Towards Robust Model Identification in Interactive Influence Diagrams

Yifeng Zeng
Dept. of Computer Science
Aalborg University, Denmark
yfzeng@cs.aau.dk

Prashant Doshi
Dept. of Computer Science
University of Georgia, USA
pdoshi@cs.uga.edu

Abstract

Modeling the perceived behaviors of other agents improves the performance of an agent in multiagent interactions. We utilize the language of interactive influence diagrams to model repeated interactions between the agents, and ascribe procedural models to other agents. Procedural models offer the benefit of understanding how others arrive at their behaviors. As model spaces are often bounded, the true models of others may not be present in the model space. In addition to considering the case when the true model is within the model space, we investigate the case when the true model may fall outside the space. We then seek to identify models that are relevant to the observed behaviors of others and show how the agent may learn to identify these models. We evaluate the performance of our methods in two repeated games and provide experimental results in support.

1 Introduction

Modeling other agents cohabiting the environment is an important topic of research in multiagent systems. Accurate behavioral models of others facilitate optimal decision-making in multiagent settings. Consequently, agent modeling finds significant applications in several areas such as robotics, interactive software and games. Because the true models of others are often private, especially in non-cooperative settings, we may discover them only by observing the actions of the other agents. However, as the space of possible models is very large, we typically restrict the models to those that can be represented using a *modeling language*. Thus, the problem of discovery is transformed into the more manageable problem of *identification* of the true model from the space of models represented using the modeling language.

Therefore, two issues are of importance when modeling other agents. First, is the selection of the modeling language. A large amount of previous work focuses on identifying directly the strategies (or agent functions) of the other agents. For example, Carmel and Markovitch [3], use finite state automatons to model agents' strategies. Saha *et al.* [12] use Chebychev polynomials to approximate agents' decision functions in negotiations. The second issue is the learning method used to gradually identify the agents' true models. Existing approaches include Bayesian learning [15], learning finite state automata [3] and polynomial approximation [12].

While knowledge of the behavioral strategies of others is indeed what is needed, it is also important to understand how others arrived at their behaviors. Besides providing intuitive reasons for the strategies, the *procedural* knowledge may help preclude certain strategies of others, deeming them impossible because of the structure of the environment. In the context of learning models of others, our focus in this paper is on identifying the likely model(s) from a set using Bayesian learning. While there are uncountably infinite numbers of agent functions, there are only countable computable models. Hence current modeling languages such as finite automata necessarily restrict the model space with the implicit assumption that the true model is contained or approximable in this space. In the absence of this assumption, Bayesian learning is not guaranteed to converge and in fact, may become undefined.

In this paper, we utilize the language of *interactive influence diagrams* (I-IDs) [5] to model interactions between agents. We ascribe procedural models to the other agents – the models may be IDs, Bayesian networks (BN) [11], or I-IDs themselves leading to recursive modeling. We use the I-IDs to model repeated games, though as mentioned in [5], they are applicable to sequential games as well. Given the assumption that the true model of the other agent lies within the set of models that we consider, standard Bayesian learning is sufficient to update the likelihood of each candidate model (also called model weight) given the observation histories of others' actions.

Perhaps, a more realistic case is when we are uncertain that the true model is indeed within the bounded model space. For this case, we present a technique that identifies a model or a weighted combination of models whose predictions are *relevant* to the observed action history. Using previous observations of others' actions and predictions of the candidate models, we learn how the predictions may be related to the observation history. In other words, we learn to *classify* the predictions of the candidate models using the previous observation history as the training set. Thus, we seek the hidden function that may possibly relate the candidate models to the true model.

We then update the likelihoods of the candidate models. As a Bayesian update may be inadequate, we utilize the similarity between the predictions of a candidate model and the observed actions as the likelihood of the model. In this context, we measure the *mutual information* of the predicted actions by a candidate model and the observed action. This provides a natural measure of the dependence between the candidate and true models, possibly due to some shared behavioral aspects. We show that under certain conditions, our approach is

guaranteed to converge. We empirically evaluate the performance of our approach on multiple problem domains and demonstrate that an agent utilizing the approach gathers larger rewards on average as it better predicts the actions of the other agent.

2 Related Work

The benefits of utilizing graphical models for representing agent interactions have been recognized previously. Suryadi and Gmytrasiewicz [15] used IDs to model other agents and Bayesian learning to update the distributions over the models based on observed behavior. Additionally, they also consider the case where none of the candidate models reflect the observed behavior. In this situation, Suryadi and Gmytrasiewicz show how certain aspects of the IDs may be altered to better reflect the observed behavior. In comparison, we seek to find the underlying dependencies that often exist between candidate models and the true model. More recently, MAIDs [8] and NIDs [7] extend IDs to multiagent settings. MAIDs objectively analyze the game, efficiently computing the Nash equilibrium profile by exploiting the independence structure. NIDs extend MAIDs to include agents' uncertainty over the game being played and over models of the other agents. MAIDs provide an analysis of the game from an external viewpoint and the applicability of both is limited to single play in static games. Although I-IDs are similar to NIDs, their dynamic extensions, I-DIDs [5], model interactions that are extended over time.

