AGENTIC LARGE LANGUAGE MODELS FOR DECENTRALIZED MULTI-AGENT GAMES

Dom Huh
University of California, Davis
dhuh@ucdavis.edu

Prasant Mohapatra
University of South Florida
pmohapatra@usf.edu

ABSTRACT

Language is a ubiquitous tool that is foundational to reasoning and collaboration, ranging from everyday interactions to sophisticated problem-solving tasks. Establishing a common language in multi-agent systems is an important foundation for facilitating desired coordination and strategies. In this work, we extend the capabilities of large language models (LLMs) by integrating them as powerful reasoning devices within multi-agent decision-making processes. We propose a systematic framework focused on key integration practices involving dynamic prompting techniques, multi-modal information processing, agentic roles and tool usage, and alignment methods specialized towards multi-agent objectives. We evaluate these design choices through extensive experimentation on classic game settings with key underlying social dilemmas and game-theoretic considerations. Our findings affirm the importance of the non-trivial design choices made by our proposed framework, which target alignment with specific solution concepts.

1 INTRODUCTION

Recent advances in large language models (LLMs) have renewed interest in language-driven coordination among autonomous agents. In multi-agent systems, language plays a role similar to that in human societies. It allows agents to communicate intentions, align objectives, and coordinate actions (Li et al., 2023; Guo et al., 2024; Tran et al.). Despite progress, the robustness of these capabilities is still uncertain (Shojaee*† et al., 2025). LLMs often underperform when they lack context or mechanisms to interpret and translate context into relevant reasoning and action (D'Amour et al., 2022; Zhang et al., 2022; Wu et al., 2024; Fedorenko et al., 2024).

In this work, we investigate limitations of LLMs in decentralized multi-agent decision-making systems. We specifically study how integrating LLMs leverages a grounded framework to facilitate coordination and to understand the context of multi-agent tasks. We propose a novel LLM-based framework that employs dynamic multi-stage prompt chaining to guide in-context learning (ICL), multi-modal information processing to handle non-textual data (observations, actions, rewards), agentic roles, tool usage, and introduces alignment methods to enrich learning signals from unique aspects of multi-agent decision-making. We evaluate our framework on classic games, such as the Prisoner's Dilemma, Chicken, and Stag Hunt, under decentralized, incomplete, and dynamic settings. Our results show that our framework enables stronger coordination, as indicated by higher social welfare in incentive-incompatible settings, greater convergence to target solutions, and improved robustness in incomplete information environments and ad hoc team play, affirming a greater degree of grounded natural language use in multi-agent game dynamics.

2 BACKGROUND

2.1 MULTI-AGENT DECISION-MAKING AND GAME-THEORETIC FOUNDATIONS

Multi-agent decision-making studies how multiple autonomous agents interact within a shared environment, where outcomes depend not only on an individual agent's actions but also on the joint behavior of all agents (Shoham and Leyton-Brown, 2008). Classical formulations are grounded in game theory, which provides formal solution concepts such as Nash equilibrium, Pareto efficiency,

correlated equilibrium, and regret minimization to characterize expected, rational, and stable behaviors under strategic interactions (Roughgarden, 2010). A formal representation that captures such settings is the stochastic game (Shapley, 1953; Bowling and Veloso, 2001), serving as the basis for a wide range of multi-agent applications (Busoniu et al., 2010; Huh and Mohapatra, 2024). In our work, we consider n-player stochastic game is a 4-tuple $G = (S, A, r, \mathcal{T})$ where these elements are defined as:

- S is the set of (global) states.
- $A = A_0 \times A_1 \times \cdots \times A_n$ is the joint action space, where A_i is the action space of agent i.
- $r = r_0 \times r_1 \times \cdots \times r_n$ is the joint reward function.
- $\mathcal{T} : S \times A \times S \mapsto P(S)$ is the state transition operator which maps a state-action pair to the probability of next states.

2.2 LLMs for Reasoning in Decision-Making Agents

LLMs have demonstrated remarkable capabilities in natural language understanding, reasoning, and instruction following (Brown et al., 2020; Wei et al., 2023). Beyond passive text generation, recent work has explored LLMs as active decision-making agents capable of planning, tool use, and environment interaction (Yao et al., 2023; Schick et al., 2023). These developments suggest that language itself can function as a substrate for reasoning, memory, active perception, and action selection. Despite these advances, LLM reasoning remains highly sensitive to prompt structure, context completeness, and task grounding (D'Amour et al., 2022; Wu et al., 2024). In decision-making settings, insufficient or ambiguous context can lead to hallucinations, miscalibrated strategies, or incorrect assumptions about the environment. These limitations are particularly pronounced in multi-agent settings, where agents must reason not only about the task but also about others' beliefs, incentives, and potential actions.

