Efficient Adaptation in Mixed-Motive Environments via Hierarchical Opponent Modeling and Planning

Yizhe Huang [1 2] Anji Liu [3] Fanqi Kong [2 4] Yaodong Yang [1 2] Song-Chun Zhu [1 2 5] Xue Feng✉ [2]

Abstract

Despite the recent successes of multi-agent reinforcement learning (MARL) algorithms, efficiently adapting to co-players in mixed-motive environments remains a significant challenge. One feasible approach is to hierarchically model co-players' behavior based on inferring their characteristics. However, these methods often encounter difficulties in efficient reasoning and utilization of inferred information. To address these issues, we propose Hierarchical Opponent modeling and Planning (HOP), a novel multi-agent decision-making algorithm that enables few-shot adaptation to unseen policies in mixed-motive environments. HOP is hierarchically composed of two modules: an opponent modeling module that infers others' goals and learns corresponding goal-conditioned policies, and a planning module that employs Monte Carlo Tree Search (MCTS) to identify the best response. Our approach improves efficiency by updating beliefs about others' goals both across and within episodes and by using information from the opponent modeling module to guide planning. Experimental results demonstrate that in mixed-motive environments, HOP exhibits superior few-shot adaptation capabilities when interacting with various unseen agents, and excels in self-play scenarios. Furthermore, the emergence of social intelligence during our experiments underscores the potential of our approach in complex multi-agent environments.

[1]Institute for Artificial Intelligence, Peking University [2]State Key Laboratory of General Artificial Intelligence, BIGAI [3]University of California, Los Angeles [4]Tsinghua University [5]PKU-WUHAN Institute for Artificial Intelligence. Correspondence to: Xue Feng <fengxue@bigai.ai>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Constructing agents being able to rapidly adapt to previously unseen agents is a longstanding challenge for Artificial Intelligence. We refer to this ability as few-shot adaptation. Previous work has proposed well-performed MARL algorithms to study few-shot adaptation in zero-sum games (Vinyals et al., 2019; Vezhnevets et al., 2020) and common-interest environments (Barrett et al., 2011; Hu et al., 2020; Mahajan et al., 2022; Mirsky et al., 2022; Bauer et al., 2023). These environments involve a predefined competitive or cooperative relationship between agents. However, the majority of realistic multi-agent decision-making scenarios are not confined to these situations and should be abstracted as mixed-motive environments (Komorita & Parks, 1995; Dafoe et al., 2020), where the relationships between agents are non-deterministic, and the best responses of an agent may change with others' behavior. A policy, that is unable to quickly adapt to co-players, may harm not only the focal agent's interest but also the entire group's benefit. Therefore, fast adapting to new co-players in mixed-motive environments warrants significant attention, but there has been little focus on this aspect.

In this paper, we focus on the few-shot adaptation to unseen agents in mixed-motive environments. Many algorithms struggle to perform well in mixed-motive environments despite success in zero-sum and pure-cooperative environments, because they use efficient techniques specific to reward structures, such as minimax (Littman, 1994; Li et al., 2019), Double Oracle (McMahan et al., 2003; Balduzzi et al., 2019) or IGM condition (Sunehag et al., 2017; Son et al., 2019; Rashid et al., 2020), which are not applicable in mixed-motive environments. The non-deterministic relationships between agents and the general-sum reward structure make decision-making and few-shot adaptation more challenging in mixed-motive environments compared with zero-sum and pure-cooperative environments.

According to cognitive psychology and related disciplines, humans' ability to rapidly solve previously unseen problems depends on hierarchical cognitive mechanisms (Butz & Kutter, 2016; Kleiman-Weiner et al., 2016; Eppe et al., 2022). This hierarchical structure unifies high-level goal reasoning with low-level action planning. Meanwhile, research on machine learning also emphasizes the importance and effectiveness of hierarchical goal-directed planning for few-shot problem-solving (Eppe et al., 2022). Inspired by the hierarchical structure, we propose an algorithm, named Hierarchical Opponent modeling and Planning (HOP), for tackling few-shot adaptation in mixed-motive environments. HOP hierarchically consists of two modules: an opponent modeling module and a planning module. The opponent modeling module infers co-players' goals and learns their goal-conditioned policies, based on Theory of Mind (ToM) - the ability to understand others' mental states (like goals and beliefs) from their actions (Baker et al., 2017). More specifically, to improve inference efficiency, beliefs about others' goals are updated both between and within episodes. Then, the information from the opponent modeling module is sent to the planning module, which is based on Monte Carlo Tree Search (MCTS), to compute the next action.

To assess the few-shot adaptation ability of HOP, we conduct experiments in Markov Stag-Hunt (MSH) and Markov Snowdrift Game (MSG), which spatially and temporally extend two classic paradigms in game theory: the Stag-Hunt game (Rousseau, 1999) and the Snowdrift game(also known as the game of chicken or hawk-dove game) (Rapoport & Chammah, 1966). Both of the two games illustrate how the best response in a mixed-motive environment is influenced by the strategy of co-players. Experimental results illustrate that in these environments, HOP exhibits superior few-shot adaptation ability compared with baselines, including the well-established MARL algorithms LOLA, social influence, A3C, prosocial-A3C, PR2, and a model-based algorithm direct-OM. Meanwhile, HOP achieves high rewards in self-play, showing its exceptional decision-making ability in mixed-motive games. In addition, we observe the emergence of social intelligence from the interaction between multiple HOP agents, such as self-organized cooperation and alliance of the disadvantaged.

2. Related Work

MARL has explored multi-agent decision-making in mixed-motive games. One approach is to add intrinsic rewards to incentivize collaboration and consideration of the impact on others, alongside maximizing extrinsic rewards. Notable examples include ToMAGA (Nguyen et al., 2020), MARL with inequity aversion (Hughes et al., 2018), and prosocial MARL (Peysakhovich & Lerer, 2018). However, many of these algorithms rely on hand-crafted intrinsic rewards and assume access to rewards of co-players, which can make them exploitable by self-interested algorithms and less effective in realistic scenarios where others' rewards are not visible (Komorita & Parks, 1995). To address these issues, Jaques et al. (2019) have included intrinsic social influence reward that use counterfactual reasoning to assess the effect of an agent's actions on its co-players' behavior.

LOLA (Foerster et al., 2018) and its extension (such as POLA (Zhao et al., 2022), M-FOS (Lu et al., 2022)) consider the impact of one agent's learning process, rather than treating them as a static part of the environment. However, LOLA requires knowledge of co-players' network parameters, which may not be feasible in many scenarios. LOLA with opponent modeling relaxes this requirement, but scaling problems may arise in complex sequential environments that require long action sequences for rewards.

Our work relates to opponent modeling (see (Albrecht & Stone, 2018) for a comprehensive review). I-POMDP (Gmytrasiewicz & Doshi, 2005) is a typical opponent modeling and planning framework, which maintains dynamic beliefs over the physical environment and beliefs over co-players' beliefs. It maximizes a value function of the beliefs to determine the next action. However, the nested belief inference suffers from serious computational complexity problems, which makes it impractical in complex environments. Unlike I-POMDP and its approximation methods (Doshi & Perez, 2008; Doshi & Gmytrasiewicz, 2009; Hoang & Low, 2013; Han & Gmytrasiewicz, 2018; 2019; Zhang & Doshi, 2022), HOP explicitly uses beliefs over co-players' goals and policies to learn a neural network model of co-players, which guides an MCTS planner to compute next actions. HOP avoids nested belief inference and performs sequential decision-making more efficiently.

Theory of mind (ToM), originally a concept of cognitive science and psychology (Baron-Cohen et al., 1985), has been transformed into computational models over the past decade and used to infer agents' mental states such as goals and desires. Bayesian inference has been a popular technique used to make ToM computational (Baker et al., 2011; Pöppel & Kopp, 2018; Wu et al., 2021; Zhi-Xuan et al., 2022). With the rapid development of the neural network, some recent work has attempted to achieve ToM using neural networks (Rabinowitz et al., 2018; Shu & Tian, 2018; Wen et al., 2019; Moreno et al., 2021). HOP gives a practical and effective framework to utilize ToM, and extend its application scenarios to mixed-motive environments, where both competition and cooperation are involved and agents' goals are private and volatile.

Monte Carlo Tree Search (MCTS) is a widely adopted planning method for optimal decision-making. Recent work, such as AlphaZero (Silver et al., 2018) and MuZero (Schrittwieser et al., 2020) have used MCTS as a general policy improvement operator over the base policy learned by neural networks. However, MCTS is limited in multi-agent environments, where the joint action space grows rapidly with the number of agents (Choudhury et al., 2022). We avoid this problem by estimating the policies of co-players and planning only for the focal agent's actions.

BAMDP (Duff, 2002) is a principled framework for handling uncertainty in dynamic environments. It maintains a posterior distribution over the transition probabilities, which is updated using Bayes' rule as new data becomes available. Several algorithms (Guez et al., 2012; Zintgraf et al., 2019; Rigter et al., 2021) have been developed based on BAMDP, but they are designed for single-agent environments. BA-MCP (Guez et al., 2012) employs the Monte Carlo Tree Search (MCTS) method to provide a sample-based approach grounded in BAMDP. However, it assumes a fixed transition function distribution to be learned interactively, posing challenges in multi-agent scenarios due to the co-player's strategy under an unknown distribution. (Ng et al., 2012) combines BAMDP with I-POMDP in an attempt to address multi-agent problems. However, this integration introduces computational complexity issues similar to those of I-POMDP, as previously discussed. In contrast, HOP efficiently handles both reward and transition uncertainties, and extends MCTS to multi-agent scenarios, offering a scalable solution for multi-agent environments.

Numerous real-world scenarios, including autonomous driving, human-machine interaction and multi-player sports, can be effectively modeled as mixed-motive games. Existing research (Fisac et al., 2018; Nakamura & Bansal, 2023; Hu et al., 2023) has explored planning and controlling robots in these real multi-agent environments, relying on predictions of other agents' behavior within the scene. These studies primarily concentrate on robot control within specific scenarios. In contrast, our environment abstracts the mixed motivation factors inherent in these scenarios, enabling representation of a broader range of scenarios and facilitating the development of more general algorithms. We believe HOP holds significant potential for application in various real-life scenarios.

