Emergence of Theory of Mind Collaboration in Multiagent Systems

Luyao Yuan
yuanluyao@ucla.edu

Zipeng Fu
fu-zipeng@engineering.ucla.edu

Linqi Zhou
Alexzhou907@gmail.com

Kexin Yang
ykx1998@ucla.edu

Song-Chun Zhu
sczhu@cs.ucla.edu

Department of Computer Science
University of California, Los Angeles

Abstract

Currently, in the study of multiagent systems, the intentions of agents are usually ignored. Nonetheless, as pointed out by Theory of Mind (ToM) [1], people regularly reason about other's mental states, including beliefs, goals, and intentions, to obtain performance advantage in competition, cooperation or coalition. However, due to its intrinsic recursion and intractable modeling of distribution over belief [2, 3], integrating ToM in multiagent planning and decision making is still a challenge. In this paper, we incorporate ToM in multiagent partially observable Markov decision process (POMDP) and propose an adaptive training algorithm to develop effective collaboration between agents with ToM. We evaluate our algorithms with two games, where our algorithm surpasses all previous decentralized execution algorithms without modeling ToM.

1 Introduction

Developing effective collaborations in a multiagent system (MAS), when there are more than one learning agents, is a challenging problem. Modeling the joint action for all agents enables single agent reinforcement learning (RL) algorithms to be applied to MAS directly, however, the joint action space grows exponentially as the number of agents increases. Decentralized execution and centralized training algorithms aim to cope with this complexity restriction [4, 5, 6, 7], where agents learn policy mapping from local observation history to actions through a centralized critic. Nevertheless, two difficulties need to be addressed in current approaches. First, simultaneous agents updates will cause non-stationary environments and impinge the convergence of these algorithms [8, 6, 7, 9, 10]. Second, because a MAS is partially observable to all agents, they must infer each other's mental status to effectively coordinate with each other, either with or without communications. For example, it is important for a group of item collecting robots to infer each other's current target, otherwise, there could be robots aiming for the same item or items collected by no one.

It has been shown that humans, during an interaction, can reason about others' beliefs, goals, intentions and predict opponents/partners' behaviors, a capability called Theory of Mind (ToM) [11, 12, 1]. In some cases, people can even use ToM recursively, and form beliefs about the way others reason about themselves [2]. Thus, to collaborate and communicate with people smoothly, artificial agents must also bear similar potentially recursive mutual reasoning capability. Despite the recent surge of multiagent collaboration modeling [13, 5, 14, 10], integrating ToM is still a nontrivial challenge.

Emergent Communication Workshop, 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

A few approaches attempted to model nested belief of other agents in general multiagent systems, but extensive computation restricts the scale of the solvable problems [3, 15]. When an agent has an incomplete observation of the environment, it needs to form a belief, a distribution over the actual state of the environment, to take actions [12, 15]. ToM agents, besides their own beliefs about the state, also model other agents' beliefs, forming belief over beliefs. They can further have beliefs about others' belief over belief, so on and so forth [3, 16, 12, 2, 17]. The intractability of distribution over distribution makes exact solving for ToM agents' nested beliefs extremely complicated [3]. Therefore, an approach to acquire the sophistication of high-level recursions without getting entangled into the curse of intractability is needed.

In this paper, we propose an adaptive training process, following which effective collaboration can emerge between agents with ToM by only modeling one level belief over belief. The complexity of higher level recursions can be preserved by the dynamic evolving of agents' tractable belief estimation functions. Intuitively, for a given agent, we don't simulate its behavior by assuming it has a certain level of recursions, which requires modeling nested beliefs all the way up to the desired level [3, 16, 2, 17]. We directly learn a function to approximate its actual belief and how to react accordingly. In cooperative games, this learning becomes mutual adaptation, with controlled exploration rate, improving the performance of the multiagent system [18]. Also, in our adaptive training procedure, one agent is trained while the other being fixed, creating a stationary environment for the learning agents. Hence, our algorithm won't be influenced by non-stationarity.

To justify the advantage of ToM agents, we evaluate our algorithm in two-player imperfect information games. In these games, there is a public game state available to all players, and each player has a piece of private information. Each agent maintains a belief about its partner's private state and an estimation of its partner's belief about its private state, namely a belief over belief. The belief over others' private information is obtained with counterfactual reasoning [19, 9]. The estimation of the partner's belief about one's private information is learned in centralized training, during which agents share their beliefs to others as supervision. We test our algorithms in two multiagent POMDPs. The first is inspired by the robot-human collaboration in the kitchen setting proposed by Fisac et al. [19] The second game simulates an appointment scheduling process between two agents, each of whom has private time schedules and wants to choose a commonly available time slot to have a meeting. By regulating the message space of the agents, this game becomes a nontrivial POMDP, requiring mutual reasoning to accomplish.

We compared our adaptive ToM algorithm with several popular multiagent benchmarks using various flavors of policy gradients [20] and Q-learning [21], and achieved robust performance close to SOTA models, which are not only trained in a centralized way, but also have centralized modules like shared Q-functions [19, 10] or meta-agent [9]. Moreover, we found that ToM can facilitate the emergence of more universal protocols decipherable across different groups. Our model beats all benchmarks with a large margin in flexible group assignment experiments.

