Learning Others' Intentional Models in Multi-Agent Settings Using Interactive POMDPs

Yanlin Han Piotr Gmytrasiewicz
Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60607

Abstract

Interactive partially observable Markov decision processes (I-POMDPs) provide a principled framework for planning and acting in a partially observable, stochastic and multi-agent environment, extending POMDPs to multi-agent settings by including models of other agents in the state space and forming a hierarchical belief structure. In order to predict other agents' actions using I-POMDP, we propose an approach that effectively uses Bayesian inference and sequential Monte Carlo (SMC) sampling to learn others' intentional models which ascribe them beliefs, preferences and rationality in action selection. For problems of various complexities, empirical results show that our algorithm accurately learns models of other agents and has superior performance in comparison with other methods. Our approach serves as a generalized reinforcement learning algorithm that learns over other agents' transition, observation and reward functions. It also effectively mitigates the belief space complexity due to the nested belief hierarchy.

Introduction

Partially observable Markov decision processes (POMDPs) (Kaelbling, Littman, and Cassandra 1998) provide a principled, decision-theoretic framework for planning under uncertainty in a partially observable, stochastic environment. An autonomous agent operates rationally in such settings by maintaining a belief of the physical state at any given time, in doing so it sequentially chooses the optimal actions that maximize the future rewards. Therefore, solutions of POMDPs are mappings from an agent's beliefs to actions. Although POMDPs can be used in multi-agent settings, it is doing so under strong assumptions that the effects of other agents' actions are implicitly treated as noise and folded in the state transitions, such as recent Bayes-adaptive POMDPs (Ross, Draa, and Pineau 2007), infinite generalized policy representation (Liu, Liao, and Carin 2011), infinite POMDPs (Doshi-Velez et al. 2013). Therefore, an agent's beliefs about other agents are not in the solutions of POMDPs.

Interactive POMDP (I-POMDP) (Gmytrasiewicz and Doshi 2005) are a generalization of POMDP to multi-agent settings by replacing POMDP belief spaces with interactive hierarchical belief systems. Specifically, it augments the plain beliefs about the physical states in POMDP by including models of other agents, which forms a hierarchical belief structure that represents an agent's belief about the physical state, belief about the other agents and their beliefs about others' beliefs. The models of other agents included in the new augmented state space consist of two types: the intentional models and subintentional models. The sophisticated intentional model ascribes beliefs, preferences, and rationality to other agents (Gmytrasiewicz and Doshi 2005), while the simpler subintentional model, such as finite state controllers (Panella and Gmytrasiewicz 2016), does not. Solutions of I-POMDPs map an agent's belief about the environment and other agents' models to actions. Therefore, it is applicable to all important agent, human, and mixed agent-human applications. It has been clearly shown (Gmytrasiewicz and Doshi 2005) that the added sophistication for modeling others as rational agents results in a higher value function which dominates the one obtained from simply treating others as noise, which implies the modeling superiority of I-POMDPs for multi-agent systems over other approaches.

However, the interactive belief modification for I-POMDPs results in a drastic increase of the belief space complexity, adding to the curse of dimensionality: the complexity of the belief representation is proportional to belief dimensions due to exponential growth of agent models with increase of nesting level. Since exact solutions to POMDPs are proven to be PSPACE-complete for finite horizon and undecidable for infinite time horizon (Papadimitriou and Tsitsiklis 1987), the time complexity of the more generalized I-POMDPs, which may contain multiple POMDPs and I-POMDPs of other agents, is greater than or equal to PSPACE-complete for finite horizon and undecidable for infinite time horizon. Due to this severe space complexity, currently no complete belief update has been accomplished using the sophisticated intentional models over entire interactive belief space. There are only partial updates on other agents' sole beliefs about the physical states (Doshi and Gmytrasiewicz 2009) and indirect approach such as subintentional finite state controllers (Panella and Gmytrasiewicz 2016). Therefore, in order to unleash the full modeling power of intentional models and apply I-POMDPs to more realistic settings, a good approximation algorithm for computing the nested interactive belief and predicting other agents' actions is crucial to the trade-off between solution quality and computation complexity.

To address this issue, we propose a Bayesian approach that utilizes customized sequential Monte Carlo sampling algorithms (Doucet, De Freitas, and Gordon 2001) to obtain approximating solutions to interactive I-POMDPs and implement the algorithms in a software package[1]. Specifically, We assume that models of other agents are unknown and learned from imperfect observations of the other agents' behaviors. We parametrize other agents' intentional models and maintain a belief over them, making sequential Bayesian updates using only observations from the environment. Since this Bayesian inference task is analytically intractable, to approximate the posterior distribution, we devise a customized sequential Monte Carlo method to descend the belief hierarchy and sample all model parameters at each nesting level, starting from the interactive particle filter (I-PF) (Doshi and Gmytrasiewicz 2009) for I-POMDP belief update.

Our approach, for the first time, successfully learns others' models over the entire intentional model space which contains their initial belief, transition, observation and reward functions, making it a generalized reinforcement learning method for multi-agent settings. Our algorithm accurately predicts others' actions on various problem settings, therefore enables the modeling agent to make corresponding optimal action to maximize its own rewards. By approximating Bayesian inference using a customized sequential Monte Carlo sampling method, we significantly mitigate the belief space complexity of the I-POMDPs.

Background

POMDP

A Partially observable Markov decision process (POMDP) (Kaelbling, Littman, and Cassandra 1998) is a general reinforcement learning model for planning and acting in a single-agent, partially observable, stochastic domain. It is defined for a single agent i as:

$$POMDP_i = \langle S, A_i, \Omega_i, T_i, O_i, R_i \rangle \tag{1}$$

Where the meaning for each element in the 6-tuple is:

- S is the set of states of the environment.
- A_i is the set of agent i's possible actions
- Ω_i is the set of agent i's possible observations
- $T_i : S \times A_i \times S \to [0, 1]$ is the state transition function
- $O_i : S \times A_i \times \Omega_i \to [0, 1]$ is the observation function
- $R_i : S \times A_i \to R_i$ is the reward function.