3. Problem Formulation

We consider multi-agent hierarchical decision-making in mixed-motive environments, which can be described as a Markov game (Littman, 1994) with goals, specified by a tuple $< N, S, \mathbf{A}, T, \mathbf{R}, \gamma, T_{max}, \mathbf{G} >$.

Here, agent $i \in N = \{1, 2, \cdots, n\}$ chooses action from action space $A_i = \{a_i\}$. $\mathbf{A} = A_1 \times A_2 \times \cdots \times A_n$ is the joint action space. The joint action $\boldsymbol{a}_{1:n} \in \mathbf{A}$ will lead to a state transition based on the transition function $T : S \times \mathbf{A} \times S \to [0, 1]$. Specifically, after agents take the joint action $\boldsymbol{a}_{1:n}$ the state of the environment will transit from s to s' with probability $T(s'|s, \boldsymbol{a}_{1:n})$. The reward function $R_i : S \times \mathbf{A} \to \mathbb{R}$ denotes the immediate reward received by agent i after joint action $\boldsymbol{a}_{1:n}$ is taken on state $s \in S$. The discount factor for future rewards is denoted as γ. T_{max} is the maximum length of an episode. $\pi_i : S \times A_i \to [0, 1]$ denotes agent i's policy, specifying the probability $\pi_i(a_i|s)$ that agent i chooses action a_i at state s.

The environments we study have a set of goals, denoted by $\mathbf{G} = G_1 \times G_2 \times \cdots \times G_n$, where $G_i = \{g_{i,1}, \cdots, g_{i,|G_i|}\}$ represents the set of goals for agent i. $g_{i,k}$ is a set of states, where $g_{i,k} \cap g_{i,k'} = \emptyset, \forall\ k \neq k'$. We would say agent i's goal is g_{i,k_0} at time t, if $\exists t' \geq 0, s^{t+t'} \in g_{i,k_0}$ and $\forall\ 0 \leq t'' < t', 0 \leq k \leq |G_i|, s^{t+t''} \notin g_{i,k}$. For any two agents i and j, i can infer j's goal based on its trajectory. Specifically, i maintains a belief over j's goals, $b_{ij} : G_j \to [0, 1]$, which is a probability distribution over G_j.

Here, algorithms are evaluated in terms of self-play and few-shot adaptation to unseen policies in mixed-motive environments. Self-play involves multiple agents using the same algorithm to undergo training from scratch. The performance of algorithms in self-play is evaluated by their expected reward after convergence. Self-play performance demonstrates the algorithm's ability to make autonomous decisions in mixed-motive environments. Few-shot adaptation refers to the capability to recognize and respond appropriately to unknown policies within a limited number of episodes. The performance of algorithms in few-shot adaptation is measured by the rewards they achieve after engaging in these brief interactions.

4. Methodology

In this section, we propose **H**ierarchical **O**pponent modeling and **P**lanning (HOP), a novel algorithm for multi-agent decision-making in mixed-motive environments. HOP consists of two main modules: an opponent modeling module to infer co-players' goals and predict their behavior and a planning module to plan the focal agent's best response guided by the inferred information from the opponent modeling module.

Based on the hypothesis in cognitive psychology that agents' behavior is goal-directed (Gergely et al., 1995; Buresh & Woodward, 2007), and that agents behave stably for a specific goal (Warren, 2006), the opponent modeling module models behavior of co-players with two levels of hierarchy. At the high-level, the module infers co-players' internal goals by analyzing their action sequences. Based on the inferred goals and the current state of the environment, the low-level component learns goal-conditioned policies to model the atomic actions of co-players.

In the planning module, MCTS is used to plan for the best response of the focal agent based on the inferred co-players' policies. To handle the uncertainty over co-players' goals, we sample multiple goal combinations of all co-players from the current belief and return the action that maximizes the average return over the sampled configurations. Following AlphaZero (Silver et al., 2018) and MuZero (Schrittwieser et al., 2020), we maintain a policy and a value network to

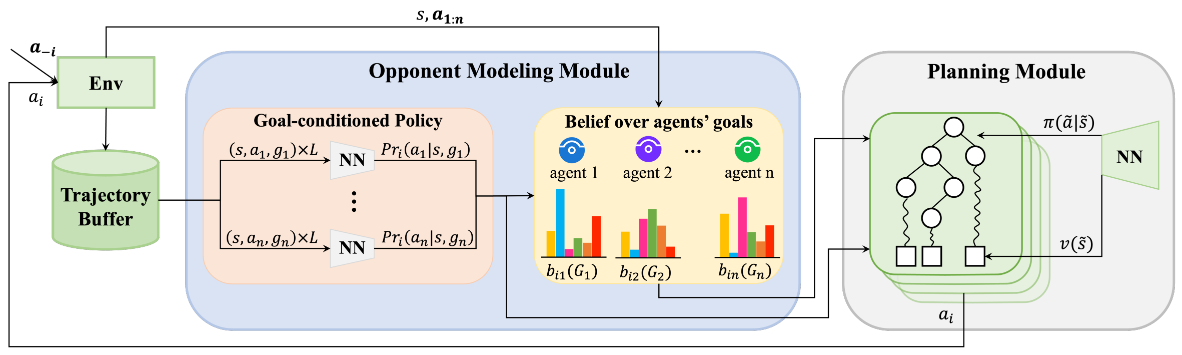


Figure 1. Overview of HOP. HOP consists of an opponent modeling module and a planning module. The opponent modeling module models the behavior of co-players by inferring co-players' goals and learning their goal-conditioned policies. Estimated behavior is then fed to the planning module to select a rewarding action for the focal agent.

boost MCTS planning and in turn use the planned action and its value to update the neural network.

Figure 1 gives an overview of HOP, and the pseudo-code of HOP is provided in Appendix A.

4.1. Opponent Modeling with Efficient Adaptation

In goal-inference (as the light yellow component shown in Figure 1), HOP summarizes the co-players' objectives based on the interaction history. However, it faces the challenge of the co-player's goals potentially changing within episodes. To solve these issues, we propose two update procedures based on ToM: intra opponent modeling (intra-OM), which infers the co-player's immediate goals within a single episode, and inter opponent modeling (inter-OM), which summarizes the co-player's goals based on their historical episodes. Intra-OM reasons about the goal of co-player j in the current episode K according to j's past trajectory in episode K. It ensures that HOP is able to quickly respond to in-episode behavior changes of co-players. Specifically, in episode K, agent i's belief about agent j's goals at time t, $b_{ij}^{K,t}(g_j)$, is updated according to:

$$\begin{aligned} b_{ij}^{K,t+1}(g_j) &= Pr(g_j\,|s^{K,0:t+1}, a_j^{K,0:t}) \\ &= Pr(g_j|s^{K,0:t}, a_j^{K,0:t-1}) \\ &\quad \cdot Pr_i(a_j^{K,t}|s^{K,0:t}, a_j^{K,0:t-1}, g_j) \\ &\quad \cdot \frac{Pr(s^{K,t+1}|s^{K,0:t}, a_j^{K,0:t}, g_j)}{Pr_i(s^{K,t+1}, a_j^{K,t}|s^{K,0:t}, a_j^{K,0:t-1})} \\ &= \frac{1}{Z_1} b_{ij}^{K,t}(g_j) Pr_i(a_j^{K,t}|s^{K,t}, g_j), \end{aligned} \tag{1}$$

where we follow the Markov assumption $Pr(s^{K,t+1}|s^{K,0:t}, a_j^{K,0:t}, g_j) = Pr(s^{K,t+1}|s^{K,t}, a_j^{K,t})$ and model the co-player j to maintain a Markov policy $Pr_i(a_j^{K,t}|s^{K,t}, g_j) = Pr_i(a_j^{K,t}|s^{K,0:t}, g_j)$, and $Z_1 = \frac{Pr_i(s^{K,t+1}, a_j^{K,t}|s^{K,t})}{Pr(s^{K,t+1}|s^{K,t}, a^{K,t})}$ is the normalization factor that makes $\sum_{g_j \in G_j} b_{ij}^{K,t+1}(g_j) = 1$. The likelihood term $Pr_i(a_j^{K,t}|s^{K,t}, g_j)$ is provided by the estimated goal-conditioned policies of co-players, which are described in the following.

However, intra-OM may suffer from inaccuracy of the prior (i.e., $b_{ij}^{K,0}(g_j)$) when past trajectories are not long enough for updates. Inter-OM makes up for this by calculating a precise prior based on past episodes. Belief update between two adjacent episodes is defined as:

$$b_{ij}^{K,0}(g_j) = \frac{1}{Z_2}[\alpha b_{ij}^{K-1,0}(g_j) + (1-\alpha)\mathbf{1}(g_j^{K-1} = g_j)], \tag{2}$$

where $\alpha \in [0, 1]$ is the horizon weight, which controls the importance of the history. As α decreases, agents attach greater importance to recent episodes. $\mathbf{1}(\cdot)$ is the indicator function. Z_2 is the normalization factor. The equation is equivalent to a time-discounted modification of the Monte Carlo estimate. Inter-OM summarizes co-players' goals according to all the previous episodes, which is of great help when playing with the same agents in a series of episodes.