Our work also contributes to the currrent work on opponent modeling besides the mentioned work in Section 1. In [1, 14], extensions of the minimax algorithm to incorporate different opponent strategies (rather than just being rational) are provided. However, this line of work focuses on improving the applicability of the minimax algorithm and uses agent functions as models. It assumes that the true model of the opponent is within the set of candidate models. As mentioned previously, Saha et al. [12] ascribe orthogonal Chebychev polynomials as agent functions. They provide an algorithm to learn the coefficients of the polynomials using the observation history. However, both the degree and the number of polynomials is fixed a'priori thereby bounding the model space.

3 Background: Interactive Influence Diagrams

We briefly describe interactive influence diagrams (I-IDs) [5] for modeling two-agent interactions and illustrate their application using a simple example.

3.1 Syntax and Solution

In addition to the usual chance, decision, and utility nodes, I-IDs include a new type of node called the *model* node (hexagon in Fig. $1(a)$). The probability distribution over the model node represents an agent, say i's, belief over the candidate models of the other agent.

In addition to the model node, I-IDs differ from IDs by having a chance node, A_j, that represents the distribution over the other agent's actions, and a dashed link, called a *policy link*.

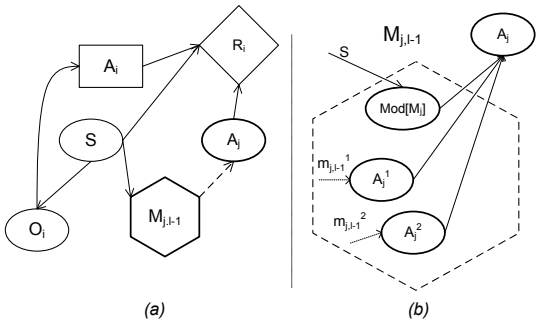


Figure 1: (a) A generic I-ID for agent i situated with one other agent j. The hexagon is the model node whose structure we show in (b). Members of the model node may be IDs,BNs or I-IDs themselves (m_j^1, m_j^2); not shown here for simplicity) whose decision nodes are mapped to the corresponding chance nodes (A_j^1, A_j^2). Depending on the value of node, $Mod[M_j]$, distribution of the chance node is assigned to A_j with some probability.

The model node $M_{j,l-1}$ contains as its values the alternative computational models ascribed by i to the other agent j in a lower level $l-1$. Formally, we denote a model of j as $m_{j,l-1}$ which resides in a lower level within an I-ID. A model in the model node, for example, may itself be an I-ID, in which case the recursion terminates when a model is an ID or a Bayesian network (BN). We observe that the model node and the dashed policy link that connects it to the chance node, A_j, could be represented as shown in Fig. 1(b). Once an I-ID or ID of j is solved and the optimal decisions are determined, the decision node is transformed into a chance node [1]. The chance node has the decision alternatives as possible states and is given a probability distribution to the states. Specifically, if OPT is the set of optimal actions obtained by solving the I-ID (or ID) like $m_{j,l-1}^1$, then $Pr(a_j \in A_j^1) = \frac{1}{|OPT|}$ if $a_j \in OPT$, 0 otherwise. The different chance nodes (A_j^1, A_j^2), one for each model, and additionally, the chance node labeled $Mod[M_j]$ form the parents of the chance node, A_j. The states of $Mod[M_j]$ denote the different models of j. The distribution over $Mod[M_j]$ is i's belief over j's candidate models (model weight) given the physical state S. The conditional distribution of the chance node, A_j, is a *multiplexer* that assumes the distribution of each of the action nodes (A_j^1, A_j^2) depending on the state of $Mod[M_j]$. In other words, when $Mod[M_j]$ has the state m_j^1, the chance node A_j assumes the distribution of the node A_j^1, and A_j assumes the distribution of A_j^2 when $Mod[M_j]$ has the state m_j^2.

Solution of an I-ID proceeds in a bottom-up manner, and is implemented recursively. We start by solving the lower level models, which are traditional IDs or BNs. Their solutions

[1] If j's models are BNs a chance node representing j's decisions will be directly mapped into a chance node in the model node

provide probability distributions over the other agents' actions, which are entered in the corresponding chance nodes found in the model node of the I-ID. Given the distributions over the actions within the different chance nodes (one for each model of the other agent), the I-ID is transformed into a traditional ID. During the transformation, the conditional distribution of the node, A_j, is populated such that the node assumes the distribution of each of the chance nodes depending on the state of the node, $Mod[M_j]$. The transformed I-ID is a traditional ID that may be solved using the standard expected utility maximization method [13].

