Anytime Point Based Approximations for Interactive POMDPs

by

Dennis D. Perez Barrenechea

(Under the direction of Prashant Doshi)

Abstract

Partially observable Markov decision processes (POMDPs) have been largely accepted as a rich-framework for planning and control problems. In settings where multiple agents interact, POMDPs fail to model other agents explicitly. The interactive partially observable Markov decision process (I-POMDP) is a new paradigm that extends POMDPs to multiagent settings. The I-POMDP framework models other agents explicitly, making exact solution unfeasible but for the simplest settings. Thus, a need for good approximation methods arises, methods that could find solutions with tight error bounds and short periods of time. We develop a point based method for solving finitely nested I-POMDPs approximately. The method maintains a set of belief points and form value functions including only the value vectors that are optimal at these belief points. Since I-POMDPs computation depends on the prediction of the actions of other agents in multiagent settings, an interactive generalization of the point based value iteration (PBVI) methods that recursively solves all models of other agents needed to be developed. We present some empirical results in domains on the literature and discuss the computational savings of the proposed method.

Index words: Markov Decision Process, Multiagent systems, Decision making, POMDP

Anytime Point Based Approximations for Interactive POMDPs

by

Dennis D. Perez Barrenechea

B.S., Universidad Peruana de Ciencias Aplicadas, 1999

M.S., Virginia Polytechnic Institute and State University - Virginia Tech, 2004

A Thesis Submitted to the Graduate Faculty

of The University of Georgia in Partial Fulfillment

of the

Requirements for the Degree

Master of Science

Athens, Georgia

2007

Anytime Point Based Approximations for Interactive POMDPs

by

Dennis D. Perez Barrenechea

Approved:

Major Professor: Prashant Doshi

Committee: Walter D. Potter
Khaled Rasheed

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
Dec 2007

Dedication

To Hugo and Nola, my unconditional loving parents, and Milagros, my soulmate.

Acknowledgments

First of all, I want to thank my advisor, Dr. Prashant Doshi, for envisioning this thesis and helping me through the many cumbersome parts of its development. Thank you also to Dr. Walter Potter and Dr. Khaled Rasheed for taking the time to read this material and being part of my committee. Finally, thank you to Dr. Covington for my work in the CASPR project during part of my time at the AI Center.

During my two years in Georgia, many things got complicated outside of the academics. There are a few people I want to thank for their help, support and word of advice, which definitely made a difference in my life during this time. To my dear friend Katia, who has a heart the size of Texas and was always there for me with a helping hand. To my dear friend Marilu, who visited me from Lima and was always there to listen. To my pal Dan, from Virginia Tech, who also visited me. And my dearest friends Dennis and Cathy Duncan, who always provided their friendship and help and never let me spend a holiday alone.

This work has been funded in part by the Perez foundation: thanks Mom and Dad for your support and patience through the hard times, you are always on my team even when I take on projects that may not make a lot of sense. The truth is that you always take an important role in every dream of mine that becomes a reality. A special thank you note is due to my cousins Carlos and Marilena, my aunt Oti, and my dearest niece and very best friend Mayra, for letting me stay with them and helping me out a lot in the last and more complicated months of this work.

Finally, to my beloved Milagros, for your infinite love and support. You really had a hard time keeping me focused and convincing me of not quitting when things weren't working out. When I am with you, I can achieve anything. This is only a little example.

Table of Contents

List of Figures

Chapter 1

Introduction

The last few years have seen an increasing interest in planning algorithms under uncertainty. This growing enthusiasm has been triggered by a significant amount of applications that have been successfully developed in different fields of engineering and computer science, applications where optimization of the course of action is needed to achieve some level of autonomy, and control procedures had to be developed to work under unreliable sensor data.

In this context, many methods have arisen to tackle the several challenges researchers had to face. Among planning problems, the sequential planning problem under uncertainty on environments where other agents, either cooperative or competitive, exists, is one of the most complex and less developed ones. Among these efforts, *Interactive partially observable Markov decision processes (I-POMDPs)* emerge as mathematical paradigms that allow for planning on multiagent settings with very few restrictions, as opposed to many other approaches that restrict their problem domains to reduce complexity (i.e. DEC-POMDPs focus only on cooperative multiagent environments where all the agents maintain common initial beliefs, see [14, 36, 31]). As may be expected, the benefits of such a framework come with the cost of computationally prohibitive solutions. It is then important to have good approximation methods under I-POMDPs to make a wider range of applications feasible.

The purpose of this thesis is to propose a new approximation method to solve I-POMDPs. Up to the time of publication of this document, there are very few methods that can be used to solve I-POMDPs, and the current computer power (i.e. memory, CPU) restrict their use to very few horizons (a very short span of time ahead in the planning sequence). The

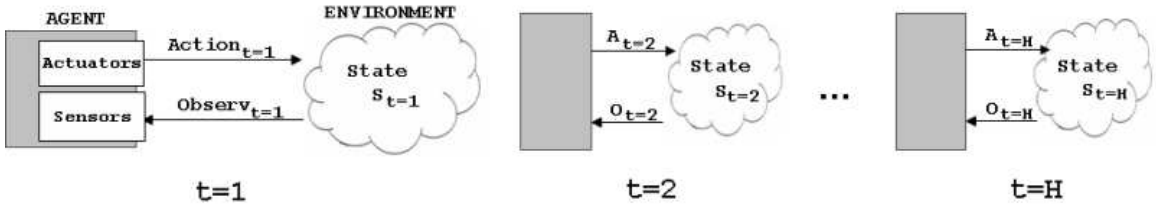


Figure 1.1: An agent interacts with the environment through sensors and actuators.

proposed method will tackle these difficulties and will provide a new approach that may extend solutions over the current limited options.

1.1 Agents and Planning Agents

An agent is something that acts; anything that can be understood as perceiving his environment through sensors and affecting it through actuators. Fig. 1.1 depicts a computer planning agent, an automated system that can be seen as taking inputs from the enviroment (sensor measurements) and producing output (actuator actions) to achieve a desired goal. The main challenge in the planning agent research area is the development of methods to obtain control policies that based on the inputs or *observations*, produce a set of outputs or *actions* that lead to better rewards (optimal behavior) in a reasonable amount of time.

Agents that are able to develop a sequence or plan of actions while pursuing a goal form the class of *rational planning agents*. There exists a sub-class of agents called *classical planners*, or agents that perform over *classical planning* environments. These environments are characterized by their full observability, and are usually deterministic, finite, static and discrete in time, actions, objects and effects. These environments are less common in nature than those that allow for uncertainty in observations (i.e. the agent is not able to fully observe the state of the world). Thus, we may not expect to see many of these environments in real world applications.

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 $\langle S, A, T, R \rangle$, 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; 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. The mathematical notation, properties and details of this framework that are relevant to this research are further explained in Chapter 2. For the purposes of this chapter, 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.

1.2 Decision-Theoretic Planning Under Uncertainty

In the last section we stated that problems where no uncertainty arises are hard to find in nature. Thus, for a 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 is a non-trivial problem.

The reader must think that a logical approach would be to extend MDPs to account for uncertain events, using probabilities. *Partially observable Markov decision processes* (POMDPs) do exactly that. POMDPs extend MDPs to partially observable environments. They provide a principled framework for sequential planning in single agent settings under uncertainty, and have been implemented with success in many applications in the last few years, applications such as assistive technologies ([2, 22]), mobile robotics ([5, 32]), preference elicitation ([3]), spoken dialog systems ([29, 37]), gesture recognition ([8]), among others.

There is another set of problems where multiple agents interact in the environment. These other agents may cooperate with us to achieve a goal, or may have antagonistic preferences,

competing with us. In these contexts, POMDPs fail to *explicitly* model the effect that these entities have on the world.

The actions other agents perform while interacting with the environment may drastically affect the performance of our agent, and an agent in the POMDP framework would hardly be able to recognize when other agents' actions affect the state of the world. Since POMDPs do not explicitly model other agents and their possible actions, the only way available in this framework to deal with multiagent settings is to model the effects of these agents in the world as noise. A new framework was developed by Gmytrasiewicz & Doshi [13], called Interactive POMDPs (I-POMDPs) in an effort to model other agents explicitly, in order to improve the performance of an agent in multiagent settings. This framework includes models of the other agents and takes into account their possible beliefs and associated preferences and capabilities. The effect of these improvements is added complexity, which make exact solutions difficult to calculate but only for the simplest problems. But I-POMDPs, analogous to POMDPs, exhibit computational difficulties by growing dimensionalities of the state space (also called course of dimensionality) and by the exponential growth of the policy space with the number of actions and observations (curse of history). An effort to find good approximation methods is necessary. It has been shown that I-POMDPs perform as good as, and in most cases better than, POMDPs [1] in multiagent settings. Therefore, a good approximation method to I-POMDPs would be greatly appreciated by the research community and may help leverage this framework to a point where implementations of I-POMDPs in real world environments are feasible.

1.3 Claims and Contributions

The previous section provides the reader with the basic concepts underlying the study of multiagent decision-theoretic planning. More in depth development of these ideas will be presented in the following chapters. The information provided so far lays the building blocks

[1] These POMDPs model other agents in the multiagent environment as noise.

for presenting the primary claims and contributions that this work will provide towards the advancing of this subject and the contribution to the literature of the Artificial Intelligence field.

- This thesis focuses on the development of an offline approximation method for finitely nested I-POMDPs, which do not assume any particular initial belief of the agent or other agents.
- The proposed solution is based on the point based solution techniques, a class of methods for solving POMDPs that reduce the impact of the curse of history and therefore scale well to large problems.
- A generalization of the PBVI methods to I-POMDPs poses many challenges. Point based techniques use an initial set of belief points, which require a representation. The state space is prohibitively large due to the possibility of an infinitely large number of other agents' models. And, since actions of an agent depend on action of other agents as well, solution of other agents' models are required.
- In order to select initial belief points, we provide a new computational representation of the nested beliefs using mixtures of Gaussians. Mixtures of Gaussians are known to approximate to an arbitrary accuracy any probability distribution.
- We show how computational representations of multiply nested beliefs are non-trivial and how restrictive assumptions have to be made to facilitate their representation.
- We limit the interactive state space by including a finite set of initial models of other agents and those models that are reachable from the initial set over time.
- We provide a generalized *point based value iteration (PBVI)* [26] method for finitely nested I-POMDPs that approximately solve the models of other agents at each level by recursion.

- We provide evaluations of the performance of the approach on multiple problem domains. We analyze the results and discuss opportunities for future research.

1.4 Structure of this work

We start outlining what the reader may expect to find in each chapter of this thesis. The first three chapters focus on presenting some background information to the reader so that the building blocks for the understanding of the key issues involving research in Markov decision processes are laid. *Chapter 1* focus is to open the issue to the reader by giving a very broad idea of the context of the research area and introduce a few general concepts.

In *Chapter 2*, we introduce the reader to the Markov decision processes(MDPs), the first building block towards the multiagent planning problem we engage to solve. This framework presents some of the basic ideas that will be used in the explanation of partially observable Markov decision processes (POMDPs), which are described next in the chapter. POMDPs extend MDPs by taking into account uncertainty in the world, characteristic of the partial observability problems they solve (in POMDPs, we are never sure what the state of the world is). This second framework extends planning agent capabilities to plan on more realistic domain problems, but the added functionality doesn't come free of charge: the computation time to solve large state-space problems is very restrictive. Thus, approximation methods are very important for applications to real problems. Among the different approximation methods in the literature, the focus of our attention is a special set of techniques, called *Point Based Value Iteration (PBVI)*, a modification of the dynamic programming value iteration, to select solutions that maximize rewards at specific belief points. We survey different implementations of PBVI and review their pros and cons, keeping in mind which of these may be applicable in our proposed method.

Up to this point, all the discussion and the presentation of methods is focused on single agent problems. In *Chapter 3*, we take the discussion to the multiagent level, by introducing the *Interactive partially observable Markov decision process* framework. We give the details

of the framework, and revise some concepts of belief representation that are of use to our research. Due to the computational challenge that represents solving I-POMDPs, we present the *Interactive particle filter*, an online approximation method that has shown the best results on I-POMDPs until now.

Chapter 4 introduces the method proposed by this research: *Interactive point based value iteration (I-PBVI)*. I-PBVI is a generalization of PBVI methods to the I-POMDP framework, and its development entails many complexities beyond the issues raised in the POMDP version (belief expansion algorithm, number of initial beliefs). The chapter solves these issues, as we present a new representation of nested beliefs in I-POMDPs using mixtures of Gaussians to solve initial belief representation issues, lay down the theory behind the bounding of interactive states to allow for computability of solutions and present a detailed explanation of the algorithms for I-PBVI and belief expansion.

Chapter 5 presents an empirical evaluation of the proposed method. We take 3 problems from the literature (i.e. the multiagent tiger problem, the multiagent machine maintenance problem and the UAV reconnaissance problem) and perform simulations to measure the time needed to achieve different levels of performance. We compare our results with the available approximation method for I-POMDPs that has shown the best results, the *Interactive particle filter*(IPF).

Finally, in *Chapter 6* we gather the conclusions of our work and provide some ideas to further develop approximation methods for I-POMDPs.

Chapter 2

Partially Observable Markov Decision Processes

Markov Decision Processes(MDPs) are planning frameworks for single agent settings where the state of the problem is always fully observable to the agent. This property is hard to be exhibited in real world domains, not only because the observability of the world is partial to the agent, but also because real world sensor are noisy. An extension of this framework to work under uncertain observability of the state are called the partially observable Markov decision processes (POMDPs). POMDPs combine maximization of expected rewards with a cognitive process called belief update.

Since both MDPs and POMDPs form an important foundation on which our work is based, we present them in some detail on this chapter. In Section 2.1, we formally define the Markov decision processes and present some properties. Section 2.2 presents the partially observable Markov decision processes with some detail, specially focusing on the value iteration process, a dynamic programming method for solving them. Section 2.3 introduce the point based value iteration (PBVI), an approximation method that drastically improves response time while solving policies for POMDPs and that will be the base of our work. Finally, section 2.4 presents some results in the laboratory when implementing and experimenting with POMDPs and PBVI.

2.1 Markov Decision Processes

The *Markov decision process (MDP)* [28] is a well-known planning paradigm used to optimize an agent behavior in fully observable environments. The solution to this framework is

composed by a policy π, which is a specification for the agent of what action to take given a specific state of the world. An MDP$\langle S, A, T, R\rangle$ is defined by

- A set of world states, S.
- A set of possible actions, A.
- A transition model, $p(s'|s, a)$. This is the probability that the world changes from state s to state s' with the execution of action a.
- A reward function, $R(s, a)$ [1].

On a given point in time, the world state is $s \in S$. The agent executes action $a \in A$, receives a reward $r = R(s, a)$ for its action and the world state changes to $s' \in S$. This behavior is captured by a value function, which is the expected reward of the agent for a given policy over time, V^π. Therefore,

$$V^\pi(s_0) = E[\sum_{t=0}^{n} \gamma^t r_{\pi(s_t)}(s_t)]. \tag{2.1}$$

where $\gamma \in [0, 1\rangle$ is called the discount factor, which weights exponentially the relevance of the rewards on time t (t steps ahead). Here, $r_{\pi(s_t)}(s_t)$ is the reward of the application of policy π at time t.

The objective of MDP planning is to find the *optimal policy*, π^*, that maximizes the expected reward of the agent for each state of the world. For this purpose, the most widely known algorithm is *value iteration*, a dynamic programming method, based on the backup of the *Bellman equation*:

$$V_t(s) \leftarrow R(s) + \gamma max_a \sum_{s'} T(s, a, s')V_{t+1}(s').$$

[1] For problems where the reward depends on the resulting state, the reward function is defined as $R(s, a, s')$.

Value iteration eventually converges to a unique set of solutions to the Bellman equations. Other methods used to find optimal policies are *policy iteration*, which uses a set of linear equations that needs to be solved, and through the application of *linear programming*. See [30] and [19] for more information on those methods, respectively.