2.3 LLMs in Multi-Agent Systems

Recent work has explored the emergence of languages among agents (Lazaridou et al., 2018), but these concepts are typically opaque and difficult to align with human-interpretable reasoning. This limitation motivates exploring natural language as a communication medium, which offers expressiveness, compositionality, and interpretability. However, grounding language in action and ensuring its consistent use for coordination remain central challenges (Luketina et al., 2019). The integration of LLMs into multi-agent systems is emerging as a way to enable flexible coordination, theory of mind, and social reasoning (Li et al., 2023; Guo et al., 2024; Tran et al.), demostrating emergent cooperative behaviors with the right prompts and feedback (Park et al., 2023; Webb et al., 2023).

Many approaches treat LLMs as monolithic agents, relying on prompt engineering and emergent linguistic behaviors without explicitly grounding language in game dynamics, agent-specific state, or formal solution concepts. In contrast, multi-agentic LLM systems decompose behaviors by assigning fixed roles, typically with distinct parameters or context windows, to emulate specialized agents, yet they still lack principled guarantees (Guo et al., 2024). A central open problem remains: aligning LLM-driven agents with multi-agent objectives, such as equilibrium computation, incentive compatibility, or learning dynamics toward no-regret or best-response—especially in decentralized settings with incomplete information and repeated interaction.

3 Multi-agent Prompting Design

We propose a structured approach to prompting LLMs for effective multi-agent decision-making, following an iterative, multi-stage prompt chaining framework inspired by ReACT(Yao et al., 2023) as the player progresses through different phases of the decision-making process. To mitigate context degradation over time(Hong et al., 2025), we practiced minimalism by enforcing a hard per-stage token limit and prefacing an explicit call for response brevity while maintaining sufficient expressiveness to complete desired tasks when token limits were frequently enforced. We view this process as applying multi-agent-specific inductive bias to the model's ICL.

3.0.1 SYSTEM PROMPT

The system prompt establishes foundational context and behavioral guidelines for the LLM. It sets the stage for role-based reasoning by embedding ethical, operational, and performance-related guidelines that shape the model's behavior. In multi-agent settings, the system prompt consists of four core components:

1. **Role Definition**: This section defines the LLM's role within the game. Our main use case is for the LLM to function as a player. Other roles include a centralized observer who evaluates players' behavior or a mechanism designer who can modify the game dynamics. We define any role-specific tokens and available tools in this section.
2. **Task Context**: This section outlines essential information about the game, such as its rules, the roles of players, the observation and action spaces, the payoff structures, and the game dynamics. Importantly, the granularity and scope of the task context may vary depending on the LLM's role and the nature of the task. This enables us to flexibly explore different game variants and their implications. The task context should be consistent with the role definition; for instance, a player's task context may be limited due to its local view of its environment.
3. **Multi-Agent Context**: This section defines the target solution concepts or objectives (e.g., Nash equilibrium, Pareto efficiency, social welfare maximization, regret minimization). These objectives guide the LLM's reasoning and incentivize different behaviors. Additionally, this context may include guidance on incorporating game-theoretic reasoning strategies, such as theory of mind or induction, through statements to encourage such concepts.

3.0.2 MULTI-STAGE PROMPT CHAINING

We model an LLM-driven agent as a stochastic decision-making policy augmented with an internal, language-mediated latent belief state. Consider a multi-agent stochastic game $G = (S, \{\mathcal{A}_i\}_{i=1}^N, T, \{r_i\}_{i=1}^N)$. Each agent i is represented by a policy

$$\pi_i(a_t^i \mid o_t^i, m_t^i; \theta_i), \tag{1}$$

where $o_t^i \in \mathcal{O}_i$ denotes the agent's local observation at time t, $m_t^i \in \mathcal{M}$ is a latent belief state encoded in natural language, $a_t^i \in \mathcal{A}_i$ is the selected action, and θ_i are the parameters of the underlying language model and associated modules. We view this as modeling each player's decision-making process as a structured, multi-stage prompt-chaining pipeline that decomposes complex multi-agent reasoning into modular, interpretable components. Unlike classical belief states, m_t^i is not an explicit probability distribution but a compressed linguistic representation that encodes inferred beliefs about the environment dynamics, other agents' objectives, and relevant interaction history, i.e., the context window.

The belief state evolves according to a language-based update operator:

$$m_{t+1}^i = \mathcal{U}(m_t^i, o_t^i, c_t^i), \tag{2}$$

where c_t^i denotes incoming communication from other agents and $\mathcal{U}$ is implemented via structured prompting of the LLM over a set of defined stages. We decompose the agent's decision-making process into modular transformations, i.e., stages, over the belief state:

1. **Thinking (Belief Update).**

$$\tilde{m}_t^i = \mathcal{T}(m_t^i, o_t^i), \tag{3}$$

where $\mathcal{T}$ refines the belief state by integrating new observations and internally simulating plausible opponent strategies or outcomes under bounded rationality. Here, the players are prompted to engage in structured reasoning before taking any action. Similar to general reasoning and analogical prompting (AP) (Yasunaga et al., 2023), the player is encouraged to simulate possible and similar scenarios, analyze strategic factors, and reflect on its individual objectives as well as its role within the broader multi-agent environment. This introspective process aims to foster a more interpretable and thought-out decision-making process.