Given the definition above, an agent's belief about the state can be represented as a probability distribution over S. The belief update can be simply done using the following formula, where α is the normalizing constant:

$$b(s') = \alpha O(o, s, a) \sum_{s \in S} T(s', a, s) b(s) \tag{2}$$

Given the agent's belief, then the optimal action, a^*, is simply part of the set of optimal actions, $OPT(b_i)$, for the belief state defined as:

$$OPT(b_i) = \underset{a_i \in A_i}{\arg\max} \Big\{ \sum_{s \in S} b_i(s) R(s, a_i) \tag{3}$$
$$+ \gamma \sum_{o_i \in \Omega_i} P(o_i | a_i, b_i) \times U(SE(b_i, a_i, o_i)) \Big\}$$

Particle Filter

The Markov Chain Monte Carlo (MCMC) method (Gilks et al., 1996) is widely used to approximate probability distributions when they are unable to be computed directly. It generates samples from a posterior distribution $\pi(x)$ over state space x, by simulating a Markov chain $p(x'|x)$ whose state space is x and stationary distribution is $\pi(x)$. The samples drawn from p converge to the target distribution π as the number of samples goes to infinity.

In order to make MCMC work on sequential inference task, especially sequential decision makings under Markov assumptions, sequential versions of Monte Carlo methods have been proposed and some of them are capable of dealing with high dimensionality and/or complexity problems, such as particle filters (Del Moral and Pierre 1996). At each time step, particle filters draw samples (or particles) from a proposal distribution, commonly $p(x_t|x_{t-1})$, which is essentially the conditional distribution of the current state x_t given the previous x_{t-1}, then use the observation function $p(y_t|x_t)$ to compute the importance weight for each particle and resample all particles according to the weights.

The Model

I-POMDP framework

An interactive POMDP of agent i, I-POMDP i, is defined as:

$$I\text{-}POMDP_i = \langle IS_{i,l}, A, \Omega_i, T_i, O_i, R_i \rangle \tag{4}$$

where $IS_{i,l}$ is the set of interactive states of the environment, defined as $IS_{i,l} = S \times M_{i,l-1}, l \geq 1$, where S is the set of states and $M_{i,l-1}$ is the set of possible models of agent j, and l is the strategy level. A specific class of models are the $(l-1)$th level *intentional* models, $\Theta_{j,l-1}$, of agent j: $\theta_{j,l-1} = \langle b_{j,l-1}, A, \Omega_j, T_j, O_j, R_j, OC_j \rangle$, $b_{j,l-1}$ is agent j's belief nested to level $(l-1)$, $b_{j,l-1} \in \Delta(IS_{j,l-1})$, and OC_j is j's optimality criterion. The intentional model $\theta_{j,l-1}$, sometimes is referred to as *type*, can be rewritten as $\theta_{j,l-1} = \langle b_{j,l-1}, \hat{\theta}_j \rangle$, where $\hat{\theta}_j$ includes all elements of the intentional model other than the belief and is called the agent j's frame.

The $IS_{i,l}$ could be defined in an inductive manner (note that when $\hat{\theta}_j$ is usually known, $\hat{\theta}_j$ reduces to b_j):

$$\begin{aligned}
&IS_{i,0} = S, && \theta_{j,0} = \{\langle b_{j,0}, \hat{\theta}_j \rangle : b_{j,0} \in \Delta(S)\} \\
&IS_{i,1} = S \times \theta_{j,0}, && \theta_{j,1} = \{\langle b_{j,1}, \hat{\theta}_j \rangle : b_{j,1} \in \Delta(IS_{j,1})\} \\
&\ldots\ldots && \\
&IS_{i,l} = S \times \theta_{j,l-1}, && \theta_{j,l} = \{\langle b_{j,l}, \hat{\theta}_j \rangle : b_{j,l} \in \Delta(IS_{j,l})\}
\end{aligned} \tag{5}$$

And all other remaining components in an I-POMDP are similar to those in a POMDP:

[1]`https://github.com/solohan22/IPOMDP.git`

- $A = A_i \times A_j$ is the set of joint actions of all agents.
- Ω_i is the set of agent i's possible observations.
- $T_i : S \times A_i \times S \rightarrow [0, 1]$ is the state transition function.
- $O_i : S \times A_i \times \Omega_i \rightarrow [0, 1]$ is the observation function.
- $R_i : IS \times A_i \rightarrow R_i$ is the reward function.

Interactive belief update

Given all the definitions above, the interactive belief up-date can be performed as follows:

$$b_i^t(is^t) = Pr(is^t|b_i^{t-1}, a_i^{t-1}, o_i^t) \tag{6}$$
$$= \alpha \sum_{is^{t-1}} b(is^{t-1}) \sum_{a_j^{t-1}} Pr(a_j^{t-1}|\theta_j^{t-1}) T(s^{t-1}, a^{t-1}, s^t)$$
$$\times O_i(s^t, a^{t-1}, o_i^t) \sum_{o_j^t} O_j(s^t, a^{t-1}, o_j^t) \tau(b_j^{t-1}, a_j^{t-1}, o_j^t, b_j^t)$$

Unlike plain belief update in POMDP, the interactive belief update in I-POMDP takes two additional sophistications into account. Firstly, the probabilities of other's actions given its models (the second summation) need to be computed since the state of physical environment now depends on both agents' actions. Secondly, the agent needs to update its beliefs based on the anticipation of what observations the other agent might get and how it updates (the third summation).