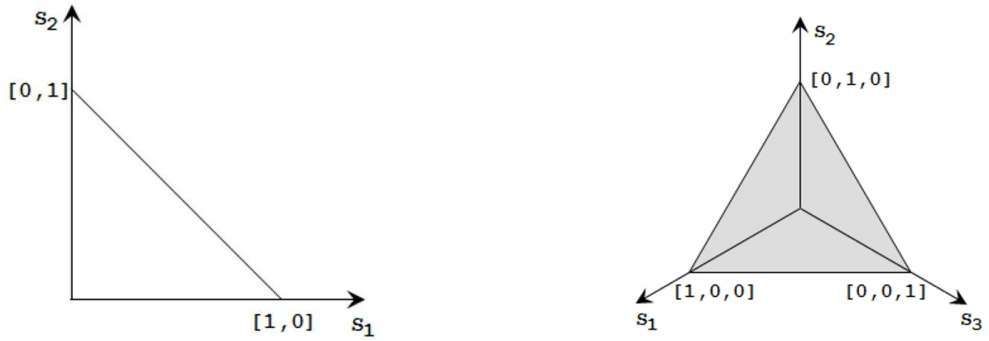
There are different optimality criteria that the agent may use to optimize its rewards:

- A *Finite horizon* approach, where the agent maximizes his expected rewards for H steps in time. In Eqn. 2.1, $\gamma^t = 1$.
- An *Infinite horizons with discount* approach, where the agent maximizes his expected discounted infinite rewards. In Eqn. 2.1, $0 < \gamma^t < 1$ and $n \to \infty$.
- An *Infinite horizons with averaging* approach, where the agent maximizes his expected average of infinite rewards: $\lim_{n\to\infty} E[\frac{1}{n}\sum_{t=0}^{n} \gamma^t r_{\pi(s_t)}(s_t)]$.

2.2 Partially Observable Markov Decision Processes

MDPs are used in problems where the agent always knows the state of the world in any point in time. However, in many real world applications this setting is not very common. The usual challenge that an agent faces is the partial observability of the environment. This forces the agent to keep track of observations somehow to infer the state of the world and therefore decide further steps. For such problems, a more complex model called partially observable Markov decision process (POMDP) is used. This paradigm extends the classical MDP by incorporating a set of observations O, where $p(o|s)$ is the probability that the agent observes $o \in \Omega$ when the world is in state $s \in S$, with Ω being the set of all possible observations. Thus, a POMDP model is defined as $POMDP\langle S, A, \Omega, T, O, R\rangle$.

In POMDPs, an agent should keep track of what he believes is the state of the system, based on his observations. Therefore, a *belief* is defined in this framework as a probability distribution over the state space (see Fig.2.1). POMDPs assume that the initial belief is



(a) (b)

Figure 2.1: Belief simplices for 2 (a) and 3 (b) states. Since this are probabilities distributions, the coordinates of each point within the belief simplex sum to 1.

known, and it is updated by the belief update equation τ given by

$$\tau(b_{t-1}, a_{t-1}, o_t) = b_t(s_t) = \frac{\sum_{s_{t-1}} O(s_t, a_{t-1}, o_t) \times T(s_{t-1}, a_{t-1}, s_t) \times b_{t-1}(s_{t-1})}{Pr(o_t|b_{t-1}, a_{t-1})} \tag{2.2}$$

Value iteration in POMDPs is more challenging to calculate, since now both the beliefs and the observation probabilities come into the picture. For POMDPs, the value function is given by

$$V_0(b) = max_a \sum_{s \in S} R(s, a)b(s). \tag{2.3}$$

$$V_t(b) = max_a[\sum_{s \in S} R(s, a)b(s) + \gamma \sum_{o \in \Omega} Pr(o|a, b) \times V_{t+1}(\tau(b, a, o))]. \tag{2.4}$$

To look for a solution, a more convenient way of representing the value function is through *alpha vectors*. Sondik [34] demonstrated that the value function at any finite horizon t can be expressed by a set of vectors $\Gamma_t = \alpha_0, \alpha_1, ..., \alpha_m$. These alpha vectors are hyperplanes that define the value function over a bounded region of the belief. Each alpha vector is associated with an action, and therefore, finding the alpha vector that maximizes the value function over a belief point defines the optimal policy. Eqn. (2.4) may be rewritten as:

$$V_t(b) = max_{\alpha \in \Gamma_t} \sum_{s \in S} \alpha(s) \times b(s). \tag{2.5}$$

To understand value iteration, we must first rewrite Eqn. (2.4) in terms of alpha vectors:

$$V_t(b) = max_a[\sum_{s\in S} R(s,a)b(s) + \gamma \sum_{o\in\Omega} max_{\alpha\in\Gamma_{t+1}} \sum_{s\in S}\sum_{s'\in S} T(s,a,s')O(s',a'o)\alpha(s')b(s)]. \quad (2.6)$$

Γ_t is generated then through a sequence of operations over Γ_{t-1}, the first of them (*Step 1*) being to generate the intermediate sets $\Gamma_t^{a,*}$ and $\Gamma_t^{a,o}$,$\forall a \in A, \forall o \in \Omega$:

$$\Gamma_t^{a,*} \leftarrow \alpha^{a,*}(s) = R(s,a)$$
$$\Gamma_t^{a,o} \leftarrow \alpha^{a,0}(s) = \gamma \sum_{s'\in S} T(s,a,s')O(s',a,o)\alpha_i(s'), \forall \alpha_i \in \Gamma_{t+1}$$

Next, Γ_t^a is the cross-sum over observations (denoted by $\oplus$;*Step 2*, Eqn.2.7). The union of these sets is the final Γ_t (*Step 3*, Eqn.2.8):

$$\Gamma_t^a = \Gamma_t^{a,*} + \Gamma_t^{a,o_1} \oplus \Gamma_t^{a,o_2} \oplus ... \quad (2.7)$$

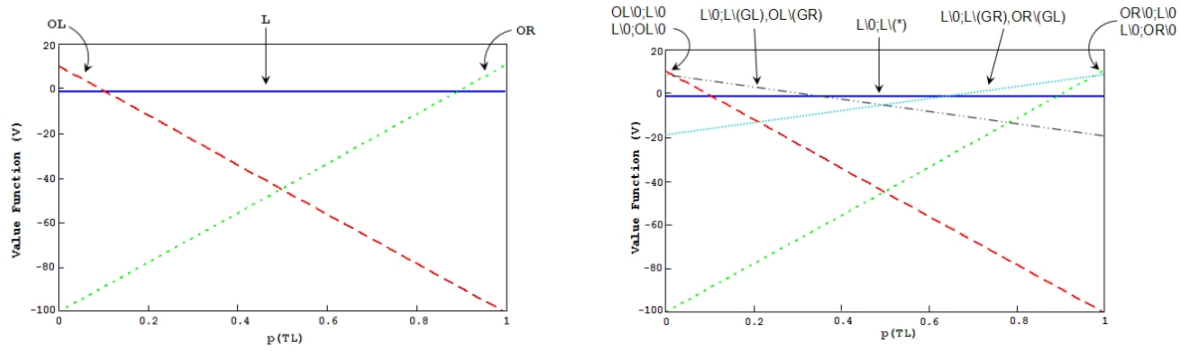
$$\Gamma_t = \bigcup_{a\in A} \Gamma_t^a. \quad (2.8)$$

We will illustrate the value iteration process and the policy construction using alpha vectors with an example. The tiger problem is presented in Cassandra et al. [6], and due to its few states is an adequate example to aid in understanding of POMDPs. It is a game show situation, where the decision maker has to decide to open one of two doors. In one door there exists a pot of gold, and is the one we aim to find. But in the other, there is a dangerous tiger. The agent possible actions are to listen for the tiger growl, or to open any of the doors. Uncertainty is present in the observation of the agent about the growl, since in some opportunities it may hear imperfectly (15% of the time for our simulations). The rewards are given as -100 for opening a door with a tiger, $+10$ for finding the pot and -1 for listening. The state space is either that the tiger is behind the left door (TL) or it is behind the right door (TR).

Fig. 2.2 shows the value function for the Tiger problem at horizons 1 (initial alpha vectors) and 2 (after one backup). These graphs plot the probability of the Tiger being in the left door ($p(TL)$) versus the value function for each horizon. The alpha vectors that maximize value determine which action the agent should perform at a given horizon. For example, in fig. 2.2(a), when $p(TL) < 0.1$ the action should open left(OL), because it is very unlikely that the tiger will be behind the left door. On the other hand, when $p(TL) > 0.9$, it is highly possible that the tiger awaits behind the left door, thus we should open right(OR). In any other case, we listen. Notice also that since we only have 2 states and a belief is a probability function, $p(TL) = 0.1$ is equal to $p(TR) = 0.9$.

Fig. 2.2(b) shows the same value function but for horizon 2. After a value iteration, some new vectors are added to the solution. Now, each vector represents horizon 2 plans. Notice that when $p(TL) < 0.02$ the agent either open left(OL) and then listens (L) no matter what observation it gets; or listens and open left no matter what. When $0.02 < p(TL) < 0.39$ the plan start listening, and depending on observing a growl left(GL) the agent will listen again(L), or on observing a growl right(GR) the agent will open left(OL). For $0.39 < p(TL) < 0.61$ the agent listen and listen again. Fig. 2.3 shows a better representation of the plans: a policy tree. In this graph, we can easily see what the plan of the agent is. Fig. PolicyTiger4 shows the policy tree for horizon 4. Notice how the dynamic programming works to build the solution backwards. At $t = 1$ we have the initial set of alpha vectors, input to the value iteration process. The process generates a set of vectors a $t = 2$, 3 and 4. In the end, we get back the solution as a set of vectors at $t = 4$. Then we construct our plan of action backwards, depending on the observations at each time step.

The insightful reader may already have noticed that the problem of finding an optimal solution becomes intractable as the number of states of the world is increased. Exact solutions can only be computed for problems with a small number of states. Due to this problem, many approximation algorithms have been developed. Many of these approximate methods gain



(a) (b)

Figure 2.2: Horizons 1 (a) and 2 (b) value functions for the Tiger problem.

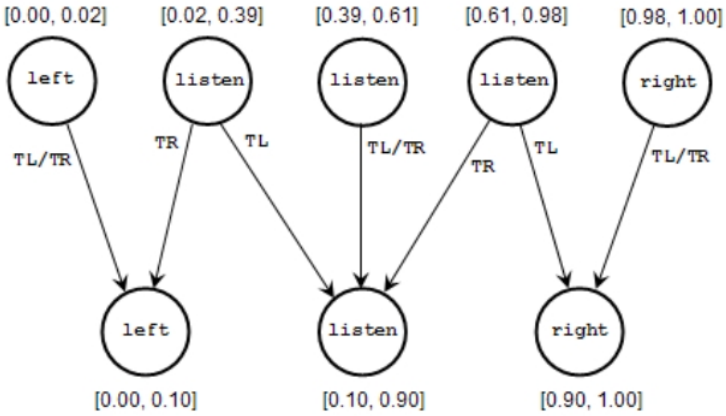


Figure 2.3: Policy tree for horizon 2 in the Tiger Problem.

computational advantage by applying value updates to a reduced number of belief points, instead of over all beliefs ([7], [38], [27]).

But first, why is it so challenging to compute solutions for POMDPs? There are two reasons: the so called *curse of dimensionality*; and the less known *curse of history*. The curse of dimensionality is defined as the increase in complexity due to the increasing number of alpha vectors needed to compute a value function. More strictly speaking, for a problem

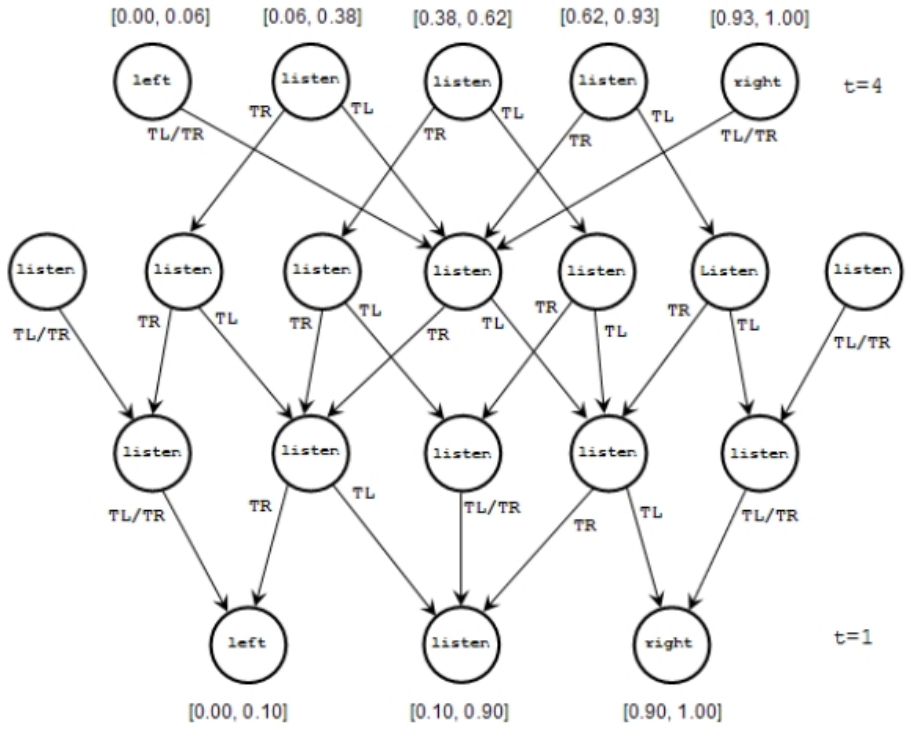


Figure 2.4: Policy tree for horizon 4 in the Tiger Problem.

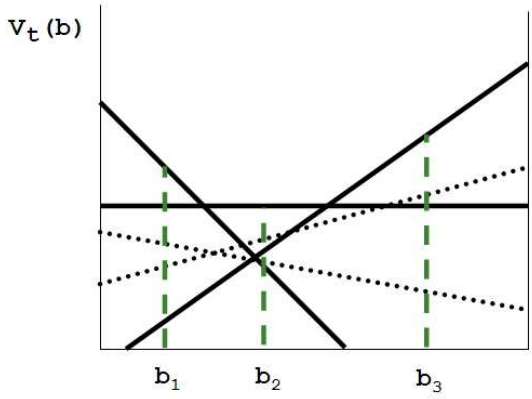


Figure 2.5: Point based value iteration. The set of beliefs determine which vectors are kept as the solution for each iteration, improving significantly the efficiency in policy search. This set may be increased as we iterate. Vectors in dots are dominated at the belief points in the set.

with n physical states, π is defined over all belief states in an $(n-1)$ dimensional continuous space. This means that the complexity of finding an optimal solution grows exponentially with horizon even when the number of states is low. The curse of history, on the other hand, is the problem that arises as the planning horizon increases during the search of an optimal solution. POMDP solving could be understood as a search over possible POMDP histories [26] (represented by beliefs), starting with short stories first and gradually increasing to long histories. As we move toward the longest histories, the number of distinct possible action-observations increases exponentially.

2.3 Point Based Value Iteration

Among the approximation methods that have been developed for POMDPs, the set of methods called *Point Based Value Iteration (PBVI)* [25] are of our particular interest. These methods have been proven to solve POMDPs in fairly short times, keeping a good level of accuracy and thus making it a good option for real world applications.

In PBVI methods, value iteration is modified in a way that significant computational gains are obtained. A set of belief points B is used by the method to find an approximate solution Γ_t (see Fig. 2.5). Given a solution set Γ_{t-1}, the exact backup operator (Eqn. (2.6)) is modified in a way that only one α-vector per belief point in B is maintained. The point-based value backup assumption is that every alpha vector should be valid for a region surrounding the belief point b, since all other points in this region should have the same action choice. The gain is obtained by eliminating the cross-sums on the regular backup operator (*Step 2*, instead of Eqn. (2.7)) and keeping only alpha vectors that are optimal on the points in the belief set (*Step 3*, instead of Eqn. (2.8)).

Step 2:

$$\Gamma_t^a \leftarrow \alpha_b^a = \Gamma_t^{a,*} + \sum_{o \in \Omega} argmax_{\alpha \in \Gamma_t^{a,o}} (\sum_{s \in S} \alpha(s) b(s)), \forall b \in B.$$

Step 3:

$$\alpha_b = argmax_{\Gamma_t^a, \forall a \in A} (\sum_{s \in S} \Gamma_t^a(s) b(s)), \forall b \in B. \Gamma_t = \bigcup_{b \in B} \alpha_b.$$

Another important process in the PBVI methods is the way new belief points are selected to be included in the set, and the number of points that should be included at every belief set expansion. The trade-offs are clear to see: the more points in the set, the more accuracy in the approximation of the value function, but also less time efficiency in calculating the approximation. Pineau [26] presented 5 different point selection strategies (also called heuristics):

- **Random Belief Selection (RA)**, a naive approach that simply selects random points uniformly over the belief simplex. It does not take into consideration reachability (beliefs that may be generated from n updates on the initial belief, forming a *belief tree* of depth n).