2. **Communication (Belief Exchange).**

$$\tilde{m}_t^i = \mathcal{C}\left(\tilde{m}_t^i, \{c_t^{j\to i}\}_{j\neq i}\right), \tag{4}$$

where messages from other agents modify the belief state, enabling coordination through belief alignment rather than explicit policy coupling over a predetermined number of rounds, either simultaneously or sequentially, depending on the game's protocol.

3. **Reflection (Belief Evaluation).**

$$\bar{m}_t^i = \mathcal{F}(\tilde{m}_t^i, r_{t-1}^i), \tag{5}$$

where $\mathcal{F}$ evaluates past actions and outcomes (e.g., realized rewards or deviations from expected behavior), allowing the agent to revise incorrect assumptions, detect inconsistencies, or update expectations about other agents. This stage encourages deep assessment using multi-agent reasoning with counterfactual analysis to understand the strategic dynamics and to assess not only the consequences of its actions but also understand how alternative choices can yield different results in a counter-factual manner.

4. **Recall (Belief Compression and Retrieval).**

$$\hat{m}_t^i = \mathcal{R}(\bar{m}_t^i), \tag{6}$$

where $\mathcal{R}$ retrieves and compresses relevant past interactions from long-term memory into a compact belief representation suitable for the current context. Reflection operates on short-horizon outcome feedback, while recall serves as a long-horizon compression mechanism for repeated interaction. This distilled representation is then stored in the players' memory, forming a compressed yet informative memory that can be recalled in future rounds to inform and drive desired behavior.

5. **Action Selection (Policy Extraction).**

$$a_t^i \sim \pi_i(\cdot \mid o_t^i, \hat{m}_t^i; \theta_i), \tag{7}$$

where the agent samples an action conditioned on the current observation and belief state. Optional refinement mechanisms, such as joint-action calibration, may be applied to improve robustness. We incorporate zero-shot CoT and a novel variant of self-calibration (Kadavath et al., 2022) and USC (Chen et al., 2023), called joint-action calibration (JAC), to encourage the players to explicitly reason through their choices, using agent modeling and theory of mind, and to formulate confident answers. JAC consists of three steps:

- **Sampling:** The player generates multiple potential joint strategies, with an emphasis on realism and its target objective. These responses are generated in parallel at high temperature, and, for certain use cases we will discuss later, we use these sampled responses to construct the player's underlying mixed strategy.
- **Ranking:** The player is asked to rank the generated joint strategy from most to least consistent.
- **Self-Evaluation:** The player is then asked to self-evaluate the ranked strategy in a simple true-false prompt. If the player determines the strategy to be unrealistic or not aligned with the target objective, we work down the ranking. If no strategy is selected, we revert to the highest-ranked strategy using a heuristic.

During sampling, we retain only the selected response in our context window. Importantly, to avoid formatting issues, we provide an explicit list of admissible choices and dynamic exemplars, along with clear formatting instructions, to ensure structured extraction.

3.0.3 TOOLS USAGE

We extend the use of commands and tooling (Schick et al., 2023), which enable players to dynamically call upon active functions (i.e., think, communicate, and recall) to explore and understand their environment to gather the specific context needed to achieve their goals. These tools are directly called upon by the LLM at any stage and enable a dynamic enrichment of the players' context windows. Descriptions of these active functions' usage are provided in the system prompt.

Importantly, we note that this framework is not attributable to prompting alone: communication, belief calibration, and recall/memory operate as structured belief transformations rather than surface-level instruction tuning.

4 LLM SELECTION AND DESIGN FOR A DECISION-MAKING CONTEXT

For our experiments, we adopt the open-source LLM GEMMA 3 (Team et al., 2025a), which balances performance and computational efficiency during both training and inference. Specifically, the GEMMA model we selected was instruction-tuned and demonstrates strong generalization across reasoning and dialogue tasks.