Then the optimal action, a^*, for the case of infinite horizon criterion with discounting, is part of the set of optimal actions, $OPT(\theta_i)$, for the belief state defined as:

$$OPT(\theta_i) = \underset{a_i \in A_i}{\arg\max} \Big\{ \sum_{is \in IS} b_{is}(s) ER_i(is, a_i) \tag{7}$$
$$+ \gamma \sum_{o_i \in \Omega_i} P(o_i|a_i, b_i) \times U(\langle SE_{\theta_i}(b_i, a_i, o_i), \hat{\theta}_i \rangle) \Big\}$$

Sampling Algorithms

The Interactive Particle Filter (I-PF) (Doshi and Gmytrasiewicz 2009) was proposed as a filtering algorithm for interactive belief update in I-POMDP. It generalizes the classic particle filter algorithm to multi-agent settings and uses the state transition function as the proposal distribution, which is usually used in a specific particle filter algorithm called bootstrap filter (Gordon, ect 1993). However, due to the enormous belief space, I-PF assumes that other agent's frame $\hat{\theta}_j$ is known to the modeling agent, therefore simplifies the belief update process from $S \times \Theta_{j,l-1}$ to a significantly smaller space $S \times b_{j,l-1}$.

The intuition of our algorithm is to assign appropriate prior distributions over agent j's all possible models $\theta_j =< b_j(s), A_j, \Omega_j, T_j, O_j, R_j, OC_j >$ and sample from each dimension of them. At each time step, we update all samples using perceived observations, namely computing and assigning weights to each sample, and resample them according to the weights. At last, since it is a randomized Monte Carlo method, to prevent the learning algorithm from converging to incorrect models, we add another resampling step to sample from the neighboring similar models given the current samples. Consequently, our algorithm is able to maintain a probability distribution of the most possible models of other agents and eventually learn the optimal actions of them.

Algorithm 1: Interactive Belief Update

```
$\tilde{b}_{k,l}^t$ = InteractiveBeliefUpdate($\tilde{b}_{k,l}^{t-1}$, $a_k^{t-1}$, $o_k^t$, $l > 0$)
1  For $is_k^{(n),t-1} =< s^{(n),t-1}, \theta_{-k}^{(n),t-1} > \in \tilde{b}_{k,l}^{t-1}$,
2    sample $a_{-k}^{t-1} \sim P(A_{-k}|\theta_{-k}^{(n),t-1})$
3    sample $s^{(n),t} \sim T_k(S^t|S^{(n),t-1}, a_k^{t-1}, a_{-k}^{t-1})$
4    for $o_{-k}^t \in \Omega_{-k}$:
5      if $l = 1$:
6        $b_{-k,0}^{(n),t}$ = Level0BeliefUpdate($b_{-k,0}^{(n),t-1}$, $a_{-k}^{t-1}$
                                   , $o_{-k}^t$, $\theta_{-k}^{(n),t-1}$)
7        $\theta_{-k}^{(n),t} =< b_{-k,0}^{(n),t}, \hat{\theta}_{-k}^{(n),t-1} >$
8        $is_k^{(n),t} =< s^{(n),t}, \theta_{-k}^{(n),t} >$
9      else:
10       $b_{-k,l-1}^{(n),t}$ = InteractiveBeliefUpdate($\tilde{b}_{-k,l-1}^{t-1}$
                                   , $a_{-k}^{t-1}$, $o_{-k}^t$, $l-1$)
11       $\theta_{-k}^{(n),t} =< b_{-k,l-1}^{(n),t}, \hat{\theta}_{-k}^{(n),t-1} >$
12       $is_k^{(n),t} =< s^{(n),t}, \theta_{-k}^{(n),t} >$
13     $w_t^{(n)} = O_{-k}^{(n)}(o_{-k}^t|s^{(n),t}, a_k^{t-1}, a_{-k}^{t-1})$
14     $w_t^{(n)} = w_t^{(n)} \times O_k(o_k^t|s^{(n),t}, a_k^{t-1}, a_{-k}^{t-1})$
15     $\tilde{b}_{k,l}^{temp} =< is_k^{(n),t}, w_t^{(n)} >$
16 normalize all $w_t^{(n)}$ so that $\sum_{n=1}^N w_t^{(n)} = 1$
17 resample from $\tilde{b}_{k,l}^{temp}$ accroding to normalized $w_t^{(n)}$
18 resample $\theta_{-k}^{(n),t}$ according to neighboring similar models
19 return $\tilde{b}_{k,l}^t = is_k^{(n),t}$
```

The interactive belief update described in Algorithm 1 is similar to I-PF in terms of the recursive Monte Carlo sampling and nesting hierarchy, but it has three major differences. Firstly, the belief update is over the entire intentional model space of other agents, therefore the initial set of N samples $\tilde{b}_{k,l}^{t-1} =< b_{-k,l-1}^{(n),t-1}, A_{-k}, \Omega_{-k}, T_{-k}^{(n)}, O_{-k}^{(n)}, R_{-k}^{(n)}, OC_j >$, where k here denotes the modeling agent and $-k$ denotes all other modeled agents. We only assume that the actions A_{-k}, observations Ω_{-k} and optimal criteria OC_j are known, as in a multi-agent game the rules are usually known to all agents or could be obtained through intelligence. Secondly, it is intuitive to see that the observation function $O_{-k}^{(n)}(o_{-k}^t|s^{(n),t}, a_k^{t-1}, a_{-k}^{t-1})$ in line 13 is now randomized as well, as each of them is a particular observation function of that agent. Lastly, we add another resampling step in line 18 in order to avoid divergence, by resampling each dimension of the model samples from a Gaussian distribution with the mean of current sample value. Intuitively, similar models are resampled from a relatively tight neighboring region of the current model samples to maintain the learning accuracy.

