Journal of Artificial Intelligence Research 32 (2008) 289–353 Submitted 09/07; published 05/08

Optimal and Approximate Q-value Functions for Decentralized POMDPs

Frans A. Oliehoek F.A.OLIEHOEK@UVA.NL
Intelligent Systems Lab Amsterdam, University of Amsterdam
Amsterdam, The Netherlands

Matthijs T.J. Spaan MTJSPAAN@ISR.IST.UTL.PT
Institute for Systems and Robotics, Instituto Superior Técnico
Lisbon, Portugal

Nikos Vlassis VLASSIS@DPEM.TUC.GR
Department of Production Engineering and Management, Technical University of Crete
Chania, Greece

Abstract

Decision-theoretic planning is a popular approach to sequential decision making problems, because it treats uncertainty in sensing and acting in a principled way. In single-agent frameworks like MDPs and POMDPs, planning can be carried out by resorting to Q-value functions: an optimal Q-value function Q^* is computed in a recursive manner by dynamic programming, and then an optimal policy is extracted from Q^*. In this paper we study whether similar Q-value functions can be defined for decentralized POMDP models (Dec-POMDPs), and how policies can be extracted from such value functions. We define two forms of the optimal Q-value function for Dec-POMDPs: one that gives a normative description as the Q-value function of an optimal pure joint policy and another one that is sequentially rational and thus gives a recipe for computation. This computation, however, is infeasible for all but the smallest problems. Therefore, we analyze various approximate Q-value functions that allow for efficient computation. We describe how they relate, and we prove that they all provide an upper bound to the optimal Q-value function Q^*. Finally, unifying some previous approaches for solving Dec-POMDPs, we describe a family of algorithms for extracting policies from such Q-value functions, and perform an experimental evaluation on existing test problems, including a new firefighting benchmark problem.

1. Introduction

One of the main goals in artificial intelligence (AI) is the development of intelligent agents, which perceive their environment through sensors and influence the environment through their actuators. In this setting, an essential problem is how an agent should decide which action to perform in a certain situation. In this work, we focus on *planning*: constructing a plan that specifies which action to take in each situation the agent might encounter over time. In particular, we will focus on planning in a cooperative multiagent system (MAS): an environment in which multiple agents coexist and interact in order to perform a joint task. We will adopt a decision-theoretic approach, which allows us to tackle uncertainty in sensing and acting in a principled way.

Decision-theoretic planning has roots in control theory and in operations research. In control theory, one or more controllers control a stochastic system with a specific output as goal. Operations research considers tasks related to scheduling, logistics and work flow and tries to optimize the concerning systems. Decision-theoretic planning problems can be formalized as *Markov decision processes* (MDPs), which have have been frequently employed in both control theory as well as operations research, but also have been adopted by AI for planning in stochastic environments. In all these fields the goal is to find a (conditional) plan, or *policy*, that is optimal with respect to the desired behavior. Traditionally, the main focus has been on systems with only one agent or controller, but in the last decade interest in systems with multiple agents or decentralized control has grown.

A different, but also related field is that of game theory. Game theory considers agents, called *players*, interacting in a dynamic, potentially stochastic process, the game. The goal here is to find optimal *strategies* for the agents, that specify how they should play and therefore correspond to policies. In contrast to decision-theoretic planning, game theory has always considered multiple agents, and as a consequence several ideas and concepts from game theory are now being applied in decentralized decision-theoretic planning. In this work we apply game-theoretic models to decision-theoretic planning for multiple agents.

1.1 Decision-Theoretic Planning

In the last decades, the *Markov decision process (MDP)* framework has gained in popularity in the AI community as a model for planning under uncertainty (Boutilier, Dean, & Hanks, 1999; Guestrin, Koller, Parr, & Venkataraman, 2003). MDPs can be used to formalize a discrete time planning task of a single agent in a stochastically changing environment, on the condition that the agent can observe the state of the environment. Every time step the state changes stochastically, but the agent chooses an action that selects a particular transition function. Taking an action from a particular state at time step t induces a probability distribution over states at time step $t + 1$.

The agent's objective can be formulated in several ways. The first type of objective of an agent is reaching a specific goal state, for example in a maze in which the agent's goal is to reach the exit. A different formulation is given by associating a certain cost with the execution of a particular action in a particular state, in which case the goal will be to minimize the expected total cost. Alternatively, one can associate rewards with actions performed in a certain state, the goal being to maximize the total reward.

When the agent knows the probabilities of the state transitions, i.e., when it knows the model, it can contemplate the expected transitions over time and construct a plan that is most likely to reach a specific goal state, minimizes the expected costs or maximizes the expected reward. This stands in some contrast to reinforcement learning (RL) (Sutton & Barto, 1998), where the agent does not have a model of the environment, but has to learn good behavior by repeatedly interacting with the environment. Reinforcement learning can be seen as the combined task of learning the model of the environment *and* planning, although in practice often it is not necessary to explicitly recover the environment model. In this article we focus only on planning, but consider two factors that complicate computing successful plans: the inability of the agent to observe the state of the environment as well as the presence of multiple agents.

In the real world an agent might not be able to determine what the state of the environment exactly is, because the agent's sensors are noisy and/or limited. When sensors are noisy, an agent can receive faulty or inaccurate observations with some probability. When sensors are limited the agent is unable to observe the differences between states that cannot be detected by the sensor, e.g., the presence or absence of an object outside a laser range-finder's field of view. When the same sensor reading might require different action choices, this phenomenon is referred to as *perceptual aliasing.* In order to deal with the introduced sensor uncertainty, a *partially observable Markov decision process (POMDP)* extends the MDP model by incorporating observations and their probability of occurrence conditional on the state of the environment (Kaelbling, Littman, & Cassandra, 1998).

