Approximate Solutions of Interactive Dynamic Influence Diagrams Using $\epsilon$-Behavioral Equivalence

by

Muthukumaran Chandrasekaran

(Under the direction of Prof. Prashant Doshi)

Abstract

Interactive dynamic influence diagrams (I-DID) are graphical models for sequential decision making in uncertain settings shared by other agents. Algorithms for solving I-DIDs face the challenge of an exponentially growing space of candidate models ascribed to other agents, over time. Pruning the behaviorally equivalent models is one way toward identifying a minimal model set. We seek to further reduce the complexity by additionally pruning models that are approximately behaviorally equivalent. Toward this, we redefine behavioral equivalence in terms of the distribution over the subject agent’s future action-observation paths, and introduce the notion of $\epsilon$-behavioral equivalence. We present a new approximation technique that reduces the candidate model space by removing models that are $\epsilon$-behaviorally equivalent with representative ones. We empirically demonstrate that our approach improves significantly in performance on previous techniques, with some limitations.

Index Words: Distributed Artificial Intelligence, Multiagent Systems, Decision making, Interactive Dynamic Influence Diagrams, Agent modeling, Behavioral equivalence
APPROXIMATE SOLUTIONS OF INTERACTIVE DYNAMIC INFLUENCE DIAGRAMS USING 
$\varepsilon$-BEHAVIORAL EQUIVALENCE

by

MUTHUKUMARAN CHANDRASEKARAN

Approved:

Major Professor: Prof. Prashant Doshi

Committee: Prof. Khaled Rasheed  
Prof. Walter D. Potter

Electronic Version Approved:

Maureen Grasso  
Dean of the Graduate School  
The University of Georgia  
December 2009
DEDICATION

To Chandrasekaran and Brinda, my loving parents, Hariharan, my supportive brother, and friends.
Acknowledgments

First, I would like to thank my major professor, Dr. Prashant Doshi, for his supervision, advice and guidance. I am indebted to him for having faith in me during difficult times and for all his encouragement.

I am grateful to Dr. Yifeng Zeng for his advice and crucial contributions, which made him the backbone of this research. I would like to particularly thank him for patiently answering all my sometimes—basic questions. I hope to continue collaborating with him in the future.

I would like to thank Nithya Vembu and Thomas Horton for helping me proofread my thesis and for sitting through infinite dry runs of my defense presentation.

I would also like to thank my brother, Hariharan, for his valuable advice on programming.

Last but not the least, I would like to thank Ekhlas, Xia, and my friends at the Institute for Artificial Intelligence at the University of Georgia for indulging in very useful discussions which benefitted me in more ways than they actually know.
# Table of Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 RELEVANCE TO AI</td>
<td>3</td>
</tr>
<tr>
<td>1.2 INTELLIGENT AGENTS</td>
<td>4</td>
</tr>
<tr>
<td>1.3 RATIONAL DECISION MAKING</td>
<td>5</td>
</tr>
<tr>
<td>1.4 MARKOV DECISION PROCESSES</td>
<td>7</td>
</tr>
<tr>
<td>1.5 GRAPHICAL MODELS</td>
<td>9</td>
</tr>
<tr>
<td>1.6 CURSES OF DIMENSIONALITY AND HISTORY</td>
<td>9</td>
</tr>
<tr>
<td>1.7 CLAIMS AND CONTRIBUTIONS</td>
<td>10</td>
</tr>
<tr>
<td>1.8 STRUCTURE OF THIS WORK</td>
<td>11</td>
</tr>
<tr>
<td>2 BACKGROUND</td>
<td>14</td>
</tr>
<tr>
<td>2.1 INTERACTIVE POMDP (I-POMDP) FRAMEWORK</td>
<td>14</td>
</tr>
<tr>
<td>2.2 INFLUENCE DIAGRAMS (IDs)</td>
<td>16</td>
</tr>
<tr>
<td>2.3 DYNAMIC INFLUENCE DIAGRAMS (DIDs)</td>
<td>20</td>
</tr>
<tr>
<td>2.4 INTERACTIVE INFLUENCE DIAGRAMS (I-IDs)</td>
<td>21</td>
</tr>
<tr>
<td>2.5 INTERACTIVE DYNAMIC INFLUENCE DIAGRAMS (I-DIDs)</td>
<td>25</td>
</tr>
<tr>
<td>3 PREVIOUS WORK</td>
<td>32</td>
</tr>
<tr>
<td>3.1 EXACTLY SOLVING I-DIDS USING BEHAVIORAL EQUIVALENCE</td>
<td>34</td>
</tr>
</tbody>
</table>
3.2 APPROXIMATELY SOLVING I-DIDs USING MODEL CLUSTERING ...... 37
3.3 APPROXIMATELY SOLVING I-DIDs USING DISCRIMINATIVE MODEL
   UPDATES ................................................................. 39
4 REDEFINING BEHAVIORAL EQUIVALENCE ............................. 41
   4.1 DEFINITION .......................................................... 42
   4.2 COMPUTING THE DISTRIBUTION OVER FUTURE PATHS ......... 43
   4.3 CORRECTNESS ........................................................ 46
5 ε-BEHAVIORAL EQUIVALENCE ........................................... 48
   5.1 DEFINITION .......................................................... 48
   5.2 APPROACH ........................................................... 49
   5.3 APPROXIMATION ALGORITHM ....................................... 51
6 TEST PROBLEM DOMAINS ................................................ 54
   6.1 MULTI-AGENT TIGER PROBLEM ...................................... 54
   6.2 MULTI-AGENT MACHINE MAINTENANCE PROBLEM ............... 56
7 EXPERIMENTAL EVALUATION ............................................ 61
   7.1 MULTI-AGENT TIGER PROBLEM ...................................... 61
   7.2 MULTI-AGENT MACHINE MAINTENANCE PROBLEM ............... 63
8 THEORETICAL ANALYSIS ................................................ 66
   8.1 COMPUTATIONAL SAVINGS .......................................... 66
   8.2 ERROR BOUND ...................................................... 68
9 CONCLUSION ............................................................... 71

BIBLIOGRAPHY .............................................................. 74
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Sample environment to show why one step greedy strategy is undesirable</td>
<td>6</td>
</tr>
<tr>
<td>2.1</td>
<td>A simple Influence Diagram (ID) representing the decision-making problem of an agent. The oval nodes representing the state (S) and the observation (Ω) reflected in the observation function, O, are the chance nodes. The rectangle is the decision node (A) and the diamond is the reward/utility function (R). Influences (links) connect nodes and symbolify the relationship between nodes.</td>
<td>17</td>
</tr>
<tr>
<td>2.2</td>
<td>An illustration of the arc reversal operation: reversing arc $a \rightarrow b$</td>
<td>19</td>
</tr>
<tr>
<td>2.3</td>
<td>An illustration of the random node removal operation: $x$ is removed by expectation</td>
<td>20</td>
</tr>
<tr>
<td>2.4</td>
<td>An illustration of the decision node removal operation: $d$ is removed by maximization</td>
<td>20</td>
</tr>
<tr>
<td>2.5</td>
<td>A two time-slice/horizon Dynamic Influence Diagram (DID) representing the decision-making problem of an agent. Here, the Influences (links) connect nodes not only within the same time slice but nodes across time slices as well.</td>
<td>21</td>
</tr>
<tr>
<td>2.6</td>
<td>(a) A generic level $l &gt; 0$ I-ID for agent $i$ situated with one other agent $j$. The hexagon is the model node ($M_{j,l-1}$) and the dashed arrow is the policy link. (b) Representing the model node and policy link using chance nodes and dependencies. The decision nodes of the lower-level I-IDs or IDs ($m_{j,l-1}^1, m_{j,l-1}^2$) are mapped to the corresponding chance nodes ($A_{j1}, A_{j2}$), which is indicated by the dotted arrows. Depending on the value of node, $Mod[M_j]$, distribution of each of the chance nodes is assigned to node $A_j$ with some probability.</td>
<td>23</td>
</tr>
<tr>
<td>2.7</td>
<td>The transformed I-ID with the model node replaced by the chance nodes and the relationships between them.</td>
<td>24</td>
</tr>
<tr>
<td>2.8</td>
<td>A generic two time-slice level $l$ I-DID for agent $i$.</td>
<td>26</td>
</tr>
</tbody>
</table>
2.9 The semantics of the model update link. Notice the growth in the number of models
at $t + 1$ shown in bold. ................................................. 27

2.10 Transformed I-DID with the model nodes and model update link replaced with the
chance nodes and the relationships (in bold). ............................... 28

2.11 Algorithm for exactly solving a level $l \geq 1$ I-DID or level 0 DID expanded over $T$
time steps. ................................................................. 30

3.1 Horizon-1 value function in the tiger game and the belief ranges corresponding to
different optimal actions. .................................................. 35

3.2 Algorithm for exactly solving a level $l \geq 1$ I-DID or level 0 DID expanded over $T$
time steps. ................................................................. 37

4.1 Future action-observation paths of agent $i$ in a 2-horizon multiagent tiger problem.
The nodes represent $i$’s action, while the edges are labeled with the possible obser-
vations. This example starts with $i$ listening. Agent $i$ may receive one of six obser-
vations conditional on $j$’s action, and performs an action that optimizes its resulting
belief. ................................................................. 43

5.1 Illustration of the iterative $\epsilon$-BE model grouping using the tiger problem. Black
vertical lines denote the beliefs contained in different models of agent $j$ included
in the initial model node, $M^{1}_{j,0}$. Decimals on top indicate $i$’s probability distribu-
tion over $j$’s models. We begin by picking a representative model (red line) and
grouping models that are $\epsilon$-BE with it. Unlike exact BE, models in a different
behavioral (shaded) region get grouped as well. Of the remaining models, another
is selected as representative. Agent $i$’s distribution over the representative models
is obtained by summing the probability mass assigned to the individual models in
each class. ................................................................. 50

5.2 Algorithm for partitioning $j$’s model space using $\epsilon$-BE. This function replaces
BehaviorEq() in Fig. 3.2. ................................................. 53
6.1 (a) Level 1 I-ID of agent $i$, (b) two level 0 IDs of agent $j$ whose decision nodes are mapped to the chance nodes, $A_1j$ and $A_2j$, in (a), indicated by the dotted arrows. The two IDs differ in the distribution over the chance node, TigerLocation. 55

6.2 Level 1 I-DID of agent $i$ for the multiagent tiger problem. The model node contains $M$ level 0 DIDs of agent $j$. At horizon 1, the models of $j$ are IDs. 56

6.3 CPD of the chance node $TigerLocation_i^{t+1}$ in the I-DID of Fig. 6.2 when the tiger (a) likely persists in its original location on opening doors, and (b) randomly appears behind any door on opening one. 57

6.4 The CPD of the chance node $Growl&Creak_i^{t+1}$ in the level 1 I-DID. 57

6.5 Reward functions of agents $i$ and $j$ for the multi-agent tiger problem. 58

6.6 Level 1 I-DID of agent $i$ for the multiagent MM problem. The hexagonal model node contains $M$ level 0 DIDs of agent $j$. At horizon 1, the models of $j$ are IDs. 59

6.7 CPD of the chance node $MachineFailure_i^{t+1}$ in the level 1 I-DID of Fig. 6.6. 59

6.8 The CPD of the chance node $Defective_i^{t+1}$ in the level 1 I-DID. 60

6.9 Reward function of agent $i$. For the level 0 agent $j$, the reward function is identical to the one in the classical MM problem. 60

7.1 Performance profile obtained by solving a level 1 I-DID for the multiagent tiger problem using the $\epsilon$-BE approach for (a) 3 horizons and (b) 4 horizons. As $\epsilon$ reduces, quality of the solution improves and approaches that of the exact. (c) Comparison of $\epsilon$-BE and DMU in terms of the rewards obtained given identical numbers of models in the initial model node after clustering and pruning. 62

7.2 Comparison of $\epsilon$-BE and DMU for the multi-agent tiger problem in terms of the rewards obtained given identical numbers of models in the initial model node after clustering and pruning. 63

7.3 Performance profile for the multiagent MM problem obtained by solving level 1 I-DIDs approximately using $\epsilon$-BE for (a) 3 horizon and (b) 4 horizon. Reducing $\epsilon$ results in better quality solutions. 64
7.4 Significant increase in rewards obtained for $\epsilon$-BE compared to DMU, given identical numbers of retained models in the initial model node.
Chapter 1

Introduction

In real world situations, decisions are being made under conditions of certainty, risk or uncertainty. Among these conditions, it can be realized that making decisions under uncertainty is most difficult and most common. Here, the decision maker has to choose among alternatives (that may have one of several consequences) where each of these alternatives are associated with a probability distribution that is unknown. There has been a lot of advancements in this field in recent years. Researchers have realized the need to device strategies that enhance the ability to deal with uncertain information in a straight forward natural way which will in turn improve the quality of planning, enable more rational responses to unexpected events, and allow a better understanding of available options. These enhancements will enable people and machines to make better decisions in less time and with lower costs. This growth in interest for developing algorithms/strategies to handle such uncertain scenarios was initiated by a large number of applications in various fields such as computer science, business, engineering, etc. Decision theory offers two main approaches for handling conditions of uncertainty. The first exploits criteria of choice developed in a broader context by game theory [46, 47, 48], for example the min-max strategy, where an alternative is chosen such that the worst possible consequence of the chosen alternative is better than (or equal to) the best possible consequence of any other alternative. The second approach is to reduce the uncertainty case to the case of risk by using subjective probabilities, based on analysis of previous decisions made in similar circumstances. Utility theory [25] helps in understanding the value of a choice made. There are three traditions in utility theory. One attempts to describe people’s utility functions and is called the descriptive approach. Another attempts to use utility in the construction of a rational model of decision making and is called the normative approach. The third
attempts to bridge the descriptive and normative approaches by considering the limitations people have with the normative goal they would like to reach; this is called the prescriptive approach. Our research is aimed at constructing rational models for decision making, or in other words, we focus on developing new and improved normative approaches (how humans ought to take decisions) for situations where decisions have to be made under conditions of uncertainty. These decisions will be more reliable as they are the most rational of the decisions that could be taken, given a scenario.

*Interactive Partially Observable Markov Decision Processes (I-POMDPs)* [15] provide a framework for planning on multi-agent settings in complex problem domains with *partially observable* (uncertain) environments that include either *cooperative* or *competitive* participating agents. They have very few restrictions as opposed to other approaches that restrict their problem domains to reduce complexity (Decentralized POMDPs [26, 27, 28] work in cooperative multi-agent settings only). However, as expected, these benefits come with a cost; they involve complex time consuming computations for arriving at a solution. *Interactive dynamic influence diagrams (I-DIDs)* [21] are the graphical counterparts of I-POMDPs. Hence, their computational complexity is comparable to that of I-POMDPs. However, since they offer an intuitive way to not only identify but also display the essential elements, including decisions, uncertainties, and objectives, and how they influence each other, they are a much better framework to model the decision problem. Hence, it is very important to have good approximation methods so that a wide range of application domains can be made feasible.

The purpose of this thesis is to propose a new approximation technique for solving I-DIDs which not only helps in improving the quality of the solution but also in making it more scalable in terms of the number of horizons (a large span of time ahead in the planning sequence) it can plan for and efficient in terms of time consumed for solving it given the current computer power (i.e., memory, CPU speed).

Generally, the quality of the solution or the limit on scalability is influenced by the curses of history and dimensionality (the factors of which explained in detail later in this chapter. This curse is caused by the exponentially increasing model space (containing unique models) of the
other agents over time. In this paper, we aim to reduce this model space by additionally pruning models that are approximately behaviorally equivalent. Toward this objective, we introduce the concept of $\epsilon$-behavioral equivalence among candidate models. In doing so, we redefine behavioral equivalence as the class of models of the other agents that induce an identical distribution over the subject agent’s future action-observation paths in the interaction. Subsequently, models that induce distributions over the paths, which are no more than $\epsilon \geq 0$ apart are termed as being $\epsilon$-behaviorally equivalent. Intuitively, this results in a lesser number of equivalence classes in the partition. If we pick a single representative model from each class, we typically end up with no more models than in the minimal set which need be solved thereby improving on approaches that utilize exact behavioral equivalence.

We begin by selecting a model at random from the model space and grouping together $\epsilon$-behaviorally equivalent models with it. We repeat this procedure for the remaining models until all models have been grouped. The retained model set consists of the representative model from each equivalence class. In the worst case ($\epsilon = 0$), our approach identifies exact behavioral equivalence and the model set consists of all the behaviorally unique models. Chapter 5 will provide a more in-depth discussion of the proposed algorithm.