- **Stochastic Simulation with Random Action (SSRA)**, where a single belief update $\tau(b, a, o)$(Eqn. (2.2)) is applied to every point in the belief set, but the action a (uniformly over A) and initial state s (multinomial over b) are selected randomly;

s' selected randomly from the transition function $T(s, a, \bullet)$; and o selected randomly from the observation function $O(s', a, \bullet)$.

- **Stochastic Simulation with Greedy Action (SSGA)**, similar to SSRA. The only difference is that instead of selecting the action randomly, the best action is greedily selected ($a = argmax_{\alpha \in \Gamma} \sum_{s \in S} \alpha(s) b(s)$).

- **Stochastic Simulation with Exploratory Action (SSEA)**, performs a one step forward for each action (instead of selecting it randomly or greedily), generating new beliefs $b_{a_0}, b_{a_1}, ...$, calculates the distance L from each point to its closest neighbor in the belief set, and finally keeps the b_{a_i} that is farthest away from any point already in B.

- **Greedy Error Reduction (GER)**, looks for the point in the belief tree (the immediate descendants of the points in the set B of belief points) that best reduces the error bounds [2]. For a new belief b' with value hyper-plane α', the error bound is given by

$$\epsilon(b') \leq (\alpha' - \alpha) \times (b' - b) \tag{2.9}$$

Since α' is not known until some backups are performed on b', the worst-case value of α' is used. Thus:

$$\epsilon(b') \leq min_{b \in B} \sum_{s \in S} \begin{cases} (\frac{R_{max}}{1-\gamma} - \alpha(s))(b'(s) - b(s)) & b'(s) \geq b(s) \\ (\frac{R_{min}}{1-\gamma} - \alpha(s))(b'(s) - b(s)) & b'(s) < b(s) \end{cases} \tag{2.10}$$

We evaluate the error at each $b \in B$, weighting the error on the fringe nodes by their reachability probability:

$$\epsilon(b) = max_{a \in A} \sum_{o \in \Omega} O(b, a, o) \epsilon(\tau(b, a, o)) \tag{2.11}$$

[2] We cannot compute the expected error before performing an iteration. An error bound is all that can be estimated.

Then we add to B the belief $b(\tilde{b}, \tilde{a}, \tilde{o})$ which maximizes error bound reduction, where

$$\tilde{b}, \tilde{a} = argmax_{b \in B, a \in A} \sum_{o \in \Omega} O(b, a, o) \epsilon(\tau(b, a, o)) \tag{2.12}$$

$$\tilde{o} = argmax_{o \in \Omega} O(\tilde{b}, \tilde{a}, o) \epsilon(\tau(\tilde{b}, \tilde{a}, o)) \tag{2.13}$$

From these four, the one that presented the best results in the simulations with several of the classical game problems was GER.

Pineau et al. [26] suggested that anytime point based planning algorithms could benefit significantly from the principled selection of belief points during the belief set expansion. James et al [15] presented an improvement for PBVI methods that implements the concept of *gain* of a point, a scalar measure that allows the algorithm to evaluate how useful a point would be. The ideal behind this solution is to identify a point to add to the belief set that given the current approximation of the value function at time t will most improve the approximation of the value function at time $t + 1$. Unluckily, exact computation of the best gain is computationally expensive, and [15] presents two approaches that do not fall into such computational complexity.

The first heuristic for belief set expansion on PBVI methods presented in [15] is called *Gain Measure by Belief Expansion*, and is a measure of the difference between the current value of point b and the value according to an α vector computed by a one-step backup (Eqn. (2.6)). The underlying idea is that if the one-step difference is large, then it will also improve the approximations of the nearby points significantly. This improvement propagates to points that lead to these (through the backup).Thus, g_B is defined as:

$$g_B(b) = max_a[r(b, a) + \gamma \sum_a Pr(o|b, a) V(b_a^o)] - V(b) \tag{2.14}$$

$$= b^T \alpha - V(b). \tag{2.15}$$

where $\alpha = backup(b)$.

The second heuristic, *Gain Based on Value-Function Bounds* is computationally more complex to calculate. To explain the idea of this measure, assume a two nominal state space. In this space, optimal values for beliefs A and B are given by $\alpha_A * A$ and $\alpha_B * B$, and are assumed to be correct. Now, consider a new point X in the belief space, between A and B, and a corresponding α_X. If α_A indeed maximizes the value function at A, and α_B does for B, then α_X cannot modify the value for A or B, but is upper-bounded by the vector that connects the maximum value at A and B (For a visual representation of this example, see [15] Fig. 1(a)). Thus, the gain g_{LB} would be the difference between the upper bound and the value of X.

From what has been explained so far, it is evident that the problem complexity is increased as states are added. For a 3 nominal state problem, the upper-bound is given by a surface. Say, for example, that the value function is defined in four points A, B, C and D. Then we would be looking for a plane that connect a combination of the values at A, B, C or D, and represents the tighter upper bound of a plane at X (For a visual representation of this example, see [15] Fig. 1(b) and 1(c)). In higher dimensional spaces, it is not a trivial task to select those points that make such a multidimensional surface. The solution proposed includes solving linear programs to find the points and the gain (see [15] for implementation specifics).

The results of the simulations presented in [15] showed an increase in accuracy due to the selection of good points over classical PBVI methods. From the two methods presented in this paper, the g_{LB} the metric had a better performance. The authors also made clear in the results the tradeoff between accuracy and time, since the selection of points adds some complexity to the problem and regular PBVI methods perform slightly better in terms of time of computation.

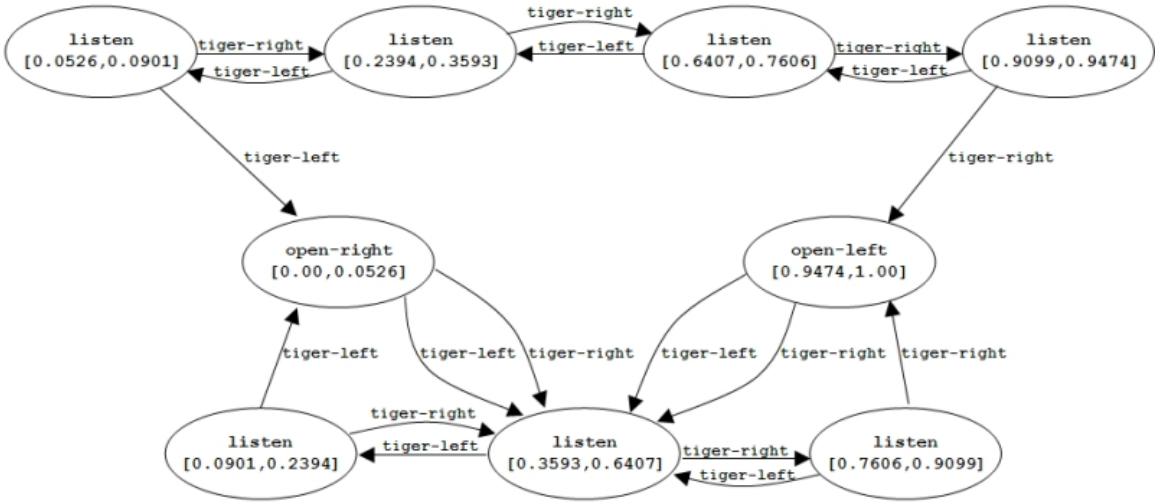


Figure 2.6: Policy graph for an exact solution of the Tiger problem, on an infinite horizon and with $\gamma = 0.9$.

2.4 Initial Experiments

A program was developed as a part of this thesis to replicate Pineau's results [26] and to serve as a basis for the new method we are proposing (see Chapter 4 for details). The tiger problem, one of the computationally friendly scenarios in the literature, was used for this purpose.

Fig. 2.6 and 2.7 show a policy graph for the tiger problem by using an exact method and a PBVI method, on an infinite horizon. The policy graphs contain the same information than policy trees, but for infinite horizons. The graph shows how to proceed when an action is performed and an observation is received. Each node represents the action to take and the arrows that start from them leads to the next action given the observation caption of the arrow. The ranges in each node are the cut-off probabilities that lead to each initial action. Note how the approximation policy "prunes" several nodes that listen. This means that the approximation will not do as good as the exact on the long run in a simulation, but we

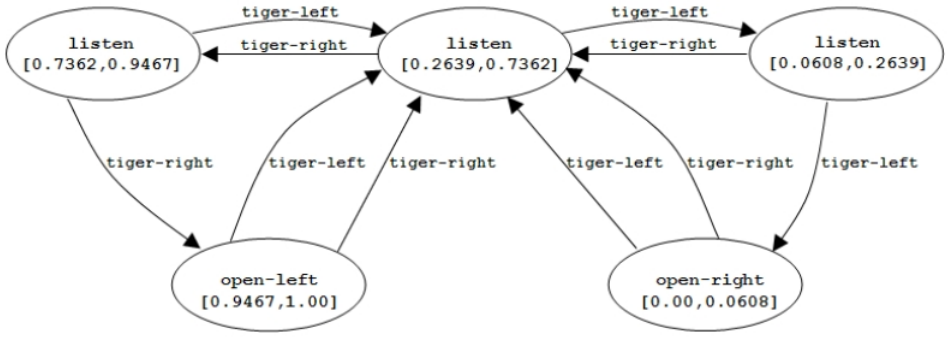


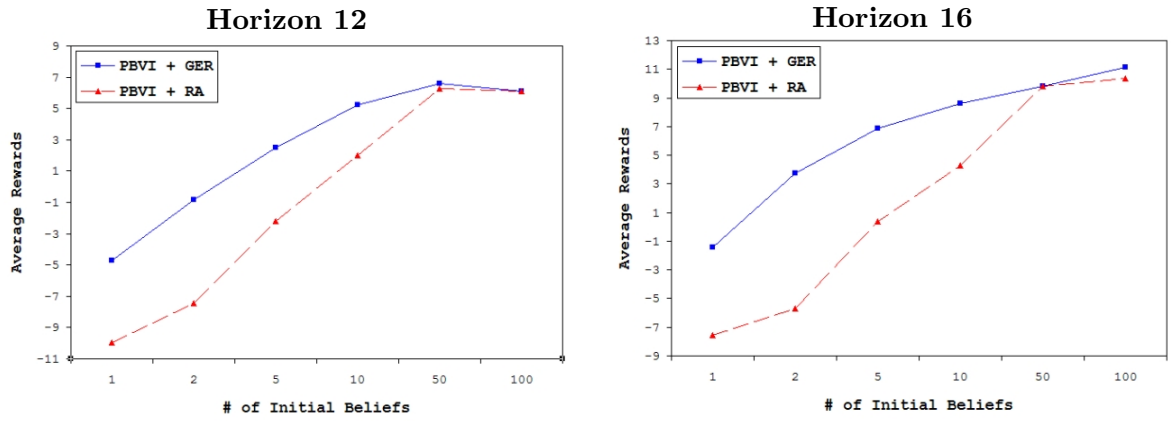
Figure 2.7: Policy graph for an approximate solution of the Tiger problem using PBVI and Greedy Error Reduction, on a infinite horizon and with $\gamma = 0.9$.

expect to have similar performance. Note also that it is impossible to do better than the exact approach in the long run.

We also run a simulation to compare the performance of two belief expansion methods empirically. Fig. 2.8 shows two graphs with comparative results between both PBVI with greedy error reduction and PBVI with random belief selection as a belief expansion method. The results show how the number of initial beliefs makes a difference in performance. When we include a small number of initial beliefs, the GER heuristic leads to better policies. But when the number of initial beliefs is sufficiently large, the performance of policies generated with each method is not significantly different. As we increase the number of initial beliefs, we get closer to the value obtained by the exact approaches.

2.5 Summary

POMDPs are decision-theoretic frameworks for planning under uncertainty in single agent settings. They address both the action outcome uncertainty and the state of the world uncertainty problems. Exact solutions under this framework are computationally expensive to calculate, and exact policies have been reported in the literature only for simple domains



(a) (b)

Figure 2.8: Average rewards of PBVI solution policies at horizon 12(a) and 16(b). Simulation averaged results of 100 policy trees, over 100 simulation runs for each of them.

that contain less than 10 states. However, an important amount of work is being done in the development of approximation techniques. These techniques have reportedly been able to minimize the two sources of complexity in POMDPs: the curse of dimensionality, and the policy state complexity or curse of history.

Chapter 3

Interactive Partially Observable Markov Decision Processes

The reader could probably think of many problems where one agent would not be interacting alone with the world. Multiagent settings have many agents either cooperating or competing to achieve a task. Planning under these environments with POMDP models would not take into account other agents as actual entities whose actions may affect directly or indirectly the performance of the agent, but they would be modeled as noise (included in the transition probabilities). It is therefore evident that a better approach would take into account other agents and their actions and beliefs as part of the model, for the agent to achieve a better performance in the planning task. Gmytrasiewicz and Doshi [13] presented a new formalism, called Interactive partially observable Markov decision processes (I-POMDPs), that takes into account other agents and their beliefs as part of the model (see Fig 3.1) and showed how this setting performs better in accuracy when compared with classic POMDPs results. This chapter will present all the details related to this new framework.

For the sake of simplicity, I-POMDPs are usually presented assuming intentional model agents, similar to those used in Bayesian games, though the framework extends to any model. Also for the sake of simplicity, theory is usually presented considering an agent i, interacting with another agent j. All results are scalable for three or more agents. An *I-POMDP* of agent i is defined as: $\textit{I-POMDP}_i = \langle IS_i, A, T_i, \Omega_i, O_i, R_i \rangle$, where

- IS_i is the set of interactive states defined as $IS_i = S \times M_j$, where S is the set of physical states of the environment, and M_j is the set of possible models of agent j. Each model $m_j \in M_j$ is a pair $m_j = \langle h_j, f_j \rangle$, where $f_j : H_j \rightarrow \Delta(A_j)$ is agent j's function, assumed

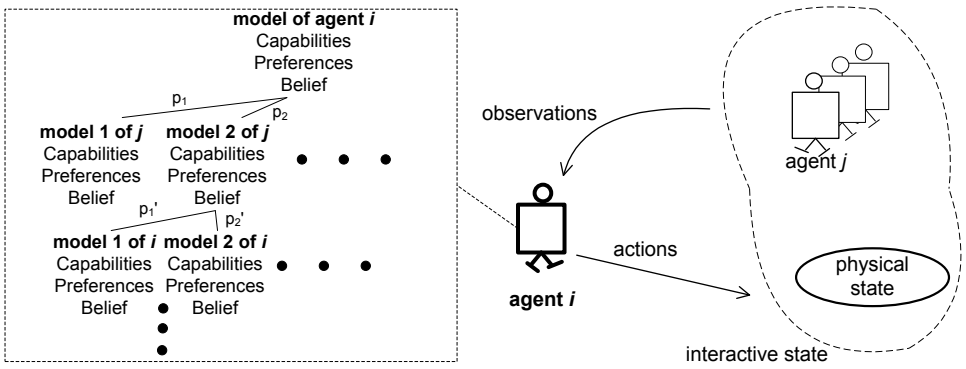


Figure 3.1: I-POMDP problem setting.

computable, which maps possible histories of j's observations to distributions over its actions. h_j is an element of H_j.

- $A = A_i \times A_j$ is the set of joint moves of all agents.
- Given the *Model Non-manipulability Assumption (MNM)*: Agents actions do not change other agents' model directly, T_i, the transition function, is defined as: $T_i : S \times A \times S \rightarrow [0, 1]$.
- Ω_i is the set of agent i's observations.
- Given the *Model Non-Observability Assumption (MNO)*: Agents cannot observe other's models directly, O_i, the observation function, is defined as : $O_i : S \times A \times \Omega \rightarrow [0, 1]$.
- $R_i : IS_i \times A \rightarrow \boldsymbol{R}$.

An important class of models, which is usually used in the literature, is *intentional models*. These models ascribe to the other agents' beliefs, preferences and rationality in action selection. Intentional models are solutions to j's types, $\theta_j = \langle b_j, \hat{\theta}_j \rangle$, under the assumption that agent j is Bayesian rational. Agent j's belief is a probability distribution over states of the environment and the models of the agent i: $b_j \in \Delta(S \times M_i)$.