The goal-conditioned policy (as the light orange component shown in Figure 1) $\pi_{\boldsymbol{\omega}}(a_j^{K,t}|s^{K,t}, g_j)$ is obtained through a neural network $\boldsymbol{\omega}$. To train the network, a set of $(s^{K,t}, a_j^{K,t}, g_j^{K,t})$ is collected from episodes and sent to the replay buffer. $\boldsymbol{\omega}$ is updated at intervals to minimize the negative log-likelihood:

$$L(\boldsymbol{\omega}) = \mathbb{E}[-\log(\pi_{\boldsymbol{\omega}}(a_j^{K,t}|s^{K,t}, g_j^{K,t}))]. \tag{3}$$

4.2. Planning under Uncertain Co-player Models

Given the policies of co-players estimated by the opponent modeling module, we can leverage planning algorithms such as MCTS to compute an advantageous action. However, a key obstacle to applying MCTS is that co-player policies estimated by the opponent modeling module contain uncertainty over co-players' goals. Naively adding such uncertainty as part of the environment would add a large bias to the simulation and degrade planning performance. To overcome this problem, we propose to sample co-players' goal combinations according to the belief maintained by the opponent modeling module, and then estimate action value by MCTS based on the samples. To balance the trade-off between computational complexity and planning performance, we repeat the process multiple times and choose actions according to the average action value. In the following, we first introduce the necessary background of MCTS. We then proceed to introduce how we plan for a rewarding action under the uncertainty over co-player policies.

MCTS Monte Carlo Tree Search (MCTS) is a type of tree search that plans for the best action at each time step (Silver & Veness, 2010; Liu et al., 2020). MCTS uses the environment to construct a search tree (right side of Figure 1) where nodes correspond to states and edges refer to actions. Specifically, each edge transfers the environment from its parent state to its child state. MCTS expands the search tree in ways (such as pUCT) that properly balance exploration and exploitation. Value and visit of every state-action (node-edge) pair are recorded during expansion (Silver et al., 2016). Finally, the action with the highest value (or highest visit) of the root state (node) is returned and executed in the environment.

Planning under uncertain co-player policies Based on beliefs over co-players' goals and their goal-conditioned policies from the opponent modeling module, we run MCTS for N_s rounds. In each round, co-players' goals are sampled according to the focal agent's belief over co-players' goals $b_{ij}(g_j)$. Specifically, at time t in episode K, we sample the goal combination $\mathbf{g}_{-i} = \{g_j \sim b_{ij}^{K,t}(\cdot), j \neq i\}$. Then at every state $\tilde{s}^k$ in the MCTS tree of this round, co-players' actions $\tilde{\mathbf{a}}_{-i}$ are determined by $\tilde{\mathbf{a}}_{-i} \sim \pi_{\boldsymbol{\omega}}(\cdot|\tilde{s}^k, \mathbf{g}_{-i})$ from the goal-conditioned policy.

In each round, MCTS gives the estimated action value of the current state $Q(s^{K,t}, a, \mathbf{g}_{-i}) = V(\tilde{s'}(a))$ $(a \in A_i)$, where $\tilde{s'}(a)$ is the next state after taking $\tilde{\mathbf{a}}_{-i}^0 \cup a$ from $\tilde{s}^0 = s^{K,t}$.

We average the estimated action value from MCTS in all N_s rounds:

$$Q_{avg}(s^{K,t}, a) = \sum\nolimits_{l=1}^{N_s} Q_l(s^{K,t}, a, \mathbf{g}_{-i}^l). \tag{4}$$

Agent i's policy follows Boltzmann rationality model (Baker et al., 2017):

$$\pi_{MCTS}(a|s^{K,t}) = \frac{\exp(\beta Q_{avg}(s^{K,t}, a))}{\sum_{a' \in A_i} \exp(\beta Q_{avg}(s^{K,t}, a'))}, \tag{5}$$

where $\beta \in [0, \infty)$ is rationality coefficient. As β increases, the policy gets more rational. We choose our action at time t of the episode K based on $\pi_{MCTS}(a|s^{K,t})$.

Note that the effectiveness of MCTS is highly associated with the default policies and values provided to MCTS. When they are close to the optimal ones, they can offer an accurate estimate of state value, guiding MCTS search in the right direction. Therefore, following Silver et al. (2018), we train a neural network $\boldsymbol{\theta}$ to predict the policy and value functions at every state following the supervision provided by MCTS. Specifically, the policy target is the policy generated by MCTS, while the value target is the true discounted return of the state in this episode.

As for state $\tilde{s}^k$ in the MCTS, the policy function provides a prior distribution over actions $\pi_{\boldsymbol{\theta}}^k(\cdot|\tilde{s}^k)$. Actions with high prior probabilities are assigned high pUCT scores, prioritizing their exploration during the search process. However, as the exploration progresses, the influence of this prior gradually diminishes (see details in Appendix E.1). The value function $v_{\boldsymbol{\theta}}^k$ estimates the return and provides the initial value of $\tilde{s}^k$ when $\tilde{s}^k$ is first reached.

The network $\boldsymbol{\theta}$ is updated based on the overall loss:

$$L(\boldsymbol{\theta}) = L_p(\pi_{MCTS}, \pi_{\boldsymbol{\theta}}) + L_v(r_i, v_{\boldsymbol{\theta}}), \tag{6}$$

where

$$L_p(\pi_1, \pi_2) = \mathbb{E}[-\sum\nolimits_{a \in A_i} \pi_1(a|s^{K,t}) \log(\pi_2(a|s^{K,t}))],$$

$$L_v(r_i, v) = \mathbb{E}[(v(s^{K,t}) - \sum\nolimits_{l=t}^{\infty} \gamma^{l-t} r_i^{K,l})^2].$$

5. Experiments

5.1. Experimental Setup

Agents are tested in Markov Stag-Hunt (MSH) and Markov Snowdrift Game (MSG).

MSH expands the environment in Peysakhovich & Lerer (2018) in terms of the number of agents. In MSH, 4 agents are rewarded for hunting prey. As shown in Figure 2(a), each agent has six actions: idle, move left, move right, move up, move down, and hunt. If there are obstacles or boundaries in an agent's moving direction, its position stays unchanged. Agents can hunt prey in their current grid. There are two types of prey: stags and hares. A stag provides a reward of 10, and requires at least two agents located at its grid to execute "hunt" together. These cooperating agents will split the reward evenly. A hare, which an agent can catch alone, provides a reward of 1. After a successful hunt, both the hunters and the prey disappear from the environment. The game terminates when the timestep reaches $T_{max} = 30$.

We conducted experiments in two different settings of MSH. In the first setting, there are 4 hares and 1 stag (**MSH-4h1s**). In this scenario, agents can cooperate in hunting the stag to maximize their profits, while also competing with co-players for the opportunity to hunt. The second setting contains 4 hares and 2 stags (**MSH-4h2s**). There are sufficient stags for agents to cooperate, but the environment will end 5 timesteps after the first successful hunting in each episode. This setup maintains the tension between payoff-dominant cooperation and risk-dominant defection, highlighting the dilemma inherent in the Stag-Hunt game.

In **MSG** (Figure 2(b)), there are six snowdrifts located randomly in an 8×8 grid. Similar to MSH, at every time step the agent can stay idle or move one step in any direction. Agents are additionally equipped with a "remove a snowdrift" action, which removes the snowdrift in the same cell as the agent. When a snowdrift is removed, removers share the cost of 4 evenly, and every agent gets a reward of 6. The game ends when all the snowdrifts are removed or the time $T_{max} = 50$ runs out. The game's essential dilemma arises from the fact that an agent can obtain a higher reward by free-riding, i.e., waiting for co-players to remove the snowdrifts, than by removing a snowdrift themselves. However, if all agents take free rides, no snowdrift is removed, and agents will not receive any reward. On the other hand, if any agent is satisfied with a suboptimal strategy and chooses to remove snowdrifts, both the group benefit and individual rewards increase.

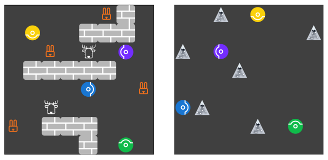
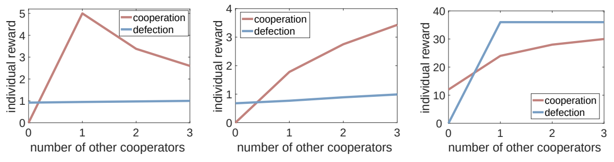

(a) MSH (b) MSG

Figure 2. Overview of Markov Stag-Hunt and Markov Snowdrift. There are four agents, represented by colored circles, in each paradigm. (a) Agents catch prey for reward. A stag with a reward of 10 requires at least two agents to hunt together. One agent can hunt a hare with a reward of 1. (b) Everyone gets a reward of 6 when an agent removes a snowdrift. When a snowdrift is removed, removers share the cost of 4 evenly.

In both environments, four agents have no access to each other's parameters, and communication is not allowed. Appendix C introduces the goal definition of these games.

Schelling diagrams Game types are determined by the relative values of elements in the payoff matrix. The Schelling diagram (Schelling, 1973; Hughes et al., 2018) is a natural generalization of the payoff matrix for two-player games to multi-player settings. As shown in Figure 3, Schelling diagrams validate our temporal and spatial extension of the matrix-form games, which maintains the dilemmas described by matrix-form games (see a detailed discussion in Appendix D). Moreover, across these three Schelling diagrams, the lines of cooperation and defection intersect. This implies that best responses change with co-players' behavior, rendering few-shot adaptation in these environments inherently challenging.



(a) MSH-4h1s (b) MSH-4h2s (c) MSG

Figure 3. Schelling diagrams for (a) MSH-4h1s, (b) MSH-4h2s, and (c) MSG.