Algorithm 1 can be viewed as two major steps. The *importance sampling* step (line 1 to line 16) samples from belief priors $\tilde{b}_{k,l}^{t-1}$ and propagates forward using related proposal distributions and computes the weights of all samples. And the *selection* or *resapmling* step (line 17 to line 18) resamples according to weights and similar models. Specifically, the algorithm starts from a set of initial priors $is_k^{(n),t-1}$, for each of them, it samples other agents' optimal action a_{-k}^{t-1} from its policy $P(A_{-k}|\theta_{-k}^{(n),t-1})$, which is solved using a very efficient POMDP solver called Perseus[2] (Spaan and Vlassis 2005). Then it samples the physical state s^t using the state transition $T_k(S^t|S^{(n),t-1}, a_k^{t-1}, a_{-k}^{t-1})$. Once a_{-k}^{t-1} and s^t are sampled, the algorithm calls for the 0-level belief update (line 5 to 8), described in Algorithm 2, to update other agents' plain beliefs $b_{-k,0}^t$ if the current nesting level l is 1, or recursively calls for itself at a lower level $l-1$ (line 9 to 12) if the current nesting level is greater than 1. The sample weights $w_t^{(n)}$ are computed according to observation likelihoods of modeling and modeled agents (line 13, 14), and then got normalized so that they sum up to 1 (line 16). Lastly, the algorithm resamples the intermediate samples according to the computed weights (line 17) and resamples another time from similar neighboring models (line 18).

Algorithm 2: Level-0 Belief Update

```
$b_{k,0}^t$ =Level0BeliefUpdate($b_{k,0}^{t-1}$,$a_k^{t-1}$,$o_k^t$, $T_k^{(n)}$,$O_k^{(n)}$)
1   $P(a_{-k}^{t-1}) = 1/a_{-k}^{t-1}$
2   for $s^t \in S$:
3       for $s^{t-1}$:
4           for $a_{-k}^{(t-1)} \in A_{-k}$:
5               $P^{(n)}(s^t|s^{t-1}, a_k^{t-1}) =$
                    $T_k^{(n)}(s^t|s^{t-1}, a_k^{t-1}, a_{-k}^{t-1}) \times P(a_{-k}^{t-1})$
6           $sum^{(n)} += P^{(n)}(s^t|s^{t-1}, a_k^{t-1}) b_{k,0}^{t-1}(s^{t-1})$
7       for $a_{-k}^{(t-1)} \in A_{-k}$:
8           $P^{(n)}(o_k^t|s^t, a_k^{t-1}) +=$
                    $O_k^{(n)}(o_k^t|s^t, a_k^{t-1}, a_{-k}^{t-1}) P(a_{-k}^{t-1})$
9       $b_{k,0}^t = sum^{(n)} \times P^{(n)}(o_k^t|s^t, a_k^{t-1})$
10  normalize and return $b_{k,0}^t$
```

The 0-level belief update, described in Algorithm 2, is similar to POMDP belief update but treats other agents' actions as noise and randomized the state transition function and observation function as input parameters. It assume other agents in the environment choose their actions according to a uniform distribution (line 1), therefore is essentially a no-information model. For each possible action $a_{-k}^{(t-1)}$, it computes the actual state transition (line 5) and actual observation function (line 8) by marginalizing over others' actions, and returns the normalized belief $b_{k,0}^t$. Notice that the transition function $T_k^{(n)}(s^t|s^{t-1}, a_k^{t-1}, a_{-k}^{t-1})$ and observation function $O_k^{(n)}(o_k^t|s^t, a_k^{t-1}, a_{-k}^{t-1})$ are now both samples from input arguments, depending on model parameters of the actual agent on the 0th level.

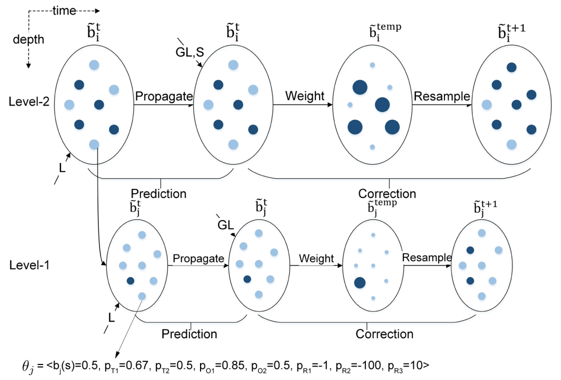

Figure 1: An illustration of interactive belief update for two agents and 1 level nesting.

In figure 1, we illustrate the interactive belief update using the problem discussed in the following section . Suppose there are two agents i and j in the environment, the sample size is 8 and the nesting level is 2, the subscripts in figure 1 denotes the corresponding agents and each dot represents a particular belief sample. The propagate step corresponds to line 2 to 12 in Algorithm 1, the weight step corresponds to line 13 to 16, the resample step corresponds to line 17 and 18. The belief update for a particular level-0 model sample ($\theta_j = \langle b_j(s) = 0.5, p_{T1} = 0.67, p_{T2} = 0.5, p_{O1} = 0.85, p_{O2} = 0.5, p_{R1} = -1, p_{R2} = -100, p_{R3} = 10\rangle$) is solved using Algorithm 2, and the optimal action is computed by calling the Perseus POMDP solver.

Experiments

Setup

We present the results using the multi-agent tiger game (Gmytrasiewicz and Doshi 2005) with various settings. The multi-agent tiger game is a generalization of the classical single agent tiger game (Kaelbling, Littman, and Cassandra 1998) with adding observations which caused by others' actions. The generalized multi-agent game contains additional observations regarding other players, while the state transition and reward function involve others' actions as well.

Let's see a specific game example with known parameters: there are a tiger and a pile of gold behind two doors respectively, two players can both listen for a growl of the tiger and a creak caused by the other player, or open doors which resets the tiger's location with equal probability. Their observation toward the tiger and the other player are both relatively high (0.85 and 0.9 respectively). No matter triggered by which player, the reward for listening action is -1, opening the tiger door is -100 and opening the gold door is 10.