3.1 Beliefs in I-POMDPs

I-POMDP's interactive beliefs infinitely nest other agents' beliefs. This poses a problem when looking for optimal solutions, since models with infinitely nesting of beliefs are not computable models. A logical solution is to constrain the nesting of beliefs, a finite nesting. Finitely nested beliefs are constructed bottom-up. First let's assume two agents, i and $-i$ for the sake of clarity. Agent i's 0^{th} level beliefs, $b_{i,0}$, are probability distributions over S (the physical states). Its 0^{th} level types, $\Theta_{i,0}$, contain its 0^{th} level beliefs, and its frames. Agent $-i$'s beliefs and types are analogously defined. Notice 0^{th} level types are POMDPs. Agent i's level-1 beliefs, $b_{i,1}$, are probability distributions over physical states and level-0 types of agent $-i$. Agent i's level-1 types consist of it's first level beliefs and frames. Level-2 beliefs are defined in terms of level-1 types and so on. The formulas below depict the recursion:

$$
\begin{aligned}
&IS_{i,0} = S, && \Theta_{-i,0} = \langle b_{-i,0}, \hat{\theta}_{-i} \rangle : b_{-i,0} \in \Delta(IS_{-i,0}), A = A_{-i}, \\
&IS_{i,1} = IS_{i,0} \times \Theta_{-i,0}, && \Theta_{-i,1} = \langle b_{-i,1}, \hat{\theta}_{-i} \rangle : b_{-i,1} \in \Delta(IS_{-i,1}), \\
&\quad \vdots && \quad \vdots \\
&IS_{i,l} = IS_{i,l-1} \times \Theta_{-i,l-1}, && \Theta_{-i,l} = \langle b_{-i,l}, \hat{\theta}_{-i} \rangle : b_{-i,l} \in \Delta(IS_{-i,l}).
\end{aligned}
$$

Thus, we define a *finitely nested I-POMDP* of agent i as

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

where the parameter l is called the *strategy level* of the finitely nested I-POMDP. The belief update, value function and optimal actions for finitely nested I-POMDPs are calculated by the equations presented before, but recursion is guaranteed to end at the 0^{th} level models.

The belief update in I-POMDPs is more complicated than in POMDPs. Since the state of the physical environment depends on the actions of all agents, a prediction on how the physical state changes has to be done taking into account the other agent's possible actions.

In order to do this, it is necessary to keep the other agent's beliefs. So I-POMDPs belief update also includes updating the belief of the other agents, represented in our model. For the sake of understanding, the belief update process may be divided into two steps:

- *Prediction:* When agent i performs action a_i^{t-1} and agent j performs action a_j^{t-1}, the predicted belief state is given by:

$$\begin{aligned} Pr(is^t|a_i^{t-1}, a_j^{t-1}, b_{i,l}^{t-1}) = \quad & \int_{IS^{t-1}} b_{i,l}^{t-1}(is^{t-1}) \times Pr(a_j^{t-1}|\theta_{j,l-1}^{t-1}) \\ & \times T_i(s^{t-1}, a_i^{t-1}, a_j^{t-1}, s^t) \times \textstyle\sum_{o_j^t} O_j(s^t, a_i^{t-1}, a_j^{t-1}, o_j^t) \\ & \times \delta(SE_{\hat{\theta}_j^t}(b_{j,l-1}^{t-1}, a_j^{t-1}, o_j^t) - b_{j,l-1}^t) d(is^{t-1}) \end{aligned} \tag{3.1}$$

  where δ is the Dirac-delta function, $SE(\bullet)$ is the belief update function, and $Pr(a_j^{t-1}|\theta_{j,l-1}^{t-1})$ is the probability that a_j^{t-1} is Bayes rational for the agent described by $\theta_{j,l-1}^{t-1}$.

- *Correction:* After receiving an observation, o_i^t, the corrected belief state is an expectation over every possible action of j (α is a normalizing constant):

$$Pr(is^t|o_i^t, a_i^{t-1}, b_{i,l}^{t-1}) = \alpha \sum_{a_j^{t-1}} O_i(s^t, a_i^{t-1}, a_j^{t-1}, o_i^t) \times Pr(is^t|a_i^{t-1}, a_j^{t-1}, b_{i,l}^{t-1}). \tag{3.2}$$

The belief update of i invokes the belief update of j (through the SE function in Eqn.(3.1)) and if j is modeled as an I-POMDP, it will call back again the belief update for i and so on, until the recursion bottoms out at the 0^{th} level. At this level, the agent's belief update reduces to a POMDP belief update.

3.2 Value Iteration in I-POMDPs

Value iteration is also more complicated than in POMDPs, since now the expectation of the actions performed by other agents must be included. The value function for I-POMDPs is

therefore given by:

$$V^t(\langle b_{i,l}, \hat{\theta}_i \rangle) = \begin{array}{l} max_{a_i \in A_i} \int_{is} ER_i(is, a_i) b_{i,l}(is) d(is) + \gamma \sum_{o_i \in \Omega_i} Pr(o_i | a_i, b_{i,l}) \\ \times V^{t-1}(\langle SE_{\hat{\theta}_i}(b_{i,l}, a_i, o_i), \hat{\theta}_i \rangle). \end{array} \tag{3.3}$$

where $ER_i(is, a_i) = \sum_{a_j} R_i(is, a_i, a_j) Pr(a_j | \theta_{j,l-1})$.

It is, though, non-trivial to calculate this function. Since the agent is unaware of the true models of the other agents, it must maintain a belief over all possible models. Thus, the problem becomes intractable for all but the simplest settings, and good approximation methods are needed.

Notice that the value function, V^t, maps $\Theta_{i,l} \rightarrow \mathbb{R}$. Because $\Theta_{i,l}$ is a continuous space (countable infinite if we limit to computable beliefs), we cannot iterate over all the models of i to compute their values. Instead, analogous to POMDPs, we may decompose the value function into its components:

$$V^t(\langle b_{i,l}, \widehat{\theta_i} \rangle) = \sum_{is \in IS_{i,l}} \alpha^t(is) \times b_{i,l}(is) \tag{3.4}$$

where,

$$\alpha^t(is) = \begin{array}{l} \max_{a_i \in A_i} \Bigg\{ ER_i(is, a_i) + \gamma \sum_{o_i} \sum_{is' \in IS_{i,l}} \Bigg\{ \sum_{a_j} Pr(a_j | \theta_{j,l-1}) \Bigg[T_i(s, a_i, a_j, s') O_i(s', a_i, a_j, o_i) \\ \sum_{o_j} O_j(s', a_i, a_j, o_j) \delta_D(SE_{\hat{\theta}_j}(b_{j,l-1}, a_j, o_j) - b'_{j,l-1}) \Bigg] \Bigg\} \alpha^{t+1}(is') \Bigg\} \end{array} \tag{3.5}$$

The proof for Eq. 3.4 is given in the Appendix of [13]. Exact computation of the α^t's using methods analogous to those of POMDPs [6, 21] is possible for the simplest settings. Here, we calculate all possible alpha vectors at time $t+1$ ($\mathcal{O}(|A_i||\Omega_i|)$ sets of $|\Gamma_{t+1}|$ new vectors, since we apply the formula for every action and observation, for each vector in the solution at t). The formulas are given below:

$$\Gamma^{a_i,*} \leftarrow \alpha^{a_i,*}(is) = \sum_{a_j \in A_j} R(s, a_i, a_j) Pr(a_j | \theta_{j,l-1}) \tag{3.6}$$

$$\Gamma^{a_i,o_i} \overset{\cup}{\leftarrow} \quad \alpha^{a_i,o_i}(is) = \gamma \sum_{is'} \sum_{a_j} Pr(a_j|\theta_{j,l-1}) T_i(s,a_i,a_j,s') O_i(s',a_i,a_j,o_i) \\ \sum_{o_j} O_j(s',a_i,a_j,o_j) \delta_D(SE_{\widehat{\theta}_j}(b_{j,l-1},a_j,o_j) - b'_{j,l-1})\ \alpha^{t+1}(is'), \forall \alpha^{t+1} \in \Gamma_{t+1} \tag{3.7}$$

Next, we build Γ^{a_i} by taking the cross-sum of the previously computed sets of alpha vectors, analogous to POMDP:

$$\Gamma^{a_i} \leftarrow \Gamma^{a_i,*} \oplus \Gamma^{a_i,o_i^1} \oplus \Gamma^{a_i,o_i^2} \oplus \cdots \oplus \Gamma^{a_i,o_i^{|\Omega_i|}} \tag{3.8}$$

We may generate $\mathcal{O}(|A_i||\Gamma^{t+1}|^{|\Omega_i|})$ many distinct intermediate alpha vectors. We need a linear program (LP) to pick those that are optimal for at least one of the belief points in our set.

$$\Gamma^t = \underset{\alpha^t}{\mathsf{prune}}\ (\bigcup_{a_i} \Gamma^{a_i})$$

Notice that Eqs. 3.6 and 3.7 require $Pr(a_j|\theta_{j,l-1})$, which involves solving the level $l-1$ intentional models of agent j. Thus, we carry out the above mentioned procedure recursively for solving models at all levels.

3.3 The Interactive Particle Filter

Doshi and Gmytrasiewicz [9] presented an approximation technique called *Interactive Particle Filter* (IPF), an approach that is based on the particle filter, and more exactly, the bootstrap filter. Using a nested polynomial representation (see [9] for details), they implemented the filter and tested it over two problems on the literature: the tiger problem and the machine maintenance problem, both adjusted to fit the multiagent setting. Their results showed that the particle filter improves over an exact approach tackling the curse of dimensionality, but it does not scale to large horizons, being affected by the curse of history.

The particle filter algorithm can be reviewed in Doucet et. al [11]. The bootstrap implementation of this method works by maintaining a set of N particles sampled from the prior

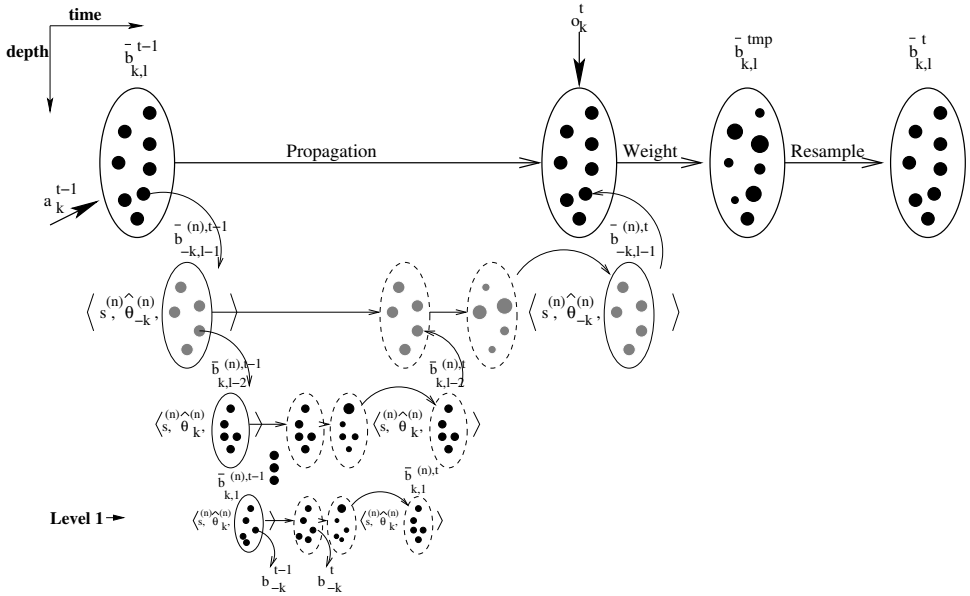


Figure 3.2: Interactive Particle Filter. The image shows how the recursive call is performed in the propagation step, and how this recursion bottoms out at the zero strategy level. Colors black and gray differentiate among agents.

belief. Each of these particles is propagated forward in time, by applying the transition function T_i. Next, each particle is weighted by the probability of perceiving an observation of the state that it represents, given by the observation function O_i. Finally, an unbiased resampling step selects those particles that will prevail proportionally to their weights and assign them a uniform weight.

The interactive particle filter implementation works similarly, but adds a recursive call in the propagation step (See Fig.3.2). Since I-POMDPs include the models of j, the propagation step must include the action of the other agent. Thus, a call to solve the model of j included in the interactive state (recall $is^t{=}\langle s^t, \theta_j^t \rangle$) must be performed.

As reported by the authors in [9], the IPF solves effectively the effects of the curse of dimensionality, but it is not possible to generate solutions for large horizons with it. This is due to the growth of the look ahead reachability tree needed for larger horizons (the curse of

history). When implemented and executed, the memory resources available will rapidly fill up. We will discuss about empirical results of the IPF in comparison with the new method presented in this thesis in Chapter 5.

3.4 Summary

I-POMDP is a framework for optimal sequential decision making suitable for autonomous agent control when these interact with other agents in a certain environment. I-POMDPs extend POMDPs by having beliefs not only about the physical environment, but also about other agents. This extension adds complexity into the model and exact solutions are even harder to compute than for POMDPs. The interactive particle filter is an approximation method that extends the classical particle filter to multiagent settings. This method aims to minimize the effects of the curse of history and the curse of dimensionality in I-POMDPs, to allow for approximate solutions in a reasonable amount of time. The method fails to address the former problem effectively.

Chapter 4

Interactive Point Based Value Iteration

Researchers in the POMDP framework have developed in the last few years many methods that offer fairly good approximations for solving POMDPs, allowing significant savings in computation time as compared to exact approaches and still providing good accuracy. One of these methods, which has proven very successful due to its good performance and the fact that can be implemented as an anytime method, is PBVI [25]. The I-POMDP framework, due to its novelty, lacks of an equivalent amount of research in approximation methods. In order to implement I-POMDPs in real world applications, an effort on the development of new approximation methods is needed, mainly because current approximation methods do not scale well when horizons are increased (namely the I-PF).

Another open issue with I-POMDPs is belief representation. Previous work has used nested polynomials to represent nested beliefs, but researchers are still on the lookout for a better representation schema that may lead to better performance of current and new approximation methods.

The aim of this research is to find a good approximation method based/extending on the PBVI methods for POMDPs. The goal is to provide offline solutions of finitely nested I-POMDPs that do not assume a particular initial belief of the agent. Since it has been shown in the context of POMDPs that point based methods provide effective offline approximations that scale well to relatively large problems, our proposal will be based on them, namely an *Interactive Point Based Value Iteration (I-PBVI)* method.

To provide generalization of the point based value iteration algorithm for I-POMDPs is not a trivial task. The main challenges that we confront are three:

- Point based techniques use a set of initial belief points. Therefore, there's a need for computational representations of the nested beliefs of I-POMDPs in order to select the initial belief points.

- The number of computable models of other agents could be infinitely large, making the state space prohibitively complex.

- The performance of an agent in multiagent settings depends on other agents' actions too. Therefore, for optimal decision making, the solutions of other agents' models are required. This requirement suggests a recursive implementation of the point based approach.

This chapter will present the proposed method in detail. As we develop the chapter, we will tackle the challenges presented before and provide solutions to each of them. Section 4.1 focuses on the computational representation of nested beliefs. Section 4.2 moves on to present the bounded interactive states, a condition necessary for computability of the I-POMDPs with point based methods. Section 4.3 presents the method with detail, including some algorithms to illustrate the procedure. Section 4.4 describes two belief expansion methods, an important part of the point based approaches, which we propose for I-PBVI. Finally, section 4.5 presents an analysis on the computational savings of the method.

4.1 Computational Representation of Nested Beliefs

In chapter 2, we characterized the nested beliefs of I-POMDPs. We also presented the definition of finitely nested beliefs, to make belief update on I-POMDPs computable. In this section, we will provide a new representation of nested beliefs, and present the use of mixtures of Gaussians as an implementation of this new notation.

For a better understanding of the representation, we will explore it starting at level 0. Please keep in mind that a couple of assumptions have to be made to promote understanding:

Agent j's frame is known and the uncertainty of agent i is only about the physical states and the belief of j.