The other complicating factor we consider is the presence of multiple agents. Instead of planning for a single agent we now plan for a team of cooperative agents. We assume that communication within the team is not possible.[1] A major problem in this setting is how the agents will have to coordinate their actions. Especially, as the agents are not assumed to observe the state—each agent only knows its own observations received and actions taken—there is no common signal they can condition their actions on. Note that this problem is in addition to the problem of partial observability, and not a substitution of it; even if the agents could freely and instantaneously communicate their individual observations, the joint observations would not disambiguate the true state.

One option is to consider each agent separately, and have each such agent maintain an explicit model of the other agents. This is the approach as chosen in the Interactive POMDP (I-POMDP) framework (Gmytrasiewicz & Doshi, 2005). A problem in this approach, however, is that the other agents also model the considered agent, leading to an infinite recursion of beliefs regarding the behavior of agents. We will adopt the *decentralized partially observable Markov decision process (Dec-POMDP)* model for this class of problems (Bernstein, Givan, Immerman, & Zilberstein, 2002). A Dec-POMDP is a generalization to multiple agents of a POMDP and can be used to model a team of cooperative agents that are situated in a stochastic, partially observable environment.

The single-agent MDP setting has received much attention, and many results are known. In particular it is known that an optimal plan, or policy, can be extracted from the optimal action-value, or *Q-value,* function $Q^*(s,a)$, and that the latter can be calculated efficiently. For POMDPs, similar results are available, although finding an optimal solution is harder (PSPACE-complete for finite-horizon problems, Papadimitriou & Tsitsiklis, 1987).

On the other hand, for Dec-POMDPs relatively little is known except that they are provably intractable (NEXP-complete, Bernstein et al., 2002). In particular, an outstanding issue is whether Q-value functions can be defined for Dec-POMDPs just as in (PO)MDPs, and whether policies can be extracted from such Q-value functions. Currently most algorithms for planning in Dec-POMDPs are based on some version of policy search (Nair, Tambe, Yokoo, Pynadath, & Marsella, 2003b; Hansen, Bernstein, & Zilberstein, 2004; Szer, Charpillet, & Zilberstein, 2005; Varakantham, Marecki, Yabu, Tambe, & Yokoo, 2007), and a proper theory for Q-value functions in Dec-POMDPs is still lacking. Given the wide range of applications of value functions in single-agent decision-theoretic planning, we expect that such a theory for Dec-POMDPs can have great benefits, both in terms of providing insight as well as guiding the design of solution algorithms.

1. As it turns out, the framework we consider can also model communication with a particular cost that is subject to minimization (Pynadath & Tambe, 2002; Goldman & Zilberstein, 2004). The non-communicative setting can be interpreted as the special case with infinite cost.

1.2 Contributions

In this paper we develop theory for Q-value functions in Dec-POMDPs, showing that an optimal Q-function Q^* *can* be defined for a Dec-POMDP. We define two forms of the optimal Q-value function for Dec-POMDPs: one that gives a normative description as the Q-value function of an optimal pure joint policy and another one that is sequentially rational and thus gives a recipe for computation. We also show that given Q^*, an optimal policy can be computed by *forward-sweep policy computation,* solving a sequence of Bayesian games forward through time (i.e., from the first to the last time step), thereby extending the solution technique of Emery-Montemerlo, Gordon, Schneider, and Thrun (2004) to the exact setting.

Computation of Q^* is infeasible for all but the smallest problems. Therefore, we analyze three different approximate Q-value functions $\mathrm{Q}_{\mathrm{MDP}}$, $\mathrm{Q}_{\mathrm{POMDP}}$ and Q_{BG} that can be more efficiently computed and which constitute upper bounds to Q^*. We also describe a generalized form of Q_{BG} that includes $\mathrm{Q}_{\mathrm{POMDP}}$, Q_{BG} and Q^*. This is used to prove a hierarchy of upper bounds: $Q^* \leq \mathrm{Q}_{\mathrm{BG}} \leq \mathrm{Q}_{\mathrm{POMDP}} \leq \mathrm{Q}_{\mathrm{MDP}}$.

Next, we show how these approximate Q-value functions can be used to compute optimal or sub-optimal policies. We describe a generic policy search algorithm, which we dub Generalized MAA* (GMAA*) as it is a generalization of the MAA* algorithm by Szer et al. (2005), that can be used for extracting a policy from an approximate Q-value function. By varying the implementation of a sub-routine of this algorithm, this algorithm unifies MAA* and forward-sweep policy computation and thus the approach of Emery-Montemerlo et al. (2004).

Finally, in an experimental evaluation we examine the differences between $\mathrm{Q}_{\mathrm{MDP}}$, $\mathrm{Q}_{\mathrm{POMDP}}$, Q_{BG} and Q^* for several problems. We also experimentally verify the potential benefit of tighter heuristics, by testing different settings of GMAA* on some well known test problems and on a new benchmark problem involving firefighting agents.

This article is based on previous work by Oliehoek and Vlassis (2007)—abbreviated OV here—containing several new contributions: (1) Contrary to the OV work, the current work includes a section on the sequential rational description of Q^* and suggests a way to compute Q^* in practice (OV only provided a normative description of Q^*). (2) The current work provides a formal proof of the hierarchy of upper bounds to Q^* (which was only qualitatively argued in the OV paper). (3) The current article additionally contains a proof that the solutions for the Bayesian games with identical payoffs given by equation (4.2) constitute Pareto optimal Nash equilibria of the game (which was not proven in the OV paper). (4) This article contains a more extensive experimental evaluation of the derived bounds of Q^*, and introduces a new benchmark problem (firefighting). (5) Finally, the current article provides a more complete introduction to Dec-POMDPs and existing solution methods, as well as Bayesian games, hence it can serve as a self-contained introduction to Dec-POMDPs.