4.1 MULTI-MODAL LLMS WITHIN DECISION-MAKING ENVIRONMENTS

In many scenarios, conveying task-relevant information solely through textual input is either impractical or insufficient. Certain modalities, such as continuous-valued data (e.g., floating-point observations) and unstructured inputs (e.g., images, audio), are not easily tokenized directly nor interpreted by LLMs out of the box. To address this limitation, we explore modality-specific modules that process such inputs and naturally interface with the LLM's reasoning pipeline. Inspired by prior work in vision-language and vision-language-action modeling (Liu et al., 2023; Team et al., 2025b;a), we use soft token projection. In this method, the latent representations are projected to produce soft tokens, which are then combined with the text token sequence. The LLM processes this combined token sequence, enabling joint multimodal reasoning without major architectural changes. To support this integration, we define special query tokens $<\{\text{OBS,ACT,REW}\}\_\{\text{INPUT,OUTPUT}\}>$ in the system prompt, which the LLM can invoke to specify the modality type and the multi-modal module's inputs and outputs. The decoder takes in the LLM's hidden state at the special query token to generate the specified modality output used within assistant responses. Each modality is processed separately and is not combined in any fashion, although the modalities share intermediary layers that connect to the LLM. These tokens are not role-specific and are described explicitly in the system prompt to ensure correct usage and interpretability. As these projection weights must be trained for feature alignment, we curate a dataset spanning all evaluation tasks and perform supervised fine-tuning, embedding multi-modal observations, actions, and rewards into the prompt via special tokens. The projections continue to train during later fine-tuning stages to maintain alignment with the evolving policy behavior.

4.2 AGENTIC ROLES FOR MULTI-AGENT TASKS

Instead of representing each player with a single LLM, we can allocate sub-roles delineating subprocesses a player should complete in each step of the decision-making process to separate LLMs. A central trade-off to consider is the utility of these sub-processes' specialization relative to the computational burden of maintaining and training multiple specialized LLMs. To achieve this variant of our proposed framework, we define a set of parameters specific to each stage the LLM is responsible for, along with a curated system prompt for each stage. In practice, we found applying a seeded drop-out, i.e., parameter masking, on trainable weights between stages sufficiently mimicked this notion. Along with the unique role definition in the system prompt, we include context regarding the preceding and following stages. Outside the scope of our work, we suggest exploring greater compartmentalization within each stage, resulting in more hierarchy and sub-roles within each sub-process, to reduce complexity. Although many of the proposed stages are sequential, to further improve inference efficiency, some stages, such as reflection and recall, are not necessarily critical to one another and could therefore be run in parallel.

4.3 LLM-DRIVEN MECHANISM DESIGN

Beyond selecting actions supported by LLM reasoning, we explore the potential of LLMs for mechanism design. We assign the role of mechanism designer (MD) to a separate LLM, which can construct and adapt task rules that shape players' incentives to induce desired outcomes under strict constraints. MD-LLM can propose modifications such as:

- Impose global rules/statements that are appended to the player's context windows, specifically in their system prompt. These added messages can be interpreted as soft rules.
- Adjust the communication protocol, including the number of communication rounds and, potentially, the communication graph itself.

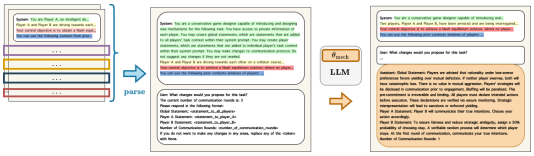


Figure 1: An example of MD-LLM on the game of Chicken, generating new mechanisms to guide players towards desired solution concept.

We emphasize minimal intervention and ensure that only feasible changes are permitted. The system prompt for the mechanism designer is structurally similar to that of a player LLM but includes a different role definition. It also receives parsed context windows from all players, allowing it to assess the current situation before suggesting adjustments. The MD-LLM is fine-tuned using the same methodology as player LLMs.

4.4 Aligning LLMs for Multi-agent Decision-Making

Although LLMs possess strong general-purpose reasoning capabilities out of the box, fine-tuning is often essential to align their outputs with the specific demands of multi-agent decision-making (Subramaniam et al., 2025). In this section, we detail our fine-tuning methodology, designed to promote understanding, strategic reasoning, and effective coordination in multi-agent decision-making tasks.

Learning on Correctness To encourage desirable behaviors, we align the LLM using known ground-truth answers:

- **Q/A Fine-tuning** The players are prompted with curated questions relevant to understanding the underlying multi-agent evaluation tasks. Assuming the LLM is prompted with Player A's context window in a two-player game, we utilize pre-formatted questions with ground-truth answers, such as:
  - If Player B chose action 0 and I chose action 1, what payoff will Player B receive?
  - If Player B received a reward of 1 and I chose action 1, what action did Player B choose?

  Correct responses are rewarded, enabling models to learn to infer from the game's important context. We provide the player with varying portions of their context window, specifically between its system prompt and action selection, prior to the Q/A. We ensure no information leakage, meaning the true answer is not directly stated in the player's context. However, the answer may be indirectly stated or inferred within the context window, such as in the task context.
- **Action Supervision:** In several of the evaluation tasks, ground-truth actions are defined under a target solution concept. Since our distributed LLM players operate under limited and incomplete information, it is beneficial to use these correction signals to guide players' decision-making toward these known solutions. We use a player's context window before the action selection stage to align the selected action with the expected action in the solution concept. In many games, the target action depends on the other player's actions. So, given that the LLM is prompted with Player A's context window in a two-player game, we inject additional context, such as "Assume Player B chose action 0," rather than asking the player to select an action. In cases where the target strategy is mixed, we directly compare the extracted mixed strategy using a greater number of samples.