3.2 Illustration

We illustrate I-IDs using an example application to the public good (PG) game with punishment (Table 1) explained in detail in [6]. Two agents, i and j, must either contribute some resource to a public pot or keep it for themselves. To make the game more interesting, we allow agents to contribute the full (FC) or partial (PC) portion of their resources though they could defect without making any contribution (D). The value of resources in the public pot is shared by the agents regardless of their action and is discounted by c_i for each agent i, where $c_i \in (0, 1)$ is the marginal private return. As defection is a dominating action, we introduce a punishment P to penalize the defecting agents and to promote contribution. In addition, a non-zero cost c_p of punishing is incurred by the contributing agents. For simplicity, we assume each agent has the same value X_T of private resources and makes a partial contribution of $\frac{1}{2}X_T$.

i, j	FC	PC	D
FC	$2c_iX_T$, $2c_jX_T$	$\frac{3}{2}X_Tc_i - \frac{1}{2}c_p$, $\frac{1}{2}X_T + \frac{3}{2}X_Tc_j - \frac{1}{2}P$	$c_iX_T - c_p$, $X_T + c_jX_T - P$
PC	$\frac{1}{2}X_T + \frac{3}{2}X_Tc_i - \frac{1}{2}P$, $\frac{3}{2}X_Tc_j - \frac{1}{2}c_p$	$\frac{1}{2}X_T + c_iX_T$, $\frac{1}{2}X_T + c_jX_T$	$\frac{1}{2}X_T + \frac{1}{2}c_iX_T - \frac{1}{2}P$, $X_T + \frac{1}{2}c_jX_T - P$
D	$X_T + c_iX_T - P$, $c_jX_T - c_p$	$X_T + \frac{1}{2}c_iX_T - P$, $\frac{1}{2}X_T + \frac{1}{2}c_jX_T - \frac{1}{2}P$	X_T, X_T

Table 1: PG game with punishment. Based on punishment, P, and marginal return, c_i, agents may choose to contribute than defect.

We let agents i and j play the PG game repeatedly a finite number of times and aim for larger average rewards. After a round, each agent observes the simultaneous action of its opponent. Except for the observation of their actions, no additional information is shared between the agents.

As discovered in field experiments with humans [2], different types of agents play the PG differently. To act rationally, i ascribes candidate behavioral models to j. We assume the models are procedural taking the form of IDs and BNs.

For illustration, let agent i consider four models of j (m_j^1, m_j^2, m_j^3, and m_j^4) in the model node at time t, as shown in Fig. 2. The first two model, m_j^1 and m_j^2, are simple IDs where

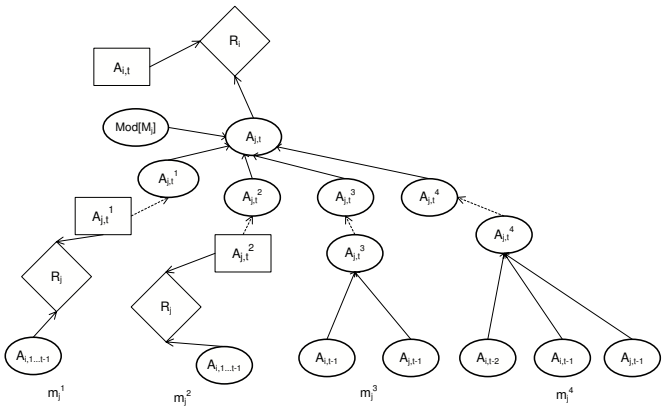


Figure 2: Example I-ID for the repeated PG game with four models ascribed to j. The dotted lines represent the mapping between decision (chance) nodes in j's models and chance nodes in the model node.

the chance node $A_{i,\langle 1,\cdots,t-1\rangle}$ represents the frequencies of the different actions of agent i in the game history (from 1 to time $t-1$). However, the two IDs have different reward functions in the value node. The model m_j^1 has a typically low marginal private return, c_j, and represents a reciprocal agent who contributes only when it expects the other agent to contribute as well. The model m_j^2 has a high c_j and represents an altruistic agent who prefers to contribute during the play. The third model, m_j^3, is a BN representing that j's behavior relies on its own action in the previous time step ($A_{j,t-1}$) and i's previous action ($A_{i,t-1}$). m_j^4 represents a more sophisticated decision process. Agent j considers not only its own and i's actions at time $t-1$ (chance nodes $A_{i,t-1}$ and $A_{j,t-1}$), but also agent i's actions at time $t-2$ ($A_{i,t-2}$). It indicates that j relies greatly on the history of the interaction to choose its actions at time t. We point out that these four models reflect typical behaviors of humans in the field experiments.

The weights of the four models are the probability distribution over the values of the chance node, $Mod[M_j]$. As agent i is unaware of the true model of j, it may begin by assigning a uniform distribution to $Mod[M_j]$. Over time, this distribution is updated to reflect any information that i may have about j's model.

4 Model Identification in I-IDs

As we mentioned previously, agent i hypothesizes a limited number of candidate models of its opponent j, $M_j = \{m_j^1, \ldots, m_j^p, \ldots, m_j^n\}$, and intends to ascertain the true model, m_j^*, of j in the course of interaction. On observing j's action, where the observation in round t is denoted by o_i^t, i may update the likelihoods (weights) of the candidate models in the model node of the I-ID. Gradually, the model that emerges as most likely may be hypothesized to be the true model of j.

However, because the space of candidate models is often bounded [2], the true model of j may not be within the model space. For this case, we may intuitively expect to identify a model or a weighted combination of models within the model space whose predictions are *relevant* in determining those of the true model. Consequently, the model identification problem in this case involves finding the weights of the models in the model space and updating the weights using the observations. We begin by exploring the traditional setting where the true model, m_j^* is in the model space, M_j, and move on to the challenge where the true model is outside the model space.