1.3 Applications

Although the field of multiagent systems in a stochastic, partially observable environment seems quite specialized and thus narrow, the application area is actually very broad. The real world is practically always partially observable due to sensor noise and perceptual aliasing. Also, in most of these domains communication is not free, but consumes resources and thus has a particular cost. Therefore models as Dec-POMDPs, which do consider partially observable environments are relevant for essentially all teams of embodied agents.

Example applications of this type are given by Emery-Montemerlo (2005), who considered multi-robot navigation in which a team of agents with noisy sensors has to act to find/capture a goal. Becker, Zilberstein, Lesser, and Goldman (2004b) use a multi-robot space exploration example. Here, the agents are Mars rovers and have to decide on how to proceed their mission: whether to collect particular samples at specific sites or not. The rewards of particular samples can be sub- or super-additive, making this task non-trivial. An overview of application areas in cooperative robotics is presented by Arai, Pagello, and Parker (2002), among which is robotic soccer, as applied in RoboCup (Kitano, Asada, Kuniyoshi, Noda, & Osawa, 1997). Another application that is investigated within this project is crisis management: RoboCup Rescue (Kitano, Tadokoro, Noda, Matsubara, Takahashi, Shinjoh, & Shimada, 1999) models a situation where rescue teams have to perform a search and rescue task in a crisis situation. This task also has been modeled as a partially observable system (Nair, Tambe, & Marsella, 2002, 2003, 2003a; Oliehoek & Visser, 2006; Paquet, Tobin, & Chaib-draa, 2005).

There are also many other types of applications. Nair, Varakantham, Tambe, and Yokoo (2005), Lesser, Ortiz Jr., and Tambe (2003) give applications for distributed sensor networks (typically used for surveillance). An example of load balancing among queues is presented by Cogill, Rotkowitz, Roy, and Lall (2004). Here agents represent queues and can only observe queue sizes of themselves and immediate neighbors. They have to decide whether to accept new jobs or pass them to another queue. Another frequently considered application domain is communication networks. Peshkin (2001) treated a packet routing application in which agents are routers and have to minimize the average transfer time of packets. They are connected to immediate neighbors and have to decide at each time step to which neighbor to send each packet. Other approaches to communication networks using decentralized, stochastic, partially observable systems are given by Ooi and Wornell (1996), Tao, Baxter, and Weaver (2001), Altman (2002).

1.4 Overview of Article

The rest of this article is organized as follows. In Section 2 we will first formally introduce the Dec-POMDP model and provide background on its components. Some existing solution methods are treated in Section 3. Then, in Section 4 we show how a Dec-POMDP can be modeled as a series of Bayesian games and how this constitutes a theory of Q-value functions for BGs. We also treat two forms of optimal Q-value functions, Q^*, here. Approximate Q-value functions are described in Section 5 and one of their applications is discussed in Section 6. Section 7 presents the results of the experimental evaluation. Finally, Section 8 concludes.

2. Decentralized POMDPs

In this section we define the Dec-POMDP model and discuss some of its properties. Intuitively, a Dec-POMDP models a number of agents that inhabit a particular environment, which is considered at discrete *time steps*, also referred to as *stages* (Boutilier et al., 1999) or *(decision) epochs* (Puterman, 1994). The number of time steps the agents will interact with their environment is called the *horizon* of the decision problem, and will be denoted by h. In this paper the horizon is assumed to be finite. At each stage $t = 0,1,2,\ldots,h-1$ every agent takes an action and the combination of these actions influences the environment, causing a state transition. At the next time step, each agent first receives an observation of the environment, after which it has to take an action again. The probabilities of state transitions and observations are specified by the Dec-POMDP model, as are rewards received for particular actions in particular states. The transition- and observation probabilities specify the dynamics of the environment, while the rewards specify what behavior is desirable. Hence, the reward model defines the agents' goal or task: the agents have to come up with a plan that maximizes the expected long term reward signal. In this work we assume that planning takes place off-line, after which the computed plans are distributed to the agents, who then merely execute the plans on-line. That is, computation of the plan is centralized, while execution is decentralized. In the centralized planning phase, the entire model as detailed below is available. During execution each agent only knows the joint policy as found by the planning phase and its individual history of actions and observations.

2.1 Formal Model

In this section we more formally treat the basic components of a Dec-POMDP. We start by giving a mathematical definition of these components.

Definition 2.1. A *decentralized partially observable Markov decision process (Dec-POMDP)* is defined as a tuple $\langle n,\mathcal{S},\mathcal{A},T,R,\mathcal{O},O,h,b^0 \rangle$ where:

- n is the number of agents.
- $\mathcal{S}$ is a finite set of states.
- $\mathcal{A}$ is the set of joint actions.
- T is the transition function.
- R is the immediate reward function.
- $\mathcal{O}$ is the set of joint observations.
- O is the observation function.
- h is the horizon of the problem.
- $b^0 \in \mathcal{P}(\mathcal{S})$, is the initial state distribution at time $t = 0$.[2]

The Dec-POMDP model extends single-agent (PO)MDP models by considering *joint* actions and observations. In particular, we define $\mathcal{A} = \times_i \mathcal{A}_i$ as the set of *joint actions*. Here, $\mathcal{A}_i$ is the set of actions available to agent i. Every time step, one joint action $a = \langle a_1,...,a_n \rangle$ is taken. In a Dec-POMDP, agents only know their own individual action; they do not observe each other's actions. We will assume that any action $a_i \in \mathcal{A}_i$ can be selected at any time. So the set $\mathcal{A}_i$ does not depend on the stage or state of the environment. In general, we will

2. $\mathcal{P}(\cdot)$ denotes the set of probability distributions over $(\cdot)$.

denote the stage using superscripts, so a^t denotes the joint action taken at stage t, a_i^t is the individual action of agent i taken at stage t. Also, we write $a_{\neq i} = \langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \rangle$ for a profile of actions for all agents but i.