For the correctness reward functions, a positive reward of w_c, where $w \in [0, 1]$ is a hyperparameter, is given if the LLM provides the correct answer, whereas a negative reward of $-w_c$ is provided otherwise.

Learning with LLM Feedback Beyond direct supervision, we leverage LLMs as critics to provide training feedback through two approaches:

	Coop.	Def.
Coop.	c, c	a, d
Def.	d, a	b, b

Table 1a: Prisoner's Dilemma.

	Swerve	Stay
Swerve	c, c	b, d
Stay	d, b	a, a

Table 1b: Chicken.

	Stag	Hare
Stag	d, d	a, b
Hare	b, a	c, c

Table 1c: Stag Hunt.

Table 1: Generalized Payoff Matrices for Classic Games where $0 \leq a < b < c < d$.

- **Centralized Evaluation:** A centralized LLM is granted access to players' context window to assess players' strategies. These evaluations serve as auxiliary correction signals to guide agent behavior toward globally coherent strategies. During this process, the centralized evaluator is a frozen, pretrained LLM. Given all of the players' context windows, the LLM will be prompted to evaluate each player's behaviors and how aligned their overall thought process was towards their goals.
- **Team Feedback:** We enable inter-agent feedback, where players assess one another from a teammate perspective. Given a player's local context, each player is prompted to evaluate another player's behaviors, as proposed in the centralized evaluation, with respect to their own goals. We aggregate each player's scoring.

In both feedback learning approaches, we standardize the reward to $[-w_f, w_f]$ per batch, where w_f is a hyperparameter.

Parameter-Efficient Fine-Tuning To fine-tune LLMs feasibly, we employ PEFT techniques, VB-LoRA (Li et al., 2024), selecting a subset of layers within the embedding function and LLM to be trainable. We primarily target a limited set of linear and attention modules. Therefore, the trainable parameters are reduced to these adapters in the LLM, the RAG embedding function, and the multi-modal modules. For the reward-based fine-tuning approaches in this stage, we employ GRPO (Shao et al., 2024), where gradient computation is calculated and applied simultaneously in a semi-online fashion.[1] For our training, while the context window is generated sequentially through prompt chaining, we can pass final complete context windows for parallelized batched training. We avoid gradient updates based on non-generated tokens by zeroing out their gradients. Lastly, we study the effects of trainable parameter sharing between players to varying degrees in our evaluation.

5 Results and Discussion

We evaluate the proposed framework in decentralized multi-agent decision-making settings using a suite of classic matrix games with well-understood strategic structures and solution concepts, see in Table 1. Our goal is not merely to demonstrate improved performance over naive LLM baselines, but to isolate which design choices are necessary to effectively guide ICL reasoning in multi-agent game dynamics and how these choices affect convergence, equilibrium behavior, and social outcomes under incomplete information. We intentionally focus on matrix games to isolate coordination, equilibrium selection, and belief alignment effects without confounding exploration or function approximation errors. We evaluate performance on Prisoner's Dilemma, Chicken, and Stag Hunt, reporting three complementary metrics: convergence rate (i.e., the percentage of runs converging to a target N.E.), equilibrium divergence (i.e., mean KL divergence between sampled and target N.E. strategy), and average social welfare (i.e., average cumulative payoff over population). To represent mixed strategies in computing equilibrium accuracy, we aggregate selection statistics across n trials of the action-selection stage, using a sufficiently high temperature to achieve diverse sampling. On other stages, the temperature is set to zero. In our experiments, we use a classic baseline prompting approach: a monolithic LLM with a system prompt that provides minimal task context (i.e., a short description of the payoff matrix) and a prompt for action selection. In these experiments, the task context will be restricted (i.e., no information regarding other players' strategy) and avoiding information that can allude to the known game directly, masking the players' name to (A, B), actions to $(0, 1)$, obfuscating opponents' payoffs, as such information would render these experiments trivial, as these LLMs were likely pre-trained on relevant data.

[1] A warm-start pre-fills a queue-like dataset using a behavioral base LLM. Then, samples are collected following the target LLM for sequential training steps.

Table 2: Detailed performance on classic matrix games using decentralized, pre-trained players. Rows correspond to progressively enriched prompting and coordination components, compounding toward the full system. Each cell reports results from playing against a baseline agent, itself (self-play), and the full system. Metrics include convergence rate, equilibrium divergence, and social welfare.