[2]http://www.st.ewi.tudelft.nl/~mtjspaan/pomdp/index_en.html

Table 1: Parameters for transition functions

S	A	TL	TR
TL	L	p_{T1}	$1 - p_{T1}$
TR	L	$1 - p_{T1}$	p_{T1}
*	OL	p_{T2}	$1 - p_{T2}$
*	OR	$1 - p_{T2}$	p_{T2}

Table 2: Parameters for observation functions

S	A	GL	GR
TL	L	p_{O1}	$1 - p_{O1}$
TR	L	$1 - p_{O1}$	p_{O1}
*	OL	p_{O2}	$1 - p_{O2}$
*	OR	$1 - p_{O2}$	p_{O2}

Table 3: Parameters for reward functions

S	A	R
*	L	p_{R1}
TL	OL	p_{R1}
TR	OR	p_{R2}
TL	OR	p_{R3}
TR	OL	p_{R3}

For the sake of brevity, we restrict the experiments to a two-agent setting and nesting level of one, but the sampling algorithm is extensible to any number of agents and nesting levels in a straightforward manner. Recall that an interactive POMDP of agent i is defined as a six tuple $I\text{-}POMDP_i = \langle IS_{i,l}, A, \Omega_i, T_i, O_i, R_i \rangle$. Thus for the specific setting of multi-agent tiger problem:

- $IS_{i,1} = S \times \theta_{j,0}$, where $S = \{$tiger on the left (TL), tiger on the right (TR)$\}$ and $\theta_{j,0} =< b_j(s), A_j, \Omega_j, T_j, O_j, R_j, OC_j >\}$.
- $A = A_i \times A_j$ is a combination of both agents' possible actions: listen (L), open left door (OL) and open right door(OR).
- Ω_i is all the combinations of each agent's possible observations: growl from left (GL) or right (GR), combined with creak from left (CL), right (CR) or silence (S).
- $T_i = T_j : S \times A_i \times A_j \times S \to [0, 1]$ is a joint state transition probability that involves both actions.
- $O_i : S \times A_i \times A_j \times \Omega_i \to [0, 1]$ becomes a joint observation probability that involves both actions. O_j is symmetric of O_i with respect to the joint actions.
- $R_i : IS \times A_i \times A_j \to R_i$: agent i gets corresponding rewards when he listens, opens the wrong door and opens the correct door respectively. They are independent of j's actions.

Parameter Space

For the experiments of multi-agent tiger game, we want to learn over all possible intentional models of the other agent j: $\theta_j =< b_j(s), A_j, \Omega_j, T_j, O_j, R_j, OC_j >$. We only make reasonable assumptions that A_j and Ω_j are known, OC_j is infinite horizon with discounting. Now what we really want to learn are as follow:

- b_j^0: the initial belief of agent j about the physical state.
- T_j: the transition function of agent j, which can be parametrized by two parameters p_{T1} and p_{T2}, as shown in Table 1.
- O_j: the observation function of agent j, which can be parametrized by two parameters p_{O1} and p_{O2}, as shown in Table 2.
- T_j: the reward function of agent j, which can be parametrized by three parameters p_{R1}, p_{R2} and p_{R3}, as shown in Table 3.

We could easily see that it is a enormous 8-dimensional parameter space to learn from: $b_j^0 \times p_{T1} \times p_{T2} \times p_{O1} \times p_{O2} \times p_{R1} \times p_{R2} \times p_{R3}$, where $b_j \in [0, 1] \subset \mathbb{R}$, $p_{T1} \in [0, 1] \subset \mathbb{R}$, $p_{T2} \in [0, 1] \subset \mathbb{R}$, $p_{O1} \in [0, 1] \subset \mathbb{R}$, $p_{O2} \in [0, 1] \subset \mathbb{R}$, $p_{R1} \in [-\infty, +\infty] \subset \mathbb{R}$, $p_{R2} \in [-\infty, +\infty] \subset \mathbb{R}$, $p_{R3} \in [-\infty, +\infty] \subset \mathbb{R}$.

We mainly reduce this huge space by two means: utilizing Monte Carlo sampling methods and giving them problem-specific priors so that they are not over informative but provide enough information for the algorithm to learn from.

Results

For the actual experiments, we fix the number of samples to be 2000 and run it on a two agent tiger game simulation as described above. We run experiments for learning three difference models of agent j:

1. $\theta_{j1} =< 0.5, 0.67, 0.5, 0.85, 0.5, -1, -100, 10 >$
2. $\theta_{j2} =< 0.5, 1.00, 0.5, 0.95, 0.5, -1, -10, 10 >$
3. $\theta_{j3} =< 0.5, 0.66, 0.5, 0.85, 0.5, 10, -100, 10 >$

These models are all very special cases and carefully chosen in order to verify the correctness and evaluate the performance of our algorithm. For instance, the first model is a sophisticated one when the other agent is actually modeling his opponent using a subintentional model, while the second is a classic single-agent POMDP and the third is a very simple one but contains a large model space. We want to investigate if our framework is able to correctly and efficiently learn these models through these experiments.

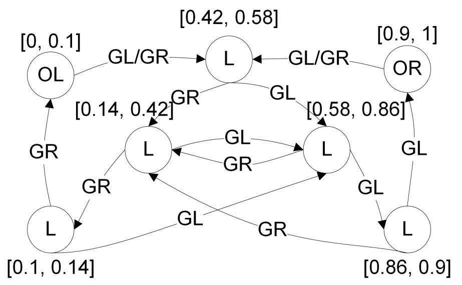

Figure 2: Optimal policy of a no-information model.