Similarly to the set of joint actions, $\mathcal{O} = \times_i \mathcal{O}_i$ is the set of joint observations, where $\mathcal{O}_i$ is a set of observations available to agent i. Every time step the environment emits one joint observation $o = \langle o_1,...,o_n \rangle$, from which each agent i only observes its own component o_i, as illustrated by Figure 1. Notation with respect to time and indices for observations is analogous to the notation for actions. In this paper, we will assume that the action- and observation sets are finite. Infinite action- and observation sets are very difficult to deal with even in the single-agent case, and to the authors' knowledge no research has been performed on this topic in the partially observable, multiagent case.

Actions and observations are the interface between the agents and their environment. The Dec-POMDP framework describes this environment by its *states* and *transitions.* This means that rather than considering a complex, typically domain-dependent model of the environment that explains how this environment works, a descriptive stance is taken: A Dec-POMDP specifies an environment model simply as the set of states $\mathcal{S} = \{s_1,...,s_{|\mathcal{S}|}\}$ the environment can be in, together with the probabilities of state transitions that are dependent on executed joint actions. In particular, the transition from some state to a next state depends stochastically on the past states and actions. This probabilistic dependence models *outcome uncertainty*: the fact that the outcome of an action cannot be predicted with full certainty.

An important characteristic of Dec-POMDPs is that the states possess the *Markov property.* That is, the probability of a particular next state depends on the current state and joint action, but not on the whole history:

$$P(s^{t+1}|s^t,a^t,s^{t-1},a^{t-1},...,s^0,a^0) = P(s^{t+1}|s^t,a^t). \tag{2.1}$$

Also, we will assume that the transition probabilities are *stationary*, meaning that they are independent of the stage t.

In a way similar to how the transition model T describes the stochastic influence of actions on the environment, the observation model O describes how the state of the environment is perceived by the agents. Formally, O is the observation function, a mapping from joint actions and successor states to probability distributions over joint observations: $O : \mathcal{A} \times \mathcal{S} \to \mathcal{P}(\mathcal{O})$. I.e., it specifies

$$P(o^t|a^{t-1},s^t). \tag{2.2}$$

This implies that the observation model also satisfies the Markov property (there is no dependence on the history). Also the observation model is assumed stationary: there is no dependence on the stage t.

Literature has identified different categories of observability (Pynadath & Tambe, 2002; Goldman & Zilberstein, 2004). When the observation function is such that the individual observation for all the agents will always uniquely identify the true state, the problem is considered *fully-* or *individually observable.* In such a case, a Dec-POMDP effectively reduces to a multiagent MDP (MMDP) as described by Boutilier (1996). The other extreme is when the problem is *non-observable,* meaning that none of the agents observes any useful

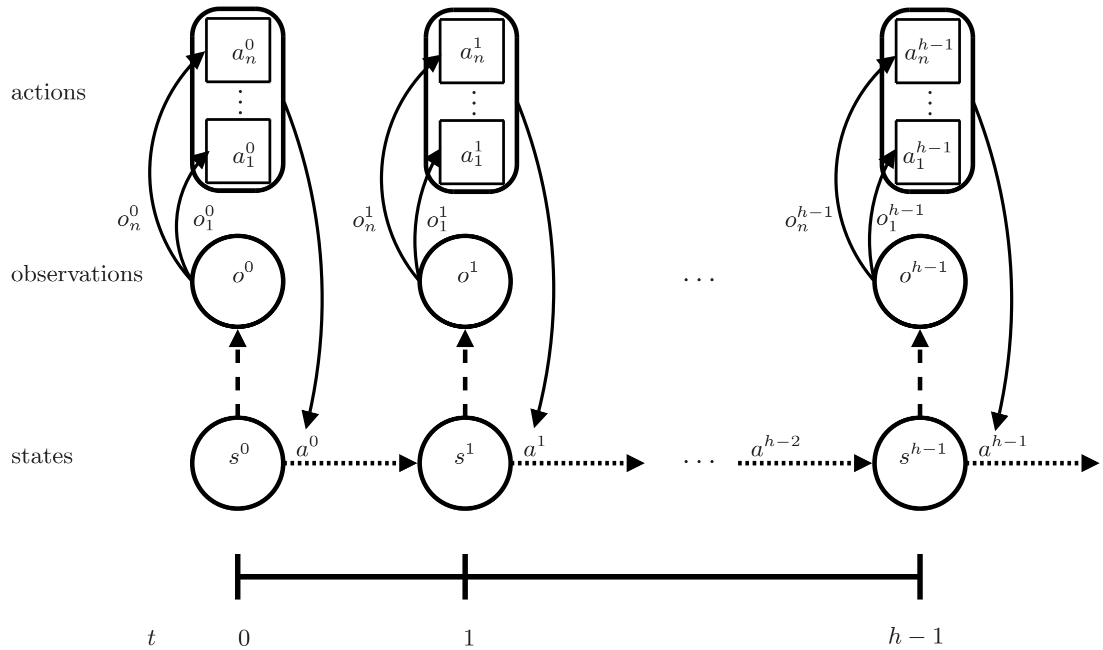


Figure 1: An illustration of the dynamics of a Dec-POMDP. At every stage the environment is in a particular state. This state emits a joint observation, of which each agent observes its individual observation. Then each agent selects an action forming the joint action.

information. This is modeled by the fact that agents always receive a null-observation, $\forall_i \ \mathcal{O}_i = \{o_{i,\emptyset}\}$. Under non-observability agents can only employ an open-loop plan. Between these two extremes there are *partially observable* problems. One more special case has been identified, namely the case where not the individual, but the joint observation identifies the true state. This case is referred to as *jointly-* or *collectively observable*. A jointly observable Dec-POMDP is referred to as a *Dec-MDP*.