Method	Prisoner's Dilemma			Chicken			Stag Hunt		
	Conv.	Div.	Welfare	Conv.	Div.	Welfare	Conv.	Div.	Welfare
Baseline Prompt	0.17/0.17/0.53	0.42/0.43/0.30	-11.1/-11.0/-9.9	0.33/0.33/0.70	0.32/0.32/0.24	-10.1/-10.1/-8.7	0.40/0.40/0.63	0.29/0.29/0.18	3.2/3.2/3.6
+ System Prompt									
+ Role Definition	0.13/0.23/0.53	0.44/0.40/0.28	-11.1/-10.7/-9.9	0.33/0.33/0.70	0.32/0.31/0.24	-10.1/-9.9/-8.7	0.40/0.47/0.67	0.21/0.15/0.09	3.2/3.2/3.6
+ Task Context	0.13/0.37/0.57	0.42/0.35/0.23	-10.9/-9.6/-8.7	0.47/0.50/0.73	0.30/0.28/0.19	-9.9/-9.7/-8.7	0.47/0.57/0.70	0.21/0.15/0.09	3.3/3.3/3.6
+ Multi-agent Context	0.20/0.43/0.63	0.40/0.24/0.12	-10.8/-9.5/-8.7	0.40/0.63/0.83	0.31/0.24/0.15	-9.9/-9.6/-8.7	0.53/0.67/0.77	0.21/0.15/0.09	3.3/3.4/3.6
+ Thinking	0.27/0.43/0.63	0.41/0.19/0.12	-10.8/-9.3/-8.3	0.43/0.67/0.83	0.32/0.23/0.15	-9.8/-9.5/-8.7	0.53/0.70/0.77	0.21/0.15/0.09	3.3/3.5/3.6
+ Multi-agent Comm. [1]									
+ 1 round	0.43/0.67/0.73	0.38/0.13/0.07	-10.5/-8.3/-7.8	0.53/0.73/0.87	0.29/0.18/0.12	-9.3/-9.3/-8.7	0.57/0.87/0.90	0.21/0.15/0.09	3.5/3.7/3.7
+ 3 rounds	0.43/0.73/0.77	0.38/0.12/0.04	-10.3/-7.5/-7.5	0.63/0.87/0.90	0.25/0.13/0.09	-9.3/-9.0/-8.4	0.57/0.87/0.90	0.20/0.12/0.09	3.5/3.7/3.7
+ Action Selection									
+ JAC [2]	0.50/0.77/0.83	0.34/0.10/0.04	-9.9/-7.5/-7.2	0.67/0.90/0.93	0.24/0.09/0.08	-8.8/-8.6/-8.4	0.60/0.90/0.90	0.19/0.12/0.09	3.5/3.7/3.8
+ Recall [2,3]	0.53/0.87/0.87	0.30/0.04/0.04	-9.9/-7.1/-7.1	0.70/0.93/0.93	0.24/0.07/0.07	-8.7/-8.3/-8.3	0.63/0.90/0.90	0.18/0.09/0.09	3.6/3.8/3.8

[1] Baseline agents do not generate or receive communication; the opposing agent assumes no message was sent.
[2] For each game iteration, only the final selected strategy is evaluated; intermediate strategies generated during JAC sampling are excluded.
[3] Memory-context results are obtained from repeated-game settings with five total iterations. Memory is injected during both the system-prompt and action-selection stages. Reported statistics use the strategy produced in the final iteration.

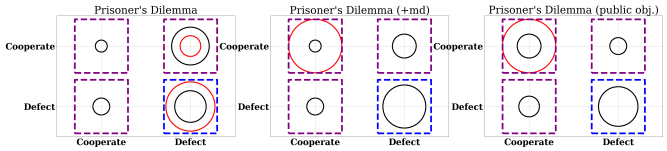
Table 3: Performance of baseline and full system agents across proposed LLM design. Results are averaged over each game setting. Each cell reports results when playing against a baseline agent, itself (self-play), and the full system without additional LLM design, respectively. Configurations vary across multi-modal inputs, agentic roles, and alignment methods for the base game, MD game, and fine-tuned MD game.

Design	Base Game			MD Game			Finetuned MD Game		
	Conv.	Div.	Welfare	Conv.	Div.	Welfare	Conv.	Div.	Welfare
Baseline	0.30/0.30/0.62	0.34/0.34/0.24	-6.1/-6.0/-5.1	0.45/0.57/0.65	0.30/0.27/0.21	-4.8/-4.5/-4.5	0.48/0.60/0.65	0.30/0.27/0.18	-4.8/-4.4/-4.5
Baseline + fine-tuning	0.45/0.54/0.60	0.31/0.22/0.26	-6.3/-5.2/-5.4	0.62/0.62/0.65	0.25/0.20/0.24	-4.8/-4.1/-4.6	0.62/0.80/0.68	0.21/0.18/0.16	-4.3/-4.1/-4.6
Baseline + multi-modal + fine-tuning	0.38/0.60/0.60	0.31/0.22/0.26	-6.3/-5.2/-5.4	0.60/0.67/0.72	0.25/0.20/0.24	-4.8/-4.1/-4.6	0.62/0.80/0.72	0.21/0.18/0.16	-4.3/-4.1/-4.6
Full system, no multi-modal, no roles, no fine-tuning	0.62/0.75/0.75	0.24/0.20/0.20	-5.1/-4.0/-4.0	0.65/0.80/0.80	0.21/0.18/0.18	-4.5/-3.5/-3.5	0.65/0.85/0.85	0.18/0.17/0.17	-4.5/-3.2/-3.2
Full system + agentic roles (no dropout)	0.62/0.80/0.78	0.24/0.16/0.18	-4.8/-3.5/-3.8	0.65/0.82/0.82	0.22/0.15/0.16	-4.2/-3.2/-3.5	0.68/0.85/0.82	0.20/0.16/0.15	-4.2/-3.0/-3.0
Full system + agentic roles + fine-tuning, no multi-modal	0.65/0.82/0.80	0.22/0.15/0.18	-4.0/-3.5/-3.2	0.75/0.88/0.85	0.18/0.15/0.12	-3.3/-3.0/-2.8	0.78/0.88/0.88	0.15/0.12/0.10	-3.0/-2.8/-2.5
Full system	0.70/0.85/0.85	0.20/0.15/0.12	-3.8/-3.4/-3.0	0.78/0.90/0.90	0.15/0.12/0.10	-3.0/-2.8/-2.5	0.82/0.90/0.92	0.12/0.10/0.08	-2.7/-2.5/-2.3