4.1 Case 1: $m_j^* \in M_j$ (Traditional)

Let $o_i^{1:t-1}$ be the history of agent i's observations up to time $t-1$. Agent i's belief over the models of j at time step $t-1$ may be written as, $Pr(M_j|o_i^{1:t-1}) \stackrel{def}{=} \langle\, Pr(m_j^1), Pr(m_j^2),\ldots,Pr(m_j^*),\ldots, Pr(m_j^n)\,\rangle$. If o_i^t is the observation at time t, agent i may update its belief on receiving the observation using a straightforward Bayesian process. We show the update of the belief for some model, m_j^n, in Eq. 1.

$$Pr(m_j^n|o_i^t) = \frac{Pr(o_i^t|m_j^n)Pr(m_j^n|o_{1:t-1})}{\sum_{m_j \in M_j} Pr(o_i^t|m_j)Pr(m_j)} \tag{1}$$

Here, $Pr(o_i^t|m_j^n)$ is the probability of j performing its observed action given that its model is m_j^n. This may be obtained from the chance node A_j^n in the I-ID of i.

Eq. 1 provides a way for updating the weights of models contained in the model node, M_j, given the observation history. In the context of the I-ID, agent i's belief over the other's models updated using the process outlined in Eq. 1 will converge in the limit. Formally,

Proposition 1 (Bayesian Learning in I-IDs) *If an agent's prior belief assigns a non-zero probability to the true model of the other agent, its posterior beliefs updated using Bayesian learning will converge with probability 1.*

Proof of Proposition 1 relies on showing that the sequence of the agent's beliefs updated using Bayesian learning is known to be a Martingale [4]. Proposition 1 then follows from a straightforward application of the Martingale convergence theorem (§4 of Chapter 7 in [4]).

The above result does not imply that an agent's belief always converges to the true model of the other agent. This is due to the possible presence of models of the other agent that are *observationally equivalent* to the true model. For example, all models of j that induce identical distributions over all possible future observation paths are said to be observationally equivalent for agent i. When a particular observation history obtains, agent i is unable to distinguish between the observationally equivalent models of j. In other words, observationally equivalent models generate distinct behaviors for histories which are never observed.

[2] Saha et al. [12] bound the space of agent functions to those that can be modeled using finite degree Chebychev polynomials and settle for a best fit.

4.2 Case 2: $m_j^* \notin M_j$

For computability purposes, the space of candidate models ascribed to j is often bounded. In the absence of prior knowledge, i may be unaware whether j's true model, m_j^*, is within the model space. If $m_j^* \notin M_j$ and in the absence of observationally equivalent models, Bayesian learning may be inadequate. For this case, we naturally expect to find a candidate model or a combination of models from the space whose predictions are *relevant* in determining those of the true model.

4.2.1 Relevant Models and Mutual Information

As the true model may lie outside the model space, our objective is to identify candidate models whose predictions exhibit a mutual pattern with the observed actions of the other agent. We interpret the existence of a mutual pattern as evidence that the candidate model shares some behavioral aspects of the true model. In order to do this, we introduce a notion of *relevance* between a model in M_j and the true model, m_j^*. Let a_j^* be the observed action of the other agent j and $\bar{a}_j^*$ denote any other action from its set of actions. Define $Pr(a_j^1|m_j^n, a_j^*)$ as the probability that a candidate model of j, m_j^n, predicts action a_j^1 given that a_j^* is observed.

Definition 1 (Relevant Model) *If for a model, m_j^n, there exists an action, a_j^1 such that, $Pr(a_j^1|m_j^n, a_j^*) \geq Pr(a_j^1|m_j^n, \bar{a}_j^*)$, where $a_j^1 \in OPT(m_j^n)$, then m_j^n is a* relevant *model.*

In predicting a_j^1, model m_j^n may utilize the past observation history. Definition 1 formalizes the intuition that a relevant model predicts an action that is likely to correlate with a particular observed action of the other agent. We note that the above definition generalizes to a relevant combination of models in a straightforward way. Given Definition 1, we need an approach that assigns large probabilities to the relevant model(s) in the node $Mod[M_j]$ over time. We proceed to show one way of computing these probabilities.

We begin by observing that the chance nodes, $Mod[M_j]$, A_j and the mapped chance nodes, $A_j^1, A_j^2, \ldots$, form a BN, as shown in Fig. $3(a)$. We seek the weights of models in $Mod[M_j]$ that would allow the distribution over A_j to resemble that of the observed actions. Subsequently, we may map the problem to one of classifying the predicted actions of the individual models to the observed action of j, and using the classification function for deriving the model weights. Because the candidate models are independent of each other, the BN is *naive* and the classification reduces to learning the parameters (conditional probability distributions) of the naive BN using say, the maximum likelihood approach with Dirichlet priors. For multiple agents, the models may exhibit dependencies in which case we learn a general BN. We show the equivalent naive BN in Fig. $3(b)$.

As relevant models hint at possible dependencies with the true model, we utilize the *mutual information* (MI) between the chance nodes A_j and say, A_j^n, as a measure of the likelihood of the model, m_j^n, in $Mod[M_j]$. MI is a well-known way of quantifying the mutual dependency between two random variables.