The aim of first experiment to try to learn relatively complicated models of agent j with $\theta_j =< 0.5, 0.67, 0.5, 0.85, 0.5, -1, -100, 10 >$, who assumes others' actions are drawn from a uniform distribution. Equivalently, agent j's actual policy, as shown in figure 2, is to look for three consecutive growls from same direction and then open the corresponding door. For this particular experiment, we simulated the observation history for agent i, for the sake of firstly verifying the correctness of our algorithm, excluding the impacts of uncertainties from hearing accuracy. The simulated observation history is as follows: {GL,S GL,S GL,S GL,CR GL,S GL,S GL,S GR,CR GL,S GL,S GL,S GR,CR GL,S GL,S GL,S GR,CR GR,S GR,S GR,S GR,CL GR,S GR,S GR,S GR,CL GR,S GR,S GR,S GR,CL GR,S GR,S GR,S GR,CL GR,S GR,S GR,S GR,CL GR,S GR,S GR,S GL,CL GL,S GL,S GL,S GR,CR GL,S GL,S GL,S GR,CR GR,S GR,S}

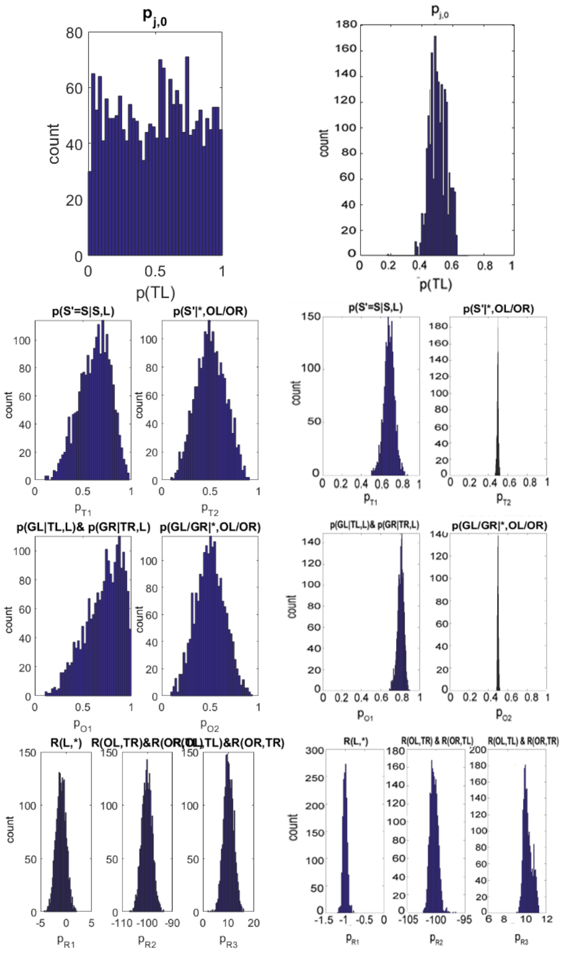

Figure 3: Assigned priors and learned posterior distributions over model parameters for model $\theta_{j1} =< 0.5, 0.67, 0.5, 0.85, 0.5, -1, -100, 10 >$.

The priors we assign to each parameters are shown in figure 3, specifically, they are uniform U(0,1) for b_j^0, Beta(5,3) with mode 0.67 for p_{T1}, Beta(5,5) for p_{T2}, Beta(3.5,1.4) with mode 0.85 for p_{O1}, Beta(5,5) for p_{O2}, Gaussian N(-1,2) for p_{R1}, N(-100,4) for p_{R2}, and N(10,2) for p_{R3}.

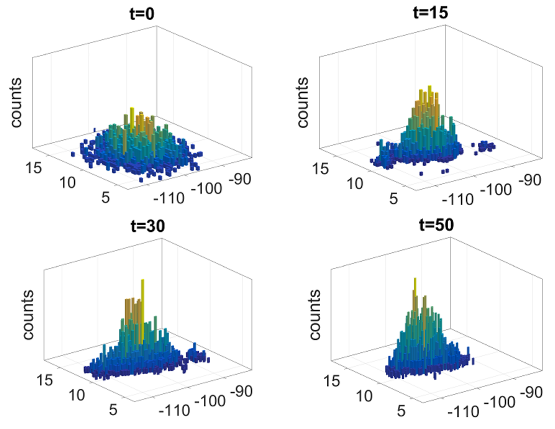

Figure 4: 3D histogram of all model samples.

After 50 time steps, the algorithm converge to a posterior distribution over agent j's intentional models, the results are also given in figure 3. Since the parameter space of agent j's models is 8-dimensional, here we only show the marginal distributions of each parameter space in histograms. We can easily see that the majority of samples are centered around the true parameter values.

We use principal component analysis (PCA) (Abdi and Williams 2010) to reduce sample dimensionality to 2-dimensional and plot them out in a 3-dimensional histogram, as shown in Figure 4. It starts from a Gaussian-like prior and gradually converges to the most likely models. Eventually the mean value of this cluster $\langle$ 0.49, 0.69, 0.49, 0.82, 0.51, -0.95, -99.23, 10.09 $\rangle$ is very close to the true model. Here we give two examples from the big cluster after 50 time steps: $\langle$0.56, 0.66, 0.49, 0.84, 0.59, -0.95, -101.37, 11.42$\rangle$ and $\langle$0.51, 0.68, 0.52, 0.89, 0.56, -1.33, -98.39, 12.55$\rangle$. The former has a corresponding optimal policy of [0—OL—0.10—L—1], while the latter has a [0—OL—0.09—L–0.91—OR—1], which are both extremely close to the optimal policy of the true model: [0—OL—0.1—L–0.9—OR—1]. Consequently, the framework is able to predict other agents' actions with high accuracy.

We tested the performance of our algorithms in terms of prediction accuracy towards others' actions. We compared the results with other modeling approaches, such as a frequency-based approach, in which agent j is assumed to choose his action according to a fixed but unknown distribution, and a no-information model which treats j's actions purely as uniform noise. The results shown in figure 5 are averaged plots of 10 random runs, each of which has 50 time steps. It shows clearly that the intentional I-POMDP approach has significantly lower error rates as agent i perceives more observations. The subintenional model assume j's action is draw from a uniform distribution, therefore has

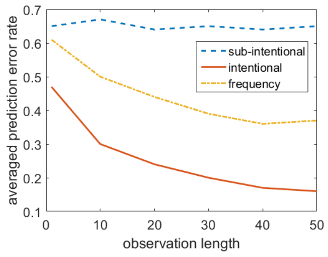

Figure 5: Prediction error rate vs observation length.

a fixed high error rate. The frequency based approach has certain learning ability but is far from enough sophisticated for modeling a fully rational agent.