2 Related Work

Most multiagent RL algorithms have their single agent origins [22]. The main challenge of generalizing single agent methods to MAS is the trade-off between complexity and optimality. Centralized execution methods guarantee optimality, but have exponential complexity wrt. the number of agents, while decentralized execution sacrifices performance for simplicity. Thus, the idea of centralized learning but decentralized execution finds its balanced position. Independent Q-based algorithms in MAS, from early works investigating small scale multiagent tabular games [23, 24] to a multiagent generalization of DQN [8, 25] on large-scale state and action spaces, emphasize decentralized execution. However, the non-stationarity caused by simultaneous agents' updates casts a shadow on the convergence of these algorithms. To relieve the non-stationarity, Foerster et al. [6] proposed stabilizing experience replay by memorizing fingerprints of opponents' policies. Another stream of multiagent algorithms originates from the actor-critic method [7, 10]. They learn a decentralized policy guided by a centralized Q-function. However, the large variance of the policy gradient [20] usually restricts the performance of these algorithms.

On top of the generic Q-learning and policy gradient methods, there are various opponent modeling and communication methods proposed to facilitate coordination and further improve the performance of specific types of tasks. COMA in [10] aims to solve the credit assignment problem in MAS, but

they require agents to have the same reward functions. [4, 5, 26] proposed multiagent communication with continuous signals during collaboration. They model communication as another type of actions and have specific modules to control it. Nevertheless, transmitting continuous signals requires channels with large bandwidth, which may not be available when execution. Our model, on the other hand, only needs communication with discrete messages in centralized training to develop effective collaboration in decentralized execution.

Emergence of communication protocol using discrete messages has been explored with various types of communication games [27, 28, 29, 30]. In these games, agents only communicate with non-suited messages [31], namely no agent-environment interactions. Nevertheless, our model integrates agent-agent messages and agent-environment actions to infer other agents' mind and collaborate.

Attempts to integrate ToM in opponent modeling has profound cognitive science origin [16, 1]. LOLA in [32] learns the best response to evolving opponents. Yet, opponents/partners' real-time believes are not considered into policy. Interactive-POMDP (I-POMDP) [3, 15] moves one more step forward by actually modeling opponents' mental states at the current moment and integrates other's belief into the agent's policy. However, I-POMDP requires extensive sampling to approximate the nested integration over the belief space, action space and observation space, limiting its scalability. The Bayesian action decoder (BAD)-MDP proposed by Foerster et al. [9] and cooperative inverse reinforcement learning (CIRL) proposed by Fisac et al. [19] also use counterfactual reasoning in their belief update, but their methods are more centralized in the testing process than ours. The BAD-agent is a super-agent controlling all other agents collectively. Deterministic partial policies can easily reveal agents' private information to the BAD-agent and make it public. CIRL requires that human and robot have their Q-function in common knowledge. Instead, our model doesn't depend on any implicit information flowing between agents during testing and assumes no common knowledge. Wen et al. [33] proposed recursive reasoning policies in MAS to accommodate ToM. The only change in their method from [7] is to use soft Q-learning [34, 35] and SVGD [36] to estimate other agents' policy conditioning on the estimator's action[1]. Our algorithm, which uses explicit belief and belief over belief in our value functions, outperforms their implicit modeling approach in partially observable games.

3 Background and Setting

Consider a partial observable multiagent game with N agents. At time t each agent $i \in \{1, ..., N\}$ takes an action from its action space $a_i^t \in A_i$ according to $\pi_i(a|\tau_i^t)$, where τ_i^t stands for agent i's observation and action history $\{o_i^0, a_i^0, \dots, o_i^t, a_i^t\}$. Here we assume that states of this MAS can be decomposed into physical states and private agent states, namely $\widetilde{S} = S \times \Omega_1 \times ... \times \Omega_N$, where S is the environment state space, Ω_i is the agent i's private state space, eg. agent's goal or intention, and $\widetilde{S}$ is the complete game space. At time t agent i's observation $o_i^t = (O_i(s, a_i^t), \omega_i)$, an observation triggered by its action from the physical state and its private agent state, $\omega_i \in \Omega_i$ which we assume constant until the end of one game. If we denote $\widetilde{s}_{\omega_i}$ as agent i's private state at state $\widetilde{s}$ and $\mathbf{a}^t = \{a_1^t, ..., a_N^t\}$, then we have the state transition function $P(\widetilde{s}^{t+1}|\widetilde{s}^t, \mathbf{a}^t) = T(s^{t+1}|s^t, \mathbf{a}^t) \prod_{i=1}^N \mathbf{1}(\widetilde{s}_{\omega_i}^{t+1} = \widetilde{s}_{\omega_i}^t)$. At time t, agent i gets reward $r_i^t = r_i(\widetilde{s}^t, \mathbf{a}^t)$. In fully cooperative games, all agents share the same reward function. Notice that rewards depend on the complete state instead of just the environment. An example will be some of the agents know the goal of a task requiring all agents' effort to accomplish, making inference of other agents' private state crucial.