1.1 RELEVANCE TO AI

Artificial Intelligence is the field that strives to program software agents that exhibit intelligence. The word artificial means something that can be built and the word intelligence describes a property of the mind that encompasses many abilities, such as the capacities to reason, to plan, to solve problems, to think abstractly, to comprehend ideas, etc. Thus, in order to create an AI agent, we end up with four possible goals:

1. Systems that think like humans also known as cognitive modeling which focuses on reasoning like humans and the human framework.
2. Systems that *think rationally* or in other words systems that are governed by the *laws of thought* which focuses on reasoning and a general concept of intelligence.

3. Systems that *act like humans* or in other words systems that pass the *turing test* [30] where the focus is on behavior of humans and the human framework.

4. Systems that *act rationally*, also called *rational agents* that focus on behavior and a general concept of intelligence.

Our research caters to the fourth goal of AI stated above. *Rationality* is an idealized concept of intelligence, which means “doing the right thing”. We will only deal with creating algorithms for modeling intelligent *software* agents. For convenience sake, throughout this paper, we will refer to *intelligent software agents* as just *agents* or *intelligent agents* unless it is mentioned otherwise.

1.2 INTELLIGENT AGENTS

An *intelligent agent* is something that *observes* its environment through sensors and *acts* intelligently upon that environment through actuators. A human agent has eyes, ears and other organs for sensors, and mouth, hands, legs and other body parts for actuators. Similarly, a software agent receives keyboard inputs, and files as sensory input and acts on the environment by displaying the output on the screen, writing files, etc. A rational agent selects an action that maximizes its performance measure given all the information it has regarding the nature of the environment and the percepts it receives from it. These environments are characterized along several dimensions – They can be fully or partially observable, deterministic or stochastic, episodic or sequential, static or dynamic, discrete or continuous, and single-agent or multi-agent. Environments that are fully observable, deterministic, finite, static or discrete in time, actions, object and effects are less common in nature when compared to those that are partially observable (allow for uncertainty in observations), and dynamic in the real world. Thus, for correct modeling of most real world problems, the method to use must account for possible actions with stochastic effects and for noisy
measurements. When the environment exhibits these properties, the planning task becomes a non-trivial problem. Solving these problems is a complex, and time-consuming procedure. Hence, the need for efficient algorithms to exactly or approximately solve them rose.

1.3 RATIONAL DECISION MAKING

The decision-making process is similar to a problem solving process which is time consuming, and context dependent. For example, consider the problem of a robot navigating in a large office building. The robot can move from hallway intersection to intersection and can make local observations of its world. Its actions are not completely reliable, however. Sometimes when it intends to move, it stays where it is, or goes too far; sometimes when it intends to turn it overshoots. It has similar problems with the observations it makes. The point here is that, any machine that is autonomous is not completely reliable because of various factors like sensor malfunction, power shut down, or even lack of adequate data or information regarding the environment it is in. Hence, these agents are faced with the problem of partially observable environments [19]. Hence, accurate analysis of the environment and rational decision-making become extremely difficult and it is interesting to see how these agents handle such scenarios.

So researchers were faced with their next challenge in the rational decision making process; making the agent understand what a good or a bad decision was. They came up with an easy solution. If an agent was going to make decisions by itself, it required some metric that it could use to differentiate good and bad choices. This was another trouble spot because even humans often found it difficult to articulate what the differences are between good and bad choices. Nevertheless, they had to do so because they desired that the agents had to know to choose between numerous options. Hence, they assumed that each state had an associated reward for performing each possible action or a decision choice in that state. This reward is the desirability of performing that decision choice, being in that particular state in the environment. These rewards need not necessarily be something that the agent gets as it operates. These rewards are a way of assigning values to different
states of the environment. Given these values, the agent attempts to make the decision that it knows has a greater advantage value.

The next problem encountered by researchers; what if planning had to be done for the future? For convenience sake, time was assumed to pass in discrete increments of some clock and the agent must choose some action to perform at each tick of the clock (it could also choose to do nothing). Say, planning had to be done for two time steps in advance. Decisions had to be made taking into consideration factors like the future and expected rewards. In other words, is it recommended to perform the action that fetches most reward for this particular step or also take into consideration the expected reward at every time step in advance and hope to find a better overall utility or desirability to execute the action at the current step? In order to understand why the later is important, consider the following scenario.

![Diagram](image.png)

**Figure 1.1: Sample environment to show why one step greedy strategy is undesirable**

As it can be seen from the above figure, if the agent chose to perform the action A (higher immediate reward) or in other words, a one step greedy strategy, it would not have ended up with a reward as high as it would have if it chose to consider two time steps in advance and made its decision to go with action B. So this situation shows an example in which the agent would probably want to take into account the rewards it might receive in the future, and not just immediate rewards.

The next challenge for the agent would be to tackle the problem when it does not know in advance how many decisions it has to make. Hence, Antony R. Cassandra et. al. formalized the
infinite horizon problem [20]. Finite and infinite horizon problems are mentioned in greater detail in [20].

Formally, a model can be created for an agent consisting of a finite set of states where it can be, a finite set of actions and a reward structure defined for each action-state pair. In other words, the set of states are the different locations in which the agent can be in the environment, the set of actions are the things that the agent can do, and the reward structure for each action-state pair is the agent’s desirability for being in the particular state after performing a particular action. For the robot navigation problem, the states can be viewed as the location of the robot in the environment. The actions are the things that it can do such as such as move forward, move left, move right, move backwards etc., and associated with each action is an immediate reward for being in a particular state. For example, if there was a pit directly in front of the robot and the robot did not know how to climb out of the pit, then the reward for moving forward would be less compared to a safe area within its reach. The advantages of making machines decide what decisions to make might seem considerable. However, the real difficulties lie in precisely that; making machines act/think rationally. Having said that, a great deal of money and resources have been spent to make this dream come true. It definitely has a great positive impact on today’s world.

1.4 Markov Decision Processes

A model in the agent knows its current state (or in other words the environment is fully observable), is known as the Markov decision process (MDP) model [31, 19]. Markov decision processes (MDPs) provide a principled framework to optimize course of action under these environments. A Markov decision process is defined by a tuple $<S, A, T, R>$, where $S$ is the set of the states in the planning problem; $A$ is the set of possible actions of the agent; $T$ is the transition function that specifies the probabilities to go from state $s$ to state $s'$ given action $a$ where, $\{s, s'\} \in S$ and $a \in A$; and $R$ is the reward function, that specifies the reward the agent gets for performing action $a$ when the world is in the $s$ state. It is important to understand that while MDP solution techniques
are able to solve large state space problems, the assumptions of classical planning (mainly the full observability assumption) make them unsuitable for most complex real world applications.

However, if a model of a participating agent cannot directly observe the underlying environmental state but instead, infers a distribution over the state based on a model of the world and some local observations, or in other words, if the environment is partially observable, then such a model is known as Partially Observable Markov Decision Process model (POMDP model) [32, 33, 34, 35, 36, 38]. POMDP is a generalization of a Markov decision process. The POMDP framework is general enough to model a variety of real-world sequential decision processes. A POMDP is really just an MDP; we have a set of states, a set of actions, transitions and immediate rewards. The actions’ effects on the state in a POMDP is exactly the same as in an MDP. The only difference is in whether or not we can observe the current state of the process. In a POMDP we add a set of observations to the model. So instead of directly observing the current state, the state gives us an observation which provides a hint about what state it is in. The observations can be probabilistic; so we need to also specify an observation function. This observation function simply tells us the probability of each observation for each state in the model. We can also have the observation likelihood depend on the action if we like. Formally, a POMDP is defined by a tuple \(<S, A, \Omega, T, O, R>\) where \(S\) is a finite set of states, \(A\) is a finite set of actions, \(\Omega\) is a finite set of observations, \(T\) is the transition function that specifies the probabilities to go from state \(s\) to state \(s'\) given action \(a\), where, \(\{s, s'\} \in S\) and \(a \in A\); \(O\) is the observation function and \(R\) is the reward function, that specifies the reward the agent gets for performing action \(a\) when the world is in the \(s\) state.

POMDPs, when generalized to multi-agent settings [17, 20] by including other agents’ computable models in the state space along with the physical environment, are known as Interactive partially observable Markov decision processes (I-POMDP) [15, 37, 39]. They provide a framework for sequential decision making in partially observable multi-agent environments. This framework will be discussed in Chapter 2.
1.5 Graphical Models

An influence diagram (ID) [22, 29, 40] is a simple visual representation of a decision problem. Influence diagrams offer an intuitive way to identify and display the essential elements, including decisions, uncertainties, and objectives, and how they influence each other. Solving an ID unrolled over many time slices is called a Dynamic ID (DID). Did’s may be viewed as structural representations of POMDPs.

Interactive dynamic influence diagrams (I-DID) [2, 21] may be viewed as graphical counterparts of interactive POMDPs (I-POMDPs) [5], providing a way to model and exploit the embedded structure often present in real-world decision making situations. I-DIDs concisely represent the problem of how an agent should act in an uncertain environment shared with others who may act in sophisticated ways. They generalize DIDs [13], which are graphical representations of POMDPs, to multi-agent settings analogously to how I-POMDPs generalize POMDPs. These graphical models will be explained in greater detail in the Chapter 2.

1.6 Curses of Dimensionality and History

The curse of dimensionality is the problem caused by the exponential increase in the model space. Since there exists limitations in the CPU speed and memory available to us, it leads to large computational costs in terms of the time needed to solve each of these models in the model space. The curse of history is the problem caused by nested modeling. As we may expect, all these frameworks possess the curses of history and dimensionality; some factors that contribute to this are as follows.

- The initial number of candidate models for the other agents; The greater the initial models considered, better are the chances of finding the exact model of the other agent and greater the computational cost as more number of models have to be solved. This problem contributes to the curse of dimensionality. As we already know, infinite models of the other
agents can exist. Hence, it is important to effectively sample models from this infinite model space in order to obtain good solutions with low computational cost.

- The number of *horizons* (look ahead steps); At time step $t$, there could be $|\mathcal{M}_j^0|^{(|A_j||\Omega_j|)}$ many models of the other agent $j$, where $|\mathcal{M}_j^0|$ is the number of models considered initially, $|A_j|$ is the number of possible actions for $j$, and $|\Omega_j|$ is the number of possible observations for $j$. As it can be seen, the number of models that have to be solved increase exponentially with increase in the number of horizons considered. This problem also contributes to the curse of dimensionality.

- The number of *strategy levels* (nested modeling); Nested modeling contributes to the curse of history and further, to the complexity since solution of each models at level $l-1$ requires solving the lower level $l-2$ models and so on recursively up to level 0.

Hence, good techniques that mitigate these curses to the greatest extent possible will enable a wider range of applications in larger problem domains.

### 1.7 Claims and Contributions

In the previous section we provided some basic concepts that are underlying the study of multi-agent decision making. This section mainly enumerates our claims and contributions to the field.

- The primary focus of this thesis is the development of an approximate solution for interactive dynamic influence diagrams helps in improving the quality of the solution.

- Algorithms for solving I-DIDs face the challenge of an exponentially growing space of candidate models ascribed to other agents, over time. Previous methods pruned the behaviorally equivalent models to identify the minimal model set. We mitigate this curse of dimensionality by reducing the candidate model space by additionally pruning models that are approximately behaviorally equivalent and replacing them with representatives.
• We redefine behavioral equivalence in terms of the distribution over the subject agent’s future action-observation paths. This offers a more rigorous way to define behavioral equivalence as it has the additional advantage that it permits us to measure the degree to which the candidate models of the other agent are behaviorally equivalent. We use symmetric Kullback Leibler (KL) divergence as the metric to measure this degree.

• We introduce the notion of $\epsilon$-behavioral equivalence as a way to approximate behavioral equivalence, which is developed based on our new definition of behavioral equivalence.

• We also propose that our $\epsilon$-behavioral equivalence approach results in at most one model for each behavioral equivalence class after pruning which results in better solutions in terms of computational efficiency and quality when compared to the model clustering approach by Zeng et al. [14].

• We theoretically analyze the error introduced by this approach in the optimality of the subject agent’s solution and also discuss its advantages over the model clustering approach.

• We empirically evaluate the performance of our approximation technique on benchmark problem domains such as the multi-agent tiger problem and the multi-agent machine maintenance problem and compare the results with previous exact and approximation techniques. We show significant improvement in performance, although with limitations.

1.8 Structure of this Work

Due to the nature of this research topic, it is necessary to perform a large literature review to get a hold of the issues and facts about the sequential decision problems that are solved using I-DIDs. It is therefore necessary to present a significant amount of background information to the reader so that the foundation is laid and the understanding of the key issues involving this research are easier to acquire. We thus, outline the structure of this thesis as follows in order to have a proper flow in understanding.
In *Chapter 1*, the focus is to give a very broad idea of the context of the research area, introduce a few general concepts, and give a basic outline of our contributions to the field.

In *Chapter 2*, we briefly review the framework of finitely nested Interactive POMDPs which provides the mathematical foundations for graphical models formalized by influence diagrams applied to multiagent settings. We will also introduce the readers to IDs and dynamic IDs which can be viewed as structural representations for POMDPs. We will also provide a detailed description of Interactive IDs and their extensions to dynamic settings - I-DIDs. Exact algorithms to solve I-DIDs will also be discussed in detail.

In *Chapter 3*, we survey different implementations of I-DIDs and revise their pros and cons, keeping in mind that some of these previous approaches; both exact and approximate, may be applicable in our proposed method. We introduce the readers to the initial concept of behavioral equivalence and discuss why this definition makes it difficult to define an approximate BE measure and also discuss exact and approximate algorithms developed for solving I-DIDs in the past.

In *Chapter 4*, We redefine behavioral equivalence more rigorously in terms of the distribution over future action-observation paths and prove its correctness with respect to the previous definition of BE. In addition to being more rigorous, the new definition of BE has the additional advantage of providing a way to measure the degree to which the models are BE. We also derive an equation that computes the distribution of the future action-observation paths which lays the foundation of our proposed approximation technique.

In *Chapter 5*, we define the notion of epsilon BE, and introduce our new and improved approximation technique.

In *Chapter 6*, we provide a detailed description of the problem domains in which our technique was applied. The reward, observation, and transition functions for each of these application domains will be presented. Also, we illustratively show how I-DIDs were applied in these problem domains.

In *Chapter 7*, we present empirical evaluations of the proposed method. We take the two problems from the literature; the multiagent tiger problem, and the multiagent machine maintenance
problem and perform simulations to measure the time needed to achieve different levels of performance. We compare our results with the other exact and approximation methods available for I-DIDs.

In *Chapter 8*, we mention the computational advantages due to our proposed approximation technique and also attempt to bound the error due to the approximation. We also theoretically analyze our method’s savings with respect to the Model clustering approximation technique.

In *Chapter 9*, we summarize our contributions, claims and results from the theoretical and experimental evaluations and also provide some ideas to further improve on our approximation method for solving I-DIDs.
CHAPTER 2

BACKGROUND

In this chapter we will briefly review the framework of finitely nested Interactive POMDPs [5] which provides the mathematical foundation for graphical models formalized by influence diagrams (IDs) applied to multi-agent settings. We also provide an overview of IDs [22] and Dynamic IDs which can be viewed as structural representations for POMDPs. In the final section that concludes this chapter we provide a detailed description of Interactive Influence Diagrams (I-IDs) and their extension to dynamic settings - I-DIDs. I-DIDs can be viewed as structural representations for I-POMDPs. Exact algorithms to solve I-DIDs will also be discussed in detail.

2.1 INTERACTIVE POMDP (I-POMDP) FRAMEWORK

The specification of a sequential decision problem for a fully observable environment with a Markovian transition model and additive rewards is called a Markov decision process or MDP. However, when the environment is only partially observable, the agent does not necessarily know which state it is in and such models are called Partially Observable Markov Decision Process or POMDPs. Furthermore, the utility of a state $s$ and the optimal action in $s$ depend not just on $s$, but also on how much the agent knows when it is in $s$. For these reasons, POMDPs are usually viewed as much more difficult than ordinary MDPs. Interactive POMDPs generalize POMDPs to multi-agent settings by including other agents’ models as part of the state space. One could possibly think of many problems where one agent would not be interacting alone with the world. Multi agent settings have many agents either cooperating or competing or just neutral to achieve a particular task. Models of other agents include their private information such as beliefs, capabilities, and preferences. Agents contain beliefs about not only the environment, but also other agents’ models
and their respective beliefs. All this information is included in the state space - called the interactive state space. For the sake of simplicity, I-POMDPs are usually presented assuming intentional model agents, similar to those used in Bayesian games [41, 42, 43] though the framework can be extended to any model. For simplicity sake, we will consider just two agents - $i$, and $j$ interacting in a common environment. All results can be scaled to three or more agents.

Mathematically, the models can be formalized using the I-POMDP framework as follows.