At level 0, agent i's belief, $b_{i,0} \in \Delta(S)$ (where i denotes the agent and 0 the level of nesting), is a vector of probabilities over each physical state:

$$b_{i,0} \stackrel{def}{=} \langle p_{i,0}(s_1), p_{i,0}(s_2), \ldots, p_{i,0}(s_{|S|}) \rangle$$

Beliefs are probability distributions, thus $\sum_{q=1}^{|S|} p_{i,0}(s_q) = 1$. We refer to this constraint as the *simplex constraint*.

A simple example of this belief in the context of the tiger problem: using the definition above, $b_{i,0} \stackrel{def}{=} \langle\ p_{i,0}(TL) = 0.7,\ p_{i,0}(TR) = 0.3\ \rangle$, would mean that agent i believes with a probability of 70% that the tiger is located in the left door (TL).

At level 1, agent i's belief, $b_{i,1} \in \Delta(S \times \Theta_{j,0})$, may be rewritten using the conditional probability formula, as:

$$b_{i,1}(s, \theta_{j,0}) = p_{i,1}(s) p_{i,1}(\theta_{j,0}|s)$$

This factorization allows for a better representation of the joint probabilities, since $P_{i,1}(s), \forall s \in S$ is a discrete distribution probability over physical states.

Therefore, i's level 1 belief would be given by a vector:

$$b_{i,1} \stackrel{def}{=} \langle (p_{i,1}(s_1), p_{i,1}(\Theta_{j,0}|s_1)), (p_{i,1}(s_2), p_{i,1}(\Theta_{j,0}|s_2)), \ldots, (p_{i,1}(s_{|S|}), p_{i,1}(\Theta_{j,0}|s_{|S|})) \rangle$$

As mentioned before, the discrete distribution $\langle\ p_{i,1}(s_1), p_{i,1}(s_2), \ldots, p_{i,1}(s_{|S|})\ \rangle$ satisfies the simplex constraint, and each $p_{i,1}(\Theta_{j,0}|s_q)$ is a single density function over j's level 0 beliefs. In the case where the belief of j is unknown, this term would be a collection of densities over j's level 0 beliefs, one for each frame. Thus, $P_{i,1}(\Theta_{j,0}|s_q)$ is a continuous probability over all level 0 models of j.

An example of level 1 beliefs of agent i could be the use of mixtures of Gaussians to model each $p_{i,1}(\Theta_{j,0}|s_q)$ (given that the frame is known) and the same representation we gave in the previous example (for level 0) to model $p_{i,1}(s_q)$. Gaussians are a well known probability

distribution with very nice properties that allow manipulability and a mixture model allows for a better approximation to the real probability model than the use of a standard distribution. McLachlan [20] shows that for a sufficiently large K^q, Gaussian mixtures approximate any density to an arbitrary accuracy. Thus, K_q Gaussians are weighted to form the mixed model:

$$p_{i,1}(b_{j,0}|s_q) = \sum_{k=1}^{K^q} w^k \mathcal{N}(\mu_{i,1}^{k,q}, \Sigma_{i,1}^{k,q})(b_{j,0})$$

Each k^{th} Gaussian component of the mixture has a mean $\mu_{i,1}^{k,q}$ and a covariance $\Sigma_{i,1}^{k,q}$.

At level 2, agent i's belief, $b_{i,2} \in \Delta(S \times \Theta_{j,1})$, is a vector of the form:

$$b_{i,2} \stackrel{def}{=} \langle (p_{i,2}(s_1), p_{i,2}(\Theta_{j,1}|s_1)), (p_{i,2}(s_2), p_{i,2}(\Theta_{j,1}|s_2)), \ldots, (p_{i,2}(s_{|S|}), p_{i,2}(\Theta_{j,1}|s_{|S|}))\rangle$$

Level 2 beliefs or those with deeper nesting present some added complexity that is not present in the lower level beliefs (level 0 and 1). Representing these beliefs is challenging because level $l > 2$ beliefs are distributions over density functions whose representations doesn't need to be finite. This effect is easier to see on the example in Fig. 4.1. Say, for example, that we use a mixture of Gaussians to represent agent j's level 1 beliefs. Then, i's level 2 beliefs over j's densities are a set of functions where the variables of these functions are the parameters of the lower level densities: the lower level mixtures of Gaussians. Since these mixtures of Gaussians can take many forms (an arbitrary number of Gaussians to form the mix, and each Gaussian an arbitrary number of variables → means and covariances). At this point it may be evident for the reader that these representations are not trivial.

Perez and Doshi [24] propose how multiply nested beliefs are necessarily partial functions that fail to assign a probability to some elements (lower level beliefs) in their domains.

Proposition 1 *Agent i's multiply nested belief, $b_{i,l}$, $l \geq 2$, is strictly a partial recursive function.*

The problem with representation of nested-beliefs is that in their general form they are partial recursive functions that are not defined for every possible lower level belief in their

$b_{i,0} \in \Delta(S)$:
$b_{i,0}(s) = \langle p_{i,0}(s_1), p_{i,0}(s_2), ..., p_{i,0}(s_{|S|}) \rangle$

$b_{i,1} \in \Delta(S \times \Theta_{i,0})$:
$b_{i,1}(s, \theta_{i,0}) = b_{i,1}(s) \times b_{i,1}(\theta_{i,0}|s)$, where
$b_{i,1}(s) = \langle p_{i,1}(s_1), p_{i,1}(s_2), ..., p_{i,1}(s_{|S|}) \rangle$
$b_{i,1}(\theta_{i,0}|s) \stackrel{def}{\longleftarrow} \sum_{1..K_1^s} w_{k_1} \times N(\mu_{i,1}^{s,k_1}; \Sigma_{i,1}^{s,k_1})(\theta_{i,0})$

$b_{i,2} \in \Delta(S \times \Theta_{i,1})$:
$b_{i,2}(s, \theta_{i,1}) = b_{i,2}(s) \times b_{i,2}(\theta_{i,1}|s)$, where
$b_{i,2} = \langle p_{i,2}(s_1), p_{i,2}(s_2), ..., p_{i,2}(s_{|S|}) \rangle$

$$
\begin{aligned}
b_{i,2}(\theta_{i,1}|s) \quad & \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(\theta_{j,1}) \\
& \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2^s} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(\langle b_{j,1}, \hat{\theta}_j \rangle) \\
& \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2^s} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(b_{j,1}) \quad \text{(assuming } \hat{\theta}_j \text{ is known)} \\
& \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2^s} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(\langle b_{j,1}(s_1), b_{j,1}(\theta_{i,0}|s_1) \rangle, \ldots, \langle b_{j,1}(s_{|S|}), b_{j,1}(\theta_{i,0}|s_{|S|}) \rangle) \\
& \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2^s} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(\langle b_{j,1}(s_1), \ldots, b_{j,1}(s_{|S|}) \rangle, \langle b_{j,1}(\theta_{i,0}|s_1), \ldots, b_{j,1}(\theta_{i,0}|s_{|S|}) \rangle) \\
& \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2^s} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(b_{j,1}(S), \langle b_{j,1}(\theta_{i,0}|s_1), \ldots, b_{j,1}(\theta_{i,0}|s_{|S|}) \rangle) \\
& \stackrel{def}{\longleftarrow} \textstyle\sum_{1..K_2^s} w_{k_2} \times N(\mu_{i,2}^{s,k_2}; \Sigma_{i,2}^{s,k_2})(b_{j,1}(S), \langle K_1^{s_1}, \langle w_1^{s_1}, .., w_{K_1}^{s_1} \rangle, \\
& \langle \mu_{j,1}^{s_1,1}, \Sigma_{j,1}^{s_1,1} \rangle, .. \langle \mu_{j,1}^{s_1,K_1}, \Sigma_{j,1}^{s_1,K_1} \rangle \rangle, \ldots, \langle K_1^{s_{|S|}}, \langle w_1^{s_{|S|}}, .., w_{K_1}^{s_{|S|}} \rangle, \\
& \langle \mu_{j,1}^{s_{|S|},1}, \Sigma_{j,1}^{s_{|S|},1} \rangle, .. \langle \mu_{j,1}^{s_{|S|},K_1}, \Sigma_{j,1}^{s_{|S|},K_1} \rangle \rangle)
\end{aligned}
$$

Figure 4.1: Representation for Agent i Level 0, 1 and 2 beliefs using mixtures of Gaussians.

domains. Thus, we need to restrict their complexity to allow for computability and so that they are well-defined. A sufficient way would be to select a limited set of models of the other agents.

4.1.1 Absolute Continuity Condition

Let $\tilde{\Theta}_{j,0}$ be a *finite* set of j's computable level 0 models. Then, define $\tilde{IS}_{i,1} = S \times \tilde{\Theta}_{j,0}$ and agent i's belief, $\tilde{b}_{i,1} \in \Delta(\tilde{IS}_{i,1})$.

As we mentioned before, i's level 1 belief may be rewritten as: $\tilde{b}_{i,1}(\tilde{is}) = p_{i,1}(s) p_{i,1}(\tilde{\theta}_{j,0}|s)$. Therefore, i's level 1 belief is a vector: $\tilde{b}_{i,1} \stackrel{def}{=} \langle (p_{i,1}(s_1), p_{i,1}(\tilde{\Theta}_{j,0}|s_1)), (p_{i,1}(s_2), p_{i,1}(\tilde{\Theta}_{j,0}|s_2)), \ldots, (p_{i,1}(s_{|S|}), p_{i,1}(\tilde{\Theta}_{j,0}|s_{|S|})) \rangle$.

Here, the discrete distribution, $\langle\ p_{i,1}(s_1),\ p_{i,1}(s_2), \ldots, p_{i,1}(s_{|S|})\ \rangle$ satisfies the simplex constraint.

Additionally, each $p_{i,1}(\tilde{\Theta}_{j,0}|s_1)$ is also a discrete distribution that satisfies the simplex constraint. We generalize to level l in a straightforward manner: Let $\tilde{\Theta}_{j,l-1}$ be a finite set of j's computable level $l-1$ models. Then, define $\tilde{IS}_{i,l} = S \times \tilde{\Theta}_{j,l-1}$ and agent i's belief, $\tilde{b}_{i,l} \in \Delta(\tilde{IS}_{i,l})$. Here, analogous to a level 1 belief,
$b_{i,l} \stackrel{def}{=} \langle\ (p_{i,l}(s_1),\ p_{i,l}(\tilde{\Theta}_{j,l-1}|s_1)),\ (p_{i,l}(s_2),\ p_{i,l}(\tilde{\Theta}_{j,l-1}|s_2)),\ldots,(p_{i,l}(s_{|S|}),\ p_{i,l}(\tilde{\Theta}_{j,l-1}|s_{|S|}))\ \rangle$.

Notice that i's belief over the physical states and other's candidate models, together with its perfect information about its own model induces a predictive probability distribution over the joint future observations in the interaction [10]. The key problem with the representation of nested beliefs by using a finite set of models, is that the actual sequence of observations may not proceed along a path that is assigned some non-zero predictive probability by i's belief. In this case, i's observations may contradict its belief and a Bayesian belief update may not be possible.

Therefore, it is desirable that agent i's belief, $\tilde{b}_{i,l}$, assign a non-zero probability to each potentially realizable observation path in the interaction – this condition has also been called the truth compatibility condition [17]. We formalize this condition mathematically using the notion of *absolute continuity* of two probability measures:

Definition 1 (Absolute Continuity) *A probability measure p_1 is absolutely continuous with p_2, denoted as $p_1 \ll p_2$, if $p_2(E) = 0$ implies $p_1(E) = 0$, for any measurable set E.*

In order to formally define the condition, let ρ_0 be the true distribution over the possible observation paths induced by perfectly knowing the true models of i and j. Let $\rho_{b_{i,l}}$ be the distribution over the observation paths induced by i's initial belief, $\tilde{b}_{i,l}$. Then,

Definition 2 (Absolute Continuity Condition (ACC)) *ACC holds for an agent, say i, if $\rho_0 \ll \rho_{b_{i,l}}$.*

A sufficient but not necessary way to satisfy the ACC is for agent i to include each possible model of j in the support of its belief. However, as Proposition 1 precludes this, we

select a finite set of j's candidate models with the partial knowledge that the true model of j is one of them.

4.1.2 Bounded Interactive States

Once the belief representation problem has been sorted out, the next issue we must face is the extremely prohibitive (interactive) state space that agent i may have to consider. The main concern here is the fact that there is an infinite amount of possible models of the other agents, which makes the interactive state space too large to allow for computability of the belief update.

We will propose to use a limited set of beliefs of j to compose a *bounded interactive state space* for our analysis. We have discussed how agent i has to satisfy the *Absolute Continuity Condition*, and proposed that it suffices to select a finite set of j's candidate models with the knowledge that the true model of j is included in this set. We denote this set as $\tilde{\Theta}_{j,l-1}$.

Note that because the models of j grow as it acts and observes, agent i must track these models over time in order to act rationally. Let $\mathsf{Reach}(\tilde{\Theta}_{j,l-1}, H)$ be the set of level $l-1$ models that j could have in the course of H steps. Note that $\mathsf{Reach}(\tilde{\Theta}_{j,l-1}, 0) = \tilde{\Theta}_{j,l-1}$. In computing $\mathsf{Reach}(\cdot)$, we repeatedly update j's beliefs in the models contained in $\tilde{\Theta}_{j,l-1}$ using Eq. 3.2. We define a bounded interactive state space as follows:

$$
\begin{aligned}
&\tilde{IS}_{i,0} = S, && \tilde{\Theta}_{j,0} = \{\langle \tilde{b}_{j,0}, \hat{\theta}_j \rangle \mid \tilde{b}_{j,0} \in \Delta(\tilde{IS}_{j,0})\} \\
&\tilde{IS}_{i,1} = S \times \mathsf{Reach}(\tilde{\Theta}_{j,0}, H), && \tilde{\Theta}_{j,1} = \{\langle \tilde{b}_{j,1}, \hat{\theta}_j \rangle \mid \tilde{b}_{j,1} \in \Delta(\tilde{IS}_{j,1})\} \\
&\vdots && \vdots \\
&\tilde{IS}_{i,l} = S \times \mathsf{Reach}(\tilde{\Theta}_{j,l-1}, H), && \tilde{\Theta}_{j,l} = \{\langle \tilde{b}_{j,l}, \hat{\theta}_j \rangle \mid \tilde{b}_{j,l} \in \Delta(\tilde{IS}_{j,l})\}
\end{aligned}
$$

For each level of the nesting, we select an initial set of beliefs for the corresponding agent randomly. We show the procedure for performing this selection in Fig. 4.2.

Function GetInitialBeliefSet(Strategy level:$l \geq 0$, $\tilde{IS}_{k,l}$, # beliefs:N)

if $l = 0$ **then**
  for $n \leftarrow 1$ to N **do**
    Randomly select a distribution, $\tilde{b}_{k,0} \in \Delta(S)$
    $\tilde{B}^N_{k,0} \overset{\cup}{\leftarrow} \tilde{b}_{k,0}$
  end for
else
  GetInitialBeliefSet $(l-1, \tilde{IS}_{-k,l-1}, N)$
  for $n \leftarrow 1$ to N **do**
    Randomly select a distribution, $p_{k,l}(S) \in \Delta(S)$
    for all $s \in S$ **do**
      Randomly select a distribution, $p_{k,l}(\tilde{\Theta}_{-k,l-1}|s) \in \Delta(\tilde{\Theta}_{-k,l-1})$
    end for
    for all $s \in S$, $\tilde{\theta}_{-k,l-1} \in \tilde{\Theta}_{-k,l-1}$ **do**
      $\tilde{b}_{k,l}(s, \tilde{\theta}_{-k,l-1}) \leftarrow p_{k,l}(s) \times p_{k,l}(\tilde{\Theta}_{-k,l-1}|s)$
    end for
    Normalize $\tilde{b}_{k,l}$, $\tilde{B}^N_{k,l} \overset{\cup}{\leftarrow} \tilde{b}_{k,l}$
  end for
end if
return the belief sets, $\tilde{B}^N_{k,l}, \ldots, \tilde{B}^N_{k,0}$

Figure 4.2: A generic recursive algorithm for randomly selecting an initial set of N beliefs at all levels of the nesting. Here, k (and $-k$) assumes agent i (and j) or j (and i) as appropriate.