We can define the value of a state as $V^\pi(\widetilde{s}^t) = \mathrm{E}_{\mathbf{a}^t, \widetilde{s}^{t+1}, \dots}[\sum_t \gamma^t r(\widetilde{s}^t, \mathbf{a}^t)]$. The goal of this multiagent game is to find a set of policies $\pi = \{\pi_1, ..., \pi_N\}$ to maximize the expected return of the game $\mathrm{E}_{\widetilde{s}^0}[V^\pi(\widetilde{s}^0)]$ starting at time 0. The optimal value of a state is defined as $V^*(\widetilde{s}) = \max_\pi V^\pi(\widetilde{s})$. We can also define Q-function as $Q^\pi(\widetilde{s}^t, \mathbf{a}) = r(\widetilde{s}^t, \mathbf{a}) + \gamma \mathrm{E}_{\widetilde{s}^{t+1} \sim P(\widetilde{s}^{t+1}|\widetilde{s}^t, \mathbf{a})}[V^\pi(\widetilde{s}^{t+1})]$ and $Q^*(\widetilde{s}^t, \mathbf{a}) = r(\widetilde{s}^t, \mathbf{a}) + \gamma \mathrm{E}_{\widetilde{s}^{t+1} \sim P(\widetilde{s}^{t+1}|\widetilde{s}^t, \mathbf{a})}[V^*(\widetilde{s}^{t+1})]$. It has been shown that Q-learning can converge to optimal Q-values with mild assumptions [37]. We also define value functions for an agent given its opponents policies:

$$V_{i|\pi_{-i}}^*(\widetilde{s}^t) = \max_{\pi_i} \mathrm{E}_{\mathbf{a}_{-i}^t \sim \pi_{-i}, a_i^t \sim \pi_i, \widetilde{s}^{t+1}, \dots} \Big[\sum_t \gamma^t r_i(\widetilde{s}^t, \mathbf{a}^t) \Big] \tag{1}$$

[1] They also assume agents to share the same reward function, as they use the estimator's Q-function to predict other agents policies.

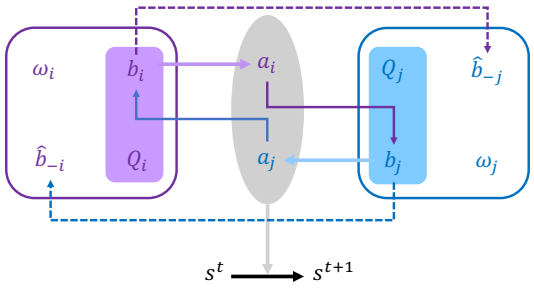


Figure 1: Execution pipeline for the case of two agents. Actions are determined by Q-values weighted by the belief. Observation of other agents' actions is utilized to update one's belief by counterfactual reasoning. Dashed lines represent belief estimation supervision, only included in centralized training. Remember that this supervision is a discretization of the real belief, eg. a sample from the belief distribution.

$$Q^*_{i|\pi_{-i}}(\tilde{s}^t, a_i) = \mathop{\mathrm{E}}_{\mathbf{a}_{-i}\sim\pi_{-i}}\Big[r_i(\tilde{s}^t, \mathbf{a}_{-i}, a_i) + \gamma \mathop{\mathrm{E}}_{P(\tilde{s}^{t+1}|\tilde{s}^t,\mathbf{a})}[V^*_{i|\pi_{-i}}(\tilde{s}^{t+1})]\Big]. \tag{2}$$

When there is no ambiguity, we can omit the π_{-i} in the subscript and only write Q_i.

4 Method

In this section, we propose our algorithm to solve the partial observable game defined in the previous section. First, we define agents' modules and introduce how they act when all modules are learned. Then, we illustrate the learning algorithm for all modules, which is a centralized process among agents.

4.1 Decentralized Execution

The execution process is purely decentralized as all agents only act according to their local observations and no direct communication other than mutual inference from action observation. For an agent i, it performs according to its value function, $Q_i : \tilde{S} \times A_i \to R$. Here we assume the physical state is fully observable to all agents, so the only belief an agent needs to hold is about other agents' private states. This assumption can be relieved by introducing an observation model for the physical state and form belief over the environment. If we denote other agents as $-i$, as their private states are not available to agent i, it must maintain a belief $b^t_i(\omega_{-i})$. Thus, we have agent i's policy following Boltzmann rationality model [1] with β as the rationality coefficient of i and quantifies the concentration of i's choices around the optimum. As $\beta \to \infty$, i becomes a perfectly rational agent, while, as $\beta \to 0$, i becomes indifferent to Q:

$$\pi_i(a|s^t, \omega^t_i, \tau^t_i) = \frac{\exp(\beta \sum_{\omega_{-i}\in\Omega_{-i}} b^t_i(\omega_{-i}|\tau^t_i) Q_i(a, s^t, \omega_i, \omega_{-i}))}{\sum_{a'\in A_i} \exp(\beta \sum_{\omega_{-i}\in\Omega_{-i}} b^t_i(\omega_{-i}|\tau^t_i) Q_i(a', s^t, \omega_i, \omega_{-i}))}. \tag{3}$$

Next problem is how agent i maintains its belief given its observation history. We utilize counterfactual reasoning [19, 9] in our belief update function. That is, agent i traverses all possible private states ω_{-i} and estimates how likely the actions it observed are taken given a specific set of private agent states are the correct one. Then, i updates its belief using Bayesian rule:

$$b^t_i(\omega_{-i}|\tau^t_i) = P(\omega_{-i}|\tau^{t-1}_i, \mathbf{a}^t_{-i}) \propto \hat{\pi}_{-i}(\mathbf{a}^t_{-i}|s^t, \omega_{-i}) b^{t-1}_i(\omega_{-i}), \tag{4}$$

where $\hat{\pi}_{-i}$ is agent i's estimation of $-i$'s policy, learned in centralized training. In this paper, we assume all actions are observable to all agents. That is, the only hidden components of the games are the agents' private state. This assumption can be relaxed by including an observation function for each agent in equation 4.

