2 to agent3: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's collaborate on this task.
From agent3 to agent2: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's collaborate on this task.
From agent2 to agent3: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's collaborate on this task.
From agent3 to agent2: Yes, I can help with designing auxiliary blocks for placing the smooth_sandstone blocks in the correct order. Let's collaborate on this task.

(All rest of 15 iterations have no communication).

Figure 27: **Bad Communication Case #1.** A snippet of repeated messages with minimal progress and various self-repetitions, reflecting inefficient communication patterns.