The reward function $R(s,a)$ is used to specify the goal of the agents and is a function of states and joint actions. In particular, a desirable sequence of joint actions should correspond to a high 'long-term' reward, formalized as the return.

Definition 2.2. Let the *return* or *cumulative reward* of a Dec-POMDP be defined as total of the rewards received during an execution:

$$r(0) + r(1) + \cdots + r(h-1), \tag{2.3}$$

where $r(t)$ is the reward received at time step t.

When, at stage t, the state is s^t and the taken joint action is a^t, we have that $r(t) = R(s^t,a)$. Therefore, given the sequence of states and taken joint actions, it is straightforward to determine the return by substitution of $r(t)$ by $R(s^t,a)$ in (2.3).

In this paper we consider as optimality criterion the *expected cumulative reward*, where the expectation refers to the expectation over sequences of states and executed joint actions. The planning problem is to find a conditional plan, or *policy,* for each agent to maximize the optimality criterion. In the Dec-POMDP case this amounts to finding a tuple of policies, called a *joint policy* that maximizes the expected cumulative reward.

Note that, in a Dec-POMDP, the agents are assumed not to observe the immediate rewards: observing the immediate rewards could convey information regarding the true state which is not present in the received observations, which is undesirable as all information available to the agents should be modeled in the observations. When planning for Dec-POMDPs the only thing that matters is the *expectation* of the cumulative future reward which is available in the off-line planning phase, not the actual reward obtained. Indeed, it is not even assumed that the actual reward can be observed at the end of the episode.

Summarizing, in this work we consider Dec-POMDPs with finite actions and observation sets and a finite planning horizon. Furthermore, we consider the general Dec-POMDP setting, without any simplifying assumptions on the observation, transition, or reward models.

2.2 Example: Decentralized Tiger Problem

Here we will describe the decentralized tiger problem introduced by Nair et al. (2003b). This test problem has been frequently used (Nair et al., 2003b; Emery-Montemerlo et al., 2004; Emery-Montemerlo, Gordon, Schneider, & Thrun, 2005; Szer et al., 2005) and is a modification of the (single-agent) tiger problem (Kaelbling et al., 1998). It concerns two agents that are standing in a hallway with two doors. Behind one of the doors is a tiger, behind the other a treasure. Therefore there are two states: the tiger is behind the left door (s_l) or behind the right door (s_r). Both agents have 3 actions at their disposal: open the left door (a_{OL}), open the right door (a_{OR}) and listen (a_{Li}). But they cannot observe each other's actions. In fact, they can only receive 2 observations. Either they hear a sound left (o_{HL}) or right (o_{HR}).

At $t = 0$ the state is s_l or s_r with probability 0.5. As long as no agent opens a door the state doesn't change, when a door is opened, the state resets to s_l or s_r with probability 0.5. The full transition, observation and reward model are listed by Nair et al. (2003b). The observation probabilities are independent, and identical for both agents. For instance, when the state is s_l and both perform action a_{Li}, both agents have a 85% chance of observing o_{HL}, and the probability of both hearing the tiger left is $0.85 \cdot 0.85 = 0.72$.

When the agents open the door for the treasure they receive a positive reward, while they receive a penalty for opening the wrong door. When opening the wrong door jointly, the penalty is less severe. Opening the correct door jointly leads to a higher reward.

Note that, when the wrong door is opened by one or both agents, they are attacked by the tiger and receive a penalty. However, neither of the agents observe this attack nor the penalty and the episode continues. Arguably, a more natural representation would be to have the episode end after a door is opened or to let the agents observe whether they encountered the tiger or treasure, however this is not considered in this test problem.

2.3 Histories

As mentioned, the goal of planning in a Dec-POMDP is to find a (near-) optimal tuple of policies, and these policies specify for each agent how to act in a specific situation. Therefore, before we define a policy, we first need to define exactly what these specific situations are. In essence such situations are those parts of the history of the process that the agents can observe.

Let us first consider what the history of the process is. A Dec-POMDP with horizon h specifies h time steps or stages $t = 0,...,h-1$. At each of these stages, there is a state s^t, joint observation o^t and joint action a^t. Therefore, when the agents will have to select their k-th actions (at $t = k-1$), the history of the process is a sequence of states, joint observations and joint actions, which has the following form:

$$\left(s^0,o^0,a^0,s^1,o^1,a^1,...,s^{k-1},o^{k-1}\right).$$

Here s^0 is the initial state, drawn according to the initial state distribution b^0. The initial joint observation o^0 is assumed to be the empty joint observation: $o^0 = o_\emptyset = \left\langle o_{1,\emptyset},...,o_{n,\emptyset}\right\rangle$.

From this history of the process, the states remain unobserved and agent i can only observe its own actions and observations. Therefore an agent will have to base its decision regarding which action to select on the sequence of actions and observations observed up to that point.