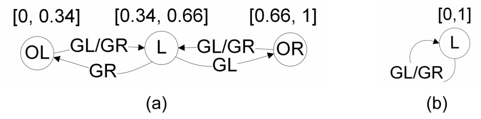

Figure 6: (a) optimal policy for $\theta_j = \langle$ 0.5, 1, 0.5, 0.95, 0.5, -1, -10, 10 $\rangle$. (b) optimal policy for $\theta_j = \langle$0.5, 0.66, 0.5, 0.85, 0.5, 10, -100, 10$\rangle$.

In the second experiment, we run our algorithm on actual observations for 30 time steps until it converges, and try to learn models of a simpler classic POMDP with high listening accuracy of 0.95 and small penalty of -10, e.g. the agent j alternately opens door and listens as shown in Figure 6 left. The actual model of j is $\theta_j = \langle$ 0.5, 1, 0.5, 0.95, 0.5, -1, -10, 10 $\rangle$, the priors assigned to b_j^0, p_{T1}, p_{T2}, p_{O1}, p_{O2}, p_{R2}, p_{R3} are U(0,1), Beta(2,0.5), Beta(10,10), Beta(19,1), Beta(10,10), N(-1,1), N(-10,2), N(10,2), and the actual observation history is {GR,S GL,CR GL,S GL,CL GL,S GL,CR GL,S GL,CL GL,S GR,S GR,CL GR,CL GL,S GR,S GR,S GL,CL GR,S GL,CR GR,S GR,CR GR,CR GR,CL GL,S GL,S GL,S GL,CR GL,S GL,CL GR,S GR,S}.

Similarly, we report the learned posterior distributions over model parameters in figure 7. We observe an interesting pattern that while some parameters, such as $b_{j,0}$, p_{T2} and p_{O2} are concentrated around the actual values, others like p_{T1} and p_{O1} become more dispersed than initial priors. The intuition behind is that the penalty and reward are -10 and 10, so one listening of reward -1 is enough for making decision of opening doors. That is to say, as long as tiger likely remains behind the same door when agent listens (the meaning of p_{T1}) and has a reliable hearing accuracy (the meaning of p_{O1}), there are many models which satisfy this particular observation sequence, hence our algorithm learns them all.

For conciseness, we show the average prediction error rates for both second and third experiments in figure 9. Both results are averaged among 10 random runs, each of which has 30 time steps. In the second experiment in figure 9(a), the intentional I-POMDP approach still has significantly lower error rates than others.

In the last experiment, we wants to learn a model of θ_j=

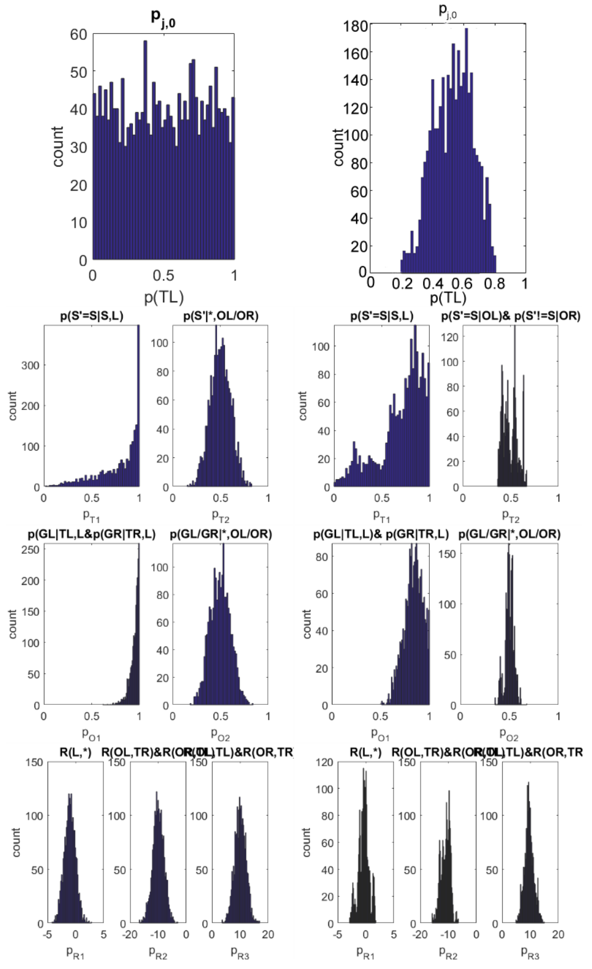

Figure 7: Learned posterior distributions for model $\theta_j = \langle$ 0.5, 1, 0.5, 0.95, 0.5, -1, -10, 10 $\rangle$.

$\langle$0.5, 0.66, 0.5, 0.85, 0.5, 10, -100, 10$\rangle$, who always listens since the listening penalty is now equal to the reward, as shown in figure 6(b). For brevity, we only show the marginal distributions over model parameters in figure 8. The priors assigned to b_j^0, p_{T1}, p_{T2}, p_{O1}, p_{O2}, p_{R2}, p_{R3} are U(0,1), Beta(5,3), Beta(10,10), Beta(3.5,1.4), Beta(10,10), N(10,1), N(-100,2), N(10,2), and the actual observation history i learns from is {GL,S GL,S GR,S GL,S GL,CL GR,S GR,S GL,CL GR,S GL,S GL,S GR,S GL,S GL,S GL,S GL,CL GR,S GL,S GL,S GL,S}. We can see that all three reward parameters are correctly learned, while samples of p_{T1}, p_{T2}, p_{O1} and p_{O2} are not very concentrated to their true values but close to their corresponding priors, since intuitively they become less important and can be in a relatively loose region due to the increased p_{R1}=10. Lastly, the performance comparison is given in figure 9(b).