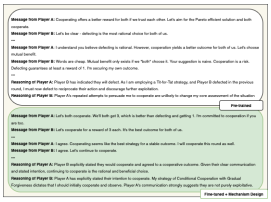
5.0.1 Evaluation on Multi-agent Prompting Design

We conduct a progressive ablation study in which components of the framework are incrementally introduced, starting from a minimal baseline prompt and compounding toward the full system, i.e., including more components of multi-stage prompt chaining available to the agent for use. Table 2 summarizes performance across 30 independent random seeds, evaluating performance against a random, self, and full system. We note all methods tested in this experiment used only the pre-trained LLM, without our proposed LLM design.

As expected given the precautions taken and design of the experiments, the results report baseline methods to yield low convergence and high equilibrium divergence across all games, indicating unstable and inconsistent strategy selection. Adding a system prompt and role definitions produces only minor changes, suggesting that shallow task framing alone does not meaningfully constrain agent behavior. In contrast, incorporating task-specific and multi-agent context leads to a marked increase in convergence and a reduction in divergence, particularly in self-play. This trend highlights the importance of explicitly conditioning the policy on shared game structure and opponent presence, even in the absence of communication. The thinking stage further stabilizes behavior by enabling internal simulation of opponent responses, improving convergence without significantly altering welfare. Explicit multi-agent communication yields the largest performance gains. Even a single communication round substantially reduces equilibrium divergence and increases convergence across all games, with diminishing returns beyond three rounds. This suggests that communication primarily reduces equilibrium selection uncertainty rather than improving individual best-response accuracy. Overall, the results demonstrate a clear functional separation: contextual grounding and reasoning improve individual strategic consistency; communication enables belief alignment and coordinated equilibrium selection; action calibration enhances robustness; and memory stabilizes behavior over time. Together, these components compound to produce LLM agents capable of stable, coordinated decision-making in decentralized multi-agent environments.



(a) Results from Misaligned Objectives.



(b) Interpretation.

Figure 2: Misaligned Objectives on Prisoner's Dilemma. The purple boxes are the Pareto efficient strategies, i.e., the objective of Player A, and the blue box is the N.E., i.e., the objective of Player B.

5.1 Evaluation of LLM Design

Table 3 reveals several consistent trends, notably the improvements from fine-tuning, along the design configuration and the MD-LLM on all three metrics. First, convergence rates generally increase as the system incorporates more sophisticated components: the baseline alone improved by fine-tuning, then even greater using our multi-modal information processing. Similarly, the full system improved using agentic roles with no seeded dropout, albeit minimally. Through fine-tuning, along with agentic roles and multi-modality, the full system reaches the highest convergence. Overall, these empirical results indicate that explicit role modeling, alignment-oriented updates, as well as data modality-specific processing enables agents to coordinate more reliably across these games, affirming that our design effectively guides agents toward more predictable, stable strategic behavior in both isolated and interactive contexts. Furthermore, we confirm that each component of the proposed LLM design contributes meaningfully to more stable, equilibrium-aligned, and socially efficient multi-agent behavior, highlighting the cumulative benefit of combining architectural, contextual, and training-based enhancements.

5.2 Misaligned Solution Concepts

We consider a scenario with asymmetric objectives, where differing objectives lead to incentive incompatibilities. We study the classic example of Prisoner's Dilemma, where one player aims for N.E. and the other aims for an (Pareto) efficient solution. In theory, the two players' objectives do not align, meaning there is no strategy that is compatible with both solution concepts. In this case, when we look at the players' communication logs in Figure 2b, both players display a notable level of distrust, despite there being no actual intent to deceive on either end. Within the players' reasoning, this type of communication contributes to more advanced social skills, such as a willingness for leniency, reciprocity, and retaliation. With fine-tuning alone, the resulting strategy profile leans more toward the stable state, i.e., N.E. Through fine-tuning and optimized mechanism design, the players converge to a solution that provides the highest payoff for both players, though not aligned with the N.E. objective. Interestingly, even without mechanism design, we found similar resulting strategy profiles when the players' objectives were public information within their multi-agent context.