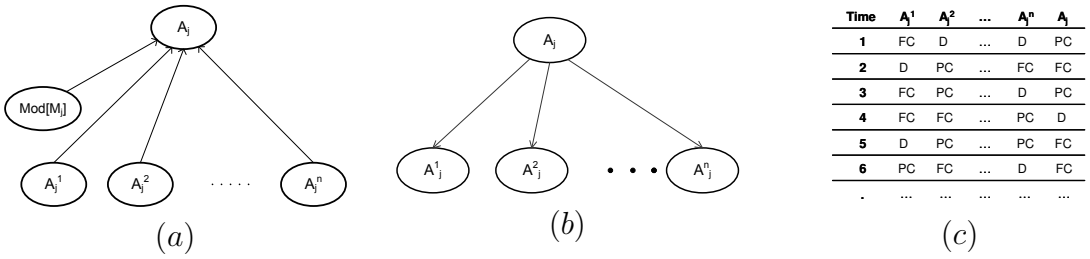


Figure 3: (a) The BN in the I-ID of agent i; (b) The equivalent naive BN that we use for classifying the outcomes of the candidate models to the observation history; (c) Example of the training set used for learning the naive BN for PG. The actions in column A_j are observations of i, while remaining columns are obtained from models.

Definition 2 (Mutual Information) *The mutual information (MI) of the true model, m_j^* and a candidate model, m_j^n, is computed as:*

$$\begin{aligned} MI(m_j^n, m_j^*) &\stackrel{def}{=} Pr(A_j^n, A_j) log[\tfrac{Pr(A_j^n, A_j)}{Pr(A_j^n)Pr(A_j)}] \\ &= Pr(A_j^n|A_j)Pr(A_j) log[\tfrac{Pr(A_j^n|A_j)}{Pr(A_j^n)}] \end{aligned} \tag{2}$$

Here, A_j^n is the chance node mapped from the model, m_j^n and A_j assumes the distribution of the observed actions generated by the true model, m_j^*.

Notice that MI is non-negative. The terms $Pr(A_j^n|A_j)$, $Pr(A_j^n)$ and $Pr(A_j)$ may be calculated from the conditional probability distributions of the naive BN. Here, the observed history of j's actions together with the predictions of the models over time may serve as the training set for learning the parameters of the naive BN. We show an example training set for PG in Fig. 3(c). Values of the columns, A_j^1, A_j^2, ..., A_j^n are obtained by solving the corresponding models and sampling the resulting distributions if needed. We utilize the normalized MI at each time step as the model weights in the chance node, $Mod[M_j]$.

4.2.2 Theoretical Results

Obviously, model m_j^n is irrelevant if $Pr(a_j|m_j^n, a_j^*) = Pr(a_j|\ m_j^n, \bar{a}_j^*)$ for each $a_j \in OPT(m_j^n)$. Then, the following proposition is trivially obtained.

Proposition 2 *If m_j^n is irrelevant, $MI(m_j^n, m_j^*) = 0$.*

Proposition 2 implies that relevant models are assigned a higher MI than irrelevant ones. To enable further analysis, we compare the relevance of a candidate model with that of another.

Definition 3 (Relevance Ordering) *Let a_j^* be some observed action of the other agent j. If for two relevant models, m_j^n and m_j^p, there exists an action, a_j^1, such that $Pr(a_j^1|m_j^n, a_j^*) \geq Pr(a_j^1|m_j^p, a_j^*)$ and $Pr(a_j^1|m_j^n, \bar{a}_j^*) \leq Pr(a_j^1|m_j^p, \bar{a}_j^*)$, where $a_j^1 \in OPT(m_j^n)$, $OPT(m_j^p)$ and $\bar{a}_j^*$ denotes all other actions of the true model, then m_j^n is a* more relevant *model than m_j^p.*

Given Definition 3, we may show that models that are more relevant are assigned a higher MI. Proposition 3 formalizes this observation (we do not show the proof due to space constraints).

Proposition 3 *If m_j^n is a more relevant model than m_j^p as per Definition 3 and m_j^* is the true model, then $MI\ (m_j^n, m_j^*) \geq MI\ (m_j^p, m_j^*)$.*

For the sake of completeness, we show that if the true model, m_j^*, is contained in the model space, our approach analogous to Bayesian learning will converge.

Proposition 4 (Convergence) *Given that the true model $m_j^* \in M_j$ and is assigned a non-zero probability, the normalized distribution of mutual information of the models converges with probability 1.*

The proof is intuitive and relies on the fact that the estimated parameters of the naive Bayes converge to the true parameters as the observation history grows (see chapter 3 of [10] for the proof when the *maximum a posteriori* approach is used for parameter estimation). Proposition 4 then follows because the terms $Pr(A_j^n, A_j)$, $Pr(A_j^n)$ and $Pr(A_j)$ used in calculating the MI are obtained from the parameter estimates.

Analogous to Bayesian learning, the distribution of MI may not converge to the true model in the presence of *MI-equivalent* models in M_j. In particular, the set of MI-equivalent models is larger and includes observationally equivalent models. However, consider the example where j's true strategy is to always select *FC*, and let M_j include the true model as well as a candidate model that generates the strategy of always selecting *D*. Though observationally distinct, the two candidate models are assigned equal MI because of the perceived dependency between the action of selecting *D* by the candidate strategy and selecting *FC* by the true one.