Baselines Here, some baseline algorithms are introduced to evaluate the performance of HOP. During the evaluation of few-shot adaptation, baseline algorithms serve a dual purpose. Firstly, they act as unfamiliar co-players during the evaluation process to test the few-shot adaptation ability of HOP. Secondly, we evaluate the few-shot adaptation ability of the baseline algorithms to demonstrate HOP's superiority. *LOLA* (Foerster et al., 2018; Zhao et al., 2022) agents consider a 1-step look-ahead update of co-players, and update their own policies according to the updated policies of co-players. *SI* (Jaques et al., 2019) agents have an intrinsic reward term that incentivizes actions maximizing their influence on co-players' actions. The influence is accessed by counterfactual reasoning. *A3C* (Mnih et al., 2016) agents are trained using the Asynchronous Advantage Actor-Critic method, a well-established reinforcement learning (RL) technique. *Prosocial-A3C* (PS-A3C) (Peysakhovich & Lerer, 2018) agents are trained using A3C but share rewards between players during training, so they optimize the per-capita reward instead of the individual reward, emphasizing cooperation between players. *PR2* (Wen et al., 2019) agents model how the co-players would react to their potential behavior, based on which agents find the best response. The ablated version of HOP, *direct-OM*, retains the planning module, but uses neural networks to model co-players directly (see details in Appendix F.2). In addition, we construct some rule-based strategies. *Random* policy takes a valid action randomly at each step. An agent that consistently adopts cooperative behavior is called *cooperator*, and an agent that consistently adopts exploitative behavior is called *defector*. In MSH, the goals of cooperators and defectors are hunting the nearest stag and hare, respectively. In MSG, cooperators keep moving to remove the nearest snowdrift, and defectors randomly take actions other than "remove a snowdrift". When evaluating few-shot adaptation, the set of unfamiliar co-players includes LOLA, A3C, and PS-A3C, serving as representatives of learning agents with explicit opponent modeling module, self-interest purpose, and prosocial purpose, respectively. The co-players also include rule-based agents: random, cooperator and defector.

5.2. Performance

The experiment consists of two phases. The first phase focuses on self-play, where agents using the same algorithm are trained until convergence. Self-play performance, showing the ability to achieve cooperation, is measured by the algorithm's average reward after convergence. The second phase evaluates the few-shot adaptation ability of HOP. Specifically, a focal agent interacts with three co-players using a different algorithm for 2400 steps. The focal agent's average reward during the final 600 steps is used to measure its algorithm's few-shot adaptation ability. At the start of the adaptation phase, any policy's parameters are the convergent parameters derived from the corresponding algorithms in self-play. During this phase, policies can update their parameters if possible. Implementation details are given in Appendix E. The results of self-play and few-shot adaptation are displayed in Figure 4 and Table 1, respectively.

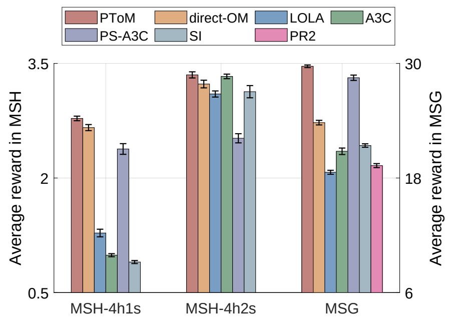
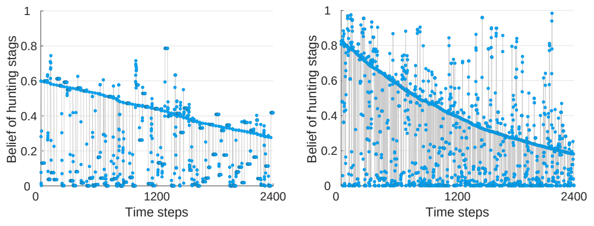


Figure 4. Self-play performance of HOP and baseline algorithms. Shown is the average reward in the self-play training phase.

MSH-4h1s In MSH-4h1s, only HOP, direct-OM, and PS-A3C learn the strategy of hunting stags (Figure 4). However, since PS-A3C can get rewards without hunting by itself, it may not effectively learn the relationship between hunting and receiving rewards, leading to a "lazy agent" problem (Sunehag et al., 2017) for PS-A3C. This results in the overall reward of PS-A3C being inferior to HOP and direct-OM. LOLA swings between hunting stags and hunting hares. SI and A3C primarily learn the strategy of hunting hares, resulting in low rewards. PR2 fails to work in MSH. In this environment, the number of agents may be reduced due to the successful hunting of agents, and this is not supported by PR2. Despite attempts to modify the algorithm accordingly, the modified version ultimately failed to learn a decent policy. As a result, the relevant results of PR2 in MSH are not shown in Figure 4 and Table 1.

HOP learns the stag hunting strategy through self-play, enabling seamless cooperation with agents like PS-A3C and cooperators, which similarly prioritize stag hunting (Table 1(a)). This compatibility stems from the fact that in the Stag-Hunt game, the best response of cooperation is cooperation. Thus, direct-OM and PS-A3C agents, who are equipped with learned cooperative strategies, also attain relatively high rewards when playing with cooperative co-players. When confronted with co-players with fluctuating strategies such as LOLA or random agents lacking fixed objectives, HOP seeks out opportunities for heightened returns through cooperation. Furthermore, when encountering co-players like A3C and defectors, known for their inclination towards hunting hares, HOP adjusts to these non-cooperative scenarios within a small amount of interaction. HOP and direct-OM achieve substantially greater rewards when confronting defectors compared to PS-A3C, who also favors cooperation. This observation highlights the pivotal role of the planning module in efficient adaptation.



(a) MSH-4h1s (b) MSH-4h2s

Figure 5. Visualization of HOP's belief in adaptation to three defectors in MSH. Every blue-filled circle represents HOP's inferred probability (i.e., belief) that a co-player hunts stags.

MSH-4h2s As depicted in Figure 4, in MSH-4h2s, all algorithms have learned the strategy of cooperatively hunting stags, among which HOP and A3C are more stable and yield higher returns. PS-A3C tends to delay hunting, as early hunting results in leaving the environment and failing to obtain the group reward from subsequent hunting. This may lead PS-A3C to suboptimal actions in the last few steps and thus fail to hunt under the 5-step termination rule.

The adaptation performance in MSH-4h2s is presented in Table 1(b). When facing with the cooperator, the best response is to hunt stags, which requires minimal adjustments to each algorithm's policies, so their returns are comparable to the Orcale reward. Similarly, when encountering learning co-players who have adopted the cooperation policy, HOP and most baselines yield high rewards. However, given that learning agents may dynamically adjust their goals, it becomes essential to discern the real-time goals of the co-players in order to find the best response. In these scenarios, HOP's performance surpasses that of other algorithms, approaching the Orcale reward. When playing with non-cooperative co-players such as random and defectors, significant strategy adjustments are necessary for each algorithm to achieve high returns. Therefore, the returns for all algorithms are notably diminished. HOP demonstrates superior adaptability compared to the other algorithms, exhibiting its ability to make substantial strategic adjustments.

We would like to provide further intuition on why HOP is capable of efficiently adapting its policy to unseen agents.

Table 1. Few-shot adaptation performance of HOP and baselines in (a) MSH-4h1s, (b) MSH-4h2s, and (c) MSG. The interaction happens between 1 agent using the row policy and 3 co-players using the column policy. Shown are the min-max normalized rewards, with normalization bounds set by the rewards of Orcale and the lowest rewards among all baselines and random policy. See detailed description and analysis of Orcale in Appendix F.1. The results are depicted for the row policy from 1800 to 2400 step. Overall best adaptation percentage shows the proportion of scenarios in which the algorithm performs optimally, while accounting for standard deviation.

(a) Performance in MSH-4h1s

	learning co-players			rule-based co-players			Overall best adaptation percentage
	LOLA	A3C	PS-A3C	random	cooperator	defector	
HOP	**0.97**± 0.06	**0.80**± 0.15	**0.93**± 0.04	**0.96**± 0.07	**0.74**± 0.06	0.51± 0.02	**83.3%**
direct-OM	0.64± 0.05	0.57± 0.07	0.79± 0.04	**0.91**± 0.10	0.56± 0.04	0.44± 0.04	16.7%
LOLA	-	0.55± 0.07	0.38± 0.04	0.45± 0.06	0.46± 0.03	0.41± 0.02	0.0%
A3C	0.35± 0.01	-	0.24± 0.02	0.85± 0.01	0.31± 0.02	**1.00**± 0.01	20.0%
PS-A3C	0.85± 0.05	0.60± 0.11	-	0.64± 0.06	0.55± 0.05	0.09± 0.04	0.0%
SI	0.32± 0.01	**0.95**± 0.04	0.22± 0.02	0.81± 0.01	0.28± 0.02	0.89± 0.04	16.7%

(b) Performance in MSH-4h2s

	learning co-players			rule-based co-players			Overall best adaptation percentage
	LOLA	A3C	PS-A3C	random	cooperator	defector	
HOP	**0.97**± 0.02	**0.99**± 0.02	**0.88**± 0.02	**0.78**± 0.07	**1.00**± 0.01	**0.36**± 0.02	**100.0%**
direct-OM	**0.95**± 0.01	0.85± 0.02	0.74± 0.03	0.62± 0.04	0.96± 0.02	0.31± 0.02	16.7%
LOLA	-	0.92± 0.04	0.82± 0.02	**0.75**± 0.04	**1.00**± 0.03	0.28± 0.03	40.0%
A3C	0.91± 0.02	-	**0.87**± 0.02	0.55± 0.05	**0.98**± 0.02	0.25± 0.02	40.0%
PS-A3C	0.24± 0.03	0.18± 0.02	-	0.29± 0.02	0.38± 0.01	0.06± 0.02	0.0%
SI	0.77± 0.02	0.83± 0.01	0.74± 0.01	0.52± 0.03	0.87± 0.03	0.27± 0.02	0.0%

(c) Performance in MSG

	learning co-players			rule-based co-players			Overall best adaptation percentage
	LOLA	A3C	PS-A3C	random	cooperator	defector	
HOP	**0.78**± 0.04	**0.39**± 0.09	**0.65**± 0.08	**0.44**± 0.03	0.48± 0.05	**0.55**± 0.01	**83.3%**
direct-OM	0.31± 0.11	0.12± 0.05	**0.55**± 0.04	**0.38**± 0.04	**0.67**± 0.05	0.34± 0.05	50.0%
LOLA	-	**0.33**± 0.07	**0.55**± 0.06	0.25± 0.08	0.43± 0.04	0.18± 0.01	40.0%
A3C	0.33± 0.04	-	**0.52**± 0.09	0.30± 0.04	0.33± 0.03	0.14± 0.01	20.0%
PS-A3C	0.67± 0.05	**0.35**± 0.04	-	0.33± 0.04	0.00± 0.08	0.38± 0.02	20.0%
SI	**0.74**± 0.08	0.00± 0.05	0.33± 0.08	0.00± 0.04	0.24± 0.07	0.24± 0.03	16.7%
PR2	0.00± 0.13	0.00± 0.08	**0.58**± 0.05	0.16± 0.05	0.43± 0.02	0.14± 0.01	16.7%

Take the experiment facing three defectors (always attempting to hunt the nearest hare) as an example. There are two goals here: hunting stags or hunting hares. At the start of the evaluation phase, HOP holds the belief that every co-player is more likely to hunt a stag because HOP has seen its co-players hunt stags more than hares during self-play. This false belief for defectors degrades HOP's performance. Both intra-OM and inter-OM correct this false belief by updating during the intereactions with defectors (see visualization of belief update in Figure 5). Intra-OM provides the ability to correct the belief of hunting stags within an episode. Specifically, as a co-player keeps moving closer to a hare, intra-OM will update the belief of the co-player toward the goal "hare", leading to accurate opponent models. In Figure 5, there are many points with values near 0, showing that HOP infers that the agent's goal is unlikely to be a stag through intra-OM. Taking these accurate co-player policies as input, the planning module can output advantageous actions. Inter-OM further accelerates the convergence towards true belief by updating the inter-episode belief, which is used as a prior for intra-OM at the start of every episode. A declining line, formed by the points from initial steps of each episode, appears in both sub-figures of Figure 5, which reflects that HOP gradually reduces the prior of the co-player hunting a stag through inter-OM.