**Definition 1 (I-POMDP$_{i,l}$).** A finitely nested I-POMDP of agent $i$ with a strategy level $l$ is

$I$-POMDP$_{i,l} = < IS_{i,l}, A, T_i, \Omega_i, O_i, R_i >$

where:

1. $IS_{i,l}$ is a set of interactive states defined as, $IS_{i,l} = S \times M_{j,l-1}$, where $M_{j,l-1} = \Theta_{j,l-1} \cup SM_j$, for $l \geq 1$, and $IS_{i,0} = S$, where $S$ is the set of states of the physical environment. $\Theta_{j,l-1}$ is the set of computable intentional models of agent $j$. The remaining set of models, $SM_j$, is the set of subintentional models of $j$;

2. $A = A_i \times A_j$, is the set of joint actions of all agents in the environment;

3. Given the Model Non-Manipulability Assumption (MNM): An agent’s actions do not change other agents’ model directly. $T_i$ is a transition function, $T_i : S \times A \times S \rightarrow [0, 1]$. It reflects the possibly uncertain effects of the joint actions on the physical states of the environment;

4. $\Omega_i$ is the set of observations of agent $i$;

5. Given the Model Non-Observability Assumption (MNO): An agent cannot observe other agents’ model directly. $O_i$ is an observation function, $O_i : S \times A \times \Omega_i \rightarrow [0, 1]$. It describes how likely it is for agent $i$ to receive the observations given the physical state and joint actions;

6. $R_i$ is a reward function, $R_i : IS_i \times A \rightarrow \mathbb{R}$. It describes agent $i$’s preferences over its interactive states and joint actions, though usually only the physical states and actions matter.
Intentional models ascribe to the other agent beliefs, preferences and rationality in action selection and are analogous to types as used in game theory [46, 48]. Each intentional model, $\theta_{j,l-1} = < b_{j,l-1}, \hat{\theta}_j >$, where $b_{j,l-1}$ is agent $j$’s belief at level $l - 1$, and the frame, $\hat{\theta}_j = < A, T_j, \Omega_j, O_j, R_j, OC_j >$. Here, $j$ is assumed Bayes rational and $OC_j$ is $j$’s optimality criterion. A subintentional model is a triple, $sm_j = < h_j, O_j, f_j >$, where $f_j : H_j \rightarrow \Delta(A_j)$ is agent $j$’s function, assumed computable, which maps possible histories of $j$’s observations to distributions over its actions. $h_j$ is an element of $H_j$ and $O_j$ gives the probability with which $j$ receives its input. We refer the reader to [5] for details regarding the belief updates and the value iteration in I-POMDPs.

2.2 Influence Diagrams (IDs)

In this section we briefly describe influence diagrams (IDs) followed by their extensions to dynamic settings, DIDs, and refer the reader to [22, 44] for more details. An influence diagram (ID) (also called a decision network) is a compact graphical and mathematical representation of a decision problem. It is a way of describing the dependencies among aleatory variables and decisions. It is a generalization of a Bayesian network, in which both probabilistic inference problems and decision making problems can be modeled and solved. An influence diagram can be used to visualize the probabilistic dependencies in a decision analysis and to specify the states of information for which independencies can be assumed to exist. The first complete algorithm for evaluating an influence diagram was developed by Shachter in 1986 [29]. An overview of his algorithm is also presented.

2.2.1 Syntax

An ID has three types of nodes and three types of arcs (or arrow) between these nodes. See the Fig. 2.1 below. We observe that an ID augments a Bayesian network with decision and utility nodes.
Figure 2.1: A simple Influence Diagram (ID) representing the decision-making problem of an agent. The oval nodes representing the state (S) and the observation (Ω) reflected in the observation function, O, are the chance nodes. The rectangle is the decision node (A) and the diamond is the reward/utility function (R). Influences (links) connect nodes and symbolify the relationship between nodes.

**Types of Nodes**

1. *Decision node* (corresponding to each decision to be made) is drawn as a rectangle. It represents points where the decision making agent has a choice of actions.

2. *Chance node* (corresponding to each uncertainty to be modeled) is drawn as an oval. These represent random variables, just as they do in Bayes nets. The agent could be uncertain about various things because of the partial observability problem faced in real world problems. Each chance node has a conditional distribution associated with it that is indexed by the state of the parent nodes.

3. *Utility node* (corresponding to a utility function) is drawn as a diamond (or an octagon). The utility node has all the variables describing the outcome that directly affect the utility, as parents. This description could be just a tabulation of the function or a multi-linear function.
**TYPES OF ARCS/ARROWS**

1. *Functional arcs* (ending in utility node) indicate that one of the components of additively separable utility function is a function of all the nodes at their tails.

2. *Conditional arcs* (ending in chance node) indicate that the uncertainty at their heads is probabilistically conditioned on all the nodes at their tails.

3. *Informational arcs* (ending in decision node) indicate that the decision at their heads is made with the outcome of all the nodes at their tails known beforehand.

**2.2.2 EVALUATING INFLUENCE DIAGRAMS**

The solution of the influence diagram is the action that is chosen to be performed for each possible setting. This decision is made in the decision node. Once the decision node is set, it behaves just like a chance node that has been set as an evidence variable. The algorithm outline for evaluating the influence diagram is as follows.

1. Set the evidence variables for the current state.

2. For each possible value of the decision node;
   
   (a) Set the decision node to that value.

   (b) Calculate the posterior probabilities for the parent nodes of the utility node, using a standard probabilistic inference algorithm.

   (c) Calculate the resulting utility for the action.

3. Return the action with the highest utility.

   **Shachter’s Algorithm.** Shachter’s algorithm takes a reduction approach. The algorithm evaluates an influence diagram by applying a series of *value-preserving reductions*. Value preserving reductions is an operation that transforms an influence diagram into another one with the same optimal expected value.
Shachter identifies four basic value-preserving reductions, namely arc reversal, barren node removal, random node removal, and decision node removal.

**Arc Reversal.** The arc reversal operation is illustrated in Fig. 2.2. Suppose \( a \rightarrow b \) is an arc in an influence diagram such that both \( a \) and \( b \) are random nodes and there is no other directed path from node \( a \) to node \( b \). The direction of that arc can be reversed and both nodes inherit each other's parents. This operation is an application of Bayes Theorem. In Fig. 2.2, we begin with conditional probability distributions \( P\{b|a, y, z\} \) and \( P\{a|x, y\} \), and end up with conditional probability distributions \( P\{a|b, x, y, z\} \) and \( P\{b|x, y, z\} \). Formally,

\[
P\{b|x, y, z\} = \sum_a P\{a|b, x, y, z\} = \sum_a P\{b|a, y, z\} \ast P\{a|x, y\}
\]

\[
P\{a|b, x, y, z\} = \frac{P\{a|b, x, y, z\}}{P\{b|x, y, z\}}
\]

![Figure 2.2: An illustration of the arc reversal operation: reversing arc \( a \rightarrow b \)](image)

**Barren Node Removal.** A node in the influence diagram is called a barren node if it has no child in the diagram. The barren node removal reduction states that any barren node that is not a value node can be removed together with its incoming arcs.

**Random Node Removal.** If the value node is the only child of a random node \( x \) in an influence diagram, then the node \( x \) can be removed by conditional expectation. As a result of this operation, the random node \( x \) is removed and the old value node is replaced with the one that inherits all the parents of both the old value node and the random node. This reduction is illustrated in Fig. 2.3 where the value function \( g' \) of the new value node \( v' \) in the resultant influence diagrams given by:

\[
g'(a, b, c) = \sum_x g(x, b, c) \ast P\{x|a, b\}
\]

**Decision Node Removal.** A decision node is called a leaf decision node if it has no decision node descendant. If a leaf decision node \( d \) has the value node \( v \) as its only child and \( \pi(v) \subseteq\)
\{d\} \cup \pi(d), \text{ then the decision node can be removed by maximization. The reduction is illustrated in Fig. 2.4 where the value function } g' \text{ of the new value node } v' \text{ in the resultant influence diagram is given by:}

\[ g'(b) = \max_d g(d, b) \]

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure2-4}
\caption{An illustration of the decision node removal operation: \(d\) is removed by maximization}
\end{figure}

The maximization operation also results in an optimal decision function \(\delta_d\) for the leaf decision node through

\[ \delta_d(b) = \arg\max_d g(d, b) \]

Note that the new value node has the same parent as the old value node. Thus, some of the parents of \(d\) may become barren nodes as a result of this reduction.

### 2.3 Dynamic Influence Diagrams (DIDs)

On solving an ID unrolled over as many time slices as the number of horizons, called a dynamic ID [19] shown in Fig. 2.5, we obtain the value of performing each action in the decision node, with the best action being the one with the largest value. Dynamic IDs provide a structural and concise
representation for large POMDPs [19]. Hence they can also be used as inputs for any POMDP algorithm. The nodes in a DID, like the one in Fig. 2.5, correspond to the elements of a POMDP.

![Dynamic Influence Diagram](image)

Figure 2.5: A two time-slice/horizon Dynamic Influence Diagram (DID) representing the decision-making problem of an agent. Here, the Influences (links) connect nodes not only within the same time slice but nodes across time slices as well.

That is, the values of the decision node $A^t$, correspond to the set of actions, $A$, in a POMDP. The values of the chance node, $S^t$ and $O^t$, correspond to the sets of states and observations, respectively, in a POMDP. The conditional probability distribution (CPD), Pr($S^{t+1}|S^t, A^t$), of the chance node, $S^{t+1}$, is analogous to the transition function, $T$ in a POMDP. The CPD, Pr($O^{t+1}|S^{t+1}, A^t$), of the chance node, $O^{t+1}$, is analogous to the observation function, $O$, and the utility table of the utility node, $U$, is analogous to the reward function, $R$, in a POMDP. DID s perform planning using a forward exploration technique. This technique explores the possible states of belief an agent may be in in the future, the likelihood of reaching each state of belief, and the expected utility of each belief state. The agent then adopts the plan which maximizes the expected utility. DID s provide exact solutions for finite horizon POMDP problems, and finite look-ahead approximations for POMDPs of infinite horizon.

2.4 **Interactive Influence Diagrams (I-IDs)**

In this section we describe interactive influence diagrams [23] for modeling two-agent interactions.

Interactive influence diagrams [23] are graphical models of decision-making in uncertain multi-agent settings. They are a naive extension of influence diagrams (IDs) to settings populated by multiple agents made possible by treating other agents as automatons, represented using chance
nodes. However, this approach that the agent’s actions are controlled using a probability distribution that does not change over time. I-IDs adopt a more sophisticated approach by generalizing influence diagrams [24] to make them applicable to settings shared with other agents, who may act, observe and update their beliefs. These formalisms seek to explicitly and transparently model the structure that is often present in real-world problems by decomposing the situation into chance and decision variables, and the dependencies between the variables. I-IDs ascribe procedural models to other agents these may be IDs, Bayesian networks (BNs), or I-IDs themselves leading to recursive modeling. Besides providing intuitive reasons for the strategies, procedural knowledge may help preclude certain strategies of others, deeming them impossible because of the structure of the environment. As agents act and make observations, beliefs over others models are updated. With the implicit assumption that the true model of other is contained in the model space, I-IDs use Bayesian learning to update beliefs, which gradually converge.

2.4.1 Syntax

In addition to the usual chance, decision, and utility nodes, I-IDs include a new type of node called the model node (hexagonal node, \( M_{j,t-1} \)). We show a general level I-IID in Fig. 2.6(a), where the model node \( (M_{j,t-1}) \) is denoted using a hexagon. We note that the probability distribution over the chance node, \( S \), and the model node together represents agent \( i \)'s belief over its interactive state space. 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 between the model node and the chance node, \( A_j \). In the absence of other agents, the model node and the chance node, \( A_j \), vanish and I-IDs collapse into traditional IDs.

The model node contains as its values the alternative computational models ascribed by \( i \) to the other agent from the set, \( \Theta_{j,t-1} \cup SM_j \), where \( \Theta_{j,t-1} \) and \( SM_j \) are as defined previously in Section 2.1. We denote the set of these models by \( M_{j,t-1} \). A model in the model node may itself be an I-ID or ID, and the recursion terminates when a model is an ID or a simple probability distribution over the actions (subintentional). Formally, we denote a model of \( j \) as, \( m_{j,t-1} = \langle b_{j,t-1}, \hat{\theta}_j \rangle \), where
Figure 2.6: (a) A generic level $l > 0$ I-ID for agent $i$ situated with one other agent $j$. The hexagon is the model node $(M_{j,l-1})$ and the dashed arrow is the policy link. (b) Representing the model node and policy link using chance nodes and dependencies. The decision nodes of the lower-level I-IDs or IDs ($m_{j,l-1}^1, m_{j,l-1}^2$) are mapped to the corresponding chance nodes ($A_j^1, A_j^2$), which is indicated by the dotted arrows. Depending on the value of node, $Mod[M_j]$, distribution of each of the chance nodes is assigned to node $A_j$ with some probability.

$b_{j,l-1}$ is the level $l - 1$ belief, and $\hat{\theta}_j$ is the agent’s frame encompassing the action, observation, and utility nodes. Because the model node contains the alternative models of the other agent as its values, its representation is not trivial. In particular, some of the models within the node are I-IDs that when solved generate the agents optimal policy in their decision nodes. Each decision node is mapped to the corresponding chance node, say $A_j^1$, in the following way: if $OPT$ is the set of optimal actions obtained by solving the I-ID (or ID), then $Pr(a_j \in A_j^1) = \frac{1}{|OPT|}$ if $a_j \in OPT$, 0 otherwise.

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. 2.6(b). The decision node of each level $l - 1$ I-ID is transformed into a chance node as we mentioned previously, so that the actions with the largest value in the decision node are assigned uniform probability in the chance node while the
rest are assigned zero probability. 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, \(A_j\). Thus, there are as many action nodes \((A_j^1, A_j^2)\) in \(M_{j,l-1}\) as the number of models in the support of agent \(i\)'s beliefs. 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 weights) given the physical state \(S\). The conditional probability table (CPT) 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 value of \(Mod[M_j]\). In other words, when \(Mod[M_j]\) has the value \(m_{j,l-1}^1\), 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 value \(m_{j,l-1}^2\). The distribution over \(Mod[M_j]\), is \(i\)'s belief over \(j\)'s models given the state. For more than two agents, we add a model node and a chance node representing the distribution over an agent’s action linked together using a policy link, for each of the other agents. Notice that in Fig. 2.6(b), the 'policy link' is shown and it can be seen how it is represented using traditional dependency links.

![Image](image.png)

Figure 2.7: The transformed I-ID with the model node replaced by the chance nodes and the relationships between them.

In Fig. 2.7, we show the transformed I-ID when the model node is replaced by the chance nodes and relationships between them. In contrast to the representation in [21], there are no special-
purpose policy links, rather the I-ID is composed of only those types of nodes that are found in traditional IDs and dependency relationships between the nodes.

2.4.2 Solution

Solution of an I-ID proceeds in a bottom-up manner, and is implemented recursively.

1. Solve the lower level models, which are traditional IDs or BNs. Their solutions 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.

2. The mapping from the level 0 models decision nodes to the chance nodes is carried out so that actions with the largest value in the decision node are assigned uniform probabilities in the chance node while the rest are assigned zero probability.

3. 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.

4. During the transformation, the CPD 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, $Mod[M_j]$.

5. The transformed I-ID is a traditional ID that may be solved using the standard expected utility maximization method [12].

6. This procedure is carried out up to the level l I-ID whose solution gives the non-empty set of optimal actions that the agent should perform given its belief. Notice that analogous to IDs, I-IDs are suitable for online decision-making when the agents current belief is known.

2.5 Interactive Dynamic Influence Diagrams (I-DIDs)

In this section we describe the interactive dynamic influence diagrams (I-DIDs) for two-agent interactions which are the extensions of interactive influence diagrams to dynamic settings, I-DIDs, and refer the reader to [2] for more details.
I-DIDs extend I-IDs to allow sequential decision making over several time steps (see Fig. 2.8). Just as DIDs are structured graphical representations of POMDPs, I-DIDs are the graphical online analogs for finitely nested I-POMDPs. I-DIDs may be used to optimize over a finite look-ahead given initial beliefs while interacting with other, possibly similar, agents.