Agent i needs to maintain a belief about other agents' private states, so do other agents need to estimate i's state. If there are two proper actions for task completion, but one can convey agent i's private state to others while the other reveals little information, then the first action should be preferred. Thus, agent i's Q-function should have another argument to accommodate others' belief about ω_i. Utilizing the obverter technique [38], we let the i holds a belief $\hat{b}_{-i}$ as the estimation of $-i$'s belief about ω_i. Here, $\hat{b}_{-i}$ is still a distribution over Ω_i. We didn't use a distribution over distribution to model this nested belief because the belief update process is deterministic for rational agents following Bayesian rule. Given $\hat{b}^0_{-i}$ a uniform distribution over candidates, $P(\hat{b}^t_{-i})$ is unimodal with uncertainty merely from the likelihood and can be approximated with a single point. All we need is a belief update function in i, $f_{-i} : \Delta(\Omega_i) \times A_i \times S \to \Delta(\Omega_i)$, where $\Delta(\Omega_i)$ represents a distribution

Algorithm 1 Adaptive ToM Collaboration Emergence

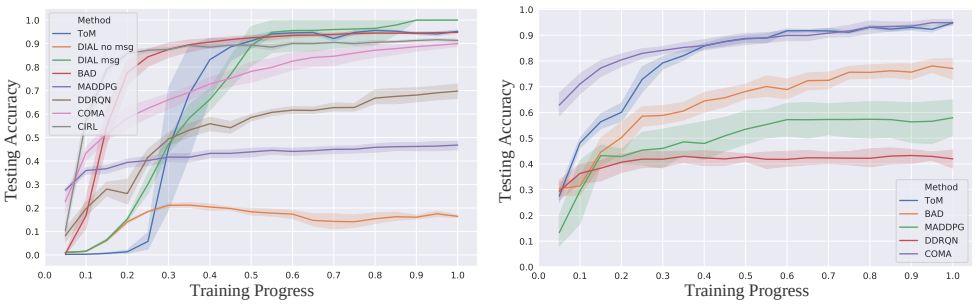
1: Randomly initialize $\theta_i, \hat{\theta}_i, \eta_Q, \eta_\pi, \eta_f, i \in \{1, ..., N\}$
2: Learning rate η, Batch size M
3: **for** each round **do**
4: **for** $i \in \{1, ..., N\}$ **do**
5: Initialize replay buffer $\mathcal{D} \leftarrow \emptyset$
6: **while** train agent i **do**
7: **repeat**
8: Agents sample actions according to equation 3
9: Agents update their beliefs
10: Agents update their estimation of partners' beliefs
11: **until** game ends
12: Update $\mathcal{D}$ with new trajectory
13: Sample M trajectories $\{(\omega_{1:N}, s_{0:T}, \mathbf{a}_{0:T}, r_{0:T})\}_{k=1}^{M}$
14: $y_i^{t,(k)} = r_i^{t,(k)} + \gamma \max_{a \in A_i} Q_{\hat{\theta}_i}(a, s^{t+1,(k)}, \omega^{(k)}, \hat{b}_{-i}^{t,(k)})$
15: $L^Q = \sum_{t,k} ||Q_{\theta_i}(a_i^{t,(k)}, s^{t,(k)}, \omega^{(k)}, \hat{b}_{-i}^{t,(k)}) - y_i^{t,(k)}||^2$
16: $L^\pi = \sum_{t,k} H(\hat{\pi}_{-i}(\mathbf{a}_{-i}^t | s^{t,(k)}, \omega_{-i}^{t,(k)}), \mathbf{a}_{-i}^{t,(k)})$
17: $L^f = KL(\bar{b}_{-i}^{t,(k)} || f_{-i}(\hat{b}_{-i}^{t-1,(k)}, a_i^{t,(k)}, s^{t,(k)}))$
18: $\theta_i \leftarrow \theta_i - \nabla_{\theta_i}(\eta_Q L^Q + \eta_\pi L^\pi + \eta_f L^f)$
19: Periodically update $\hat{\theta}_i \leftarrow \theta_i$ for Q-learning
20: **end while**
21: **end for**
22: **end for**

over Ω_i. Belief update function f takes in the old belief, agent i's action, the physical state and returns a new belief as i's new estimation of others' belief over ω_i. If ω_i and ω_{-i} are not independent, then b_i should be an additional argument of f, but in this paper, we only explore the scenarios where ω_i and ω_{-i} are independent of each other. The ability to correctly infer others' private states from their actions and predict others belief about oneself introduces ToM into our agents. Figure 1 visualizes the execution pipeline.

4.2 Centralized Training

As introduced in section 4.1, there are three components that every agent needs to learn during centralized training, $Q_i, \hat{\pi}_{-i}$ and f_{-i}. To learn Q_i, we apply the deep Q-learning algorithm [21, 39]. To avoid the non-stationarity caused by simultaneous agent updates, we fix all other agents when one agent is being trained. Thus, all the fixed agents can be considered as part of the environment, so the convergence of the Q-function of the learning agent is still guaranteed. In a fully observable multiagent game where all agents share the same reward, suppose we denote $\pi = \{\pi_1, ..., \pi_{i-1}, \pi_i, \pi_{i+1}, ..., \pi_N\}$ as the joint policy before agent i is being trained and $\pi' = \{\pi_1, ..., \pi_{i-1}, \pi_i', \pi_{i+1}, ..., \pi_N\}$ as the new policy after i is trained, where $\pi_i' = \arg\max_{a \in A_i} Q_i^*(s, a)$. One can show from the convergence of the Q-learning that $E_{s_0}[V^{\pi'}(s_0)] \geq E_{s_0}[V^\pi(s_0)]$ using the same procedure as in [37]. Therefore, our adaptive training algorithm guarantees monotonic performance improvement. As history is important in Q-functions of POMDP, we saved trajectories in the replay buffer and sample complete trajectories in the training [40].