Definition 2.3. We define *the action-observation history for agent i,* $\vec{\theta}_i$, as the sequence of actions taken by and observations received by agent i. At a specific time step t, this is:

$$\vec{\theta}_i^t = \left(o_i^0,a_i^0,o_i^1 \ldots,a_i^{t-1},o_i^t\right).$$

The *joint action-observation history*, $\vec{\theta}$, is the action-observation history for all agents:

$$\vec{\theta}^t = \langle \vec{\theta}_1^t, \ldots ,\vec{\theta}_n^t\rangle.$$

Agent i's set of possible action-observation histories at time t is $\vec{\Theta}_i^t = \times_t(\mathcal{O}_i \times \mathcal{A}_i)$. The set of all possible action-observation histories for agent i is $\vec{\Theta}_i = \cup_{t=0}^{h-1}\vec{\Theta}_i^t$.[3] Finally the set of all possible *joint* action-observation histories is given by $\vec{\Theta} = \cup_{t=0}^{h-1}(\vec{\Theta}_1^t \times ... \times \vec{\Theta}_n^t)$. At $t = 0$, the action-observation history is empty, denoted by $\vec{\theta}^0 = \vec{\theta}_\emptyset$.

We will also use a notion of history only using the observations of an agent.

Definition 2.4. Formally, we define *the observation history for agent i,* $\vec{o}_i$, as the sequence of observations an agent has received. At a specific time step t, this is:

$$\vec{o}_i^t = \left(o_i^0,o_i^1, \ldots ,o_i^t\right).$$

The *joint observation history,* $\vec{o}$, is the observation history for all agents:

$$\vec{o}^t = \langle \vec{o}_1^t, \ldots ,\vec{o}_n^t\rangle.$$

3. Note that in a particular Dec-POMDP, it may be the case that not all of these histories can actually be realized, because of the probabilities specified by the transition and observation model.

The set of observation histories for agent i at time t is denoted $\vec{\mathcal{O}}_i^t = \times_t \mathcal{O}_i$. Similar to the notation for action-observation histories, we also use $\vec{\mathcal{O}}_i$ and $\vec{\mathcal{O}}$ and the empty observation history is denoted $\vec{o}_\emptyset$.

Similarly we can define the action history as follows.

Definition 2.5. The *action history for agent i, $\vec{a}_i$*, is the sequence of actions an agent has performed. At a specific time step t, we write:

$$\vec{a}_i^{\,t} = \left(a_i^0, a_i^1, \ldots, a_i^{t-1}\right).$$

Notation for joint action histories and sets are analogous to those for observation histories. Also write $\vec{o}_{\neq i}, \vec{\theta}_{\neq i}$, etc. to denote a tuple of observation-, action-observation histories, etc. for all agents except i. Finally we note that, clearly, an (joint) action-observation history consists of an (joint) action- and an (joint) observation history: $\vec{\theta}^{\,t} = \left\langle \vec{o}^{\,t}, \vec{a}^{\,t} \right\rangle$.

2.4 Policies

As discussed in the previous section, the action-observation history of an agent specifies all the information the agent has when it has to decide upon an action. For the moment we assume that an individual policy π_i for agent i is a deterministic mapping from action-observation sequences to actions.

The number of possible action-observation histories is usually very large as this set grows exponentially with the horizon of the problem. At time step t, there are $(|\mathcal{A}_i| \cdot |\mathcal{O}_i|)^t$ action-observation histories for agent i. As a consequence there are a total of

$$\sum_{t=0}^{h-1} (|\mathcal{A}_i| \cdot |\mathcal{O}_i|)^t = \frac{(|\mathcal{A}_i| \cdot |\mathcal{O}_i|)^h - 1}{(|\mathcal{A}_i| \cdot |\mathcal{O}_i|) - 1}$$

of such sequences for agent i. Therefore the number of policies for agent i becomes:

$$|\mathcal{A}_i|^{\frac{(|\mathcal{A}_i| \cdot |\mathcal{O}_i|)^h - 1}{(|\mathcal{A}_i| \cdot |\mathcal{O}_i|) - 1}}, \tag{2.4}$$

which is doubly exponential in the horizon h.

2.4.1 Pure and Stochastic Policies

It is possible to reduce the number of policies under consideration by realizing that a lot of policies specify the same behavior. This is illustrated by the left side of Figure 2, which clearly shows that under a deterministic policy only a subset of possible action-observation histories are reached. Policies that only differ with respect to an action-observation history that is not reached in the first place, manifest the same behavior. The consequence is that in order to specify a deterministic policy, the observation history suffices: when an agent takes its action deterministically, he will be able to infer what action he took from only the observation history as illustrated by the right side of Figure 2.

Definition 2.6. A *pure* or *deterministic policy*, π_i, for agent i in a Dec-POMDP is a mapping from observation histories to actions, $\pi_i : \vec{\mathcal{O}}_i \to \mathcal{A}_i$. The set of pure policies of agent i is denoted Π_i.

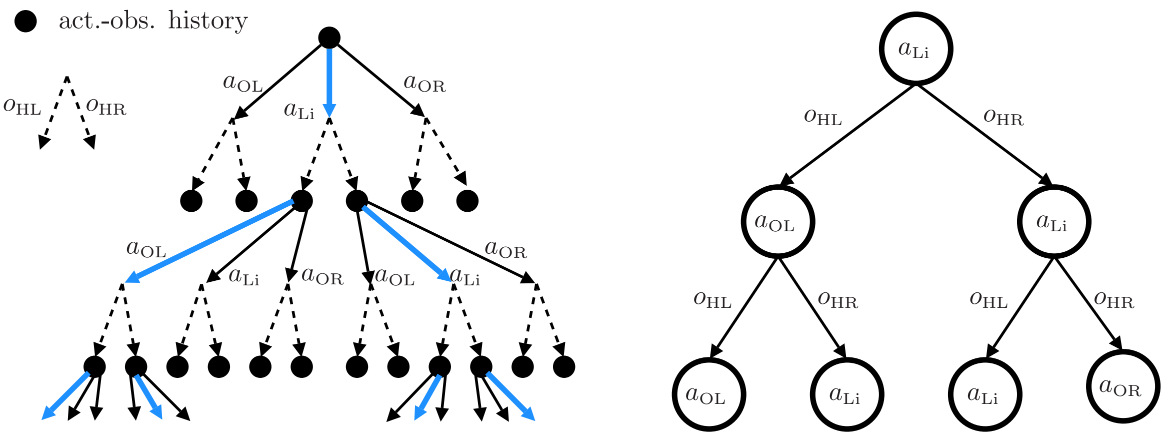


Figure 2: A deterministic policy can be represented as a tree. Left: a tree of action-observation histories $\vec{\theta}_i$ for one of the agents from the Dec-Tiger problem. An arbitrary deterministic policy π_i is highlighted. Clearly shown is that π_i only reaches a subset of of histories $\vec{\theta}_i$. ($\vec{\theta}_i$ that are not reached are not further expanded.) Right: The same policy can be shown in a simplified policy tree.