2.5.1 Syntax

We depict a general two time-slice I-DID in Fig. 2.8. In addition to the model nodes and the dashed policy link, what differentiates an I-DID from a DID is the model update link shown as a dotted arrow in Fig. 2.8. We briefly explain the semantics of the model update next. The update of the model node over time involves two steps: First, given the models at time $t$, we identify the updated set of models that reside in the model node at time $t + 1$. Because the agents act and receive observations, their models are updated to reflect their changed beliefs. Since the set of optimal actions for a model could include all the actions, and the agent may receive any one of $|\Omega_j|$ possible observations, the updated set at time step $t + 1$ will have up to $|\mathcal{M}_{j,t-1}||A_j||\Omega_j|$ models. Here, $|\mathcal{M}_{j,t-1}|$ is the number of models at time step $t$, $|A_j|$ and $|\Omega_j|$ are the largest spaces of actions and observations respectively, among all the models. The CPT of $\text{Mod}[\mathcal{M}^{t+1}_{j,t-1}]$ encodes the function, 

$$\tau(b_{j,t-1}^t, a_j^t, o_{j,t+1}^{t+1}, b_{j,t-1}^{t+1})$$

which is 1 if the belief $b_{j,t-1}^t$ in the model $m_{j,t-1}^t$ using the action $a_j^t$ and observation $o_{j,t+1}^{t+1}$ updates to $b_{j,t-1}^{t+1}$ in a model $m_{j,t-1}^{t+1}$; otherwise it is 0. Second, we compute the
new distribution over the updated models, given the original distribution and the probability of the agent performing the action and receiving the observation that led to the updated model. The dotted model update link in the I-DID may be implemented using standard dependency links and chance nodes, as shown in Fig. 2.9 transforming it into a flat DID. In Fig. 2.9, we show how the dotted model update link is implemented in the I-DID. If each of the two level $l - 1$ models ascribed to $j$ at time step $t$ results in one action, and $j$ could make one of two possible observations, then the model node at time step $t + 1$ contains four updated models ($m_{j,l-1}^{t+1,1}$, $m_{j,l-1}^{t+1,2}$, $m_{j,l-1}^{t+1,3}$, and $m_{j,l-1}^{t+1,4}$). These models differ in their initial beliefs, each of which is the result of $j$ updating its beliefs due to its action and a possible observation. The decision nodes in each of the I-DIDs or DIDs that represent the lower level models are mapped to the corresponding chance nodes, as mentioned previously. Next, we describe how the distribution over the updated set of models (the distribution over the chance node $\texttt{Mod}[M_j^{t+1}]$ in $M_{j,l-1}^{t+1}$) is computed. The probability that $j$’s updated model is, say $m_{j,l-1}^{t+1,1}$, depends on the probability of $j$ performing the action and receiving the observation that led to this model, and the prior distribution over the models at time step $t$. Because the chance node $A_j^t$ assumes the distribution of each of the action nodes based on the value of $\texttt{Mod}[M_j^t]$, the
probability of the action is given by this chance node. In order to obtain the probability of \( j \)'s possible observation, we introduce the chance node \( O_j \), which depending on the value of \( \text{Mod}[M_j^t] \) assumes the distribution of the observation node in the lower level model denoted by \( \text{Mod}[M_j^t] \). Because the probability of \( j \)'s observations depends on the physical state and the joint actions of both agents, the node \( O_j \) is linked with \( S^{t+1}, A_j^t, \) and \( A_j^t \). Analogous to \( A_j^t \), the conditional probability table of \( O_j \) is also a multiplexer modulated by \( \text{Mod}[M_j^t] \). Finally, the distribution over the prior models at time \( t \) is obtained from the chance node, \( \text{Mod}[M_j^t] \) in \( \text{Mod}[M_{j,t-1}^t] \). Consequently, the chance nodes, \( \text{Mod}[M_j^t], A_j^t, \) and \( O_j \), form the parents of \( \text{Mod}[M_j^{t+1}] \) in \( M_{j,t-1}^{t+1} \). Notice that the model update link may be replaced by the dependency links between the chance nodes that constitute the model nodes in the two time slices. In Fig. 2.10 we show the two time-slice I-DID with the model nodes replaced by the chance nodes and the relationships between them. Chance nodes and dependency links that not in bold are standard, usually found in DIDs. Expansion of the I-DID over more time steps requires the repetition of the two steps of updating the set of models.
that form the values of the model node and adding the relationships between the chance nodes, as many times as there are model update links. We note that the possible set of models of the other agent $j$ grows exponentially with the number of time steps. For example, after $T$ steps, there may be at most $|\mathcal{M}_{j|l-1}^T|(|A_j||\Omega_j|)^{T-1}$ candidate models residing in the model node.

**Solution**

Analogous to I-IDs, the solution to a level $l$ I-DID for agent $i$ expanded over $T$ time steps may be carried out recursively. For the purpose of illustration, let $l = 1$ and $T = 2$. The solution method uses the standard look-ahead technique, projecting the agents action and observation sequences forward from the current belief state [19], and finding the possible beliefs that $i$ could have in the next time step. Because agent $i$ has a belief over $j$’s models as well, the lookahead includes finding out the possible models that $j$ could have in the future. Consequently, each of $j$’s subintentional or level 0 models (represented using a standard DID) in the first time step must be solved to obtain its optimal set of actions. These actions are combined with the set of possible observations that $j$ could make in that model, resulting in an updated set of candidate models (that include the updated beliefs) that could describe the behavior of $j$. Beliefs over this updated set of candidate models are calculated using the standard inference methods using the dependency relationships between the model nodes as shown in Fig. 2.9. We note the recursive nature of this solution: in solving agent $i$’s level 1 I-DID, $j$’s level 0 DIDs must be solved. If the nesting of models is deeper, all models at all levels starting from 0 are solved in a bottom-up manner. We briefly outline the recursive algorithm for solving agent $i$’s level $l$ I-DID expanded over $T$ time steps with one other agent $j$ in Fig. 2.11. We adopt a two-phase approach: Given an I-ID of level $l$ (described previously in Section 2.4) with all lower level models also represented as I-IDs or IDs (if level 0), the first step is to expand the level $l$ I-ID over $T$ time steps adding the dependency links and the conditional probability tables for each node. We particularly focus on establishing and populating the model nodes (lines 3-11). In the second phase, we use a standard look-ahead technique projecting the action and observation sequences over $T$ time steps in the future, and backing up the utility values
**I-DID EXACT** (level $l \geq 1$ I-ID or level 0 ID, $T$)

**Expansion Phase**

1. For $t$ from 1 to $T - 1$ do
2. If $l \geq 1$ then
   - Populate $M_{j,l-1}^{t+1}$
3. For each $m_j^t$ in $M_{j,l-1}^t$ do
4. Recursively call algorithm with the $l-1$ I-ID (or ID) that represents $m_j^t$ and the horizon, $T - t + 1$
5. Map the decision node of the solved I-ID (or ID), $OPT(m_j^t)$, to the chance node $A_j^t$
6. For each $a_j$ in $OPT(m_j^t)$ do
   - For each $o_j$ in $O_j$ (part of $m_j^t$) do
     - Update $j$'s belief, $b_{j+1}^t \leftarrow SE(b_j^t, a_j, o_j)$
     - $m_{j+1}^t$ $\leftarrow$ New I-ID (or ID) with $b_{j+1}^t$ as belief
   - $M_{j,l-1}^{t+1} \cup \{m_j^t\}$
11. Add the model node, $M_{j,l-1}^{t+1}$, and the model update link between $M_{j,l-1}^t$ and $M_{j,l-1}^{t+1}$
12. Add the chance, decision and utility nodes for $t+1$ time slice and the dependency links between them
13. Establish the CPTs for each chance node and utility node

**Look-Ahead Phase**

14. Apply the standard look-ahead and backup method to solve the expanded I-DID (other solution approaches may also be used)

---

Figure 2.11: Algorithm for exactly solving a level $l \geq 1$ I-DID or level 0 DID expanded over $T$ time steps.

of the reachable beliefs. Similar to I-IDs, the I-DIDs reduce to DIDs in the absence of other agents. As we mentioned previously, the 0-th level models are the traditional DIDs. Their solutions provide probability distributions over actions of the agent modeled at that level to I-DIDs at level 1. Given probability distributions over other agents actions the level 1 IDIDs can themselves be solved as DIDs, and provide probability distributions to yet higher level models. Assume that the number of models considered at each level is bound by a number, $M$. Solving an I-DID of level $l$ is then equivalent to solving $O(M_l)$ DIDs.
2.5.2 CURSES OF DIMENSIONALITY AND HISTORY IN I-DIDs

As we may expect, I-DIDs acutely suffer from both the curses of dimensionality and history. This is because the state space in I-DIDs includes the models of other agents in addition to the traditional physical states. These models encompass the agents’ beliefs, action and sensory capabilities, and preferences, and may themselves be formalized as I-DIDs. The nesting is terminated at the 0th level where the other agents are modeled using DIDs. As the agents act, observe, and update beliefs, I-DIDs must track the evolution of the models over time. Consequently, I-DIDs not only suffer from the curse of history that afflicts the modeling agent, but more so from that exhibited by the modeled agents. The exponential growth in the number of models over time also further contributes to the dimensionality of the state space. This is complicated by the nested nature of the space. This is why most researchers have diverted their focus to finding new and better exact and approximation techniques to solve such complex sequential decision making problems and hence, account for larger number of states (bigger problems), look ahead steps/horizons and deeper nesting.

Previous approaches for approximating I-DIDs [1, 14] focus on reducing the dimensionality of the state space by limiting the number of candidate models of other agents. Using the insight that beliefs that are spatially close are likely to be behaviorally equivalent [10], Zeng et al. [14] cluster the models of other agents and select representative models from each cluster. Intuitively, a cluster contains models that are likely to be behaviorally equivalent and hence may be replaced by a subset of representatives without a significant loss in the optimality of the decision maker. However, this approach often retains more models than needed. Doshi and Zeng ([1]) formalize the concept of a minimal set of models, previously discussed by Pynadath and Marsella [9]. At each time step, only those models are updated which will result in predictive behaviors that are distinct from others in the updated model space. The initial set of models are solved and merged to obtain a policy graph, which assists in discriminating between model updates. These approaches will be dealt in greater detail in Chapter 3.
Suryadi and Gmytrasiewicz [12] in an early piece of related work, proposed modeling other agents using IDs. The approach proposed ways to modify the IDs to better reflect the observed behavior. However, unlike I-DIDs, other agents did not model the original agent and the distribution over the models was not updated over time based on the actions and observations.

I-DIDs contribute to a growing line of work that includes multiagent influence diagrams (MAIDs) [6], and more recently, networks of influence diagrams (NIDs) [3, 4]. These formalisms seek to explicitly and transparently model the structure that is often present in real-world problems by decomposing the situation into chance and decision variables, and the dependencies between the variables. MAIDs provide an alternative to normal and extensive forms of games, using a graphical formalism to represent games of imperfect information with a decision node for each agent’s actions and chance nodes capturing the agent’s private information. 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. Each model is a MAID and the network of MAIDs is collapsed, bottom up, into a single MAID for computing the equilibrium of the game keeping in mind the different models of each agent.

Both MAIDs and NIDs provide an analysis of the game from an external viewpoint, and adopt Nash equilibrium as the solution concept. However, equilibrium is not unique – there could be many joint solutions in equilibrium with no clear way to choose between them – and incomplete – the solution does not prescribe a policy when the policy followed by the other agent is not part of the equilibrium. Specifically, MAIDs do not allow us to define a distribution over non-equilibrium
behaviors of other agents. Furthermore, their applicability is limited to static single play games. Interactions are more complex when they are extended over time, where predictions about others’ future actions must be made using models that change as the agents act and observe. I-DIDs seek to address this gap by offering an intuitive way to extend sequential decision making as formalized by DIDs to multiagent settings. They allow the explicit representation of other agents’ models as the values of a special model node. Other agents’ models and the original agent’s beliefs over these models are then updated over time.

As we mentioned, a dominating cause of the complexity of I-DIDs is the exponential growth in the candidate models over time. Using the heuristic that models whose beliefs are spatially close are likely to be behaviorally equivalent, Doshi et al. [2] utilized a $k$-means approach to cluster models together and select $K$ models closest to the means in the model node at each time step. While this approach requires all models to be expanded before clustering is applied, a recent approach [1] preemptively avoids expanding models that will turn out to be behaviorally equivalent to others. By discriminating between model updates, the approach guarantees a minimal set of models in each non-initial model node. To avoid solving all of the initial models, the approach clusters models whose beliefs are within a parameter, $\epsilon$. Minimal sets of models were previously discussed by Pynadath and Marsella in [9], which used the concept of behavioral equivalence, introduced earlier [10], to form the space.

In this chapter, we will also discuss the following exact and approximation techniques in some level of detail and refer the readers to their respective papers for more information:

1. Exact algorithm to solve I-DIDs

   Using Behavioral Equivalence (BE).

2. Approximate algorithms to solve I-DIDs

   Using Model Clustering (MC).

   Using Discriminative Model Updates (DMU).
3.1 EXACTLY SOLVING I-DIDS USING BEHAVIORAL EQUIVALENCE

Since the BE approach lays the foundation for our new approximation technique ($\epsilon$-Behavioral Equivalence), we will discuss this approach in greater detail. However, an overview of the approximate algorithms will also be presented to enable the readers understand the need for an improved approximation method.

3.1.1 BEHAVIORAL EQUIVALENCE (BE)

In order to reduce the dimensionality of the state space, it is required to reduce the number of models being solved at every time step. At the same time, doing so will reduce the optimality of the solution if the actual models of the other agents were pruned before they were solved. Hence, it is important to carefully prune models from the infinitely large model space. Some methods limit the maximum number of models they solve at each time step as a way to mitigate the impact of the history that afflicts the other modeled agent. Although the space of possible models is very large/infinite, not all models need to be considered. This is because some models in the model node of the I-DID have behavioral predictions for the other agent that are identical. These models are classified as behaviorally equivalent [9, 10]. Thus, all such models could be pruned and a single representative model could be considered. This is because the solution of the subject agent’s I-DID is affected by the predicted behavior of the other agent only; thus we need not distinguish between behaviorally equivalent models.

The main idea of the exact algorithm to solve I-DIDs using behavioral equivalence is to aggregate such behaviorally equivalent models into a finite number of equivalence classes and instead of reasoning over the infinite set of interactive states, they operate over the finite set of equivalence classes each having one representative model.

In order to clearly understand the construction of behavioral equivalence classes, let us consider a simple example - the classical tiger problem introduced in [20]. According to the problem, there is an agent waiting to open one of two doors. Behind one of the doors, there is a tiger that would eat the agent that opens that door and behind the other is a pot of gold. There is a reward of +10 to get
the gold and -100 when the agent is eaten by the tiger. There are two states signifying the location of the tiger - TL, when the tiger is behind the left door and TR, when the tiger is behind the right door. The agent can choose to perform one of three actions - opening the left door (OL), opening the right door (OR), and listen (L). The agent can receive three observations when it chooses to listen that will guide it to making the right decision - GL, when it hears a Tiger’s growl from behind the left door, and GR, when it hears a growl from behind the right door each with 85% certainty. The value function gives the value of performing the optimal plan given the belief. In Fig. 3.1, we show the value function in the tiger game and the belief ranges corresponding to different optimal actions. We note that the agent opens the right door if it believes the probability that the tiger is behind the right door is less than 0.1. It will listen if the probability is between 0.1 and 0.9 and open the left door if the probability is greater than 0.9. We observe that each optimal action spans over multiple belief points. For example, opening the right door is the optimal action for all beliefs in the set [0-0.1). Thus, the beliefs in the set [0-0.1) are equivalent in that it induces the same optimal behavior. Such beliefs are \textit{behaviorally equivalent}. The collection of the equivalence classes forms a partition in the belief space. For finite horizons, and a finite number of actions and observations,
the number of distinct optimal actions and therefore the number of equivalence classes is also finite.

Using this insight, behavioral equivalence was used to solve I-DIDs exactly by pruning the models that induced the same optimal behavior and replacing all the models in the behavioral equivalence class with one representative model. This way at every time step, the number of models that have to be solved is reduced to only the number of these equivalence classes. Let \( \text{BehavioralEq}(\mathcal{M}_{j,l-1}) \) be the procedure that prunes the behaviorally equivalent models from \( \mathcal{M}_{j,l-1} \) returning the set of representative models. The algorithm for exactly solving I-DIDs using Behavioral equivalence (BE) is given below.