4.2 The Interactive Point Based Value Iteration Algorithm

In this section, we present the generalization of the point based value iteration methods for POMDPs for solving I-POMDPs. But first, we have to note a few facts about I-POMDPs. Because I-POMDPs include all possible models of other agents, and these models need to be solved in order to perform value iteration and belief updates, the curse of history is especially powerful. The curse manifests itself in the generation of the $|A_j||\Gamma_j^{t+1}|^{|\Omega_j|}$ alpha vectors during an agent j's model solution at time t (see Eqn. 3.4), and in the application of linear programming to select the optimal of those vectors that will conform a solution.

Point based approaches have been applied in POMDPs (see [26] for PBVI methods, and [35] for Perseus) and in DEC-POMDPs (see [36]) successfully. These methods utilize a finite

set of belief points and evaluate the alpha vectors generated in each epoch during value iteration to select those that will form the solution. This optimization avoids the use of linear programming, saving substantially in response time.

Fig. 4.3 shows the algorithm of the *Interactive Point Based Value Iteration (I-PBVI)* method. Parameters include the number of horizons H, the expansion period E (how many epochs should pass until the next belief expansion), the number of initial beliefs N, and the strategy level parameter of the I-POMDP, l. Note that this procedure reduces to a POMDP Point Based Value Iteration when the $l = 0$.

The first few steps are important because they implement what has been explained so far about the bounded interactive state. It is necessary to perform a sampling of the beliefs of other agents that we will consider in our interactive space. Not only so, but we may consider all reachability trees that are generated from our initial beliefs of j. Thus, our interactive state space will be formed by the combination of all physical states and all models of j composed by all reachable beliefs. Another important detail is that we could use prior knowledge about probable beliefs, to give a head start to the algorithm, instead of sampling using the algorithm in Fig.4.2.

Once we have the belief set, we will perform the backup procedure. We iterate H times, calling the backup procedure (see Fig. 4.5) every time and the belief expansion method of our choice every E epochs. We may decide to expand the initial belief set in every iteration (in which case $E = 1$), or reduce the computational overhead of the expansion method by performing it more sparsely. The details about the different expansion methods are given in the next section.

The initial alpha vectors are obtained in a manner analogous to PBVI methods in Pineau [26]. They are initialized at horizon 1, or time H, to their lower bounds: $\frac{R_{min}}{1-\gamma}$. This is a sufficient condition to ensure that the repeated backups will improve the value function. A simple algorithm is presented in Fig. 4.4.

Function I-PBVISolveModel(Horizon:$H > 0$, Expansion Period:$E > 0$, # Beliefs:$N > 0$, Strategy level:$l \geq 0$).

if $l > 0$ **then**
  $B_{-k,l-1} \leftarrow$ *GetInitialBeliefSet*$(N, l-1)$
  $B^{reach}_{-k,l-1} \leftarrow$ *GetOtherAgentSetofReachableBeliefTrees*$(B_{-k,l-1}, H)$
  $B_k \leftarrow$ *GetInitialBeliefSet*$(B_{-k,l-1}, N, l \geq 0)$
else
  $B_k \leftarrow$ *GetInitialBeliefSet*$(N, l \geq 0)$
end if
$\Gamma_0 \leftarrow$ *GetInitialAlphaVectors*()
for $t \leftarrow 1$ to H **do**
  if $l = 0$ **then**
    $\Gamma_t \leftarrow$ *PBVIBackup*(B_k, Γ_{t-1})
  else
    $\Gamma_t \leftarrow$ *I-PBVIBackup*$(B_k, \Gamma_{t-1}, H, E, N, l)$
  end if
  Expand the previous set of beliefs at all levels using techniques from Section 4.3
  Add the expanded beliefs to the existing sets
end for
return Γ_t

Figure 4.3: I-PBVI Algorithm for model solution.

The I-PBVI backup procedure is shown in Fig. 4.5. We defined a bounded interactive state in section 4.1.2. Then, equations 3.6 and 3.7 have to be rewritten as:

$$\tilde{\Gamma}^{a_i,*} \leftarrow \alpha^{a_i,*}(\tilde{is}) = \sum_{a_j \in A_j} R(s, a_i, a_j) Pr(a_j | \tilde{\theta}_{j,l-1}) \tag{4.1}$$

$$\begin{aligned} \tilde{\Gamma}^{a_i,o_i} \overset{\cup}{\leftarrow} \quad & \alpha^{a_i,o_i}(\tilde{is}) = \gamma \sum_{\tilde{is}'} \sum_{a_j} Pr(a_j | \tilde{\theta}_{j,l-1}) T_i(s, a_i, a_j, s') O_i(s', a_i, a_j, o_i) \\ & \sum_{o_j} O_j(s', a_i, a_j, o_j) \delta_D(SE_{\widehat{\theta}_j}(\tilde{b}_{j,l-1}, a_j, o_j) - \tilde{b}'_{j,l-1})\ \alpha^{t+1}(\tilde{is}'), \quad \forall \alpha^{t+1} \in \Gamma_{t+1} \end{aligned} \tag{4.2}$$

where $\tilde{is}, \tilde{is}' \in \tilde{IS}_{i,l}$ and $\tilde{is} = \langle s, \tilde{\theta}_{j,l-1} \rangle$, $\forall a_i \in A_i$ and $o_i \in \Omega_i$.

The I-PBVI Method considers a belief set $B_{i,l}$, a finite set of level l belief points at some time t. This set will provide us with the evaluation points for the alpha vector generated

Function GetInitialAlphaVectors($l \geq 0$)
1: **if** $l > 0$ **then**
2: **for all** $is \in IS'$ **do**
3: $\alpha_0(is) \leftarrow \frac{R_{min}}{1-\gamma}$
4: **end for**
5: **else**
6: **for all** $s \in S$ **do**
7: $\alpha_0(s) \leftarrow \frac{R_{min}}{1-\gamma}$
8: **end for**
9: **end if**
10: $\Gamma_0 \leftarrow \alpha_0$
11: **return** Γ_0

Figure 4.4: Generic algorithm for initial alpha vector selection.

during each iteration, since we will keep those that are optimal at these points. The cross-sum operation (shown in Eq. 3.8) may be simplified by not considering all the vectors in the set Γ^{a_i,o_i^1}, but only those that are optimal at some belief point, $b_{i,l} \in B_{i,l}$ (Eq. 4.3).

$$\tilde{\Gamma}^{a_i} \leftarrow \tilde{\Gamma}^{a_i,*} \bigoplus_{o_i \in \Omega_i} \underset{\tilde{\Gamma}^{a_i,o_i}}{argmax} \left(\alpha^{a_i,o_i} \cdot b_{i,l}\right) \quad \forall b_{i,l} \in B_{i,l} \tag{4.3}$$

We use $B_{i,l}$ to select the alpha vectors that form the set Γ^t:

$$\Gamma^t \leftarrow \underset{\alpha^t \in \bigcup_{a_i} \Gamma^{a_i}}{argmax} \left(\alpha^t \cdot b_{i,l}\right) \qquad \forall b_{i,l} \in B_{i,l}$$

The difference between this method and the exact approach is evident. With Eq. 4.3, we generate at most $\mathcal{O}(|A_i||\Gamma^{t+1}|^{|\Omega_i|})$ alpha vectors (in practice usually less). Furthermore, an LP is not required to select the optimal vectors, since the set Γ^t contains unique alpha vectors that are optimal for at least one of the belief points in $B_{i,l}$. Hence, Γ^t contains at most $|B_{i,l}|$ many alpha vectors. Because the number of alpha vectors depends on the set of belief points, we may limit the latter to a constant size.

Finally, we have to compute the term $Pr(a_j|\tilde{\theta}_{j,l-1})$ in Eqs. 4.1 and 4.2, which is the action of the other agent(s). To obtain this term we have to solve agent j's I-POMDP (for levels

I-PBVI Backup ($\langle \tilde{B}_{k,l}, \ldots, \tilde{B}_{k,0} \rangle$, $\tilde{\Gamma}_k^{t+1}$, h, l)

1: $\tilde{\Gamma}_{-k}^{t} \leftarrow$ I-PBVI ($\langle \tilde{B}_{-k,l-1}, \ldots, \tilde{B}_{k,0} \rangle$, h, $l-1$)
2: **for all** $a_k \in A_k$ **do**
3: Compute $\alpha_k^{a_i,*}$ (Eq. 4.1) where $Pr(a_{-k}|\tilde{\theta}_{-k,l-1}) \leftarrow$ GetAction ($\tilde{\theta}_{-k,l-1}, \tilde{\Gamma}_{-k}^{t+1}$) and add $\alpha_k^{a_i,*}$ to $\tilde{\Gamma}^{a_i,*}$
4: **for all** $o_k \in \Omega_k$ **do**
5: Compute $\alpha_k^{a_i,o_i}$ (Eq. 4.2), where $Pr(a_{-k}|\tilde{\theta}_{-k,l-1}) \leftarrow$ GetAction ($\tilde{\theta}_{-k,l-1}$, $\tilde{\Gamma}_{-k}^{t+1}$), add $\alpha_k^{a_i,o_i}$ to $\tilde{\Gamma}^{a_i,o_i}$
6: **end for**
7: **end for**
8: **for all** $\tilde{b}_{k,l} \in \tilde{B}_{k,l}$ **do**
9: Compute $\alpha_k^{a_i}$ (Eq. 4.3) and add $\alpha_k^{a_i}$ to $\tilde{\Gamma}^{a_i}$
10: **end for**
11: $\tilde{\Gamma}^{t} \leftarrow \bigcup_{a_i} \tilde{\Gamma}^{a_i}$
12: **for all** $\tilde{b}_{k,l} \in \tilde{B}_{k,l}$ **do**
13: $\alpha_k^* \leftarrow \underset{\alpha_k \in \tilde{\Gamma}^n}{argmax}\ \alpha_k \cdot \tilde{b}_{i,l}$
14: **if** $\alpha_k^* \notin \tilde{\Gamma}_*^t$ **then**
15: Add α_k^* to $\tilde{\Gamma}_*^t$
16: **end if**
17: **end for**
18: **return** $\tilde{\Gamma}_*^t$

Figure 4.5: I-PBVI Value Backup Algorithm.

> 1) or POMDP (for level $= 1$) in an analogous way. Therefore, we need to recurse through all levels of nesting, using a new set of belief points at each level to generate the alpha vectors part of j's solution.

4.3 Belief Expansion methods

Next, we need to select the expansion methods to implement in the new I-PBVI algorithm. We have revised some of the literature available in chapter 2, and we will take into account three issues that will drive the performance of the whole method:

- An agent's beliefs often follows certain trajectories. Selecting belief points that lie on the trajectories may result in good performance solutions.

- Selecting a belief point that is in proximity to another belief point already in the set may not result in a new alpha vector in the solution. This belief point would be redundant in the set.

- In comparison to single agent settings, generating beliefs in a setting populated by other agents may require predicting their actions as well.

Some discussion has been made in Chapter 2 about previous publications on point based methods. A method we haven't described yet called Perseus [35], utilizes a fixed set of beliefs obtained by randomly exploring the environment. During the backups, they progressively filter out the belief points considering only those for which the previously back projected alpha vectors are not a better policy. Pineau et al. results [26] have been discussed before in Chapter 2, and include the performance of several belief expansion methods they propose. Among these methods, the greedy error reduction algorithm is a steepest ascent algorithm that leads the direction of the search of a new belief towards the one that offers the greatest error reduction. James et al. [15] results have also been discussed before. Their methods incrementally introduce belief points that have the potential of providing the largest gain, where gain is the difference between the current value of the policy at that point as obtained from previously selected alpha vectors and a minimal upper bound.

While deciding what methods to implement for our purposes, we had to rule out the methods presented in [15]. As the authors conclude, finding the minimal upper bound needed for the methods is computationally expensive and for large belief spaces like the interactive states of I-POMDPs it is expected to offset the runtime savings provided by the point based approach. Thus, we utilize two approaches to expand the sets of belief points over time that are used to select the alpha vectors:

- **Stochastic trajectory simulation:** For each belief in a belief set, $\tilde{B}_{i,l}$, we sample a physical state and the other agent's model. We then uniformly sample i's action, a_i, and in combination with the sampled physical state and j's action obtained from

solving j's model, we sample the next physical state using the transition function. Given the updated physical state and joint actions, we sample an observation of i, o_i, from the observation function. Agent i's belief is then updated given its action, a_i, and observation, o_i, using the belief update (Eq. 3.2). The algorithm for this method is given in Fig. 4.6.

- **Greedy error minimization:** the performance of a point based method depends heavily on the density of the set of belief points. It is important to generate new belief points $b_{i,l}^{t+1}$, such that the optimal alpha vector at that point is furthest in value from the alpha vector at an existing belief that is the closest to the generated belief. This is because in the absence of such a point, a large error would be incurred at that point. Since the optimal alpha vector at $b_{i,l}^{t+1}$ is not known before applying the I-PBVI method, we utilize the maximum (or minimum) value, $\frac{R_{max}}{1-\gamma}$ for each interactive state is, instead. The algorithm for this method is given in Fig. 4.7.

These approaches are similar to those used in [26] for expanding beliefs in point based methods in the context of single agent POMDPs, and they demonstrated good results. As explained before, the error minimization showed the best for POMDPs performance [26], improving on Perseus [35] as well. Note also that the recursion of the I-PBVI method guarantees the expansion of beliefs at all strategy levels.

4.4 Computational Savings

If the strategy level is 0, the I-POMDP$_i$ collapses into a POMDP and we generate in the worst case $\mathcal{O}(|A_i||\Gamma^{t+1}|^{|\Omega_i|})$ many alpha vectors at time t in order to solve the POMDP exactly. Let $M_{j,l-1} = \mathsf{Reach}(\tilde{\Theta}_{j,l-1}, H)$. We first consider solving the I-POMDP of i at level 1. Because we include $|M_{j,0}|$ many models of j in the state space, we need obtain $|M_{j,0}|$ alpha vectors assuming j's frame is known. These are used in solving the I-POMDP of i, which in the worst case generates $\mathcal{O}(|A_i||\Gamma^{t+1}|^{|\Omega_i|})$ vectors. Note that these vectors are of size $|\tilde{IS}_{i,l}|$

Function I-PBVIExpandStochSim$(B_k, \Gamma_k, l > 0)$

$B_{k,new} \leftarrow B_k$
for all $b_k^{(n)} \in B_k$ **do**
  $is_k \leftarrow random_{multinomial}(b_k^{(n)})$
  $a_k^{t-1} \leftarrow random_{uniform}(A)$
  $is'_k \leftarrow random_{multinomial}(T(s_k, a^{t-1}, \bullet))$
  $o_k^t \leftarrow random_{nultinomial}(O(s'_k, a^{t-1}, \bullet))$
  for all $is^t = \langle s^t, \theta^t_{-k} \rangle \in IS$ **do**
    $b_{k,new}^{(n)}(is^t) \leftarrow Pr(is^t | o_k^t, a_k^{t-1}, b_k^{(n)})$
  end for
  $B_{k,new} \overset{\cup}{\leftarrow} b_{k,new}^{(n)}$
end for
return $B_{k,new}$

Figure 4.6: Algorithm for belief expasion with the *stochastic trajectory simulation* method. $Pr(is^t|o, a, b)$ is detailed in Eqn.3.2

Function I-PBVIExpandGreedyEM$(B_k, \Gamma_k, l > 0)$

$B_{k,new} \leftarrow B_k$
for all $b_k^{(n)} \in B_k$ **do**
  $\tilde{b}_{k,l}^{t-1}, \tilde{a}_k^{t-1} \leftarrow argmax_{b \in B, a \in A}(\sum_{o \in \Omega} O(b, a, o) \epsilon(Pr(is^t|o, a, b)))$
  $\tilde{o}_k^t \leftarrow argmax_{o \in \Omega}(O(\tilde{b}_{k,l}^{t-1}, \tilde{a}_k^{t-1}, o) \times \epsilon(Pr(is^t|o, \tilde{a}_k^{t-1}, \tilde{b}_{k,l}^{t-1})))$
  for all $is^{t-1} = \langle s^{t-1}, \theta^{t-1}_{-k} \rangle \in IS$ **do**
    $b_{k,new}^{(n)}(is^t) \leftarrow Pr(is^t | \tilde{o}_k^t, \tilde{a}_k^{t-1}, \tilde{b}_{k,l}^{t-1})$
  end for
  $B_{k,new} \overset{\cup}{\leftarrow} b_{k,new}^{(n)}$
end for
return $B_{k,new}$

Figure 4.7: Algorithm for belief expasion with the *greedy error minimization* method. $Pr(is^t|o, a, b)$ is detailed in 3.2, and $\epsilon(\bullet)$ is defined in Eqn. 2.10

compared to size $|S|$ of the vectors for POMDPs. Thus, a total of $\mathcal{O}(|A_i||\Gamma^{t+1}|^{|\Omega_i|} + |M_{j,0}|)$ alpha vectors are obtained at level 1. Generalizing to level l and assuming, for the sake of simplicity, that the same number of models of the other agent are included at any level, $|M|$, we need $\mathcal{O}(|A_i||\Gamma^{t+1}|^{|\Omega_i|}+|M|l)$ alpha vectors to solve the I-POMDP$_{i,l}$ exactly. In the context of the I-PBVI, if at most N belief points are used at any level, the approximate solution of a level 0 I-POMDP generates $\mathcal{O}(N)$ alpha vectors. For level 1, because solutions of $|M|$ models are obtained approximately using N belief points, we need to obtain only $\mathcal{O}(N)$ vectors for j and another $\mathcal{O}(N)$ vectors to solve the I-POMDP of i at level 1 approximately. Generalizing to level l, we generate at most $\mathcal{O}(N(l+1))$ many alpha vectors. For the case where $N << |M|$, significant computational savings are obtained. Of course, for more than two agents, the number of alpha vectors is exponential in the number of agents.