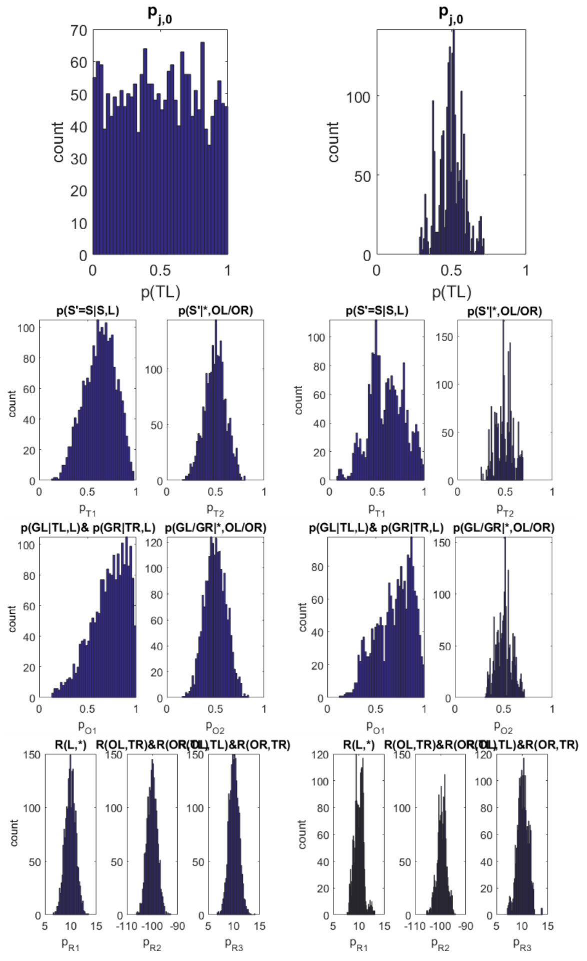

Figure 8: Learned posterior distributions for model $\theta_j = \langle 0.5, 0.66, 0.5, 0.85, 0.5, 10, -100, 10\rangle$.

Conclusions and Future Work

We have described a new approach to learn other agents' models by approximating the interactive belief update using Bayesian inference and Monte Carlo sampling methods. Our framework correctly learns others' models over the entire intentional model space and therefore is a generalized reinforcement learning algorithm for multi-agent settings. It also effectively mitigates the belief space complexity and has a significant better performance than other approaches in terms of predicting others' actions.

In the future, in order to fully evaluate the practicability on larger problem space, more multi-agent problems of various sizes could be tested. Due to computation complexity, experiments on higher nesting levels are currently limited. Thus, more efforts could be made on utilizing nonparametric Bayesian methods which inherently deal with nested belief structures.

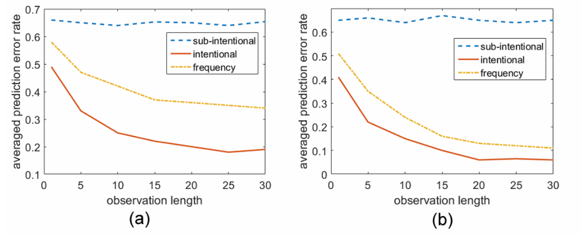

Figure 9: (a) Prediction error rate vs observation length for $\theta_j = \langle$ 0.5, 1, 0.5, 0.95, 0.5, -1, -10, 10 $\rangle$. (b) for $\theta_j = \langle$0.5, 0.66, 0.5, 0.85, 0.5, 10, -100, 10$\rangle$.

References

Abdi, H. and Williams, L.J., 2010. Principal component analysis. Wiley interdisciplinary reviews: computational statistics, 2(4), pp.433-459.

Doucet, A., De Freitas, N. and Gordon, N., 2001. An introduction to sequential Monte Carlo methods. In *Sequential Monte Carlo methods in practice* (pp. 3-14). Springer New York.

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

Doshi-Velez, F. and Konidaris, G., 2013. Hidden parameter Markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. *arXiv:1308.3513*.

Del Moral, P., 1996. Non-linear filtering: interacting particle resolution. Markov processes and related fields, 2(4), pp.555-581.

Doshi, P., Zeng, Y., and Chen, Q. 2009. Graphical models for interactive POMDPs: representations and solutions. Autonomous Agents and Multi-Agent Systems 18.3: 376-416.

Gilks, W.R., Richardson, S. and Spiegelhalter, D.J., 1996. Introducing markov chain monte carlo. Markov chain Monte Carlo in practice, 1, p.19.

Gmytrasiewicz, P. J., and Doshi, P. 2005. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research* 24(1): 49-79.

Gmytrasiewicz, P. J., and Doshi, P. 2005. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research* 24(1): 49-79.

Gordon, N.J., Salmond, D.J. and Smith, A.F., 1993, April. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In IEE Proceedings F (Radar and Signal Processing) (Vol. 140, No. 2, pp. 107-113). IET Digital Library.

Kaelbling, L.P., Littman, M.L. and Cassandra, A.R., 1998. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1), pp.99-134.

Spaan, M.T. and Vlassis, N., 2005. Perseus: Randomized point-based value iteration for POMDPs. Journal of artificial intelligence research, 24, pp.195-220.

Liu, M., Liao, X. and Carin, L., 2011. The infinite regionalized policy representation. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (pp. 769-776).

Panella, A. and Gmytrasiewicz, P., 2016, March. Bayesian Learning of Other Agents' Finite Controllers for Interactive POMDPs. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Pearl, J., 1988. Probabilistic reasoning in intelligent systems: Networks of plausible reasoning.

Ross, S., Chaib-draa, B. and Pineau, J., 2007. Bayes-adaptive pomdps. In *Advances in neural information processing systems* (pp. 1225-1232).