The solution of an I-DID (and I-ID) proceeds in a bottom-up manner, and is implemented recursively as shown in Fig. 3.2. We start by solving the level 0 models, which may be traditional DIDs. Their solutions provide probability distributions over the other agents’ actions, which are entered in the corresponding action nodes found in the model node of the level 1 I-DID. The mapping from the level 0 models’ decision nodes to the chance nodes is carried out so that actions with the largest value in the decision node are assigned uniform probabilities in the chance node while the rest are assigned zero probability. The solution method uses the standard look-ahead technique, projecting the agent’s action and observation sequences forward from the current belief state, and finding the possible beliefs that \( i \) could have in the next time step. Because agent \( i \) has a belief over \( j \)'s models as well, the look-ahead includes finding out the possible models that \( j \) could have in the future. Consequently, each of \( j \)'s level 0 models represented using a standard DID in the first time step must be solved to obtain its optimal set of actions. These actions are combined with the set of possible observations that \( j \) could make in that model, resulting in an updated set of candidate models (that include the updated beliefs) that could describe the behavior of \( j \). \( SE(b_j^l, a_j, o_j) \) is an abbreviation for the belief update. The updated set is minimized by excluding the behaviorally equivalent models. Beliefs over these updated set of candidate models are calculated using the standard inference methods through the dependency links between the model nodes (Fig. 2.9). The algorithm in Fig. 3.2 may be realized using the standard implementations of DIDs.
| I-DID EXACT (level $l \geq 1$ I-DID or level 0 DID, $T$) |
| Expansion Phase |
| 1. For $t$ from 1 to $T - 1$ do |
| 2. If $l \geq 1$ then |
| Populate $M_{j,l-1}^{t+1}$ |
| 3. For each $m_j^{t}$ in $M_{j,l-1}^{t}$ do |
| 4. Recursively call algorithm with the $l - 1$ I-DID (or DID) that represents $m_j^{t}$ and the horizon, $T - t$ |
| 5. Map the decision node of the solved I-DID (or DID), $OPT(m_j^{t})$, to the chance node $A_j^{t}$ |
| 6. For each $a_j$ in $OPT(m_j^{t})$ do |
| 7. For each $o_j$ in $O_j$ (part of $m_j^{t}$) do |
| 8. Update $j$’s belief, $b_j^{t+1} \leftarrow SE(b_j^{t}, a_j, o_j)$ |
| 9. $m_j^{t+1} \leftarrow$ New I-DID (or DID) with $b_j^{t+1}$ as belief |
| 10. $M_{j,l-1}^{t+1} \leftarrow \{m_j^{t+1}\}$ |
| 11. Add the model node, $M_{j,l-1}^{t+1}$, and the model update link between $M_{j,l-1}^{t}$ and $M_{j,l-1}^{t+1}$ |
| 12. Add the chance, decision and utility nodes for $t+1$ time slice and the dependency links between them |
| 13. Establish the CPTs for each chance node and utility node |

| Solution Phase |
| 14. If $l \geq 1$ then |
| 15. Represent the model nodes and the model update link as in Fig. 2.9 to obtain the DID |
| Minimize model spaces |
| 16. For $t$ from 1 to $T$ do |
| 17. $M_{j,l-1}^{t} \leftarrow BehavioralEq(M_{j,l-1}^{t})$ |
| 18. Apply the standard look-ahead and backup method to solve the expanded DID (other solution approaches may also be used) |

Figure 3.2: Algorithm for exactly solving a level $l \geq 1$ I-DID or level 0 DID expanded over $T$ time steps.

3.2 APPROXIMATELY SOLVING I-DIDS USING MODEL CLUSTERING

This approach was introduced by Zeng et al. [14]. They presented a method to reduce the dimensionality of the interactive state space and mitigate the impact of the curse of history. This is done by limiting and holding a constant number of models, $0 < K << M$, where $M$ is the possibly large
number of candidate models of the other agent included in the state space. Using the insight that beliefs that are spatially close are likely to be behaviorally equivalent [10], Rathnas et al. cluster the models of the other agents and select representative models from each cluster. They utilize the popular k-means clustering method, which gives an iterative way to generate the clusters. Intuitively, the clusters contain models that are likely to be behaviorally equivalent and hence may be replaced by a subset of representative models without a significant loss in the optimality of the decision maker. \( K \) representative models from the clusters are selected and updated over time.

### 3.2.1 Model Clustering Approach

The approximation technique is based on clustering the agent models and selecting \( K \), where \( 0 < K \ll M \), representative models from the clusters. In order to initiate clustering, the initial means was identified around which the models were clustered. The selection of the initial means is crucial as it is hoped to select them minimally and avoid discarding models that are behaviorally distinct from the representative ones. The initial means were selected as those that lie on the intersections of the behaviorally equivalent regions (see previous section for an illustration to help understand how these regions were constructed). This allows models that are likely to be behaviorally equivalent to be grouped on each side of the means. These intersection points are called sensitivity points (SPs). In order to compute the SPs, we observe that they are the beliefs at the non-dominated intersection points (or lines) between the value functions of pairs of policy trees. A linear program (LP) shown in [14] provides a straightforward way of computing the SPs. If the intersections were lines, then the LP returned a point on this line. The initial clusters group together models of the other agent possibly belonging to multiple behaviorally equivalent regions. Additionally, some of the SPs may not be candidate models of the other agent \( j \) as believed by the subject agent \( i \). In order to promote clusters of behaviorally equivalent models and segregate the non-behaviorally equivalent ones, the means are updated using an iterative method often utilized by the \( k\text{-means} \) clustering approach. This iterative technique converges because over increasing iterations less new models will be added to a cluster, thereby making the means gradually invariant. Given the stable clusters, a
total of $K$ representative models are selected from them. Depending on its population, each cluster contributes a proportion $k$ of models to the set. The $k$ models whose beliefs are the closest to the mean of the cluster are selected for inclusion in the set of models that are retained. Remaining models in the cluster are discarded. The selected models provide representative behaviors for the original set of models included in the cluster. The algorithm for approximately solving I-DIDs using model clustering is a slight variation of the one in Fig. 2.7 that solves I-DIDs exactly. In particular, on generating the candidate models in the model node during the expansion phase, $K$ models are selected after clustering using the procedure $KModelSelection$ explained in [14]. It can be noted that models at all levels will be clustered and pruned. Also, this approach is more suited to situations where agent $i$ has some prior knowledge about the possible models of others, thereby facilitating the clustering and selection. We refer the readers to [14] for more details on this approach.

As mentioned earlier, the insight for this approach comes from the fact that behaviorally equivalent models are spatially closer to each other than the behaviorally distinct ones. However, this approach first generates all possible models before reducing the space at each time step, and utilizes an iterative and often time-consuming k-means clustering method. Despite its favorable results when compared to the exact approaches, it can be noted that there is no way to show the degree to which models are behaviorally equivalent. Our approximation technique ($\epsilon$-Behavioral Equivalence), provides a new definition for behavioral equivalence in terms of the distribution of the future action-observation paths, that allows a way to measure the degree to which the models are behaviorally equivalent. Apart from this, the chapter on Theoretical Analysis contains more information that will magnify the advantages of our approach over the model clustering approach.

3.3 APPROXIMATELY SOLVING I-DIDs USING DISCRIMINATIVE MODEL UPDATES

This approximation method was introduced by Doshi and Zeng [1]. They formalize a minimal set of models of others, a concept discussed in [9]. They present a new approach for approximating
I-DIDs that significantly reduces the space of possible models of other agents that needs to be con-
sidered by discriminating between model updates. In this approximation technique, solving all the
initial models is avoided. First, $K$ models are randomly selected from the candidate model space
and solved. For each of the remaining models, if the belief of that model is close to that of one of
the solved models by atleast a threshold (supplied as input), then that model assumes the solution of
the solved model. Otherwise, the model is solved. That is, at each time step, only those models are
selected for updating which will result in predictive behaviors that are distinct from others in the
updated model space. In other words, models that on update would result in predictions which are
identical to those of existing models are not selected for updating. For these models, their revised
probability masses are simply transfered to the existing behaviorally equivalent models. The solu-
tions of the solved models are then merged bottom up to obtain the policy graph. Intuitively, this
approach improves on the previous one that uses model clustering (discussed earlier) because it
does not generate all possible models prior to selection at each time step; rather it results in min-
imal sets of models. In order to avoid updating all models, the regions of the belief space are found
so that models whose beliefs fall in these regions will be behaviorally equivalent on update. Note
that these regions need not be in spatial proximity. Because obtaining the exact regions is comput-
tationally intensive, we approximately obtain these regions by solving a subset of the models and
utilizing their combined policies. We empirically compare this approach with our approximation
method in terms of the the average rewards obtained and results are shown in Chapter 7. For more
details on this approach, we refer the readers to [1].
Our aim is to identify models that are approximately BE. While a pair of policy trees may be checked for equality, disparate policy trees do not directly permit intuitive behavioral comparisons. In this chapter, we provide a new definition for behavioral equivalence in terms of the distribution of the future action-observation paths, that allows a way to measure the degree to which the models are behaviorally equivalent which helps formulate our approximation technique, \((\varepsilon\text{-Behavioral Equivalence})\). We first assume that the models of the other agent \(j\) have identical frames and differ only in their beliefs. As we mentioned previously, two models of the other agent are behaviorally equivalent (BE)\(^1\) if they produce identical behaviors for the other agent. More formally, models \(m_{j,l-1}, \hat{m}_{j,l-1} \in M_{j,l-1}\) are BE if and only if \(OPT(m_{j,l-1}) = OPT(\hat{m}_{j,l-1})\), where \(OPT(\cdot)\) denotes the solution of the model that forms the argument. If the model is a DID or an I-DID, its solution is a policy tree.

As mentioned earlier, since our aim is to identify models that are approximately BE, and since disparate policy trees do not directly permit intuitive behavioral comparisons making it difficult to define a measure of approximate BE, this motivates us to investigate into a more rigorous formalization of BE.

Recall that BE models impact the decision-making of the modeling agent similarly, thereby motivating interest in grouping such models together. We utilize this insight toward introducing a new definition of BE.

\(^1\)We will use BE as an acronym for both, behaviorally equivalent (adjective form) and behavioral equivalence (noun form). Appropriate usage will be self-evident.
4.1 Definition

Let $h = \{a^t_i, o^{t+1}_i\}_{t=1}^T$ be the action-observation path for the modeling agent $i$, where $o^{T+1}_i$ is null for a $T$ horizon problem. If $a^t_i \in A_i$ and $o^{t+1}_i \in \Omega_i$, where $A_i$ and $\Omega_i$ are $i$’s action and observation sets respectively, then the set of all paths is, $H = \Pi^T_1(A_i \times \Omega_i)$, and the set of action-observation histories up to time $t$ is $H^t = \Pi^{t-1}_1(A_i \times \Omega_i)$. The set of future action-observation paths is, $H_{T-t} = \Pi^T_t (A_i \times \Omega_i)$, where $t$ is the current time step.

We show example action-observation paths of agent $i$ in Fig. 4.1. The figure shows the future action-observation paths of agent $i$ in a 2-horizon multiagent tiger problem. The nodes represent $i$’s action, while the edges are labeled with the possible observations that $i$ can perceive. In this example, agent $i$ starts with listening and then $i$ may receive one of six observations conditional on $j$’s action. Note that we have considered the future action-observation path of agent $i$ and not the joint future action-observation path of agents $i$ and $j$. The reason being the fact that the subject agent, $i$’s actions are conditioned on the other agent $j$’s actions and observation. Hence, changes in the policy of the candidate models of agent $j$ will indirectly affect the policy of agent $i$. Hence, using just the future action-observation paths of agent $i$ is justified. Each of these paths have a probability associated with it. This probability is the chance with which that particular path is chosen by the subject agent $i$. The sum of each of these future action-observation path probabilities is 1. It may also be noted that, as the number of time steps increase the number of action-observation paths and hence the size of the distribution table containing individual path probabilities, increase exponentially. This was one of the main reasons for memory issues when the algorithm was executed. Also, the size of the distribution is directly proportional to the the number of actions and observations for agent $i$.

We observe that agent $j$’s model together with agent $i$’s perfect knowledge of its own model and its action-observation history induces a predictive distribution over $i$’s future action-observation paths such as those in Fig. 4.1. This distribution plays a critical role in our approach and we denote it as, $Pr(H_{T-t} | h^t, m_{i,l}, m_{j,l-1}^t)$, where $h^t \in H^t$, $m_{i,l}$ is $i$’s level $l$ I-DID and $m_{j,l-1}^t$ is the level $l-1$ model of $j$ in the model node at time $t$. For the sake of brevity, we rewrite the distribution term as,
Figure 4.1: Future action-observation paths of agent $i$ in a 2-horizon multiagent tiger problem. The nodes represent $i$'s action, while the edges are labeled with the possible observations. This example starts with $i$ listening. Agent $i$ may receive one of six observations conditional on $j$'s action, and performs an action that optimizes its resulting belief.

$$Pr(H_{T-t}|m_{i,l}^t, m_{j,l-1}^t),$$ where $m_{i,l}^t$ is $i$'s horizon $T - t$ I-DID with its initial belief updated given the actions and observations in $h^t$. We will present a way to compute this distribution later in the next section. We define BE below:

**Definition 2** (Behavioral Equivalence). Two models of agent $j$, $m_{j,l-1}^t$ and $\tilde{m}_{j,l-1}^t$, are behaviorally equivalent if and only if $Pr(H_{T-t}|m_{i,l}^t, m_{j,l-1}^t) = Pr(H_{T-t}|m_{i,l}^t, \tilde{m}_{j,l-1}^t)$, where $H_{T-t}$ and $m_{i,l}^t$ are as defined previously.

In other words, BE models are those that induce an identical distribution over agent $i$’s future action-observation history. This reflects the fact that such models impact agent $i$’s behavior similarly.

### 4.2 Computing the Distribution over Future Paths

As mentioned earlier, each of the future action-observation paths have a probability associated with it. This probability is the chance with which that particular path is chosen by the subject agent $i$. These probabilities of all the paths put together constitute future distribution over the action-observation paths of agent $i$. The sum of each of these future action-observation path probabilities in a distribution is 1. Because our method uses this distribution in redefining BE and hence, in the approximation technique, its exponential increase in size over increasing time steps, is a cause for
worry as memory issues may arise after a limit has been reached. The size of the distribution is also directly proportional to the the number of actions and observations for agent \( i \). Let \( h_{T-t} \) be some future action-observation path of agent \( i \), \( h_{T-t} \in H_{T-t} \). In Proposition 1, we provide a recursive way to arrive at the probability, \( Pr(h_{T-t}|m_{i,t}, m_{j,t-1}) \). Of course, the probabilities over all possible paths sum to 1.

**Proposition 1.**

\[
Pr(h_{T-t}|m_{i,t}, m_{j,t-1}) = Pr(a_i^t, o_i^{t+1}|m_{i,t}, m_{j,t-1}) \sum a_j^{t+1} Pr(h_{T-t-1}|a_i^t, o_i^{t+1}, m_{i,t}, a_j^{t+1}, m_{j,t-1})
\]

\[
\times Pr(a_j^{t+1}|a_i^t, m_{i,t}, m_{j,t-1})
\]

\[
= Pr(a_i^t, o_i^{t+1}|m_{i,t}, m_{j,t-1}) \sum a_j^{t+1} Pr(h_{T-t-1}|m_{i,t}^{t+1}, m_{j,t-1}) \times Pr(a_j^{t+1}|a_i^t, m_{i,t}, m_{j,t-1})
\]

where

\[
Pr(a_i^t, o_i^{t+1}|m_{i,t}, m_{j,t-1}) = Pr(a_i^t|OPT(m_{i,t})) \sum a_j^t Pr(a_j^t)
\]

\[
OPT(m_{j,t-1})) \sum s_{t+1} O_i(s_{t+1}, a_i^t, a_j^{t+1})
\]

\[
\times \sum s, m_j T_i(s, a_i^t, a_j^{t+1}, s_{t+1}) b_{i,t}(s, m_j)
\]

and

\[
Pr(a_j^{t+1}|a_i^t, m_{i,t}, m_{j,t-1}) = Pr(a_j^t|OPT(m_{j,t-1})) \sum s_{t+1}
\]

\[
O_j(s_{t+1}, a_i^t, a_i^t, a_j^{t+1}) \sum s, m_j T_i(s, a_i^t, a_j^{t+1}, s_{t+1}) b_{i,t}(s, m_j)
\]

In Eq. 4.1, \( O_i(s_{t+1}, a_i^t, a_j^{t+1}) \) is \( i \)'s observation function contained in the CPT of the chance node, \( O_i^{t+1} \), in the I-DID, \( T_i(s, a_i^t, a_j^{t+1}, s_{t+1}) \) is \( i \)'s transition function contained in the CPT of the chance node, \( S_{t+1} \), \( Pr(a_i^t|OPT(m_{i,t})) \) is obtained by solving agent \( i \)'s I-DID, \( Pr(a_j^t|OPT(m_{j,t-1})) \) is obtained by solving \( j \)'s model and appears in the CPT of node, \( A_j \).

In Eq. 4.2, \( O_j(s_{t+1}, a_i^t, a_i^t, a_j^{t+1}) \) is \( j \)'s observation function contained in the CPT of the chance node, \( O_j^{t+1} \), given \( j \)'s model is \( m_{j,t-1} \). We give the proof of Proposition 1 below.