Both $\hat{\pi}_{-i}$ and f_{-i} are learned with supervision. To be more specific, during training agents $-i$ will reveal their private agent states to the public after each game is finished. Then $\hat{\pi}_{-i}$ can be learned with MLE. The supervision of f_{-i} is more complicated, as beliefs are continuous variables representing distributions, but communicating discrete messages are more realistic. Therefore, we discretize beliefs b_{-i} and pass that discretization $\bar{b}_{-i}$ to i as supervision for f_{-i}. The exact discretizing procedure may vary according to different tasks, as we'll elaborate in later section 5.1 and 5.2. In our future work, we consider using a specifically trained message decoder to further enrich communication. The detailed learning process is in algorithm 1. Here we abuse the notation a little by encompassing all parameters of agent i into θ_i, which contains three sets of parameters for the Q-function, $\hat{\pi}_{-i}$ and f_{-i} respectively. In line 16, H represents softmax cross-entropy loss, as here we have A_{-i} as discrete actions. In line 17 we use Kullback-Leibler (KL) divergence to measure the distance between the



(a) Kitchen Collaboration. BAD:0.15M, COMA:0.25M, ToM: 1.6M, DIAL:0.1M, MADDPG:3M, DDRQN:4M, CIRL:0.4M. When the number of possible states is smaller than the number of actions, BAD can easily form an one-to-one mapping to force agents to reveal their private information. When there are fewer actions than possible states (e.g. in appointment scheduling), BAD has a worse performance.

(b) Appointment Scheduling. BAD: 0.1M, COMA: 0.1M, ToM: 1.6M, MADDPG: 1.0M, DDRQN: 20K. Using two separate Q-functions causing MADPPG more likely to have at least one agent stuck at local optimum. DDRQN updates all agents simultaneously, suffering from non-stationarity. Our model cannot outperform COMA probably due to the difficulty of learning high dimensional belief precisely.

Figure 2: Different algorithms take different training iterations to converge, so we normalize the iterations as training progress from 0 to 1 as the x-axis. y-axis represents testing accuracy. Due to the time limit, we stop training a model after it converges. We halt when the best training accuracy stops increasing for 10000 iterations. We average the training accuracy of mini-batches every 1000 iterations to avoid fluctuation. We'll report results for training all models to a fixed iterations in our later version. In captions above, M stands for 10^6 and K stands for 10^3.

predicted partners' belief and the partners' belief supervision, $\bar{b}_{-i}$. This distance measure can be replaced with other functions given different tasks and forms of belief supervision because we use discretized belief instead of continuous vectors. As in line 18, different loss may use different step size η to update.

5 Experiments

We used two multiagent games with imperfect information including two agents to evaluate our experiments. For both experiments, we generated three exclusive training/testing splits and repeat two experiments with different random initialization on each split. We report average performance from six experiments. We compare with several benchmarks including CIRL [19], COMA [10], DDRQN [8], MADDPG [7], BAD [9] and DIAL [4]. For MADDPG, we integrate PR2's policy modeling from [33] by taking history into the policy network. We further simplify policy estimation by letting agents using their opponents' actual policies in training. Notice that in the original setting of DIAL, agents require communication in both training and testing. Our model, on the other hand, only needs communication during training. So we report DIAL performance for testing without communication. When implementing these benchmarks, we stick to their papers to adapt all models to our tasks. We also take the author released code for reference if there is. The detailed structure and hyper-parameters can be found in supplementary together with our codes.

5.1 Kitchen Collaboration

This multiagent game is inspired by the ChefWorld proposed in [19]. Suppose there are two agents A and B, with A as the chef and B as the assistant. For each game, a recipe with K dishes is publicly provided to the agents. The purpose of the game is for both agents to prepare ingredients for the target dish, which is only known by the chef. We use a categorical number to denote an ingredient and represent every dish on the recipe as a set of ingredients. Agents take turns to act by selecting an ingredient and put it on the workplace, starting from the chef, A. A game ends if either a wrong ingredient is selected or all ingredients are prepared. We allow repetitive ingredients in a dish but preparing more than enough also leads to failure. Suppose we have a recipe, $[0, 1, 2, 6]$, $[2, 2, 4, 8, 9]$, **[2, 3, 6, 7, 7]** and $[1, 2, 2, 8, 9]$, and the target dish is the third dish. Then a successful action sequence

BAD	39.56 ± 2.62
CIRL	38.20 ± 0.41
COMA	33.26 ± 1.35
DDRQN	34.19 ± 1.32
MADDPG	34.49 ± 0.26
ToM-noBoB	53.52 ± 2.60
ToM	$\mathbf{62.46 \pm 3.79}$