4.2.3 Algorithm

We briefly outline the algorithm that uses MI for model identification in Fig. 4. In each round t, agent i receives an observation of its opponent j's action (line 1). This observation together with solutions from candidate models of j (line 2), compose one sample in the training set D (line 3; see Fig. $3(c)$). The training set is used for learning the parameters of the naive BN (line 4) and subsequently for computing the model weights in the I-ID. Given the learned parameters, we compute the MI of each candidate model m_j^p and m_j^* (line 6).

The posterior probabilities are also used in the conditional probability distribution (CPD) of the chance node A_j in the I-ID (line 9). Notice that the CPD, $Pr(A_j|A_j^p, m_j^p)$, describes the relation between the predicted actions by candidate models and the observed actions. The normalized MI is assigned as the CPD of the chance node $Mod[M_j]$ in the I-ID (Lines 10-11). This distribution represents the updated weight over the candidate models of j. Given the updated model weights and the populated CPDs of the chance node A_j, we solve the I-ID of agent i to obtain its action.

Model Weight Update
Input: I-ID of agent i, observation o_i^t, training set D
1. Agent i receives an observation $o_{i,t}$
2. Solve the model, $m_{j,t}^p$ ($p = 1, \ldots, n$) to get actions for the chance nodes $A_{j,t}^p$ ($p = 1, \cdots, n$)
3. Add ($A_{j,t}^1$,$\cdots$, $A_{j,t}^p$, $\cdots$, $A_{j,t}^n$, $o_{i,t}$) as a sample into the training set *D*
4. Learn the parameters of the *naive BN* including the chance nodes, A_j^1,. . ., A_j^n, and A_j
5. **For each** A_j^p ($p = 1, \ldots, n$) **do**
6. Compute $MI(m_j^p, m_j^*)$ using Eq. 2
7. Obtain $Pr(A_j|A_j^p)$ from the learned *naive BN*
8. Populate CPD row of the chance node A_j using $Pr(A_j|A_j^p, m_j^p)$
9. **end for**
10. Normalize $MI(m_j^p, m_j^*)$
11. Populate CPD of the chance node $Mod[M_j]$ using MI

Figure 4: Algorithm revises the model weights in the model node, $Mod[M_j]$, on observing j's action using MI as a measure of likelihood, and populates CPDs of the chance node, A_j, by propagating the learned naive BN.

5 Performance Evaluation

We evaluate the effectiveness of the algorithm outlined in Fig. 4 in the context of the repeated public good game (Section 3.2) and repeated one-shot negotiations as in [12] though simplified. As we mentioned previously, if the true model falls outside the model space, Bayesian learning (BL) may be inadequate. Therefore, in addition to using BL for comparison, we also employ an adapted BL method (A-BL) that restarts the BL process when the likelihoods become zero by assigning candidate models prior weights using the frequency with which the actions are observed so far. Additionally, we also utilize the KL-Divergence (KL) to assign the likelihood of a candidate model. Lower is the KL between the distributions over A_j^n and A_j, larger is the likelihood of the corresponding model, m_j^n.

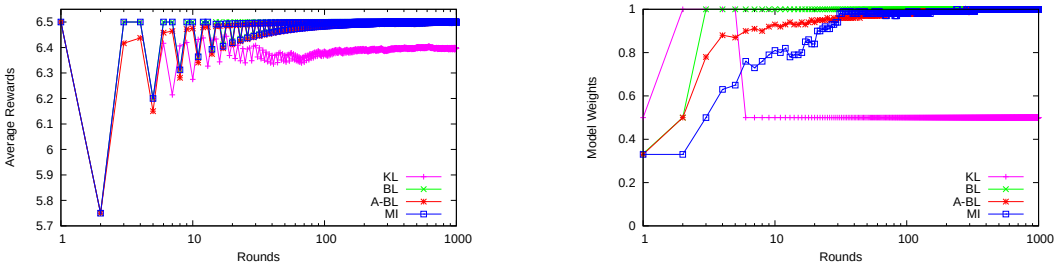
We let agents i and j play 1000 rounds of each game and report i's average rewards. To facilitate analysis, we also show the changing model weights across rounds that are assigned to the relevant and true models for the two cases – Case 1: $m_j^* \in M_j$, and Case 2: $m_j^* \notin M_j$.

5.1 Repeated Public Good Game

In the aforementioned PG game, we utilize the I-ID in Fig. 2 to model the interaction. Agent i plays with the opponent j multiple rounds of PG and aims to gain more rewards in the long run by discovering j's true behaviorial model. For case 1, we let the model space, M_j, contain three models, m_j^1, m_j^3, and m_j^4, that represent a reciprocal and two deliberative agents, respectively. We let agent j play using the true model, m_j^4. Fig. 5 (a) demonstrates the favorable performances of MI, BL and A-BL, which quickly converge to the true model and gain almost the same average rewards.