**Proof of Proposition 1.**

\[
Pr(h_{T-t}|m_{i,t}, m_{j,t-1}) =
Pr(h_{T-t-1}, a_i^t, o_i^{t+1}|m_{i,t}, m_{j,t-1}) = Pr(h_{T-t-1}|a_i^t, o_i^{t+1}, m_{i,t}, m_{j,t-1})
\]

\[
\times Pr(a_i^t, o_i^{t+1}|m_{i,t}, m_{j,t-1})
\]

(using Bayes rule)

We focus on the first term next:

\[
Pr(h_{T-t-1}|a_i^t, o_i^{t+1}, m_{i,t}, m_{j,t-1}) = \sum a_j^{t+1} Pr(h_{T-t-1}|a_i^t),
\]
In the above equation, the first term results due to an update of the models at time step $t$ with actions and observations. This term is computed recursively. For the second term, $j$’s level $l - 1$ actions and observations are independent of $i$’s observations.

We now focus on the term, $Pr(a_i', o_t^{l+1}|m_{i,t}, m_{j,t-1})$:

\[
Pr(a_i', o_t^{l+1}|m_{i,t}, m_{j,t-1}) = Pr(a_i^{l+1}|a_i', m_{i,t}, m_{j,t-1}) \\
\times Pr(a_j'|OPT(m_{j,t-1})) \quad (i$’s action is conditionally independent of $j$ given its model)
\]

\[
= Pr(a_i'|OPT(m_{j,t-1})) \sum_{a_j} Pr(a_j^{l+1}|a_i', a_j', m_{i,t}, m_{j,t-1}) \\
\times Pr(a_j'|OPT(m_{j,t-1})) \quad (i$’s observation is conditionally independent of $j$’s model)
\]

\[
= Pr(a_i'|OPT(m_{j,t-1})) \sum_{a_j} Pr(a_j'|OPT(m_{j,t-1})) Pr(o_{l+1}|a_i', a_j', b_{t,l}) \quad (b_{t,l} is i$’s belief in model, $m_{i,t}$)
\]

\[
= Pr(a_i'|OPT(m_{j,t-1})) \sum_{a_j} Pr(a_j'|OPT(m_{j,t-1})) \\
\times \sum_{s^{l+1}} Pr(o_{l+1}|s^{l+1}, a_i', a_j') Pr(s^{l+1}|a_i', a_j', b_{t,l}) \\
= Pr(a_i'|OPT(m_{j,t-1})) \sum_{a_j} Pr(a_j'|OPT(m_{j,t-1})) \sum_{s^{l+1}} O_i(s^{l+1}, a_i', a_j', a_j^{l+1}) \sum_{s,m_j} T_i(s, a_i', a_j', s^{l+1}) b_{t,l}(s, m_j)
\]

where $O_i$ and $T_i$ are $i$’s observation and transition functions respectively, in the I-DID denoted by model, $m_{i,t}$. This proves Eq. 4.1 in Proposition 1.

Finally, we move to the term, $Pr(a_j', o_t^{l+1}|a_i', m_{i,t}, m_{j,t-1})$, to obtain Eq. 4.2:

\[
Pr(a_j', o_t^{l+1}|a_i', m_{i,t}, m_{j,t-1}) = Pr(o_{l+1}|a_i', m_{i,t}, m_{j,t-1}) \\
\times Pr(a_j'|OPT(m_{j,t-1})) \quad (j$’s action is conditionally independent of $i$ given its model)
\]

\[
= Pr(a_j'|OPT(m_{j,t-1})) \sum_{s^{l+1}} Pr(o_{l+1}|a_j', a_i', s^{l+1}) \\
\times Pr(s^{l+1}|a_j', a_i', m_{i,t}, m_{j,t-1})
\]
= Pr(a_j^t|OPT(m_{j,l-1}^t)) \sum_{s \in T} O_j(s \mid a_j^t, a_i^t, a_j^{t+1}) \sum_{s,m_j} \\
Pr(s \mid a_j^t, a_i^t, s) \ b_i^t(s, m_j) \ (b_i^t \text{ is } i\text{'s belief in } m_{i,l}^t) \\
= Pr(a_j^t|OPT(m_{j,l-1}^t)) \sum_{s \in T} O_j(s \mid a_j^t, a_i^t, a_j^{t+1}) \sum_{s,m_j} \\
T_i(s, a_i^t, a_j^t, s^{t+1}) \ b_i^t(s, m_j) \ (agent i\text{'s I-DID is used}) \\
where O_j is j\text{'s observation function in model } m_{j,l-1}^t, \text{ which is a part of } i\text{'}s I-DID. \ \blacksquare 

4.3 Correctness

Now that we have a way of computing the distribution over the future paths, we may relate Definition 2 to our previous understanding of BE models and show that both definitions of BE can be interchangeably used:

**Proposition 2 (Correctness).** \textit{Pr}(H_{T-l}|m_{i,l}^t, m_{j,l-1}^t) = \textit{Pr}(H_{T-l}|m_{i,l}^t, \hat{m}_{j,l-1}^t) if and only if OPT(m_{j,l-1}^t) = OPT(\hat{m}_{j,l-1}^t), where m_{j,l-1}^t and \hat{m}_{j,l-1}^t are j\text{'s models.}

**Proof.** The proof is reducible to showing the above for some individual path, h_{T-l} \in H_{T-l}.

(\Rightarrow) From Proposition 1, we observe that the probability over i\text{'s path depends on the policy trees of } i, i and j\text{'s observation functions, } i\text{'s transition function and } i\text{'s belief over the chance node } S \text{ and model node. Given that the probabilities over the path are identical for two different models of } j \text{ (identical frames), let us assume that the solutions of these models are distinct, OPT(m_{j,l-1}^t) \neq OPT(\hat{m}_{j,l-1}^t). Therefore, there exists } a_j^t \text{ for which, } Pr(a_j^t|OPT(m_{j,l-1}^t)) \neq Pr(a_j^t|OPT(\hat{m}_{j,l-1}^t)). Because all other terms in Eqs. 4.1 and 4.2 either pertain to } i \text{ or to } j\text{'s frame, and remain unchanged for both models, we deduce that the two probabilities over the path must be different. However, this is a contradiction of our initial premise, and therefore the two models must have the same solution.}

(\Leftarrow) Given OPT(m_{j,l-1}^t) = OPT(\hat{m}_{j,l-1}^t), we may write,

Pr(a_j^t|OPT(m_{j,l-1}^t)) = Pr(a_j^t|OPT(\hat{m}_{j,l-1}^t)) for all a_j^t. Because all other terms in Eqs. 4.1 and 4.2 are identical, it follows that Pr(h_{T-l}|m_{i,l}^t, m_{j,l-1}^t) must be same as Pr(h_{T-l} \mid m_{i,l}^t, \hat{m}_{j,l-1}^t). \ \blacksquare
A simple method for computing the distribution over the paths given models of $i$ and $j$ is to replace agent $i$’s decision nodes in the I-DID with chance nodes so that $Pr(a_i \in A^i_t) = \frac{1}{|OPT(m^i_{t,j})|}$ and remove the utility nodes, thereby transforming the I-DID into a dynamic Bayesian network (DBN). The desired distribution is then the marginal over the chance nodes that represent $i$’s actions and observations with $j$’s model entered as evidence in the Mod node at $t$. 
Chapter 5

$\epsilon$-Behavioral Equivalence

The new definition of BE described in the previous section has the advantage of being more rigorous, in addition to the merit of permitting us to measure the degree to which models are BE, thereby allowing us to introduce approximate BE.

5.1 Definition

We introduce the notion of $\epsilon$-behavioral equivalence ($\epsilon$-BE) and define it as follows:

**Definition 3 ($\epsilon$-BE).** Given $\epsilon \geq 0$, two models, $m_{j,t-1}^t$ and $\hat{m}_{j,t-1}^t$, are $\epsilon$-BE if the divergence between the distributions $Pr(H_{T-t}|m_{i,t}^{t}, m_{j,t-1}^{t})$ and $Pr(H_{T-t}|m_{i,t}^{t}, \hat{m}_{j,t-1}^{t})$ is no more than $\epsilon$.

Here, the distributions over $i$'s future paths are computed as shown in Proposition 1. While multiple ways to measure the divergence between distributions exist, we utilize the well-known Kullback-Leibler (KL) divergence [7] in its symmetric form, in this paper. KL Divergence is one of the most well known information theoretic measures of divergence of probability distributions. There is a strong precedent of using KL Divergence successfully in agent research to measure distance between distributions. As KL Divergence is not symmetric, we use a symmetric version thereby providing added ease of use. Consequently, the models are $\epsilon$-BE if,

$$D_{KL}(Pr(H_{T-t}|m_{i,t}^{t}, m_{j,t-1}^{t}) || Pr(H_{T-t}|m_{i,t}^{t}, \hat{m}_{j,t-1}^{t})) \leq \epsilon$$

where $D_{KL}(p||p')$ denotes the symmetric KL divergence between distributions, $p$ and $p'$, and is calculated as:

$$D_{KL}(p||p') = \frac{1}{2} \sum_k \left( p(k) \log \frac{p(k)}{p'(k)} + p'(k) \log \frac{p'(k)}{p(k)} \right)$$
If $\epsilon = 0$, $\epsilon$-BE collapses into exact BE. Sets of models exhibiting $\epsilon$-BE for some non-zero but small $\epsilon$ do not differ significantly in how they impact agent $i$’s decision making. As we mention in the next section, these models could be candidates for pruning.

5.2 Approach

We first compute the distributions over the future observation paths of all the initial models in the candidate model space. We then proceed by picking a model of $j$ at random, $m_{j,l-1}^{t=1}$, from the model node in the first time step, which we call the representative. The divergences between the distributions of each of the remaining models is computed with respect to that of the representative. All other models in the model node whose divergence values are less than or equal to $\epsilon$ are classified as $\epsilon$-BE with $m_{j,l-1}^{t=1}$ and are grouped together with it. Of the remaining models, another representative is picked at random and the previous procedure is repeated. The procedure terminates when no more models remain to be grouped. It is seen that this iteration converges very quickly because there are only finite number of behavioral equivalence classes as we had already assumed a finite horizon problem with finite number of actions and observations. We illustrate the process in Fig. 5.1. We point out that for $\epsilon > 0$, in general, more models will likely be grouped together than if we considered exact BE. This will result in a fewer number of classes in the partition and at most as many representatives as there are behaviorally equivalent classes at each time step after pruning.

We first observe that the outcome is indeed a partition of the model set into $\epsilon$-BE classes. This is because we continue to pick representative models and build classes until no model remains ungrouped. There is no overlap between classes since new ones are built only from the models that did not get previously grouped. We observe that the representatives of different classes are $\epsilon$-behaviorally distinct, otherwise they would have been grouped together. However, this set is not unique and the partition could change with different representatives. Furthermore, let $\hat{M}_j$ be the largest set of behaviorally distinct models, also called the minimal set [1]. Then, the following proposition holds:
Proposition 3 (Cardinality). The $\epsilon$-BE approach results in at most $|\hat{M}_j|$ models after pruning.

Intuitively, the Proposition follows from the fact that in the worst case, $\epsilon = 0$, resulting in behaviorally distinct models.

Figure 5.1: Illustration of the iterative $\epsilon$-BE model grouping using the tiger problem. Black vertical lines denote the beliefs contained in different models of agent $j$ included in the initial model node, $M_{j,0}^1$. Decimals on top indicate $i$’s probability distribution over $j$’s models. We begin by picking a representative model (red line) and grouping models that are $\epsilon$-BE with it. Unlike exact BE, models in a different behavioral (shaded) region get grouped as well. Of the remaining models, another is selected as representative. Agent $i$’s distribution over the representative models is obtained by summing the probability mass assigned to the individual models in each class.

5.2.1 Transfer of Probability Mass

From each class in the partition, the previously picked representative is retained and all other models are pruned. The representatives are distinguished in that all models in its group are $\epsilon$-BE with it. Unlike exact BE, $\epsilon$-BE relation is not necessarily transitive. Consequently, we may not select any model from each class as the representative since others may not be $\epsilon$-BE with it.

Recall that agent $i$’s belief assigns some probability mass to each model in the model node. A consequence of pruning some of the models is that the mass assigned to the models would be lost. Disregarding this probability mass may introduce further error in the optimality of the solution. We avoid this error by transferring the probability mass over the pruned models in each class to the $\epsilon$-BE representative that is retained in the model node (see Fig. 5.1). A transfer of probability mass
needs to happen in any approach which prunes some models of agent $j$, so that the mass assigned to those models is not lost. Hence, it would also be done in an exact approach when models that are exactly BE are pruned.

### 5.2.2 Sampling actions and observations

Recall that the predictive distribution over $i$’s future action-observation paths, $Pr(H_{T-t}|h^t, m_{i,l}, m_{j,l-1}^l)$, is conditioned on the history of $i$’s observations, $h^t$, as well. Because the model grouping is performed while solving the I-DID when we do not know the actual history, we obtain a likely $h^t$ by sampling $i$’s actions and observations for subsequent time steps in the I-DID.

Beginning with the first time step, we pick an action, $a^t_i$, at random assuming that each action is equally likely. An observation is then sampled from the distribution given $i$’s sampled action and belief, $o^{t+1}_i \sim Pr(\Omega_i|a^t_i, b^t_{i,l})$, where $b^t_{i,l}$ is the prior belief. We utilize this sampled action and observation pair as the history, $h^t \leftarrow \text{sample}(a^t_i, o^{t+1}_i)$. We may implement this procedure by entering as evidence $i$’s action in the chance node, $A^t_i$, of the DBN (mentioned in Section 4) and sampling from the inferred distribution over the chance node, $O^{t+1}_i$.

Finally, we note that in computing the distribution over the paths, solution to agent $i$’s I-DID is needed as well ($Pr(a^t_i|OPT(m_{i,l}^t))$ term in Eq. 4.1). As we wish to avoid this, we observe that $\epsilon$-BE is based on the comparative impact that $j$’s models have on $i$, which is independent of $i$’s decisions. Therefore, we assume a uniform distribution over $i$’s actions, $Pr(a^t_i|OPT(m_{i,l}^t)) = \frac{1}{|\mathcal{A}_i|}$, which does not change the $\epsilon$-BE of models.

### 5.3 Approximation Algorithm

We present the algorithm for partitioning the models in the model node of the I-DID at each time step according to $\epsilon$-BE, in Fig. 5.2. The procedure, $\epsilon$-BehaviorEquivalence replaces the procedure, BehaviorEq, in the algorithm in Fig. 3.2. The procedure takes as input, the set of $j$’s models, $\mathcal{M}_j$, agent $i$’s DID, $m_i$, current time step and horizon, and the approximation parameter, $\epsilon$. The
algorithm begins by computing the distribution over the future paths of \( i \) for each model of \( j \). If the time step is not the initial one, the prior action-observation history is first sampled. We may compute the distribution by transforming the I-DID into a DBN as mentioned in Section 4 and entering the model of \( j \) as evidence – this implements Eqs. 4.1 and 4.2.

We then pick a representative model at random, and using the cached distributions group together models whose distributions exhibit a divergence less than \( \epsilon \) from the distribution of the representative model. We iterate over the models left ungrouped until none remain. Each iteration results in a new class of models including a representative. In the final selection phase, all models except the representative are pruned from each class in the partition. The set of representative models, which are \( \epsilon \)-behaviorally distinct, are returned.
\textbf{\(\epsilon\)-BEHAVIOR EQUIVALENCE} (Model set \(\mathcal{M}_j\), DID \(m_i\), current time step \(tt\), horizon \(T\), \(\epsilon\)) \textbf{return} \(\mathcal{M}'_j\)
1. Transform DID \(m_i\) into DBN by replacing \(i\)'s decision nodes with chance nodes having uniform distribution
2. \textbf{For} \(t\) \textbf{from} 1 \textbf{to} \(tt\) \textbf{do}
3. Sample, \(a_t^i \sim Pr(A_t^i)\)
4. Enter \(a_t^i\) as evidence into chance node, \(A_t^i\), of DBN
5. Sample, \(o_t^{i+1} \sim Pr(O_t^{i+1})\)
6. \(h^t \leftarrow \langle a_t^i, o_t^{i+1} \rangle\)
7. \textbf{For each} \(m_j^k\) \textbf{in} \(\mathcal{M}_j\) \textbf{do}
8. Compute the distribution, \(P[k] \leftarrow Pr(H_T^{T-t}|h^t, m_i, m_j^k)\), obtained from the DBN by entering \(m_j^k\) as evidence (Proposition 1)

\begin{itemize}
\item \textbf{Clustering Phase}
\begin{enumerate}
\item \textbf{While} \(\mathcal{M}_j\) not empty
\item Select a model, \(m_j^k \in \mathcal{M}_j\), at random as representative
\item Initialize, \(\mathcal{M}'_j \leftarrow \{m_j^k \}\)
\item \textbf{For each} \(m_j^k\) \textbf{in} \(\mathcal{M}_j\) \textbf{do}
\item If \(D_{KL}(P[k]|\|P[k]) \leq \epsilon\)
\item \(\mathcal{M}'_j \leftarrow m_j^k\), \(\mathcal{M}_j \leftarrow m_j^k\)
\end{enumerate}

\item \textbf{Selection Phase}
\begin{enumerate}
\item \textbf{For each} \(\mathcal{M}'_j\) \textbf{do}
\item Retain the representative model, \(\mathcal{M}'_j \leftarrow m_j^k\)
\end{enumerate}
\end{itemize}

\textbf{Return} \(\mathcal{M}'_j\)

Figure 5.2: Algorithm for partitioning \(j\)'s model space using \(\epsilon\)-BE. This function replaces \textbf{BehaviorEq()} in Fig. 3.2.
In order to illustrate the usefulness of I-DIDs, we apply them to two illustrative problems. We describe, in particular, the formulation of the I-DIDs for these examples.