Table 1: Unique identifier using percentage. ToM agents can simulate partners' reaction of self actions and choose the most ideal action to perform. In kitchen collaboration, the chef needs to clarify the target dish to the assistant as soon as possible so that only the correct ingredients are prepared. The most straightforward way is for the chef to choose the unique ingredient only included by the target dish. We calculate the probability for the chef to choose a unique ingredient if there is one. We claim that the frequency to choose the unique ingredient is related to the switch group accuracy in 5.3. As if the chef doesn't use unique ingredients to indicate the target dish, there have to be some other protocols formed between agents, which are usually group-specific.

can be $[3_A, 7_B, 6_A, 2_B, 7_A]$[2]. The order of the ingredients being prepared and who prepared what ingredient does not matter for the completion of the task. Some failure trajectories can be $[2_A, 0_B]$ or $[3_A, 7_B, 7_A, 7_B]$. At the end of each game, both agents get a reward ± 1 for success or failure, and there is no step reward or cost as the game proceeds. In this experiment, agent B's beliefs are k-dimensional distribution vectors. Hence, in centralized training, we let agent B sample an index k between 1 to K from its belief and send a one-hot vector with the k-th number being 1 to A as a discretized belief supervision. KL-divergence is used as the distance function for L^f.

There are W ingredients in total and we limit the maximum number of ingredients in a dish by M. In a recipe, all dishes are different from other dishes by at least one ingredient. A target dish is a one-hot K-dimensional vector, indicating which dish should be prepared. Only the chef knows the target, so we have ω_A as a K-dimensional one-hot vector and ω_B as an empty dumpy variable. In later of the paper, we refer a game as the combination of a recipe and a target.

All recipes are randomly generated. We synthesize a dish by randomly picking M numbers from $\{0, 1, ..., W\}$ with replacement. If W is selected, we remove it from the dish as only $\{0, 1, ..., W-1\}$ are valid ingredients. We include W in the selection process to have dishes with fewer than M ingredients. The order of the ingredients doesn't matter, and we will resample a dish if two dishes are duplicates in one recipe. There are 3335 possible dishes and 5.1×10^{12} possible games in total. To verify the robustness of our model, we keep the training dishes exclusive from the testing dishes. There are 2335 unique dishes in the training set and 1000 in the testing set. There are 7 and 3 million games in our training and testing set respectively. We evaluated our model and benchmarks using $K = 4, M = 5, W = 10$.

Results are shown in table 2 and figure 2a. BAD [9] achieves very close performance with our model. Their model has a BAD-agent that controls all other agents by sending partial policies to agents and using equation 1 in their paper to infer agents' private observations. Nevertheless, when the private observation space is small, the BAD-agent can form an one-to-one mapping between actions to private observations, forcing agents to reveal their private states by doing actions as told. Then the private states can flow among all agents through the BAD-agent, centralizing the execution process. In other words, their agents don't do counterfactual inference but tell their private states to the BAD-agent, who determine the joint action as a meta-agent. We refer it as a weakly centralized execution. One drawback of their work is that mappings between actions and private states are usually arbitrary, so the emerged protocols are exclusive to one group of agents. To test our hypothesis, in section 5.3, we conduct switch group experiments. The significant performance reduction illustrates the sub-optimality of the protocols developed by Foerster el al. [9]

In table 1, we also report the probability of the chef using the unique ingredients in the target dish in the first round. Using the unique identifier to disambiguate distractors is very common in human communication [2, 17]. Our model developed a protocol very compatible with human communication. In the future, we consider adding human-agent experiments to quantify the compatibility. Using unique ingredients to identify the target dish is not the only possible protocol between collaborative agents, but it is the most universal way across different groups.

[2]This is a trajectory generated by our model, notice the unique identifier of the target dish is selected by the teacher in the first round. Agents' usage of unique identifier is shown in table 1.

Table 2: Accuracies for kitchen collaboration and appointment scheduling tasks. In the first experiment, BAD has slightly better performance, but it is weakly centralized execution as the private agent states can flow among agents through the BAD-agent. COMA achieves 0.1% higher accuracy than ours in the second experiment. There is a centralized critic in COMA, so agents must have the same reward function. In section 5.3 we show that both of these methods learn group-specific protocols, while ToM can emerge task level protocol. DIAL with messages can achieve perfect performance as agents will share private states directly through the communication channel. We include this algorithm as an oracle. If we remove the communication channel from DIAL, its performance degrades considerably. As the communication channel enables agents to collide easily in appointment scheduling, DIAL is not used for the second experiment. CIRL is applicable when only one agent has hidden information. To show the convergence of the models, test curves are plotted in figure 2a and 2b.

	Kitchen	Appointment
BAD	$\mathbf{95.47 \pm 0.88}$	78.27 ± 3.57
CIRL	91.36 ± 0.80	N/A
COMA	89.98 ± 1.58	$\mathbf{94.90 \pm 1.65}$
DDRQN	69.77 ± 4.30	41.97 ± 4.91
DIAL	100.0 ± 0.00	N/A
DIRL no msg	17.99 ± 1.17	N/A
MADDPG	46.79 ± 2.45	57.94 ± 9.27
Random	5.67 ± 0.27	25.00
ToM	$\mathbf{95.19 \pm 1.01}$	$\mathbf{94.80 \pm 0.05}$

Table 3: Accuracy of switching the agents' partners. We also did ablation study for our model by replacing belief over belief in agent Q-functions with a hidden variable output from GRU (ToM-noBoB). ToM with partner's belief modeling has the highest performance in both experiments. BAD's performance drop proves our analysis of weakly centralized training and group-specific protocol. All models including ToM witness significant performance decrease for appointment scheduling. Since each agent only has 256 time tables to choose from, agents' action patterns toward calendars are easier to remember and take advantage of by their partners. So, a portion of the emerged protocol is inevitably group-specific. Yet, ToM still managed to form the most universal protocol.