MSG As shown in Figure 4, HOP achieves the highest reward during self-play and it is close to the theoretically optimal average reward in this environment (i.e. when all snowdrifts are removed, resulting in a group average reward of 30.0). This outcome is a remarkable achievement in a fully decentralized learning setting and highlights the high propensity of HOP to cooperate. In contrast, LOLA, A3C, SI, and PR2 prioritize maximizing their individual profits, which leads to inferior outcomes due to their failure to coordinate and cooperate effectively. PS-A3C performs exceptionally well in self-play, ranking second only to HOP. Like in MSH, it fails to achieve the maximum average reward due to the coordination problem, which is prominent when only one snowdrift is left. This issue highlights the instability of the policy due to the absence of action planning.

HOP demonstrates the most effective few-shot adaptation performance (Table 1(c)). Specifically, when adapting to three defectors, HOP receives substantially higher rewards than other policies. This highlights the effectiveness of HOP in quickly adapting to non-cooperative behavior, which differs entirely from behavior of co-players in HOP's self-play. In contrast, A3C and PS-A3C do not explicitly consider co-players. They have learned the strategies tending to exploit and cooperate, respectively. Therefore, A3C performs effectively against agents that have a higher tendency to cooperate, such as the cooperator. However, its performance is relatively poor when facing non-cooperative agents. Conversely, PS-A3C exhibits the opposite behavior.

Overall, the above experiments demonstrate the remarkable adaptation ability of HOP across all environments (see last columns in Table 1). Other algorithms can only achieve the best adaptation performance when facing some specific co-players, to whom the best response is close to the policies learned by the algorithms in self-play. HOP can achieve the best adaptation level in most test scenarios, where co-players perform either familiar or completely unfamiliar behavior. Meanwhile, HOP exhibits advantages during self-play.

Ablation study indicates that inter-OM and intra-OM play crucial roles in adapting to agents with fixed goals and agents with dynamic goals, respectively. Moreover, if opponent modeling is not conditioned on goals, the self-play and few-shot adaptation abilities are greatly weakened. Further details are provided in Appendix F.2.

We observe the emergence of social intelligence, including self-organized cooperation and an alliance of the disadvantaged, during the interaction of multiple HOP agents in mixed-motive environments. Further details can be found in Appendix G.

6. Conclusion and Discussion

We propose Hierarchical Opponent modeling and Planning (HOP), a hierarchical algorithm for few-shot adaptation to unseen co-players in mixed-motive environments. It consists of an opponent modeling module for inferring co-players' goals and behavior and a planning module guided by the inferred information to output the focal agent's best response. Empirical results show that HOP performs better than state-of-the-art MARL algorithms, in terms of dealing with mixed-motive environments in the self-play setting and few-shot adaptation to previously unseen co-players.

Whilst HOP exhibits superior abilities, there are several limitations illumining our future work. First, in any environment, a clear definition of goals is needed for HOP. To enhance HOP's ability to generalize to various environments, a technique that can autonomously abstract goal sets in various scenarios is needed, which (Ashwood et al., 2022) has attempted to explore. Second, we use Level-0 ToM, which involves "think of what they think." However, a more complex form of ToM, such as Level-1 ToM that considers "what I think they think about me," has the potential to improve our predictions about co-players. Nevertheless, incorporating nested inference introduces a higher computational cost. Consequently, it becomes imperative to develop advanced planning methods that can effectively and rapidly leverage the insights provided by high-order ToM. Third, we investigate mix-motive environments with the expectation that HOP can facilitate effective decision-making and adaptation in human society. Despite selecting diverse well-established algorithms as co-players, none of them adequately model human behavior. It would be interesting to explore how HOP can perform in a few-shot adaptation scenario involving human participants. As HOP is self-interested, it may not always align with the best interest of humans. One way to mitigate this risk is leveraging HOP's ability to infer and optimize for human values and preferences during interactions, thereby assisting humans in complex environments.

Acknowledgements

This project is supported by the National Key R&D Program of China (2022ZD0114900). We would like to acknowledge the project "Wuhan East Lake High-Tech Development Zone (also known as the Optics Valley of China, or OVC) National Comprehensive Experimental Base for Governance of Intelligent Society".

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

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

Ashwood, Z., Jha, A., and Pillow, J. W. Dynamic inverse reinforcement learning for characterizing animal behavior. *Advances in Neural Information Processing Systems*, 35: 29663–29676, 2022.

Baker, C., Saxe, R., and Tenenbaum, J. Bayesian theory of mind: Modeling joint belief-desire attribution. In *Proceedings of the annual meeting of the cognitive science society*, volume 33, 2011.

Baker, C. L., Jara-Ettinger, J., Saxe, R., and Tenenbaum, J. B. Rational quantitative attribution of beliefs, desires and percepts in human mentalizing. *Nature Human Behaviour*, 1(4):1–10, 2017.

Balduzzi, D., Garnelo, M., Bachrach, Y., Czarnecki, W., Perolat, J., Jaderberg, M., and Graepel, T. Open-ended learning in symmetric zero-sum games. In *International Conference on Machine Learning*, pp. 434–443. PMLR, 2019.

Baron-Cohen, S., Leslie, A. M., and Frith, U. Does the autistic child have a "theory of mind"? *Cognition*, 21(1): 37–46, 1985.

Barrett, S., Stone, P., and Kraus, S. Empirical evaluation of ad hoc teamwork in the pursuit domain. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 567–574, 2011.

Bauer, J., Baumli, K., Behbahani, F., Bhoopchand, A., Bradley-Schmieg, N., Chang, M., Clay, N., Collister, A., Dasagi, V., Gonzalez, L., et al. Human-timescale adaptation in an open-ended task space. In *International Conference on Machine Learning*, pp. 1887–1935. PMLR, 2023.

Bloembergen, D., De Jong, S., and Tuyls, K. Lenient learning in a multiplayer stag hunt. In *Proceedings of 23rd Benelux Conference on Artificial Intelligence (BNAIC 2011)*, pp. 44–50, 2011.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

Buresh, J. S. and Woodward, A. L. Infants track action goals within and across agents. *Cognition*, 104(2):287–314, 2007.

Butz, M. V. and Kutter, E. F. *How the mind comes into being: Introducing cognitive science from a functional and computational perspective*. Oxford University Press, 2016.

Choudhury, S., Gupta, J. K., Morales, P., and Kochenderfer, M. J. Scalable online planning for multi-agent mdps. *Journal of Artificial Intelligence Research*, 73:821–846, 2022.

Dafoe, A., Hughes, E., Bachrach, Y., Collins, T., McKee, K. R., Leibo, J. Z., Larson, K., and Graepel, T. Open problems in cooperative ai. *arXiv preprint arXiv:2012.08630*, 2020.

Doshi, P. and Gmytrasiewicz, P. J. Monte carlo sampling methods for approximating interactive pomdps. *Journal of Artificial Intelligence Research*, 34:297–337, 2009.

Doshi, P. and Perez, D. Generalized point based value iteration for interactive pomdps. In *AAAI*, pp. 63–68, 2008.

Duff, M. O. *Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes*. University of Massachusetts Amherst, 2002.

Eppe, M., Gumbsch, C., Kerzel, M., Nguyen, P. D., Butz, M. V., and Wermter, S. Intelligent problem-solving as integrated hierarchical reinforcement learning. *Nature Machine Intelligence*, 4(1):11–20, 2022.

Fisac, J. F., Bajcsy, A., Herbert, S. L., Fridovich-Keil, D., Wang, S., Tomlin, C. J., and Dragan, A. D. Probabilistically safe robot planning with confidence-based human predictions. In *14th Robotics: Science and Systems, RSS 2018*. MIT Press Journals, 2018.

Foerster, J., Chen, R. Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., and Mordatch, I. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 122–130, 2018.

Gergely, G., Nádasdy, Z., Csibra, G., and Bíró, S. Taking the intentional stance at 12 months of age. *Cognition*, 56 (2):165–193, 1995.

Gmytrasiewicz, P. J. and Doshi, P. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24:49–79, 2005.

Guez, A., Silver, D., and Dayan, P. Efficient bayes-adaptive reinforcement learning using sample-based search. *Advances in neural information processing systems*, 25, 2012.

Han, Y. and Gmytrasiewicz, P. Learning others' intentional models in multi-agent settings using interactive pomdps. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 5639–5647, 2018.

Han, Y. and Gmytrasiewicz, P. Ipomdp-net: A deep neural network for partially observable multi-agent planning using interactive pomdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 6062–6069, 2019.