6.1 Multi-agent Tiger Problem

We begin our illustrations of using I-IDs and I-DIDs with a slightly modified version of the multiagent tiger problem [15]. It differs from other multiagent versions of the same problem [16] by assuming that the agents not only hear growls to know about the location of the tiger, but also hear creaks that may tell if the other agent has opened a door. The problem has two agents, each of which can open the right door (OR), the left door (OL) or listen (L). In addition to hearing growls (from the left (GL) or from the right (GR)) when they listen, the agents also hear creaks (from the left (CL), from the right (CR), or no creaks (S)), which noisily indicate the other agents opening one of the doors or listening. When any door is opened, the tiger persists in its original location with a probability of 95%. Agent $i$ hears growls with a reliability of 65% and creaks with a reliability of 95%. Agent $j$, on the other hand, hears growls with a reliability of 95%. Thus, the setting is such that agent $i$ hears agent $j$ opening doors more reliably than the tiger’s growls. This suggests that $i$ could use $j$’s actions as an indication of the location of the tiger. Each agents preferences are as in the single agent game discussed in the original version [20].

Let us consider a particular setting of the tiger problem in which agent $i$ considers two distinct level 0 models of $j$. This is represented in the level 1 I-ID shown in Fig. 6.1. The two IDs could differ, for example, in the probability that $j$ assigns to the tiger being behind the left door as modeled by the node TigerLocation. Given the level 1 I-ID, we may expand it into the I-DID shown in
Figure 6.1: (a) Level 1 I-ID of agent $i$, (b) two level 0 IDs of agent $j$ whose decision nodes are mapped to the chance nodes, $A_1^j$ and $A_2^j$, in (a), indicated by the dotted arrows. The two IDs differ in the distribution over the chance node, TigerLocation.

Fig. 6.2. The model node, $M_{t,0}^j$, contains the different DIDs that are expanded from the level 0 IDs in Fig. 6.1(b). The DIDs may have different probabilities about the tiger location at time step $t$. We get the probability distribution of $j$’s actions in chance node $A_t^j$ by solving the level 0 DIDs of $j$. On performing the optimal action(s) at time step $t$, $j$ may receive observations of the tiger’s growls. This is reflected in new beliefs on the tiger’s position within $j$’s DIDs at time step $t + 1$. Consequently, the model node, $M_{t+1,0}^j$, contains more models of $j$ and $i$’s updated belief on $j$’s possible DIDs.

We showed the nested I-DID unrolled over two time steps for the multiagent tiger problem in Fig. 6.2. Agent $i$ at level 1 considers $M$ models of agent $j$ of level 0 which, for example, differ in the distributions over the chance node TigerLocation. In agent $i$’s I-DID, we assign the marginal distribution over the tiger’s location to the CPD of the chance node TigerLocation$. In the next time step, the CPD of the chance node TigerLocation conditioned on TigerLocation$, A_i^i$,
and $A_j$ is the transition function, shown in Fig. 6.3. We show the CPD of the observation node, $Growl&Creak^{t+1}_i$, in Fig. 6.4. The CPDs of the observation nodes in level 0 DIDs are identical to the observation function in the single agent tiger problem.

The decision node $A_i$ includes possible actions of agent $i$ in the scenario such as listening (L), opening the left door (OL), and opening the right door (OR). The utility node $R_i$ in the level 1 I-DID relies on both agents actions, $A_i$ and $A_j$, and the physical states, $TigerLocation_t^i$. We show the utility table in Fig. 6.5. The utility tables for level 0 models are identical to the reward function in the single agent tiger problem which assigns a reward of 10 if the correct door is opened, a penalty of 100 if the opened door is the one behind which is a tiger, and a penalty of 1 for listening.

6.2 Multi-agent Machine Maintenance Problem

The multiagent machine maintenance problem (MM) [15] is a multi-agent variation of the original machine maintenance problem presented in [17]. In this version, we have two agents that cooperate. The non-determinism of the original problem is increased to make it more realistic, allowing
Figure 6.3: CPD of the chance node \( \text{TigerLocation}_{t+1} \) in the I-DID of Fig. 6.2 when the tiger (a) likely persists in its original location on opening doors, and (b) randomly appears behind any door on opening one.

<table>
<thead>
<tr>
<th>( \langle a_i^t, a_j^t \rangle )</th>
<th>( \text{TigerLocation}_{t+1} )</th>
<th>TL</th>
<th>TR</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \langle OL, * \rangle )</td>
<td>TL</td>
<td>0.95</td>
<td>0.05</td>
</tr>
<tr>
<td>( \langle OR, * \rangle )</td>
<td>TR</td>
<td>0.05</td>
<td>0.95</td>
</tr>
<tr>
<td>( \langle OL, * \rangle )</td>
<td>TL</td>
<td>0.95</td>
<td>0.05</td>
</tr>
<tr>
<td>( \langle OR, * \rangle )</td>
<td>TR</td>
<td>0.05</td>
<td>0.95</td>
</tr>
<tr>
<td>( \langle *, OL \rangle )</td>
<td>TL</td>
<td>0.95</td>
<td>0.05</td>
</tr>
<tr>
<td>( \langle *, OR \rangle )</td>
<td>TR</td>
<td>0.05</td>
<td>0.95</td>
</tr>
<tr>
<td>( \langle L, L \rangle )</td>
<td>TL</td>
<td>1.0</td>
<td>0</td>
</tr>
<tr>
<td>( \langle L, L \rangle )</td>
<td>TR</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>(b)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \langle OL, * \rangle )</td>
<td>*</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>( \langle OR, * \rangle )</td>
<td>*</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>( \langle *, OL \rangle )</td>
<td>*</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>( \langle *, OR \rangle )</td>
<td>*</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>( \langle L, L \rangle )</td>
<td>TL</td>
<td>1.0</td>
<td>0</td>
</tr>
<tr>
<td>( \langle L, L \rangle )</td>
<td>TR</td>
<td>0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Figure 6.4: The CPD of the chance node \( Growl\&Creak_{t+1} \) in the level 1 I-DID.

<table>
<thead>
<tr>
<th>( \langle a_i^t, a_j^t \rangle )</th>
<th>( \text{GrLoc}_{t+1} )</th>
<th>( \langle GL, CL \rangle )</th>
<th>( \langle GL, CR \rangle )</th>
<th>( \langle GL, S \rangle )</th>
<th>( \langle GR, CL \rangle )</th>
<th>( \langle GR, CR \rangle )</th>
<th>( \langle GR, S \rangle )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \langle L, L \rangle )</td>
<td>TL</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.9</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.9</td>
</tr>
<tr>
<td>( \langle L, L \rangle )</td>
<td>TR</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.9</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.9</td>
</tr>
<tr>
<td>( \langle L, OL \rangle )</td>
<td>TL</td>
<td>0.85 * 0.9</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.05</td>
<td>0.15 * 0.9</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.05</td>
</tr>
<tr>
<td>( \langle L, OL \rangle )</td>
<td>TR</td>
<td>0.15 * 0.9</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.05</td>
<td>0.85 * 0.9</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.05</td>
</tr>
<tr>
<td>( \langle L, OR \rangle )</td>
<td>TL</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.9</td>
<td>0.85 * 0.05</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.9</td>
<td>0.15 * 0.05</td>
</tr>
<tr>
<td>( \langle L, OR \rangle )</td>
<td>TR</td>
<td>0.15 * 0.05</td>
<td>0.15 * 0.9</td>
<td>0.15 * 0.05</td>
<td>0.85 * 0.05</td>
<td>0.85 * 0.9</td>
<td>0.85 * 0.05</td>
</tr>
<tr>
<td>( \langle OL, * \rangle )</td>
<td>*</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
</tr>
<tr>
<td>( \langle OR, * \rangle )</td>
<td>*</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
</tr>
</tbody>
</table>

for more interesting policy structures when solved. The original MM problem involved a machine containing two internal components operated by a single agent. Either one or both components of the machine may fail spontaneously after each production cycle. The machine that is under main-
Tenance can have three possible states: 0-fail implying that none of the internal components in that machine failed; 1-fail implying that one of the internal components in that machine failed; and 2-fail implying that two of the internal components in that machine failed. If an internal component has failed, then there is some chance that when operating upon the product, it will cause the product to be defective. An agent may choose to manufacture the product (M) without examining it, examine the product (E), inspect the machine (I), or repair it (R) before the next production cycle. On an examination of the product, the subject may find it to be defective. Of course, if more components have failed, then the probability that the product is defective is greater.

We show the design of a level 1 I-DID for the multiagent MM problem in Fig. 6.6. We consider $M$ models of agent $j$ at level 0 which differ in the probability that $j$ assigns to the chance node $MachineFailure_j$. In the I-DID, the chance node, $MachineFailure_{i+1}$, has incident arcs from the nodes $MachineFailure_i$, $A_i^t$, and $A_j^t$. The CPD of the chance node is shown in Fig. 6.7.

For the observation chance node, $Defective_{i+1}$, we associate the CPD shown in Fig. 6.8. Note that arcs from $MachineFailure_{i+1}$ and the nodes, $A_i^t$, and $A_j^t$, in the previous time step are incident to this node. The observation nodes in the level 0 DIDs have CPDs that are identical to the observation function in the original MM problem.

<table>
<thead>
<tr>
<th>$(a_i^t, a_j^t)$</th>
<th>TL</th>
<th>TR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;OR, OR&gt;$</td>
<td>10</td>
<td>-100</td>
</tr>
<tr>
<td>$&lt;OL, OL&gt;$</td>
<td>-100</td>
<td>10</td>
</tr>
<tr>
<td>$&lt;OR, OL&gt;$</td>
<td>10</td>
<td>-100</td>
</tr>
<tr>
<td>$&lt;OL, OR&gt;$</td>
<td>-100</td>
<td>10</td>
</tr>
<tr>
<td>$&lt;L, L&gt;$</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>$&lt;L, OR&gt;$</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>$&lt;OR, L&gt;$</td>
<td>10</td>
<td>-100</td>
</tr>
<tr>
<td>$&lt;L, OL&gt;$</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>$&lt;OL, L&gt;$</td>
<td>-100</td>
<td>10</td>
</tr>
</tbody>
</table>

Figure 6.5: Reward functions of agents $i$ and $j$ for the multi-agent tiger problem.
Figure 6.6: Level 1 I-DID of agent $i$ for the multiagent MM problem. The hexagonal model node contains $M$ level 0 DIDs of agent $j$. At horizon 1, the models of $j$ are IDs.

<table>
<thead>
<tr>
<th>$(a_i^t, a_j^t)$</th>
<th>Mch Fail$^{t+1}_i$</th>
<th>0-fail</th>
<th>1-fail</th>
<th>2-fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(M/E, M/E)$</td>
<td>0-fail</td>
<td>0.81</td>
<td>0.18</td>
<td>0.01</td>
</tr>
<tr>
<td>$(M/E, M/E)$</td>
<td>1-fail</td>
<td>0.0</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>$(M/E, M/E)$</td>
<td>2-fail</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>$(M/I, R)$</td>
<td>0-fail</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$(M/I, R)$</td>
<td>1-fail</td>
<td>0.95</td>
<td>0.05</td>
<td>0.0</td>
</tr>
<tr>
<td>$(M/I, R)$</td>
<td>2-fail</td>
<td>0.95</td>
<td>0.0</td>
<td>0.05</td>
</tr>
<tr>
<td>$(E/I, R)$</td>
<td>0-fail</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$(E/I, R)$</td>
<td>1-fail</td>
<td>0.95</td>
<td>0.05</td>
<td>0.0</td>
</tr>
<tr>
<td>$(E/I, R)$</td>
<td>2-fail</td>
<td>0.95</td>
<td>0.0</td>
<td>0.05</td>
</tr>
<tr>
<td>$(I/R, ^c)$</td>
<td>0-fail</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>$(I/R, ^c)$</td>
<td>1-fail</td>
<td>0.95</td>
<td>0.05</td>
<td>0.0</td>
</tr>
<tr>
<td>$(I/R, ^c)$</td>
<td>2-fail</td>
<td>0.95</td>
<td>0.0</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Figure 6.7: CPD of the chance node $Machine Failure_{t+1}^i$ in the level 1 I-DID of Fig. 6.6.

The decision node, $A_i$, consists of agent $i$’s actions including manufacture ($M$), examine ($E$), inspect ($I$), and repair ($R$). It has one information arc from the observation node $Defective_i^t$ indicating that $i$ knows the examination results before making the choice. The utility node $R_i$ is asso-
\begin{tabular}{|c|c|c|c|} 
\hline \( \langle a_i^t, a_j^t \rangle \) & Mech fail\(^{t+1}_i \) & Not-defective & Defective \\
\hline (M,M/E) & * & 0.5 & 0.5 \\
(M,I/R) & * & 0.95 & 0.05 \\
(E,M/E) & 0-fail & 0.75 & 0.25 \\
(E,M/E) & 1-fail & 0.5 & 0.5 \\
(E,M/E) & 2-fail & 0.25 & 0.75 \\
(E,I/R) & * & 0.95 & 0.05 \\
(I/R,*) & * & 0.95 & 0.05 \\
\hline
\end{tabular}

Figure 6.8: The CPD of the chance node \textit{Defective}\(^{t+1}_i \) in the level 1 I-DID.

\begin{tabular}{|c|c|c|c|} 
\hline \( \langle a_i^t, a_j^t \rangle \) & 0-fail & 1-fail & 2-fail \\
\hline (M,M) & 1.805 & 0.95 & 0.5 \\
(M,E) & 1.555 & 0.7 & 0.25 \\
(M,I) & 0.4025 & -1.025 & -2.25 \\
(M,R) & -1.0975 & -1.525 & -1.75 \\
(E,M) & 1.5555 & 0.7 & 0.25 \\
(E,E) & 1.305 & 0.45 & 0.0 \\
(E,I) & 0.1525 & -1.275 & -2.5 \\
(E,R) & -1.3475 & -1.775 & -2.0 \\
(I,M) & 0.4025 & -1.025 & -2.25 \\
(I,E) & 0.1525 & -1.275 & -2.5 \\
(I,I) & -1.0 & -3.00 & -5.00 \\
(I,R) & -2.5 & -3.5 & -4.5 \\
(R,M) & -1.0975 & -1.525 & -1.75 \\
(R,E) & -1.3475 & -1.775 & -2.0 \\
(R,I) & -2.5 & -3.5 & -4.5 \\
(R,R) & -4 & -4 & -4 \\
\hline
\end{tabular}

Figure 6.9: Reward function of agent \( i \). For the level 0 agent \( j \), the reward function is identical to the one in the classical MM problem.

Associated with the utility table in Fig. 6.9. The CPD of the chance node, \textit{Mod}\( [M_j^{t+1}] \), in the model node, \( M_j^{t+1} \), reflects which prior model, action and observation of \( j \) results in a model contained in the model node.
We implemented the algorithms in Figs. 3.2 and 5.2 utilizing the HUGIN Java API for DIDs. HUGIN is a software used for delivering advanced solutions for decision making under uncertainty. The Hugin Decision Engine (HDE) contains all functionality related to handling and using knowledge bases in a programming environment [49]. We show results for the well-known problems in the literature: the two-agent tiger problem \(|S| = 2, |A_i| = |A_j| = 3, |\Omega_i| = 6, |\Omega_j| = 3|\) [5] and the multiagent version of the machine maintenance (MM) problem \(|S| = 3, |A_i| = |A_j| = 4, |\Omega_i| = 2, |\Omega_j| = 2|\) [11] described in the previous chapter. These problems are popular but relatively small, having a physical state space size of 2 and 3 respectively. But keep in mind that in an interactive state space, we must consider all possible models of other agents, thus making the interactive state space (IS) considerably larger. We formulate level 1 I-DIDs of increasing time horizons for the problems, and solve it approximately for varying \(\epsilon\). We show that, (i) the quality of the solution generated using our approach (\(\epsilon\)-BE) improves as we reduce \(\epsilon\) for given numbers of initial models of the other agent, \(M_0\), and approaches that of the exact solution. This is indicative of the flexibility of the approach; (ii) in comparison to the approach of updating models discriminatively (DMU) [1], which is the current efficient technique, \(\epsilon\)-BE is able to obtain larger rewards for an identical number of initial models. This indicates a more informed clustering and pruning using \(\epsilon\)-BE in comparison to DMU, although it is less efficient in doing so.