Note that also for pure policies we sometimes write $\pi_i(\vec{\theta}_i)$. In this case we mean the action that π_i specifies for the observation history contained in $\vec{\theta}_i$. For instance, let $\vec{\theta}_i = \langle \vec{o}_i, \vec{a}_i \rangle$, then $\pi_i(\vec{\theta}_i) = \pi_i(\vec{o}_i)$. We use $\pi = \langle \pi_1,...,\pi_n \rangle$ to denote a *joint policy*, a profile specifying a policy for each agent. We say that a pure joint policy is an *induced* or *implicit* mapping from joint observation histories to joint actions $\pi : \vec{\mathcal{O}} \to \mathcal{A}$. That is, the mapping is induced by individual policies π_i that make up the joint policy. Also we use $\pi_{\neq i} = \langle \pi_1, \dots, \pi_{i-1}, \pi_{i+1}, \dots, \pi_n \rangle$, to denote a profile of policies for all agents but i.

Apart from pure policies, it is also possible to have the agents execute *randomized policies*, i.e., policies that do not always specify the same action for the same situation, but in which there is an element of chance that decides which action is performed. There are two types of randomized policies: mixed policies and stochastic policies.

Definition 2.7. A *mixed policy, μ_i,* for an agent i is a set of pure policies, $\mathcal{M} \subseteq \Pi_i$, along with a probability distribution over this set. Thus a mixed policy $\mu_i \in \mathcal{P}(\mathcal{M})$ is an element of the set of probability distributions over $\mathcal{M}$.

Definition 2.8. A *stochastic* or *behavioral policy*, ς_i, for agent i is a mapping from action-observation histories to probability distributions over actions, $\varsigma_i : \vec{\Theta}_i \to \mathcal{P}(\mathcal{A}_i)$.

When considering stochastic policies, keeping track of only the observations is insufficient, as in general all action-observation histories can be realized. That is why stochastic policies are defined as a mapping from the full space of action-observation histories to probability distributions over actions. Note that we use π_i and Π_i to denote a policy (space) in general, so also for randomized policies. We will only use π_i, μ_i and ς_i when there is a need to discriminate between different types of policies.

A common way to represent the temporal structure in a policy is to split it in *decision rules* δ_i that specify the policy for each stage. An individual policy is then represented as a sequence of decision rules $\pi_i = (\delta_i^0, \ldots ,\delta_i^{h-1})$. In case of a deterministic policy, the form of the decision rule for stage t is a mapping from length-t observation histories to actions $\delta_i^t : \vec{\mathcal{O}}_i^t \to \mathcal{A}_i$.

2.4.2 Special Cases with Simpler Policies.

There are some special cases of Dec-POMDPs in which the policy can be specified in a simpler way. Here we will treat three such cases: in case the state s is observable, in the single-agent case and the case that combines the previous two: a single agent in an environment of which it can observe the state.

The last case, a single agent in a fully observable environment, corresponds to the regular MDP setting. Because the agent can observe the state, which is Markovian, the agent does not need to remember any history, but can simply specify the decision rules δ of its policy $\pi = \big(\delta^0, \ldots ,\delta^{h-1}\big)$ as mappings from states to actions: $\forall t\ \delta^t : \mathcal{S} \to \mathcal{A}$. The complexity of the policy representation reduces even further in the infinite-horizon case, where an optimal policy π^* is known to be *stationary*. As such, there is only one decision rule δ, that is used for all stages.

The same is true for multiple agents that can observe the state, i.e., a fully observable Dec-POMDP as defined in Section 2.1. This is essentially the same setting as the *multiagent Markov decision process (MMDP)* introduced by Boutilier (1996). In this case, the decision rules for agent i's policy are mappings from states to actions $\forall t\ \delta_i^t : \mathcal{S} \to \mathcal{A}_i$, although in this case some care needs to be taken to make sure no coordination errors occur when searching for these individual policies.

In a POMDP, a Dec-POMDP with a single agent, the agent cannot observe the state, so it is not possible to specify a policy as a mapping from states to actions. However, it turns out that maintaining a probability distribution over states, called *belief*, $b \in \mathcal{P}(\mathcal{S})$, is a Markovian signal:

$$P(s^{t+1}|a^t,o^t,a^{t-1},o^{t-1}, \ldots ,a^0,o^0) = P(s^{t+1}|b^t,a^t),$$

where the belief b^t is defined as

$$\forall_{s^t} \quad b^t(s^t) \ \equiv \ P(s^t|o^t,a^{t-1},o^{t-1}, \ldots ,a^0,o^0) = P(s^t|b^{t-1},a^{t-1},o^t).$$

As a result, a single agent in a partially observable environment can specify its policy as a series of mappings from the set of beliefs to actions $\forall t\ \delta^t : \mathcal{P}(\mathcal{S}) \to \mathcal{A}$.

Unfortunately, in the general case we consider, no such space-saving simplifications of the policy are possible. Even though the transition and observation model can be used to compute a *joint* belief, this computation requires knowledge of the joint actions and observations. During execution, the agents simply have no access to this information and thus can not compute a joint belief.