	Kitchen	Appointment
BAD	37.97 ± 1.76	46.89 ± 3.16
CIRL	49.65 ± 2.16	N/A
COMA	43.90 ± 9.98	52.32 ± 2.00
DDRQN	27.20 ± 2.19	42.38 ± 4.62
MADDPG	15.18 ± 2.35	31.32 ± 13.99
ToM-noBoB	87.96 ± 0.64	61.18 ± 1.66
ToM	$\mathbf{92.51 \pm 0.95}$	$\mathbf{70.29 \pm 4.09}$

5.2 Appointment Scheduling

In the kitchen collaboration game, only the chef has a private agent state. We now propose a game where both agents have private information. Suppose there are two agents A and B, each having a private time table, and they want to schedule a meeting time available to both of them. We code private time table as a D-dimension binary vector with 0 meaning free and 1 meaning occupied. Both agents can perform three types of actions, inform, propose and reject. Rejecting ends the game indicating no common available time to the agents. Proposing ends the game indicating a time slot to meet. Informing stands for the speaker informs the other agent that an interval is occupied and unable to meet. Notice that we regulate the message space for informing to avoid trivial conversation. That is, agents are only allowed to say continuous occupied intervals. For instance, if agent A has a schedule $[0, 0, 1, 1, 1, 0, 1, 1]$, then it is allowed to say $(2), (3), (4), (2, 3), (3, 4), (2, 3, 4), (6), (7)$, and $(6, 7)$, but not $(2, 4), (4, 6), (0, 2)$ etc. Rejecting meetable schedules, proposing occupied time slots or informing wrong messages[3] all lead to game failure and every informing has a message cost to prevent long-lasting games. At the end of each game, both agents get +1 or -2 for success and failure, respectively, and every correct message has a cost of -0.1. Failures caused by all reasons: wrong

[3]Agents cannot send invalid information. Here wrong messages mean valid but wrong messages. For example, $(0, 1, 2)$ in the previous example, because slot 0 and 1 are not occupied. Numbers are 0-index in the example.

proposal, rejection or message, trigger the same failure reward. In this experiment, agents' beliefs should be 2^D-dimensional beliefs. Yet, even if $D = 8$, they are high-dimensional vectors, so we use a D-dimensional vector as a concatenated belief, with each dimension representing the probability for partner's that slot being occupied[4]. Continuous vectors are discretized to be supervisions by rounding each number to the closest decimal, eg. $[0.43, 0.67]$ to $[0.4, 0.7]$. The distance function used for L^f is L2-norm between the supervision vector and the predicted concatenated belief.

Both agents' private time tables are randomly generated with each slot sampled from a Bernoulli distribution with $p = 0.5$. As there are only 2^{2D} games, to prevent overfitting, we make sure all schedules in the testing sets are not included in the training sets. There are about 180 schedules, 30k games, and 80 schedules, 6k games in the training and testing sets respectively. We used $D = 8$ in our experiments. Notice that the baseline of random guess accuracy is 25%; the baseline of random propose based on self time table is 50%. Our results and analysis are reported in captions of table 2 and figure 2b.

5.3 Flexible Group Assignment

Another contribution of our work is that with ToM considered in action selections, universal protocols compatible cross groups are more likely to be developed. That is to say, our model can integrate task level essence into the protocol so that even if the partner in testing is different from that in training, they can still achieve good performance using their protocols learned separately. Task level universal protocol is the prerequisite of flexible agents assignment. Suppose there is a group of agents collaborating to complete tasks. If one of them is broken and needs to be substituted, we expect another agent to take the broken agent's place at once. The new agent doesn't need to have experience working with this particular group if they have a task level protocol. Most centralized training approaches only learn group-specific protocols.

Since for each split, we run two experiments, we switch the agents between the two experiments. Let agent A_1 trained with agent B_1 to collaborate with agent B_2 and let agent A_2 to collaborate with agent B_1 in testing. Only when agents understand the task rather than form group-specific tacit agreements can they maintain the performance after switch partners. See table 3 for results. Methods with centralized structures like BAD and COMA suffer from significant performance deduction, revealing the limitation of their group protocols.

6 Conclusion

In this paper, we proposed an adaptive multiagent learning algorithm, with which cooperative agents can develop effective collaborations in MAS with imperfect information. Agents learn how to infer others' hidden mental states with only observations of partners' actions and no verbal communications. Our algorithm outperforms all other decentralized execution approaches and shows the least performance drop in group switching experiments, demonstrating that our agents form strategies for the game instead of emerging ad-hoc protocols only compatible to specific partners. In the future, we aim to explore games involving continuous agent state space, where beliefs cannot be modeled as vectors but have to be parameterized.

References

[1] C. L. Baker, J. Jara-Ettinger, R. Saxe, and J. B. Tenenbaum, "Rational quantitative attribution of beliefs, desires and percepts in human mentalizing," *Nature Human Behaviour*, vol. 1, no. 4, p. 0064, 2017.

[2] H. De Weerd, R. Verbrugge, and B. Verheij, "Higher-order theory of mind in the tacit communication game," *Biologically Inspired Cognitive Architectures*, vol. 11, pp. 10–21, 2015.