For the evaluation of case 2, agent i considers three candidate models of j, m_j^2, m_j^3, and m_j^4, while j uses the model m_j^1. We observe that MI obtains the largest average rewards over the long run in comparison to other updating methods. This is because MI finds the deliberative model, m_j^4, to be most relevant to the true model, m_j^1, that represents a reciprocal agent and acts according to the frequency of i's actions in the history. We note that MI does not monotonically increase but assigns the largest weight to the most relevant model at any point in time. Notice that both m_j^1 and m_j^4 consider actions of the other agent, and identical actions of the agents as promoted by a reciprocal model are more valuable. Both the A-BL and KL methods recognize the altruistic model, m_j^2, as the most likely.

(a) **Case 1**: $m_j^* = m_j^4$, M_j=$\{m_j^1, m_j^3, m_j^4\}$



(b) **Case 2**: $m_j^* = m_j^1$, M_j=$\{m_j^2, m_j^3, m_j^4\}$

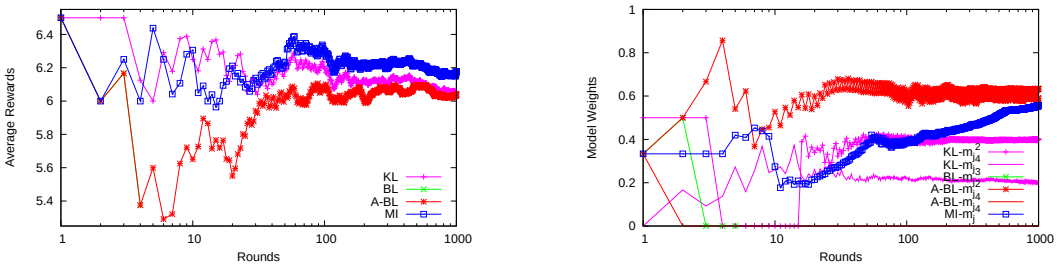


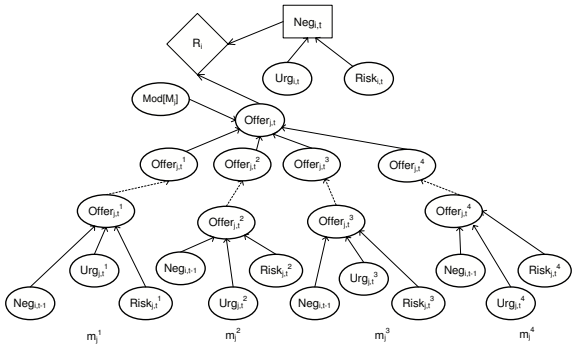
Figure 5: Performance profiles for the two cases in the repeated PG game. Notice that, for case 2, the model weight assigned using BL drops to zero

5.2 Repeated One-shot Negotiations

A seller agent i wants to sell an item to a buyer agent j. The buyer agent bargains with the seller and offers a price that ranges from Low, Mid, to $High$. The seller agent decides whether to $accept$ the offer (A), to $reject$ it immediately (R), or to $counter$ the offer (C). If i counters the offer, it expects a new price offer from agent j. Once the negotiation is completed successfully or fails, the agents restart a new one on a different item; otherwise, they continue to bargain. Figure 6(a) shows the payoffs of the seller agent when interacting with the buyer. The seller aims to profit in the bargaining process. As in most cases of negotiations, here the seller and the buyer are unwilling to share their preferences with the other. For example, from the perspective of the seller, some types of buyer agents have different bargaining strategies based on their risk preferences.

j, i	A	R	C
Low	-1	1	1
Mid	1	0	1
High	3	1	-1

(a)



(b)

Figure 6: (a) Single shot play of a negotiation between the seller i and buyer j. The numbers represent the payoffs of the seller. (b) I-ID for the negotiation with four models ascribed to j.

The idea of using probabilistic graphical models in multiagent negotiation was previously explored in [9]. In the same vein, we model agent i using the I-ID shown in Fig. 6(b). Analogous to [12], we consider four types of the buyer agent j. Each of them is represented using a BN. They differ in the probability distributions for the chance nodes *Risk* that represents the risk attitude and *Urg*, which represents the urgency of the situation to the agent. Let model m_j^1 represent a buyer of a risk averse type. A risk averse agent has an aversion to losing the deal and hence always proposes a high offer. The second model, m_j^2, is a risk seeking buyer that adopts a risky strategy by intending to offer a low price. Model m_j^3 is a risk neutral buyer that balances its low and high offers in the negotiation. The final model, m_j^4, is a buyer that is risk neutral but in an urgent situation is eager to acquire the item. Consequently, it is prone to offering a high price. Note that the chance node $\mathit{Neg}_{i,t-1}$ represents i's previous action in the negotiation.

We let agent i consider three candidate models for j, m_j^1, m_j^2, and m_j^3, and agent j use the model m_j^1 for case 1. Fig. 7(a) reveals that all the different updating methods correctly identify the true model after some steps and gather similar rewards. However, KL assigns

non-zero weights to other models as the distribution from those candidates is somewhat close to that of the true model. Because j is risk averse, it often offers a high price that the seller chooses to accept thereby incurring a payoff of 3.