Hoang, T. N. and Low, K. H. Interactive pomdp lite: towards practical planning to predict and exploit intentions for interacting with self-interested agents. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pp. 2298–2305, 2013.

Hu, H., Lerer, A., Peysakhovich, A., and Foerster, J. "other-play" for zero-shot coordination. In *International Conference on Machine Learning*, pp. 4399–4410. PMLR, 2020.

Hu, H., Zhang, Z., Nakamura, K., Bajcsy, A., and Fisac, J. F. Deception game: Closing the safety-learning loop in interactive robot autonomy. In Tan, J., Toussaint, M., and Darvish, K. (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3830–3850. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/hu23b.html.

Hughes, E., Leibo, J. Z., Phillips, M., Tuyls, K., Dueñez-Guzman, E., García Castañeda, A., Dunning, I., Zhu, T., McKee, K., Koster, R., et al. Inequity aversion improves cooperation in intertemporal social dilemmas. *Advances in neural information processing systems*, 31, 2018.

Jaques, N., Lazaridou, A., Hughes, E., Gulcehre, C., Ortega, P., Strouse, D., Leibo, J. Z., and De Freitas, N. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International conference on machine learning*, pp. 3040–3049. PMLR, 2019.

Kleiman-Weiner, M., Ho, M. K., Austerweil, J. L., Littman, M. L., and Tenenbaum, J. B. Coordinate to cooperate or compete: abstract goals and joint intentions in social interaction. In *CogSci*, 2016.

Komorita, S. S. and Parks, C. D. Interpersonal relations: Mixed-motive interaction. *Annual review of psychology*, 46(1):183–207, 1995.

Li, S., Wu, Y., Cui, X., Dong, H., Fang, F., and Russell, S. Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4213–4220, 2019.

Littman, M. L. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Liu, A., Chen, J., Yu, M., Zhai, Y., Zhou, X., and Liu, J. Watch the unobserved: A simple approach to parallelizing monte carlo tree search. *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020.

Lu, C., Willi, T., De Witt, C. A. S., and Foerster, J. Model-free opponent shaping. In *International Conference on Machine Learning*, pp. 14398–14411. PMLR, 2022.

Mahajan, A., Samvelyan, M., Gupta, T., Ellis, B., Sun, M., Rocktäschel, T., and Whiteson, S. Generalization in cooperative multi-agent systems. *arXiv preprint arXiv:2202.00104*, 2022.

McMahan, H. B., Gordon, G. J., and Blum, A. Planning in the presence of cost functions controlled by an adversary. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 536–543, 2003.

Mirsky, R., Carlucho, I., Rahman, A., Fosong, E., Macke, W., Sridharan, M., Stone, P., and Albrecht, S. V. A survey of ad hoc teamwork research. In *European Conference on Multi-Agent Systems*, pp. 275–293. Springer, 2022.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Moreno, P., Hughes, E., McKee, K. R., Pires, B. A., and Weber, T. Neural recursive belief states in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.02274*, 2021.

Nakamura, K. and Bansal, S. Online update of safety assurances using confidence-based predictions. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12765–12771. IEEE, 2023.

Ng, B., Boakye, K., Meyers, C., and Wang, A. Bayes-adaptive interactive pomdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pp. 1408–1414, 2012.

Nguyen, D., Venkatesh, S., Nguyen, P., and Tran, T. Theory of mind with guilt aversion facilitates cooperative reinforcement learning. In *Asian Conference on Machine Learning*, pp. 33–48. PMLR, 2020.

Peysakhovich, A. and Lerer, A. Prosocial learning agents solve generalized stag hunts better than selfish ones. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2043–2044, 2018.

Pöppel, J. and Kopp, S. Satisficing models of bayesian theory of mind for explaining behavior of differently uncertain agents: Socially interactive agents track. In *Proceedings of the 17th international conference on autonomous agents and multiagent systems*, pp. 470–478, 2018.

Rabinowitz, N., Perbet, F., Song, F., Zhang, C., Eslami, S. A., and Botvinick, M. Machine theory of mind. In *International conference on machine learning*, pp. 4218–4227. PMLR, 2018.

Rapoport, A. and Chammah, A. M. The game of chicken. *American Behavioral Scientist*, 10(3):10–28, 1966.

Rashid, T., Samvelyan, M., De Witt, C. S., Farquhar, G., Foerster, J., and Whiteson, S. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1):7234–7284, 2020.

Rigter, M., Lacerda, B., and Hawes, N. Risk-averse bayes-adaptive reinforcement learning. *Advances in Neural Information Processing Systems*, 34:1142–1154, 2021.

Rousseau, J.-J. *Discourse on the Origin of Inequality*. Oxford University Press, USA, 1999.

Schelling, T. C. Hockey helmets, concealed weapons, and daylight saving: A study of binary choices with externalities. *Journal of Conflict resolution*, 17(3):381–428, 1973.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Shu, T. and Tian, Y. M^{3}rl: Mind-aware multi-agent management reinforcement learning. *arXiv preprint arXiv:1810.00147*, 2018.

Silver, D. and Veness, J. Monte-carlo planning in large pomdps. *Advances in neural information processing systems*, 23, 2010.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pp. 5887–5896. PMLR, 2019.

Souza, M. O., Pacheco, J. M., and Santos, F. C. Evolution of cooperation under n-person snowdrift games. *Journal of Theoretical Biology*, 260(4):581–588, 2009.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

Vezhnevets, A., Wu, Y., Eckstein, M., Leblond, R., and Leibo, J. Z. Options as responses: Grounding behavioural hierarchies in multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 9733–9742. PMLR, 2020.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Warren, W. H. The dynamics of perception and action. *Psychological review*, 113(2):358, 2006.

Wen, Y., Yang, Y., Luo, R., Wang, J., and Pan, W. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Wu, S. A., Wang, R. E., Evans, J. A., Tenenbaum, J. B., Parkes, D. C., and Kleiman-Weiner, M. Too many cooks: Bayesian inference for coordinating multi-agent collaboration. *Topics in Cognitive Science*, 13(2):414–432, 2021.

Zhang, G. and Doshi, P. Sipomdplite-net: Lightweight, self-interested learning and planning in posgs with sparse interactions. *arXiv e-prints*, pp. arXiv–2202, 2022.

Zhao, S., Lu, C., Grosse, R. B., and Foerster, J. Proximal learning with opponent-learning awareness. *Advances in Neural Information Processing Systems*, 35:26324–26336, 2022.

Zhi-Xuan, T., Gothoskar, N., Pollok, F., Gutfreund, D., Tenenbaum, J. B., and Mansinghka, V. K. Solving the baby intuitions benchmark with a hierarchically bayesian theory of mind. *arXiv preprint arXiv:2208.02914*, 2022.

Zintgraf, L., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

A. Pseudo Code of HOP

Algorithm 1 HOP

```
**Input:** Number of MCTS tree $N_s$, update interval $T_u$, capacity of the trajectory buffer $L$, goal set $G_j$ $(j \neq i)$, initial belief of agents' goals $b_{ij}^{0,0}(g_j)$.
**Output:** Actions $a_i^{K,t}$, planning module network $\boldsymbol{\theta}$, goal-conditioned policy network $\boldsymbol{\omega}$.
for each episode $K$ do
    generate initial state of this episode $s^{K,0}$ randomly
    for $t = 0$ to $T_{max} - 1$ do
        repeat
            sample $\mathbf{g}_{-i}^{l}$ from $b_{ij}^{K,t}(g_j)(j \neq i)$
            get $Q_l(s^{K,t}, a, \mathbf{g}_{-i}^{l})$ $(\forall a)$ via MCTS
        until $N_s$ times
        calculate $Q_{avg}(s^{K,t}, a)$ $(\forall a)$ [Equation (4)]
        choose action $a_i^{K,t}$ from $\pi_{MCTS}(a|s^{K,t})$ [Equation (5)]
        intra-OM update $b_{ij}^{K,t+1}$ [Equation (1)]
        collect data of this step to the trajectory buffer
    end for
    if the trajectory buffer is full then
        update $\boldsymbol{\omega}$ [Equation (3)]
    end if
    if $K \times T_{max} \equiv 0$ (mod $T_u$) then
        update $\boldsymbol{\theta}$ [Equation (6)]
    end if
    inter-OM update $b_{ij}^{K+1,0}$ [Equation (2)]
end for
```

B. Theoretical Analysis

We aim to offer a concise theoretical analysis. Due to the complexity of environments characterized by both temporal and spatial structures, attaining theoretical guarantees in such environments can be inherently challenging. To strike a balance, we have undertaken a verification of the theoretical guarantee associated with HOP in the matrix games. These games encapsulate the same dilemma of sequential games. For clarity, our analysis will be conducted in the context of a two-player game, and the analysis can be extended to games involving a greater number of agents. Consider a two-player game where both players have two goals: "Cooperate" and "Defect," resulting in a utility matrix shown in Table 2.

Table 2. Utility matrix for a two-player game. Each element in the table represents the utility of the row player (first value) and the utility of the column player (second value). The utility values R, S, T, and P determine different game paradigms.

	Cooperate	Defect
Cooperate	R, R	S, T
Defect	T, S	P, P

Suppose HOP is the row player. At a certain timestep, the column player selects its goal g_{column} to be "Cooperate" with a probability of p and to be "efect" with a probability of $1 - p$. We sample the co-player's goal to simulate using Monte Carlo Tree Search (MCTS), with a frequency of $p + \epsilon$ to "Cooperate" and a frequency of $1 - p - \epsilon$ to "Defect."