[3] P. Doshi and P. J. Gmytrasiewicz, "Monte carlo sampling methods for approximating interactive pomdps," *Journal of Artificial Intelligence Research*, vol. 34, pp. 297–337, 2009.

[4] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 2137–2145.

[4]This is only for b_{-i}, we still use 2^D-dimensional beliefs for b_i.

[5] S. Sukhbaatar, R. Fergus *et al.*, "Learning multiagent communication with backpropagation," in *Advances in Neural Information Processing Systems*, 2016, pp. 2244–2252.

[6] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1146–1155.

[7] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.

[8] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate to solve riddles with deep distributed recurrent q-networks," *arXiv preprint arXiv:1602.02672*, 2016.

[9] J. N. Foerster, F. Song, E. Hughes, N. Burch, I. Dunning, S. Whiteson, M. Botvinick, and M. Bowling, "Bayesian action decoder for deep multi-agent reinforcement learning," *arXiv preprint arXiv:1811.01458*, 2018.

[10] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[11] D. Premack and G. Woodruff, "Does the chimpanzee have a theory of mind?" *Behavioral and brain sciences*, vol. 1, no. 4, pp. 515–526, 1978.

[12] W. Yoshida, R. J. Dolan, and K. J. Friston, "Game theory of mind," *PLoS computational biology*, vol. 4, no. 12, p. e1000254, 2008.

[13] M. Kinney and C. Tsatsoulis, "Learning communication strategies in multiagent systems," *Applied intelligence*, vol. 9, no. 1, pp. 71–91, 1998.

[14] A. Das, S. Kottur, J. M. Moura, S. Lee, and D. Batra, "Learning cooperative visual dialog agents with deep reinforcement learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2951–2960.

[15] Y. Han and P. Gmytrasiewicz, "Learning others' intentional models in multi-agent settings using interactive pomdps," in *Advances in Neural Information Processing Systems*, 2018, pp. 5634–5642.

[16] H. De Weerd, R. Verbrugge, and B. Verheij, "Theory of mind in the mod game: An agent-based model of strategic reasoning." in *ECSI*, 2014, pp. 128–136.

[17] H. De Weerd, D. Diepgrond, and R. Verbrugge, "Estimating the use of higher-order theory of mind using computational agents," *The BE Journal of Theoretical Economics*, vol. 18, no. 2, 2017.

[18] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," *AAAI/IAAI*, vol. 1998, pp. 746–752, 1998.

[19] J. F. Fisac, M. A. Gates, J. B. Hamrick, C. Liu, D. Hadfield-Menell, M. Palaniappan, D. Malik, S. S. Sastry, T. L. Griffiths, and A. D. Dragan, "Pragmatic-pedagogic value alignment," *arXiv preprint arXiv:1707.06354*, 2017.

[20] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[21] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, 1989.

[22] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. M. de Cote, "A survey of learning in multiagent environments: Dealing with non-stationarity," *arXiv preprint arXiv:1707.09183*, 2017.

[23] E. Yang and D. Gu, "Multiagent reinforcement learning for multi-robot systems: A survey," tech. rep, Tech. Rep., 2004.

[24] L. Bu, R. Babu, B. De Schutter *et al.*, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 2, pp. 156–172, 2008.

[25] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," *PloS one*, vol. 12, no. 4, p. e0172395, 2017.

[26] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[27] A. Lazaridou, A. Peysakhovich, and M. Baroni, "Multi-agent cooperation and the emergence of (natural) language," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=Hk8N3Sclg

[28] S. Havrylov and I. Titov, "Emergence of language with multi-agent games: Learning to communicate with sequences of symbols," in *Advances in neural information processing systems*, 2017, pp. 2149–2159.

[29] K. Evtimova, A. Drozdov, D. Kiela, and K. Cho, "Emergent language in a multi-modal, multi-step referential game," *arXiv preprint arXiv:1705.10369*, 2017.

[30] A. Lazaridou, K. M. Hermann, K. Tuyls, and S. Clark, "Emergence of linguistic communication from referential games with symbolic and pixel input," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=HJGv1Z-AW

[31] K. Wagner, J. A. Reggia, J. Uriagereka, and G. S. Wilkinson, "Progress in the simulation of emergent communication and language," *Adaptive Behavior*, vol. 11, no. 1, pp. 37–69, 2003.

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

[33] Y. Wen, Y. Yang, R. Luo, J. Wang, and W. Pan, "Probabilistic recursive reasoning for multi-agent reinforcement learning," *arXiv preprint arXiv:1901.09207*, 2019.

[34] E. Wei, D. Wicke, D. Freelan, and S. Luke, "Multiagent soft q-learning," in *2018 AAAI Spring Symposium Series*, 2018.

[35] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1352–1361.

[36] D. Wang and Q. Liu, "Learning to draw samples: With application to amortized mle for generative adversarial learning," *arXiv preprint arXiv:1611.01722*, 2016.

[37] T. Jaakkola, M. I. Jordan, and S. P. Singh, "Convergence of stochastic iterative dynamic programming algorithms," in *Advances in neural information processing systems*, 1994, pp. 703–710.

[38] E. Choi, A. Lazaridou, and N. de Freitas, "Compositional obverter communication learning from raw visual input," *arXiv preprint arXiv:1804.02341*, 2018.

[39] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[40] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 AAAI Fall Symposium Series*, 2015.