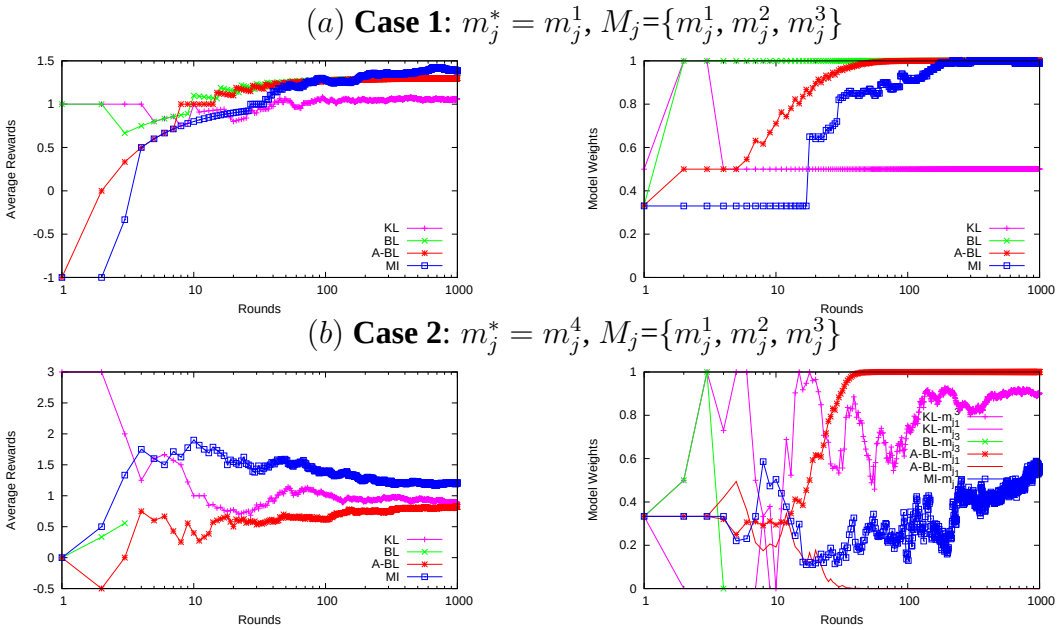


Figure 7: Performance profiles and the changing model weights for the two cases while repeatedly playing the negotiation game.

In case 2, agent j plays the game using the model, m_j^4, and i assumes the remaining three models as candidates. Notice that MI eventually assigns the largest weight (≈ 0.5) to the risk averse agent, m_j^1, that always offers a high price in the negotiation. This behavior is consistent with the model, m_j^4, that represents an urgent buyer who is also prone to offering a high price. The remaining two candidate models are MI-equivalent. In comparison, both KL and A-BL methods incorrectly identify the risk neutral agent m_j^3, which leads to lower average rewards.

6 Discussion

Our experimental results in multiple problem domains demonstrate that the normalized distribution of MI of the candidate models learned by classifying their predictions exhibits a comparable performance to Bayesian learning when the true model is within the set of candidate models. In particular, the experiments verify that the distribution of MI converges for this case. Perhaps more importantly, it improves on the other heuristic approaches for the plausible case that the true model is outside the model space. Thus, our approach shows potential as a general purpose technique for modeling other agents when we are uncertain

whether the model space is exhaustive. However, an important limitation to consider is a possibly large set of MI-equivalent models.

Our use of I-IDs to model the repeated interactions between agents provides an intuitive way to ascribe procedural models such as IDs, I-IDs and BNs to other agents. In comparison to agent functions, these models promote an understanding of the behavior of the agent and allow the explicit representation of domain structure and knowledge. However, these models must be solved to obtain their predictions thereby posing computational challenges.

References

[1] N. Bard and M. Bowling. Particle filtering for dynamic agent modelling in simplified poker. In *AAAI*, pages 515–521, 2007.

[2] C. Camerer. *Behavioral Game Theory: Experiments in Strategic Interaction*. Princeton University Press, 2003.

[3] D. Carmel and S. Markovich. Learning models of intelligent agents. In *AAAI*, pages 62–67, 1996.

[4] J. L. Doob. *Stochastic Processes*. John Wiley and Sons, 1953.

[5] P. Doshi, Y. Zeng, and Q. Chen. Graphical models for online solutions to interactive pomdps. In *AAMAS*, pages 809–816, 2007.

[6] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, 1991.

[7] Y. Gal and A. Pfeffer. A language for modeling agent's decision-making processes in games. In *AAMAS*, 2003.

[8] D. Koller and B. Milch. Multi-agent influence diagrams for representing and solving games. In *IJCAI*, pages 1027–1034, 2001.

[9] C. Mudgal and J. Vassileva. An influence diagram model for multi-agent negotiation. In *ICMAS*, pages 451–452, 2000.

[10] J. D. Rennie. Improving multi-text classification using naive bayes. Technical Report AI TR 2001-04, MIT, 2001.

[11] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (Second Edition)*. Prentice Hall, 2003.

[12] S. Saha, A. Biswas, and S. Sen. Modeling opponent decision in repeated one-shot negotiations. In *AAMAS*, pages 397–403, 2005.

[13] R. D. Shachter. Evaluating influence diagrams. *Operations Research*, 34(6):871–882, 1986.

[14] N. Sturtevant and M. Bowling. Robust game play against unknown opponents. In *AAMAS*, pages 713–719, 2006.

[15] D. Suryadi and P. Gmytrasiewicz. Learning models of other agents using influence diagrams. In *Intl. Conf. on User Modeling*, pages 223–232, 1999.