The loss in optimality or error due to approximately solving the I-POMDP using I-PBVI is due to two reasons:

- The alpha vectors that are optimal at selected belief points may be suboptimal at other points.
- Models of the other agent are solved approximately as well.

4.5 Summary

The interactive point based value iteration is a novel extension of the PBVI methods for approximately solving I-POMDPs. The extension is not straightforward because we must face the complexity of interactive state space and we must solve other agent models during the value iteration process. To make solutions computationally feasible, we bound the interactive state space to a finite set of models of the other agents. To solve the other agents' models, we do a recursive call that bottoms out at the zero strategy level I-POMDP. The computational savings are significant compared to the exact solution, and we expect to show improvement

in handling the curse of history, which is the main limitation of competing approximation methods.

Chapter 5

Empirical Performance

In the next chapter, we will present some empirical results of the proposed method. I-PBVI was tested on several problems of the literature of I-POMDPs. The experimental set-up is described first, on section 5.1. Sections 5.2 and 5.3 present the results for I-POMDPs levels 1 and 2 on the tiger and machine maintenance multiagent problem, and level 1 UAV reconnaissance problem, respectively. Some discussion is included in each section.

5.1 Experimental Design

The new algorithm was implemented in C++ and compiled and run on a Linux platform with dual processor Xeon 3.4GHz with 4GB memory. The simulations were performed on two known problems in the literature: The multiagent tiger problem, presented in [13], and a multiagent version of the machine maintenance (MM) problem, presented in [33]. These problems are popular but relatively small, having a physical state space size of 2 and 3 respectively. But keep in mind that for interactive states we must consider all possible models of other agents. Therefore, the interactive state space is considerably bigger.

Additionally, Perez and Doshi [24] presented a new problem domain inspired on Unmanned Aerial Vehicles (UAV). This scenario is a competitive game where the agent has to find a target under noisy communications, while avoiding being spotted by other hostile agents. This problem is much bigger, and a few results are presented. The main issue when dealing with larger state space problems like this is that the competing method (the IPF) can't handle many horizons due to limitations with memory.

Another difficulty arose when trying to develop a method for comparison among the main approximation method actually available for I-POMDPs (the IPF) and the new proposed method (I-PBVI). Since the IPF is an online method (it starts with a initial belief of i and projects it forward in time, building a policy given this initial belief of i) and the new I-PBVI method is an offline method (it generates a policy (solution) for any initial beliefs of i and j), we had to come up with a fair way of comparing them. Our proposed simulation is the comparison of performance vs. time for a fixed number of solutions. Say for example, that we need to get different policies for 10 possible initial beliefs of i on a given problem. We will have to run the IPF 10 times to obtain this result. On the other hand, the I-PBVI method only needs to run once to obtain the (approximate) solution to all possible initial beliefs. We generate policy trees in IPF for as many initial beliefs of i as the number of belief points used in I-PBVI.

We gradually increased the number of horizons, initial belief points and models and simulated the performance of the resulting policies over 10 trials of 50 runs each, and the results were averaged out. Since we also need to simulate a competing agent j, we used POMDP value iteration for level 1 exercises (j is a level 0 I-POMDP: a POMDP), and the interactive particle filter for level 2 exercises (j is a level 1 I-POMDP).

Two key variables have to be set up when running the I-PBVI algorithm. First, as discussed in Chapter 4, the number of initials beliefs of j had to be set up to a fixed value. For these experiments, we used 5 initial beliefs of j. Notice that we generate the reachability tree (which includes all reachable beliefs) up to horizon H, a parameter of the I-PBVI method. Thus, the interactive space grows in every time step proportional to the branching factor of the trees times the number of initial beliefs. Second, the number of initial beliefs of i that will be sampled had to be set. For the purposes of the simulations we use 10 initial beliefs.

Notice also that even though we are giving information about some of the parameters used in the simulations, our main measurement of performance is time vs. average rewards.

Thus, a "significant" result would be that I-PBVI performs better in time than the IPF, and at the same time achieves better performance in terms of average rewards.

5.2 The multiagent tiger problem and the multiagent machine maintenance problem

We utilize the *multiagent tiger problem* presented in [13]. It differs from other multiagent versions of the same problem [23] by assuming that the agent not only hear growls to know about the location of the tiger, but also hear creaks that tell him if the other agent may have opened a door. Creaks indicate which door was opened by the other agent. Furthermore, agents need not have the same payoffs. 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)). Both agents, i and j, hear growls with a reliability of 85% and creaks with a reliability of 95%. Each agent's preferences are as in the single agent game, having -100 for opening the door with the tiger, +10 for opening the door with the prize, and -1 for listening.

On the other hand, the *multiagent machine maintenance problem* (MM) [13] is a multiagent variation of the original machine maintenance problem presented in [33]. In this version, we have two agents that cooperate. The non-determinism of the original problem is increased to make it more realistic, allowing for more interesting policy structures when solved. The problem has three physical states; the possible actions are $A = A_i \times A_j$, where $A_i = A_j = M, E, I, R$; and the observation space for both agents is either the machine is *defective* or *non-defective*.

The results of level 1 experiments on both problems are presented in Fig. 5.1 and Fig. 5.2 for the Tiger and MM. We look for lower values on the *y-axis*, since this is the time taken for computing the policies. It is noticeable how the greedy error minimization (labeled *IPBVI+Min* in the plot) improves in results to the stochastic trajectory simulation (labeled

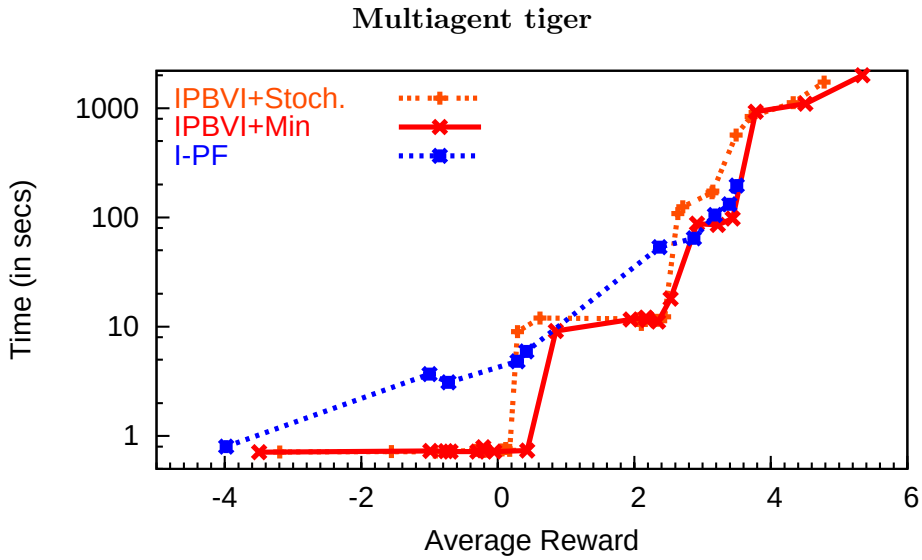


Figure 5.1: Level 1 (j's models are POMDPs) plot of time consumed in achieving a desired performance on the *Multiagent Tiger problem*. Note that the y-axis is in log scale. The I-PBVI significantly improves on the I-PF.

IPBVI+Stoch in the plot): it takes less time for the I-PBVI with the first expansion method to reach the same performance (rewards) than the other method. These results are expected, analogous to the performance of the different expansion techniques presented by Pineau for POMDPs [26]. Both clearly improve over the IPF results.

At level 2 (see Fig. 5.3 and Fig. 5.4, for Tiger and Machine Maintenance multiagent problems, respectively), the differences exhibited in the previous results are not so evident. This is due to the fact that level 2 I-POMDPs are computationally more expensive to solve, and the extra computations incurred on calculating the error minimization start to gain significance. When we compare results from level 2 to level 1, there is not much change in terms of performance (rewards). Solutions to level 2 I-POMDPs are computationally more demanding, as can be seen in Figs. 5.3 and 5.4. We carry out this comparison by holding constant the number of initial beliefs and comparing the average rewards obtained at each level. This leads to a conclusion that higher level I-POMDPs need more starting belief points

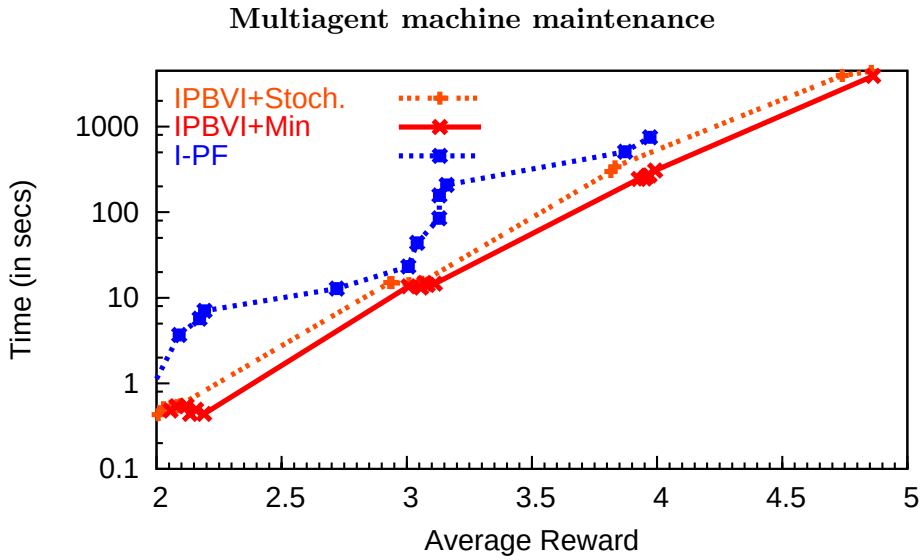


Figure 5.2: Level 1 (j's models are POMDPs) plot of time consumed in achieving a desired performance on the *Multiagent Machine Maintenance problem*. Note that the y-axis is in log scale. The I-PBVI significantly improves on the I-PF.

to improve performance over lower level I-POMDPs when using point based methods to solve them.

As has been pointed out before, due to an absence of other offline approximation techniques for I-POMDPs, we compared the performance of I-PBVI with the interactive particle filter (IPF) based approximation [9]. One characteristic of the IPF is its ability to mitigate the curse of dimensionality; but the IPF fails to address the curse of history since it must generate the full reachability tree to compute the policy. The better performance of I-PBVI in comparison to I-PF demonstrates that point based value iteration is able to mitigate the impact of the curse of history that affects the solutions of both the agents' decision processes. Furthermore, we were unable to run the IPF beyond a few time horizons due to excessive memory consumption.

Even though the I-PBVI method improves over the IPF, the I-PBVI method also reaches a limit of calculation horizon-wise. There is a point where calculation of a solution for horizon

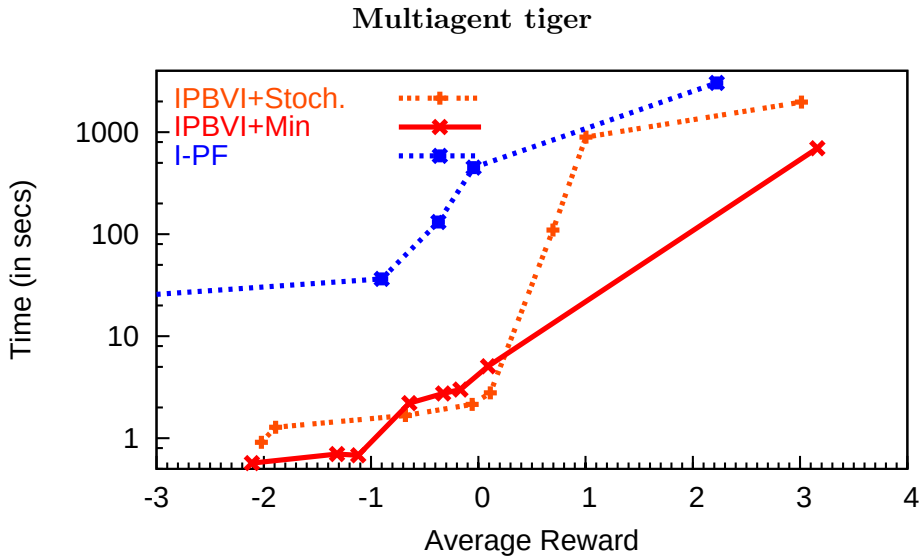


Figure 5.3: Level 2 (j's models are level 1 I-POMDPs) plot of time consumed in achieving a desired performance on the *Multiagent Tiger problem*. Note that the y-axis is in log scale. The I-PBVI significantly improves on the I-PF at this level too.

H is not possible, for large H. This gives us an idea that the curse of history is not as efficiently resolved as we may expect from the results of this class of methods in the POMDP domain. Analyzing possible places in the algorithm that may be causing this effect, we concluded that the exponential growth of the interactive state, even when we sample a few initial beliefs of j, is causing the problems.

A final comment must be done about this issue. Our implementation of the method included the dimension of all interactive states reachable through the backups (all models of j reachable at all horizons) as the dimension of the alpha vectors. When H is sufficiently large, the vast amount of vectors we generate may fill up the memory. A dynamic approach during implementation, would make the alpha vectors and beliefs of different size for different horizons, and may be the solution for the memory problems in the programs. Nevertheless, as we claim in the previous paragraph, the curse of history is not completely solved. Thus,

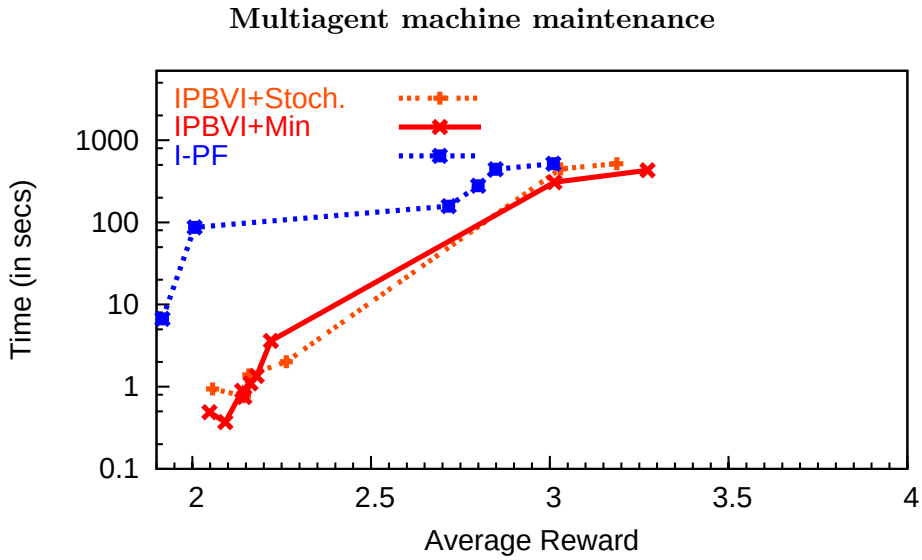


Figure 5.4: Level 2 (j's models are level 1 I-POMDPs) plot of time consumed in achieving a desired performance on the *Multiagent Machine Maintenance problem*. Note that the y-axis is in log scale. The I-PBVI significantly improves on the I-PF at this level too.

even with this change in the handling of the interactive spaces in the implementation, we wouldn't still be able to solve problems for much larger horizons.