### 7.1 Multi-agent Tiger Problem

In Fig. 7.1(a, b), we show the average rewards gathered by executing the policies obtained from solving level 1 I-DIDs approximately within a simulation of the problem domain. Each data point
Figure 7.1: Performance profile obtained by solving a level 1 I-DID for the multiagent tiger problem using the ε-BE approach for (a) 3 horizons and (b) 4 horizons. As ε reduces, quality of the solution improves and approaches that of the exact. (c) Comparison of ε-BE and DMU in terms of the rewards obtained given identical numbers of models in the initial model node after clustering and pruning.

is the average of 300 runs where the true model of j is picked randomly according to i’s belief. Notice that as we reduce ε the policies tend to converge to the exact (denoted by flat lines) and this remains true for different numbers of initial models and across horizons. Values of these policies increase as i considers greater numbers of models thereby improving it’s chances of modeling j correctly.

Next, we compare the performance of this approach with that of DMU. While both approaches cluster and prune models, DMU does so only in the initial model node, thereafter updating only those models which on update will be behaviorally distinct. Thus, we compare the average rewards obtained by the two approaches when an identical number of models remain in the initial model node after clustering and selection. This is done by varying ε in both approaches until the desired number of models are retained. This allows us to compare between the clustering and selection techniques of the two approaches. From Fig. 7.2, we observe that ε-BE results in better quality policies that obtain significantly higher average reward. This indicates that the models pruned by DMU were more valuable than those pruned by ε-BE, thereby testifying to the more informed way in which we compare between models by directly gauging the impact on i’s history. DMU’s
Figure 7.2: Comparison of $\epsilon$-BE and DMU for the multi-agent tiger problem in terms of the rewards obtained given identical numbers of models in the initial model node after clustering and pruning.

approach of measuring simply the closeness of beliefs in models for clustering results in significant models being pruned. However, the trade-off is the increased computational cost in calculating the distributions over future paths. To illustrate, $\epsilon$-BE consumed an average of 9.1 secs in solving a 4 horizon I-DID with 25 initial models and differing $\epsilon$, which represents approximately a three-fold increase compared to DMU.

7.2 MULTI-AGENT MACHINE MAINTENANCE PROBLEM

We show a similar set of results for the MM problem in Fig. 7.3. The MM problem differs in having one more physical state and action in comparison to the tiger problem, and less observations. We observe a similar convergence toward the performance of the exact solution as we gradually reduce $\epsilon$. This affirms the flexibility in selecting $\epsilon$ provided by the approach.

Furthermore, in Fig. 7.4, we again note the significant increase in average reward exhibited by $\epsilon$-BE compared to DMU given an identical number of retained models. This clearly illustrates the improvement in clustering models that are truly approximately similar, in comparison to using heuristics such as closeness of beliefs. Thus, models retained by $\epsilon$-BE are significantly more valuable than those retained by DMU translating into greater reward, albeit at the cost of efficiency.
Figure 7.3: Performance profile for the multiagent MM problem obtained by solving level 1 I-DIDs approximately using $\epsilon$-BE for (a) 3 horizon and (b) 4 horizon. Reducing $\epsilon$ results in better quality solutions.

Figure 7.4: Significant increase in rewards obtained for $\epsilon$-BE compared to DMU, given identical numbers of retained models in the initial model node.

The approach exhibited a five-fold increase in time taken compared to DMU in order to solve a horizon 4 I-DID with 25 initial models.

In summary, experiments on two multiagent problem domains indicate that the $\epsilon$-BE approach models behavioral similarity between models of the other agent more accurately resulting in favorable performance in terms of quality of the solutions, but at the expense of computational efficiency. As a part of the evaluation, we also theoretically analyze the performance of our approx-
imation approach with that of the model clustering approach (described previously in *Chapter 3*) in the next chapter.
Our main motivation toward the proposed approximation technique was to mitigate the curse of history and dimensionality by considerably reducing the size of the state space and at the same time preserving the quality of the solution. We were quite successful in that endeavor. In this chapter, however, we will focus on specifying how exactly we achieved high computational savings and also bounding the error due to the approximation. We will also theoretically analyze our savings with respect to exact BE algorithm and the Model Clustering approach.

8.1 Computational Savings

The computational complexity of solving I-DIDs is primarily due to the large number of models that must be solved over $T$ time steps. At time step $t$, there could be $|\mathcal{M}_j^0|(|A_j||\Omega_j|)^t$ many models of the other agent $j$, where $|\mathcal{M}_j^0|$ is the number of models considered initially. The nested modeling further contributes to the complexity since solution of each model at level $l-1$ requires solving the lower level $l-2$ models, and so on recursively up to level 0. In an $N+1$ agent setting, if the number of models considered at each level for an agent is bound by $|\mathcal{M}|$, then solving an I-DID at level $l$ requires the solutions of $O(\langle N|\mathcal{M}|^l)$ many models. If the models are intentional, exact solutions of the models are at least NP Complete. This complexity precludes practical implementations of I-DIDs beyond simple problems. As we mentioned in Proposition 3, the $\epsilon$-BE approximation reduces the number of agent models at each level to at most the size of the minimal set, $|\hat{\mathcal{M}}^t|$. In doing so, it solves $|\mathcal{M}_j^0|$ many models initially and incurs the complexity of performing inference in a DBN for computing the distributions. This complexity while significant is less than that of solving DIDs. Consequently, we need to solve at most $O(\langle N|\hat{\mathcal{M}}^*|^l)$ number of models at each
non-initial time step, typically less, where $\hat{\mathcal{M}}^*$ is the largest of the minimal sets, in comparison to $\mathcal{O}(\lceil N|\mathcal{M}| \rceil^l)$. Here $\mathcal{M}$ grows exponentially over time. In general, $|\hat{\mathcal{M}}| \ll |\mathcal{M}|$, resulting in a substantial reduction in the computation. Additionally, a reduction in the number of models in the model node also reduces the size of the state space, which makes solving the upper-level I-DID more efficient.

We will now compare our approach with that of the Model clustering (MC) approach.

1. In the MC approach, always, a constant number ($K$) of models is solved at every time step whereas in our $\epsilon$-BE approach, all initial models are solved in order to compute the distribution over the future action-observation paths. From the next step onwards, only a maximum of as many models as there are behavioral equivalence classes have to be solved.

2. In MC, in order to partition BE regions, it is required to find the sensitivity points which involves complex computations that are almost as hard as even solving the problem itself whereas the process of partitioning BE regions in our approach is easy. All we do is pick a model randomly and simultaneously cluster all $\epsilon$-BE models with it. Hence, when another model is picked randomly from those that remain after the grouping, it is assured that it is $\epsilon$-behaviorally equivalent. However, computing the distributions of all the candidate models which is required for the clustering process, is time consuming.

3. In MC, the $K$ means clustering process is known to take a long time to converge whereas in $\epsilon$-BE the clustering methodology is extremely simple and the convergence is quick due to the presence of only finite number of BE classes.

4. In MC, when $K$ models are selected we may end up having more than one model from the same behaviorally equivalent region. This results in redundancy (because two BE models are effectively the same model as they prescribe the same behavior to the subject agent) and unnecessary computations. Instead, if these redundant models were from a different BE region, the solution quality could have improved. However, in the $\epsilon$-BE approach, redundancy never occurs.
It can be shown theoretically that the $\epsilon$-behavioral equivalence approach always performs better or equal to, but never worse, than the Model Clustering approach in terms of the number of candidate models ascribed to the other agents. This claim followed from the analysis that we conducted shown below.

For the purpose of this analysis, let us consider $R$ to be the number of behaviorally equivalent classes at any particular time step $t$ and $K$ to be the number of models picked in the Model Clustering approach. We present results for three exhaustive cases as follows:

1. $R < K$: In this case, the $\epsilon$-BE approach ends up solving at most $R$ models. Hence even the worst case in this approach is better in terms of the number of candidate models solved with respect to the model clustering approach. In terms of quality, in the worst case of $\epsilon = 0$, the $\epsilon$-BE approach can guarantee that no redundancy in the models picked but the MC approach does not. This way better quality is more probable with the former approach.

2. $R = K$: In this case, the MC approach and the worst case of the BE approach (when $\epsilon = 0$), end up solving the same number of models. Even in terms of quality, the worst case of the BE approach guarantees at least one representative per behaviorally equivalent region thus producing an exact solution but the MC approach does not as there may be redundancy.

3. $R > K$: In this case the $\epsilon$-BE approach ends up solving greater number of models. But quality-wise, the $\epsilon$-BE approach performs better than the MC approach because a greater number of $\epsilon$-behaviorally distinct models are solved in the former and a there exists atleast $R-K$ regions without a representative model in the latter.

8.2 ERROR BOUND

In the $\epsilon$-BE approach, it is necessary to bound the error that arises, due to the approximation. We assume that the lower-level models of the other agent are solved exactly and also assume that we limit the pruning of $\epsilon$-BE models to the initial model node. The error is due to transferring the probability mass of the pruned model to the representative, effectively replacing the pruned
model with the representative. In other words, there is an error when $\epsilon$ is such that representative models are picked such that, all the models from some behaviorally equivalent regions get clustered with representative models from other regions thereby leaving some regions without representative models.

For example, say there are $R$ behaviorally equivalent regions; and $k$ representative models remain after the clustering process converges at a particular time step from $M$ candidate models of agent $j$ that are initially considered. Note that the value of $k$ is dynamic; it keeps changing every time step. We can bound the error for excluding all but $k$ models. This presents us with two situations where errors can occur:

1. When $k = R$: In this case, there is a model representing each and every $\epsilon$-behaviorally equivalent region $R$. Hence, in the trivial case where $\epsilon = 0$, there is no optimality error. However, when $\epsilon > 0$, an error may exist because there could be more than 0 regions without a representative model.

2. When $k < R$: In this trivial case where $\epsilon = 0$, there will be $R-k$ regions without a representative model and in the case where $\epsilon > 0$, an error arises because there may be more than or equal to $R-k$ regions without a representative model.

Note that our approach can never result in a situation where $k > R$ (see Proposition 3).

Our definition of BE provides us with a unique opportunity to bound the error for $i$. We observe that the expected value of the I-DID could be obtained as the expected reward of following each path weighted by the probability of that path. Let $\rho_{b_i,l}(H_T)$ be the vector of expected rewards for agent $i$ given it’s belief when each path in $H_T$ is followed. Here, $T$ is the horizon of the I-DID. The expected value for $i$ is:

$$EV_i = Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i,l}(H_T)$$

where $m_{j,l-1}$ is the model of $j$. If the above model of $j$ is pruned in the Mod node, let model $\hat{m}_{j,l-1}$ be the representative that replaces it. Then $\hat{b}_{i,j}$ is $i$’s belief in which model $m_{j,l-1}$ is replaced with the representative. Expected value for $i$, $EV_i$, is:
\[ \hat{E}V_i = Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

Then, the effective error bound is:

\[ \Delta = ||\hat{EV}_i - EV_i||_{\infty} \]

\[ = ||Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ - Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T)||_{\infty} \]

\[ = ||Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ - Pr(H_T|m_{i,l}, \hat{m}_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ + Pr(H_T|m_{i,l}, \hat{m}_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ - Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T)||_{\infty} \text{ (add zero)} \]

\[ \leq ||Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ - Pr(H_T|m_{i,l}, \hat{m}_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ + Pr(H_T|m_{i,l}, \hat{m}_{j,l-1}) \cdot \rho_{b_i}(H_T) \]

\[ - Pr(H_T|m_{i,l}, m_{j,l-1}) \cdot \rho_{b_i}(H_T)||_{\infty} \text{ (|\rho_{b_i,\hat{l}}| \leq |\rho_{b_i,l}|)} \]

\[ \leq \rho_{b_i}(H_T) - \rho_{b_i}(H_T)||_{\infty} \cdot ||Pr(H_T|m_{i,l}, m_{j,l-1}) \]

\[ - Pr(H_T|m_{i,l}, \hat{m}_{j,l-1})||_{1} \text{ (Hölder’s inequality)} \]

\[ \leq (R_{i}^{max} - R_{i}^{min})T \times 2\epsilon \text{ (Pinsker’s inequality)} \]

Matters become more complex when we additionally prune models in the subsequent model nodes as well. This is because rather than comparing over distributions given each history of \( i \), we sample \( i \)'s action-observation history. Consequently, additional error incurs due to the sampling, which is difficult to bound. Finally, Doshi and Zeng [1] show that it is difficult to usefully bound the error if lower-level models are themselves solved approximately. This limitation is significant because approximately solving lower level models could bring considerable computational savings.

In summary, error in \( i \)'s behavior due to pruning \( \epsilon \)-BE models in the initial model node may be bounded, but we continue to investigate how to usefully bound the error due to multiple additional approximations.
Chapter 9

Conclusion

Interactive dynamic influence diagrams (I-DIDs) provide a graphical formalism for modeling the sequential decision making of an agent in an uncertain multiagent setting. In this thesis, we present a new approximation method, called $\epsilon$ Behavioral Equivalence ($\epsilon$-BE), to solve interactive dynamic influence diagrams (I-DIDs). This is an approximation technique that allows an agent to plan sequentially in multi-agent scenarios, which could be cooperative, competitive or even neutral. The main motivation behind the development of this method was that the curses of dimensionality and history of I-DIDs, limited the existing algorithms from scaling to larger multi-agent problem domains. This curse was due to the exponentially growing space of candidate models ascribed to other agents over time. Hence, our goal was to come up with an approximation technique that could mitigate this curse better than those that already existed.

Existing approximation techniques used clustering and pruning of behaviorally equivalent models as the way to identify the minimal model set. Our approximation technique further reduces the complexity by additionally pruning models that are approximately behaviorally equivalent. Toward this objective, we redefined behavioral equivalence in terms of the distribution over the subject agent’s future action-observation paths that allowed a way to measure the degree to which the models are behaviorally equivalent which thereby helped formulate our approximation technique, ($\epsilon$-BE). Redefining BE by explicitly focusing on the impact that the other agents models have on the subject agent in the interaction allowed us to better identify behavioral similarity. This translated into solutions of better quality given a limit on the number of models that could be held in memory. Consequently, other approaches needed more models to achieve comparable quality, which translated into better efficiencies for our approach.

71
We showed the implementation of our approach for two test problems; the multi-agent tiger problem, and the multi-agent machine maintenance problem and compared the results of our approach with the existing best technique for solving I-DIDs (the DMU method) and also the exact BE method. Highlights of the results obtained are presented below:

1. The quality of the solution generated using our approach improves as we reduce $\epsilon$ for given numbers of initial models of the other agent, and approaches that of the exact solution. This is indicative of the flexibility of the approach.

2. In comparison to the approach of updating models discriminatively (DMU), which is the current efficient technique, $\epsilon$-BE is able to obtain larger rewards for an identical number of initial models. This indicates a more informed clustering and pruning using $\epsilon$-BE in comparison to DMU, although it is less efficient in doing so. The trade off was the increased computational cost in calculating the distributions over future paths. $\epsilon$-BE consumed three times the average time consumed by DMU in solving a 4 horizon I-DID with 25 initial models and differing $\epsilon$ for the multi-agent tiger problem and a four fold increase in the time consumed with the same setting for the multi-agent machine maintenance problem.

We also theoretically analyzed the savings of our approach with that of the model clustering approach. We claim that our $\epsilon$-BE method performs equally well or better but never worse than the model clustering approach in terms of the number of candidate models ascribed to the other agents over time.

9.0.1 Limitations and Future Work

Scalability to higher horizons using our approximation technique is limited mainly by the curse of history due to the exponential increase in the size of the distribution over the future paths over increasing number of horizons. We are investigating ways to mitigate the impact of this curse. We are also investigating ways of reducing the computational cost, for example, by directly computing the distributions instead of using the DBN and preemptively discriminating between model
updates. The new definition showed potential when bounding the final error when replacing some candidate models with an approximate representative. However, this error bound only applies when lower level models are solved exactly. This is a severe problem as it is the lower levels which offer the greatest potential for savings. We are also currently working on ways to usefully bound the error when these lower level models are solved approximately. We are optimistic that this can be done in a systematic way, and this will facilitate application to larger multiagent problem domains.


74