In the current state s, we have two possible actions: a_1 for cooperation and a_2 for defection. During the MCTS planning process, when the co-player aims to "Cooperate," we have:

$$Q(s, a_1 | g_{\text{column}} = \text{"Cooperate"}) = R(1 + \epsilon_R)$$

$$Q(s, a_2 | g_{\text{column}} = \text{"Cooperate"}) = T(1 + \epsilon_T)$$

When the co-player aims to "Defect," we have:

$$Q(s, a_1 | g_{\text{column}} = \text{“Defect”}) = S(1 + \epsilon_S)$$

$$Q(s, a_2 | g_{\text{column}} = \text{“Defect”}) = P(1 + \epsilon_P)$$

Thus, we can calculate the overall Q-values as follows:

$$Q(s, a_1) = (p + \epsilon)R(1 + \epsilon_R) + (1 - p - \epsilon)S(1 + \epsilon_S)$$

$$Q(s, a_2) = (p + \epsilon)T(1 + \epsilon_T) + (1 - p - \epsilon)P(1 + \epsilon_P)$$

In the learning process, the goal-conditioned policy network is trained using supervised learning, and its accuracy significantly improves with sufficient rounds of observation. Consequently, the accuracy of the environment simulation within the Monte Carlo Tree Search (MCTS) algorithm becomes exceedingly high. In such a scenario, the convergence guarantee of MCTS remains intact, resulting in a final precision of MCTS that is remarkably high. Specifically, we have $|\epsilon_R|, |\epsilon_S|, |\epsilon_T|, |\epsilon_P| \ll |\epsilon|$, and these small error terms can be safely ignored.

Then, when

$$\frac{T + S - R - P}{p(R - T) + (1 - p)(S - P)}\epsilon < 1,$$

the optimal strategy that HOP obtains is consistent with the true optimal strategy. Two factors affect the size of $|\epsilon|$: the accuracy in inferring the co-player's goals and the deviation between frequency and probability when sampling the goal. To address the accuracy issue, we employ two layers of modules, intra-OM and inter-OM, to make accurate predictions as early as possible in each episode. For the deviation between frequency and probability, we increase the value of N_s to reduce this deviation. In practical applications, the choice of an appropriate N_s depends on the trade-off between computational speed and sampling accuracy.

C. Goal Definition

In MSH, we define two goals: g^C as hunting stags and g^D as hunting hares.

In MSG, we define two goals: g^C as removing the drifts, and g^D as staying lazy (i.e. not attempting to remove any snowdrifts). For inter-OM, the goal g^C is decomposed into 6 parts: g^{Ck} $(1 \leq k \leq 6)$, where g^{Ck} represents removing k snowdrift(s) in one episode. $b_{ij}^{K,0}(g^{Ck})$ and $b_{ij}^{K,0}(g^D)$ will be updated according to Equation (2). During an episode, if the co-player j has removed m snowdrift(s) at time t of the episode K, our belief $b_{ij}^{K,t}(g_j^C) = \sum_{k=m+1}^{6} b_{ij}^{K,0}(g_j^{Ck})$.

For intra-OM, each snowdrift s is defined as a subgoal $g^{C[s]}$. We use Equation (1) conditioned on g^C to update our belief:

$$b_{ij}^{K,t+1}(g_j^{C[s]} | g_j^C) = \frac{1}{Z_1} b_{ij}^{K,t}(g_j^{C[s]} | g_j^C) Pr_i(a_j^{K,t} | s^{K,0:t}, g_j^{C[s]}),$$

where Z_1 is the normalization factor. We can update our belief of an agent removing a snowdrift s:

$$b_{ij}^{K,t}(g^{C[s]}) = b_{ij}^{K,t}(g_j^{C[s]} | g_j^C) b_{ij}^{K,t}(g_j^C).$$

At the start of an episode, $b_{ij}^{K,0}(g_j^{C[s]} | g_j^C)$ is set to be uniform, which means $b_{ij}^{K,0}(g_j^{C[s]} | g_j^C) = \frac{1}{6}$. We train the goal-conditioned policy network $\boldsymbol{\omega}$ conditioned on $g^{C[s]}$.

D. Schelling Diagram

The Schelling diagram compares the rewards of different potential strategies (i.e., cooperation and defection here) given a fixed number of other cooperators. It is a natural generalization of the payoff matrix for two-player games to multi-player settings. Here, we use Schelling diagrams to validate our temporal and spatial extension of the matrix-form games.

Figure 3(a) and Figure 3(b) show the Schelling diagrams of MSH. Defection (i.e., hunting hare) is a safe strategy as a reasonable reward is guaranteed independent of the co-players' strategies. Cooperation (i.e., hunting stag) poses the risk of being left with nothing (when there are no others hunting stag), but is more rewarding if at least one co-player hunts stag. That is to say, hunting hare is risk dominant, and hunting stag is reward dominant. This is consistent with the dilemma described by the matrix-form stag-hunt game (Bloembergen et al., 2011). In the “4h1s” setting, when there are more than two cooperators, the choice to act as a cooperator carries the risk of not being able to successfully hunt. In the “4h2s” setting, the income of cooperators increases with the number of cooperators, resulting in a lower risk of choosing to hunt stag compared to the “4h1s” setting.

In the matrix-form snowdrift game, cooperation incurs a cost to the cooperator and accrues benefits to both players regardless of whether they cooperate or not (Souza et al., 2009). There are two pure-strategy Nash equilibria: player 1 cooperates and player 2 defects; player 1 defects and player 2 cooperates. That is, the best response is playing the opposite strategy from what the coplayer adopts. As shown in Figure 3(c), in MSG, one agent’s optimal strategy is cooperation (i.e., removing snowdrifts) when no co-players cooperate, but when there are other cooperators, the optimal strategy is defection (i.e., free-riding). Our MSG is an appropriate extension of the matrix-form snowdrift game.

E. Implementation Details

E.1. MCTS Simulation Details

As introduced in Section 4.2, we run MCTS for N_s rounds. In each round, we run N_i search iterations (see Browne et al. (2012) for details of each iteration). The score of an action a at state $\tilde{s}^k$ is:

$$Score(\tilde{s}^k, a) = Q(\tilde{s}^k, a) + c\pi_{\boldsymbol{\theta}}(a|\tilde{s}^k)\frac{\sqrt{\sum_{a'} N(\tilde{s}^k, a')}}{1 + N(\tilde{s}^k, a)}$$

where $Q(\tilde{s}^k, a)$ denotes the average return obtained by selecting action a at state $\tilde{s}^k$ in the previous search iterations. $N(\tilde{s}^k, a)$ represents the number of times action a has been selected at state $\tilde{s}^k$ in the previous search iterations. $\pi_{\boldsymbol{\theta}}(a|\tilde{s}^k)$ refers to the policy provided by the network $\boldsymbol{\theta}$. c is the exploration coefficient. We select the action which has the highest score when reaching $\tilde{s}^k$ at the selection phase of one search iteration.

E.2. Network Architecture

The goal-conditioned policy network $\boldsymbol{\omega}$ and the policy-value network for MCTS $\boldsymbol{\theta}$ both start with three convolutional layers with the kernel size 3 and the stride size 1. Three layers have 16, 32, and 32 output channels, respectively. They are connected to two fully connected layers. The first layer has an output of size 512, and the second layer gives the final output.

E.3. Hyperparameters

For each result in Figure 4, Table 1, Table 4 and Table 5, we performed 10 independent experiments using different random seeds. The left-hand side of $\pm$ represents the average reward of the 10 trials, and the right-hand side represents the standard error.

Hyperparameters for HOP are listed in Table 3(a). α and T_u are tuned in the adaptation phase to achieve fast adaptation. As α decreases, agents attach greater importance to recent episodes, which will speed up the adaptation to new behaviors of the co-players. It is not advisable to adjust α too small, otherwise the update may be unstable due to the randomness of the co-player’s strategy.

Hyperparameters for baselines are listed in Table 3. Some hyperparameters are tuned in the adaptation phase to achieve fast adaptation.

F. Supplementary Results

F.1. Orcale Agents

To compare and evaluate the performance of few-shot adaptation between HOP and learning baselines, we train an Orcale agent to see how well a well-established RL agent can perform in adaptation to co-players through extensive interactions

Specifically, for every type of co-players, one Orcale agent interacts with them and is trained via A3C to converge from scratch. During the training phase, co-players’ parameters are fixed, which are the convergent parameters in their self-play.

In the subsequent adaptation phase, the trained Orcale agent is tested in the same way as HOP and baseline algorithms. This process ensures that the Orcale agent engages in extensive interactions with the agents it would encounter during the adaptation phase. Over an extended duration of interaction, Orcale effectively acquires a robust and high-quality policy. We use the Orcale agent's performance in the adaptation phase as a reference point to explain HOP's performance.

F.2. Ablation Study

To test the importance and necessity of each component in HOP, we construct three partially ablated versions of HOP. The agent without inter-OM (w/o inter-OM) does not execute the inter-episode update expressed as Equation (2). W/o inter-OM begins each episode with a uniform belief prior. The agent without intra-OM (w/o intra-OM) does not execute the intra-episode update expressed as Equation (1). That is, for w/o intra-OM, $b_{ij}^{K,t}(g_j) = b_{ij}^{K,0}(g_j), \forall t$. The direct-OM agent removes the whole opponent modeling module of HOP, and utilizes neural networks to model co-players directly. The co-player policies are mappings from states to actions, and not conditioned on goals. Experimental results for HOP and its three ablation versions in MSH-4h2s are shown in Table 5.

In self-play, HOP have an advantage over direct-OM agents. It suggests that utilizing a goal as a high-level representation of agents' behavior is beneficial to opponent modeling in complex environments. On the other hand, compared with w/o inter-OM and w/o intra-OM, HOP does not exhibit a significant advantage in self-play. The inter-OM and intra-OM modules may not be effective in the self-play setting, where a large number of interactions happen.

In the experiments testing few-shot adaptation, HOP outperforms its ablation versions. W/o inter-OM agents struggle when facing agents with fixed goals, such as cooperators and defectors. As the goals of cooperators and defectors are fixed, correct actions can be taken immediately if the focal agent has accurate goal priors. W/o inter-OM agents lack accurate goal priors at the beginning of an episode. In every episode, they have to use multiple interactions to infer co-players' goals and thus miss out on early opportunities to maximize their interests.