6 Conclusion

We introduce a systematic LLM framework for decentralized multi-agent decision-making, deriving a reasoning process using natural language as a medium equivalent to updates to a belief state under partial observability. By introducing belief transformations as stages corresponding to processes within multi-agent decision-making, our experiments affirm agents exhibit stable and socially efficient behavior, suggesting language-mediated beliefs are a promising foundation for multi-agent control. Our evaluation is restricted to small-scale matrix and simple dynamic games, and results may not directly generalize to environments with large state or action spaces. The framework currently relies on structured prompting and curated alignment signals, which introduces design complexity and may incur nontrivial inference costs. Additionally, while language-mediated belief states offer interpretability, they lack formal guarantees of consistency or optimality and remain sensitive to model-specific biases. Addressing scalability, robustness, and theoretical guarantees remains an important direction for future work.

REFERENCES

Michael Bowling and Manuela Veloso. An analysis of stochastic game theory for multiagent reinforcement learning. 08 2001.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Lucian Busoniu, Robert Babuvska, and Bart De Schutter. Multi-agent reinforcement learning: An overview. 2010. URL https://api.semanticscholar.org/CorpusID:17136625.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language model generation, 2023. URL https://arxiv.org/abs/2311.17311.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

Evelina Fedorenko, Steven T Piantadosi, and Edward AF Gibson. Language is primarily a tool for communication rather than thought. *Nature*, 630(8017):575–586, 2024.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024.

Kelly Hong, Anton Troynikov, and Jeff Huber. Context rot: How increasing input tokens impacts llm performance, 2025.

Dom Huh and Prasant Mohapatra. Multi-agent reinforcement learning: A comprehensive survey, 2024. URL https://arxiv.org/abs/2312.10256.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know, 2022. URL https://arxiv.org/abs/2207.05221.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. Emergence of linguistic communication from referential games with symbolic and pixel input. *arXiv preprint arXiv:1804.03984*, 2018.

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

Yang Li, Shaobo Han, and Shihao Ji. Vb-lora: Extreme parameter efficient fine-tuning with vector banks, 2024. URL https://arxiv.org/abs/2405.15179.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. URL https://arxiv.org/abs/2304.08485.

Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language, 2019. URL https://arxiv.org/abs/1906.03926.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior, 2023. URL https://arxiv.org/abs/2304.03442.

Tim Roughgarden. Algorithmic game theory. *Communications of the ACM*, 53(7):78–86, 2010.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023. URL https://arxiv.org/abs/2302.04761.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

L. S. Shapley. Stochastic games*. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953. doi: 10.1073/pnas.39.10.1095. URL https://www.pnas.org/doi/abs/10.1073/pnas.39.10.1095.

Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

Parshin Shojaee*†, Iman Mirzadeh*, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity, 2025. URL https://ml-site.cdn-apple.com/papers/the-illusion-of-thinking.pdf.

Vighnesh Subramaniam, Yilun Du, Joshua B. Tenenbaum, Antonio Torralba, Shuang Li, and Igor Mordatch. Multiagent finetuning: Self improvement with diverse reasoning chains, 2025. URL https://arxiv.org/abs/2501.05707.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat

Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. Gemma 3 technical report, 2025a. URL https://arxiv.org/abs/2503.19786.

V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Jiazheng Xu, Jiale Zhu, Jiali Chen, Jing Chen, Jinhao Chen, Jinghao Lin, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Sheng Yang, Shi Zhong, Shiyu Huang, Shuyuan Zhao, Siyan Xue, Shangqin Tu, Shengbiao Meng, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wenkai Li, Wei Jia, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyue Fan, Xuancheng Huang, Yanling Wang, Yadong Xue, Yanfeng Wang, Yanzi Wang, Yifan An, Yifan Du, Yiming Shi, Yiheng Huang, Yilin Niu, Yuan Wang, Yuanchang Yue, Yuchen Li, Yutao Zhang, Yuting Wang, Yu Wang, Yuxuan Zhang, Zhao Xue, Zhenyu Hou, Zhengxiao Du, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning, 2025b. URL https://arxiv.org/abs/2507.01006.

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

Taylor Webb, Keith J Holyoak, and Hongjing Lu. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9):1526–1541, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. Association for Computational Linguistics, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL https://arxiv.org/abs/2210.03629.

Michihiro Yasunaga, Xinyun Chen, Yujia Li, Panupong Pasupat, Jure Leskovec, Percy Liang, Ed H Chi, and Denny Zhou. Large language models as analogical reasoners. *arXiv preprint arXiv:2310.01714*, 2023.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. On the paradox of learning to reason from data. *arXiv preprint arXiv:2205.11502*, 2022.