5.3 The UAV Reconnaissance problem

The UAV Reconnaissance problem was first introduced in [24]. The problem proposes a scenario where other agents have antagonistic preferences. The objective of the agent is to perform a low altitude reconnaissance of a potentially hostile theater that may be populated by other agents with conflicting objectives and ground reconnaissance targets (see Fig. 5.5). The task is further complicated because the agent may possess noisy sensors and unreliable actuators.

The analysis is carried out on a 4×4 grid (16 sectors) and one ground reconnaissance target (T) is located in one of them. Of course, the agent's goal is to locate the target that

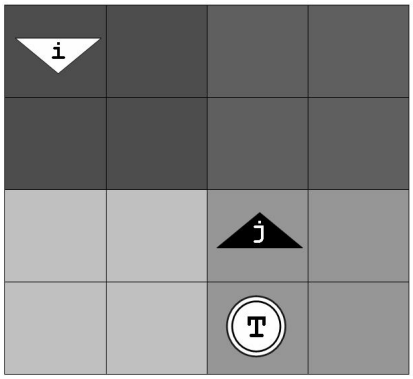


Figure 5.5: UAV Reconnaissance problem, in a grid of 4 x 4 sectors.

may be hidden in any of the sectors. To assist in its goal of locating the target, he receives a noisy communication with information about the quadrants (similar colored sectors in Fig. 5.5) but with some uncertainty. The actions that the UAV can perform after receiving these observations are either to move to any of the other 15 sectors or to hover in its current location waiting for new communications. The other agent in the theater, j, is hostile and has the option of informing T in any of the sectors that it is in danger of being spotted by the UAV, in which case T moves to an adjacent or diagonal sector. The other agent is also unaware of the location of T, receives its own communication informing it of where the target may be located, though with some uncertainty. Note that the actions of both, the UAV and agent j may affect the physical state of the problem.

The UAV problem is described below:

- A physical state, s=$\{sector_T, spotted_T\}$, where $sector_T$ and $spotted_T$ indicate the sector location of target T and whether T has been spotted, respectively.

- The joint action space, $A = A_i \times A_j$, where $A_i = \{move_{1,1},\ldots,move_{4,4},listen\}$ and $A_j = \{inform_{1,1},\ldots, inform_{4,4},listen\}$. Here, $move_{x,y}$ moves the UAV to the sector indexed

by (x, y), $inform_{x,y}$ informs the safe house in sector (x, y) that the UAV is overhead, and *listen* denotes the act of receiving communications about the location of T;

- Observation space of the UAV is, $\Omega_i = \{$top-left(TL), top-right(TR), bottom-left(BL), bottom-right(BR)$\}$, where for example, TL indicates that the corresponding target is in one of the four sectors in the top left quadrant;

- Transition function is, $T_i : S \times A \times S \rightarrow [0, 1]$. T_i models the fact that if the UAV moves to a sector containing a target, T's status is updated to being spotted, and if the other agent j informs a sector containing a target of danger, then T reappears in any of the surrounding sectors. If the UAV moves to a sector containing a target that has been informed by j, we assume that the target is spotted;

- Observation function, $O_i : S \times A \times \Omega_i \rightarrow [0, 1]$ gives the likelihood that the UAV will be informed of the correct quadrant in which the target is located;

- Reward function, $R_i : S \times A \rightarrow [0, 1]$ formalizes the goal of spotting the reconnaissance target.

In this scenario, the fact that I-POMDPs keep track of other agents beliefs becomes extremely useful. Since j's actions may cause a target to move, it is important for i to keep track and anticipate j's actions. We assume that the UAV is aware of j's conflicting objectives, the probabilities of its observations, and therefore j's frame. A level 1 solution of the problem is presented in Fig. Level1UAV. As expected, policies larger than a few horizons are needed to spot the dynamic target. Another thing to notice is that the IPF couldn't go beyond horizon 3 for this problem, due to the enormous consumption of memory that the large physical state space and large branching factor of the reachability tree in this problem involves. The I-PBVI method, on the other hand, provided some good results for larger horizons.

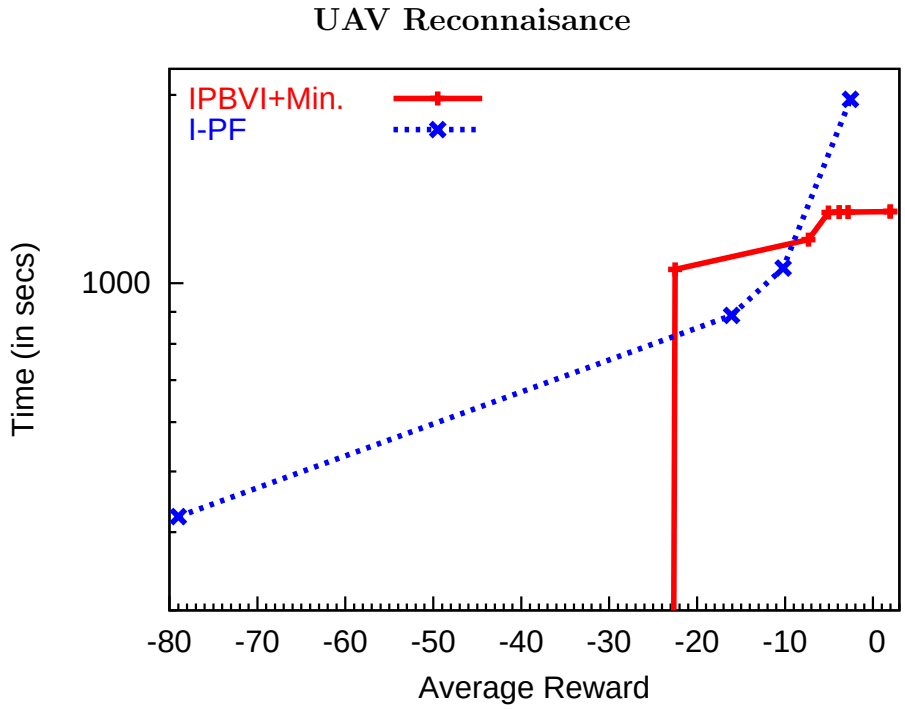


Figure 5.6: Performance of the level 1 I-PBVI on the UAV Reconnaisance problem.

5.4 Summary

The simulations included in this chapter show how I-PBVI methods reduce the effect of the curse of history when compared to competing approximation methods for I-POMDPs. I-PBVI shows better results in time efficiency and in horizon length for different problems of the literature. Among I-PBVI expansion methods, the greedy error reduction improves over stochastic selection of actions. This difference is more evident for level 1 I-POMDPs. From these results we can conclude that at this level, the overhead in computations added by calculating error bounds for the first method is paid off by a faster increase in value.

Chapter 6

Conclusion

The last chapter of this thesis will summarize our achievements, discuss some findings and outline some avenues for future work. Section 6.1 present some conclusions obtained from the development process of the new method. Finally, section 6.2 presents some ideas that may lead to future research in the approximation methods for I-POMDPs.

6.1 Conclusions

In this thesis, we present a new approximation method, called *Interactive point-based value iteration* (I-PBVI), to solve interactive partially observable Markov decision processes (I-POMDPs). This is an approximation algorithm that allows an agent to plan sequentially over a multiagent scenario, either cooperative or competitive. The main idea in the development of the method was to improve current approximation techniques to solving I-POMDPs by providing one that is less affected by the curse of history than current available options. From the literature of POMDPs, point based value iteration methods provide good results over large states/horizons problems, resolving the limitations of the curse of history in these kinds of problems. Thus, our approach is a generalization of these methods to the I-POMDP arena.

We also present a specification of the belief representation for nested beliefs in I-POMDPs. We show an implementation of this specification using mixtures of Gaussians, statistical models that provide a (demonstrated) adequate representation for probability distributions.

Finally, we provide some empirical results of the performance of the new method, compared to the Interactive particle filter. Some conclusions could be gathered from these experiments, as well as during the design and implementation of the IPBVI method, and are presented below:

- The Interactive Point Based Value Iteration generalizes point based approaches for POMDPs to the I-POMDP framework. The generalization solves I-POMDPs and provide offline approximate solutions in the form of alpha vectors (policies for any initial belief of i).

- The sampling of beliefs to look for solutions proposed in I-PBVI mitigates the curse of history, allowing the method to solve problems with larger state space for larger horizons than the existing methods for I-POMDPs.

- I-PBVI improves over the IPF in mitigating the curse of history, but there are still more opportunities for improvement. Even though larger horizon problems can be calculated with I-PBVI, the maximum horizon problem that can be solved is still not very large for large states problems. The cause may be the interactive state space itself.

- The results we showed in I-PBVI are not conclusive of the maximum capability of the method. Some improvements can be done in order to increase the maximum horizon policies that the method can solve.

As a final remark, we must be critical of our own method. I-PBVI improves over actual approximation methods, but is still not completely mature. Some work could be done in the areas of selection of initial beliefs of j to aid in improving performance. The next section will develop on some other ideas for further improvement.

6.2 Future work

We want to conclude this chapter presenting a few ideas that may provide avenues to further work posterior to the publication of this thesis. It is clear that I-POMDP researchers have to

focus in continuing improving approximation methods that reduce error bounds and tackle more efficiently both curses of dimensionality and history. Furthermore, improvements in the implementation (programming) of these methods may be performed. Due to the large consumption of memory that this method entails, some techniques to mitigate the impact of memory management on the programs could improve results.

First of all, our understanding of the implications of the assumptions made here for the I-PBVI method to be possible has to be further improved. It is not clear at this point what would be the result of a simulation if the true model of agent j is not included in the set of initial models of j when we solve the I-POMDP for agent i. An empirical and theoretical analysis could be performed to aid in the understanding of the implications of having a scenario where this assumption is not met. This would provide us with an evaluation of the robustness of this method when assumptions are not met.

Another point of improvement for this method and the whole I-POMDP framework is how to tackle effectively the growth of the interactive state over time. Since the interactive state size depends on the number of models of other agents that we want to include in our solution, the space will grow exponentially as we increase the horizon. Thus, when we look for larger horizons (which is desirable) we have a huge interactive space composed by models of j that include all beliefs in the reachability trees that are generated from the initial belief set of j for the horizon H. It is easy to see how for large state spaces, this growth can be prohibitive. Perhaps a clustering of all possible policies could help in keeping the interactive state growth to a constant size, but such an implementation is not trivial and needs some solid theoretical grounds to accompany it.

Finally, we look to other possible approximate methods. The Interactive particle filter is a powerful tool for solving I-POMDPs online, and tackles fairly well the curse of dimensionality, but fails to address the curse of history. One of the main problems when executing the particle filter are the recursive calls to solve models of j. It would be interesting to develop an algorithm that mixes the strengths of I-PBVI to tackle the curse of history, and the IPF.

Another approach could use an already solved policy of j in I-PBVI to predict the actions of it in the IPF.

Bibliography

[1] R. J. Aumann. Interactive epistemology i: Knowledge. *International Journal of Game Theory*, 28:263–300, 1999.

[2] J. Boger, P. Poupart, J. Hoey, C. Boutilier, G. Fernie and A. Mihailidis. A decision-theoretic approach to task assistance for persons with dementia. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1293–1299, Edinburgh, Scotland, 2005.

[3] C. Boutilier. A POMDP formulation of preference elicitation problems. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, 239–246, Edmonton, AB, 2002.

[4] A. Brandenburger and E. Dekel. Hierarchies of beliefs and common knowledge. *Journal of Economic Theory*, 59:189–198, 1993.

[5] A. R. Cassandra, L. P. Kaelbling and J. A. Kurien. Acting under uncertainty: discrete Bayesian models for mobile robot navigation. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 963–972, 1996.

[6] A. R. Cassandra, M. L. Littman, and N. L. Zhang. Incremental pruning: A simple, fast, exact method for partially observable markov decision processes. In *Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence (UAI-97)*, 54–61, San Francisco, CA, 1997.

[7] H. T. Cheng. *Algorithms for Partially Observable Markov Decision Processes.* PhD Thesis. University of British Columbia, 1988.

[8] T. Darrell and A. Pentland. Active Gesture Recognition using partially observable Markov decision processes. In *IEEE International Conference on Pattern Recognition*, 984–988, Vienna, Austria, 1996.

[9] P. Doshi and P. J. Gmytrasiewicz. Approximating state estimation in multiagent settings using particle filters. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pages 320–327, 2005.

[10] P. Doshi and P. J. Gmytrasiewicz. On the difficulty of achieving equilibrium in interactive pomdps. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 1131–1136, Menlo Park, CA, 2006.

[11] A. Doucet, N. D. Freitas and N. Gordon. *Sequential Montecarlo Methods in Practice*. Springer Verlag.

[12] D. Fudenberg and D. K. Levine. *The Theory of Learning in Games*. MIT Press, 1998.

[13] P. Gmytrasiewicz and P. Doshi. A framework for sequential planning in multiagent settings. *Journal of Artificial Intelligence Research (JAIR)*, 24:49–79, 2005.

[14] E. Hansen, D. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, 709–715, 2004.

[15] M. James, M. Samples, and D. Dolgov. Improving anytime point based value iteration using principled point selections. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 865–870, 2007.

[16] L. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[17] E. Kalai and E. Lehrer. Rational learning leads to nash equilibrium. *Econometrica*, 61(5):1019–1045, 1993.

[18] M. Li and P. Vitanyi. *An Introduction to Kolmogorov Complexity and its Applications.* Springer, 1997.

[19] M. Littman, A. Cassandra and L. Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning (ICML)*, 1995.

[20] G. J. McLachlan and K. E. Basford. *Mixture Models: Inference and Applications to Clustering.* Marcel Dekker, New York, 1988.

[21] G. E. Monahan. A survey of partially observable markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[22] M. Montemerlo, J. Pineau, N. Roy, S. Thrun and V. Verma. Experiences with a mobile robotic guide for the elderly. In *Proceedings of the National Conference on Artificial Intelligence*, 587–592, Edmonton, AB, 2002.

[23] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella. Taming decentralized pomdps : Towards efficient policy computation for multiagent settings. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

[24] D. D. Perez and P. Doshi. Approximate Solutions of Interactive POMDPs Using Point Based Value Iteration. In *Proceedings of the International Symposium on Artificial Intelligence and Mathematics*, Ft.Lauderdale, Florida, 2008.

[25] J. Pineau, G. Gordon and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

[26] J. Pineau, G. Gordon, and S. Thrun. Anytime point-based approximations for pomdps. *Journal of Artificial Intelligence Research (JAIR)*, 27:335–380, 2006.

[27] M. Poon. *A fast heuristic algorithm for decision-theoretic planning.* Masters Thesis. The Hong-Kong University of Science and Technology, 2001.

[28] M. L. Puterman. *Markov Decision Processes—Discrete Stochastic Dynamic Programming.* John Wiley & Sons, Inc., New York, NY, 1994.

[29] N. Roy, J. Pineau and S. Thrun. Spoken dialog management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 93–100, Hong Kong, 2000.

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

[31] S. Seuken and S. Zilberstein. Memory bounded dynamic programming for dec-pomdps. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[32] R. Simmons and S. Koenig. Probabilistic Robot Navigation in Partially Observable Environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1080–1087, 1995.

[33] R. Smallwood and E. Sondik. The optimal control of partially observable markov decision processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[34] E. Sondik. *The Optimal Control of Partially Observable Markov Processes.* PhD Thesis, Stanford University, Stanford, California, 1971.

[35] M. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for pomdps. *Journal of Artificial Intelligence Research (JAIR)*, 24:195–220, 2005.

[36] D. Szer and F. Charpillet. Point based dynamic programming for dec-pomdps. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 21:1233-1238, 2006.

[37] B. Zhang, Q. Cai, J. Mao and B. Guo. Planning and acting under uncertainty: a new model for spoken dialogue systems. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, 572–579, San Francisco, CA, 2001.

[38] N. L. Zhang and W. Zhang. Speeding Up the Convergence of Value Iteration in Partially Observable Markov Decision Processes. In *Journal of Artificial Intelligence Research (JAIR)*, 14: 29–51, 2001.