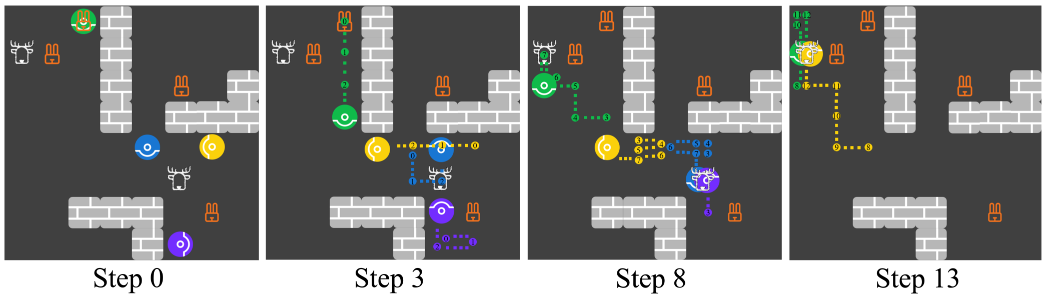
W/o intra-OM agents exhibit poor performance when facing agents with dynamic behavior such as LOLA, PS-A3C, and random. These co-players have multiple goals. But in a given episode, the specific goals of a co-player can be gradually determined by analyzing its trajectory in this episode. However, w/o intra-OM agents can only count on inter-OM, which only takes the past episodes into account, but does not consider the information from the current episode. It results in inaccurate goal estimates in a given episode, which hurts the performance in few-shot adaptation.

Direct-OM agents are at an overall disadvantage. Their opponent modeling solely relies on the neural network, which makes it challenging to obtain significant updates during a short interaction. This leads to inaccurate opponent modeling during the adaptation phase. Furthermore, direct-OM agents utilize end-to-end opponent modeling, which introduces a higher degree of uncertainty compared to the goal-conditioned policy. This uncertainty can reduce the precision of the simulated co-player behavior during planning.

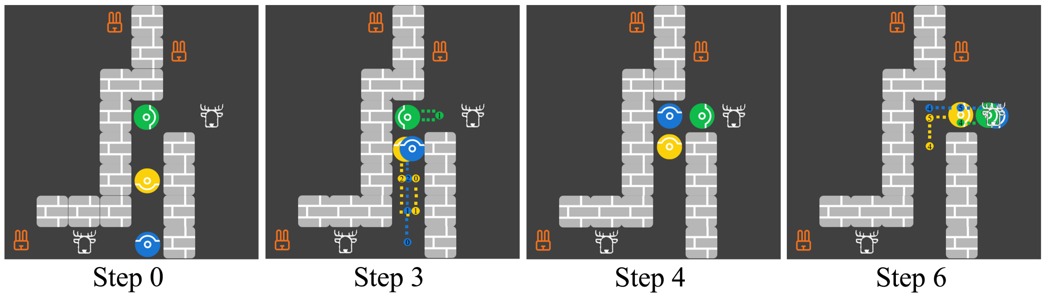
G. Emergence of Social Intelligences

There are two kinds of social intelligence, self-organized cooperation and the alliance of the disadvantaged, emerging from the interaction between multiple HOP agents in MSH. We make a minor modification to the game: the game terminates only when the time $T_{max} = 30$ runs out.

Self-organized cooperation. As shown in Figure 6(a), at the start of the game, three agents (blue, yellow, and purple) are two steps away from the stag at the bottom-right side, and the last agent (green) is spawned alone in the upper left corner. One simple strategy for the three agents located at the bottom-right corner is to hunt the nearby stag together. Although this is a riskless strategy, the three agents each only obtain a reward of $10/3$. Instead, if one agent chooses to collaborate with the green agent at the top-left corner, all four agents each get a reward of 5. This strategy is riskier since if the green agent chooses to hunt a nearby hare, the collaborative agent will not be able to catch any stag. We show that HOP is able to achieve the aforementioned risky but rewarding collective strategy. Specifically, the green agent refuses to catch the hare at his feet and shows the intention of cooperating with others (see screenshots at step 3 and step 8 in Figure 6(a)). The yellow agent refuses to catch the stag at the bottom-right corner and chooses to collaborate with the green agent to hunt the stag in the top-left corner. In this process, all four agents receive the maximum profit. Here, agents achieve pairwise cooperation through independent decision-making, without centralized assignment of goals. Thus, we call this phenomenon self-organized cooperation.



(a) Self-organized cooperation



(b) Alliance of the disadvantaged

Figure 6. Screenshots for the emergence of (a) self-organized cooperation and (b) alliance of the disadvantaged. Each panel shows agents' locations at the current step and the trajectories between the current step and the previously stated step.

Alliance of the disadvantaged. In addition to the aforementioned game rules, we assume agents are heterogeneous. Specifically, the yellow agent (Y) is three times greedier than the blue agent (B) and the green agent (G). That is, when the three agents cooperate to hunt a stag successfully, Y will get a reward of 6, and the others get 2 each. When Y cooperates with one of B and G, Y will obtain 7.5, the other one gets 2.5. As shown in Figure 6(b), at the start of the game, Y locates between B and G. Neither B nor G would like to cooperate with Y. Hence they need to move past Y to cooperate with each other. To achieve this, agents B and G first move closer to each other in the first few steps. However, to maximize its own profit, agent Y also moves toward B and G and hopes to hunt a stag with them. To avoid collaboration with agent Y, after agents B and G are close enough to each other, they move back and forth to mislead Y (see step 3 of Figure 6(b)). Once agent Y makes a wrong guess of the directions agents B and G move, B and G will get rid of Y, and move to the nearest stag to achieve cooperation (see Step 4 and 6 of Figure 6(b)), which maximizes the profit of agents B and G.

From the above two cases, we find that although HOP aims to maximize self-interest, cooperation emerges from the interaction between multiple HOP agents in mixed-motive environments. This shows that it may be helpful in solving mixed-motive environments by equipping agents with the ability to infer others' goals and behavior and the ability to fast adjust their own responses.

Table 3. Hyperparameters
(a) HOP

	self-play phase		adaptation phase	
	MSH	MSG	MSH	MSG
horizon weight α	0.99	0.99	0.95	0.95
rationality coefficient β	2	2	5	5
discount factor γ	0.95	0.95	0.95	0.95
update interval T_u	2000	2000	200	200
capacity of the trajectory buffer L	5000	5000	5000	5000
number of MCTS rounds N_s	8	5	8	5
number of search iterations for each MCTS N_i	200	200	200	200
exploration coefficient c	2	12	2	12
learning rate	10^{-4}	10^{-4}	5×10^{-4}	5×10^{-4}
OM learning rate	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}

(b) A3C and PS-A3C

	self-play phase		adaptation phase	
	MSH	MSG	MSH	MSG
learning rate	10^{-4}	10^{-4}	5×10^{-4}	5×10^{-4}
batch size	2000	2000	200	200
discount factor	0.99	0.99	0.99	0.99
value function loss coefficient	0.5	0.5	0.5	0.5
gradient clip	40	40	40	40
entropy coefficient	0.01	0.01	0.01	0.01

(c) LOLA

	self-play phase		adaptation phase	
	MSH	MSG	MSH	MSG
learning rate	10^{-4}	10^{-4}	5×10^{-4}	5×10^{-4}
OM learning rate	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
batch size	2000	2000	200	200
discount factor	0.99	0.99	0.99	0.99

(d) Social Influence

	self-play phase		adaptation phase	
	MSH	MSG	MSH	MSG
learning rate	10^{-4}	10^{-4}	5×10^{-4}	5×10^{-4}
batch size	2000	2000	200	200
Influence weight	1.0	1.0	1.0	1.0
MOA loss weight	3.0	3.0	10.0	10.0
entropy coefficient	0.01	0.01	0.01	0.01

(e) PR2

	self-play phase		adaptation phase	
	MSH	MSG	MSH	MSG
learning rate	10^{-4}	10^{-4}	5×10^{-4}	5×10^{-4}
batch size	2000	2000	200	200
soft update parameter	0.99	0.99	0.99	0.99

Table 4. Few-shot adaptation performance of Orcale in all three sequential social dilemma paradigms. The interaction happens between 1 Orcale agent and 3 co-players using the column policy. Shown is the average reward for Orcale from 1800 to 2400 step.

	learning co-players			rule-based co-players		
	LOLA	A3C	PS-A3C	random	cooperator	defector
MSH-4h1s	2.44± 0.03	0.88± 0.01	3.57± 0.03	1.10± 0.00	2.73± 0.02	0.93± 0.01
MSH-4h2s	3.23± 0.02	3.46± 0.01	3.97± 0.02	1.22± 0.01	3.42± 0.02	0.70± 0.01
MSG	20.9± 0.12	22.7± 0.17	32.5± 0.12	16.0± 0.08	36.0± 0.00	12.0± 0.00

Table 5. Performance of HOP and its ablation versions in MSH-4h2s. In (a) self-play, 4 agents of the same kind are trained to converge. Shown is the normalized score after convergence. In (b) few-shot adaptation, the interaction happens between 1 agent using the row policy and 3 co-players using the column policy. Shown are the min-max normalized scores, with normalization bounds set by the rewards of Orcale and the random policy. The results are depicted for the row policy from 1800 to 2400 step.

(a) Self-play performance

HOP	w/o inter-OM	w/o intra-OM	direct-OM
0.9767± 0.0117	0.9708± 0.0146	0.9738± 0.0117	0.9417± 0.0146

(b) Few-shot adaptation performance

	learning co-players			rule-based co-players		
	LOLA	A3C	PS-A3C	random	cooperator	defector
HOP	**0.97**± 0.02	**0.99**± 0.02	**0.88**± 0.02	**0.78**± 0.07	**1.00**± 0.01	**0.36**± 0.02
w/o inter-OM	**0.97**± 0.02	0.92± 0.03	0.87± 0.02	**0.78**± 0.03	0.96± 0.02	0.31± 0.02
w/o intra-OM	0.95± 0.02	0.98± 0.02	0.84± 0.01	0.65± 0.04	0.99± 0.02	0.34± 0.03
direct-OM	0.95± 0.01	0.85± 0.02	0.74± 0.03	0.62± 0.04	0.96± 0.02	0.31± 0.02