2.4.3 The Quality of Joint Policies

Clearly, policies differ in how much reward they can expect to accumulate, which will serve as a criterion of a joint policy's quality. Formally, we consider the expected cumulative reward of a joint policy, also referred to as its *value*.

Definition 2.9. The *value* $V(\pi)$ of a joint policy π is defined as

$$V(\pi) \;\equiv\; E\Big[\sum_{t=0}^{h-1} R(s^t,a^t)\Big|\pi,b^0\Big], \tag{2.5}$$

where the expectation is over states, observations and—in the case of a randomized π—actions.

In particular we can calculate this expectation as

$$V(\pi) = \sum_{t=0}^{h-1} \sum_{\vec{\theta}^t \in \vec{\Theta}^t} \sum_{s^t \in \mathcal{S}} P(s^t,\vec{\theta}^t|\pi,b^0) \sum_{a^t \in \mathcal{A}} R(s^t,a^t) P_\pi(a^t|\vec{\theta}^t), \tag{2.6}$$

where $P_\pi(a^t|\vec{\theta}^t)$ is the probability of a as specified by π, and where $P(s^t,\vec{\theta}^t|\pi,b^0)$ is recursively defined as

$$P(s^t,\vec{\theta}^t|\pi,b^0) = \sum_{s^{t-1} \in \mathcal{S}} P(s^t,\vec{\theta}^t|s^{t-1},\vec{\theta}^{t-1},\pi) P(s^{t-1},\vec{\theta}^{t-1}|\pi,b^0), \tag{2.7}$$

with

$$P(s^t,\vec{\theta}^t|s^{t-1},\vec{\theta}^{t-1},\pi) = P(o^t|a^{t-1},s^t) P(s^t|s^{t-1},a^{t-1}) P_\pi(a^{t-1}|\vec{\theta}^{t-1}) \tag{2.8}$$

a term that is completely specified by the transition and observation model and the joint policy. For stage 0 we have that $P(s^0,\vec{\theta}_\emptyset|\pi,b^0) = b^0(s^0)$.

Because of the recursive nature of $P(s^t,\vec{\theta}^t|\pi,b^0)$ it is more intuitive to specify the value recursively:

$$V_\pi(s^t,\vec{\theta}^t) = \sum_{a^t \in \mathcal{A}} P_\pi(a^t|\vec{\theta}^t) \left[R(s^t,a^t) + \sum_{s^{t+1} \in \mathcal{S}} \sum_{o^{t+1} \in \mathcal{O}} P(s^{t+1},o^{t+1}|s^t,a^t) V_\pi(s^{t+1},\vec{\theta}^{t+1}) \right], \tag{2.9}$$

with $\vec{\theta}^{t+1} = (\vec{\theta}^t,a^t,o^{t+1})$. The value of joint policy π is then given by

$$V(\pi) = \sum_{s^0 \in \mathcal{S}} V_\pi(s^0,\vec{\theta}_\emptyset) b^0(s^0). \tag{2.10}$$

For the special case of evaluating a pure joint policy π, eq. (2.6) can be written as:

$$V(\pi) = \sum_{t=0}^{h-1} \sum_{\vec{\theta}^t \in \vec{\Theta}^t} P(\vec{\theta}^t|\pi,b^0) R(\vec{\theta}^t,\pi(\vec{\theta}^t)), \tag{2.11}$$

where

$$R(\vec{\theta}^t,a^t) = \sum_{s^t \in \mathcal{S}} R(s^t,a^t) P(s^t|\vec{\theta}^t,b^0) \tag{2.12}$$

denotes the expected immediate reward. In this case, the recursive formulation (2.9) reduces to

$$V_\pi^t(s^t,\vec{o}^t) = R\left(s^t,\pi(\vec{o}^t)\right) + \sum_{s^{t+1} \in \mathcal{S}} \sum_{o^{t+1} \in \mathcal{O}} P(s^{t+1},o^{t+1}|s^t,\pi(\vec{o}^t)) V_\pi^{t+1}(s^{t+1},\vec{o}^{t+1}). \tag{2.13}$$

Note that, when performing the computation of the value for a joint policy recursively, intermediate results should be cached. A particular $(s^{t+1},\vec{o}^{t+1})$-pair (or $(s^{t+1},\vec{\theta}^{t+1})$-pair for a stochastic joint policy) can be reached from $|\mathcal{S}|$ states s^t of the previous stage. The value $V_\pi^{t+1}(s^{t+1},\vec{o}^{t+1})$ is the same, however, and should be computed only once.

2.4.4 Existence of an Optimal Pure Joint Policy

Although randomized policies may be useful, we can restrict our attention to pure policies without sacrificing optimality, as shown by the following.

Proposition 2.1. *A Dec-POMDP has at least one optimal pure joint policy.*

Proof. See appendix A.1. □

3. Overview of Dec-POMDP Solution Methods

In order to provide some background on solving Dec-POMDPs, this section gives an overview of some recently proposed methods. We will limit this review to a number of finite-horizon methods for general Dec-POMDPs that are related to our own approach.

We will not review the work performed on infinite-horizon Dec-POMDPs, such as the work by Peshkin, Kim, Meuleau, and Kaelbling (2000), Bernstein, Hansen, and Zilberstein (2005), Szer and Charpillet (2005), Amato, Bernstein, and Zilberstein (2006, 2007a). In this setting policies are usually represented by finite state controllers (FSCs). Since an infinite-horizon Dec-POMDP is undecidable (Bernstein et al., 2002), this line of work, focuses on finding ϵ-approximate solutions (Bernstein, 2005) or (near-) optimal policies for given a particular controller size.

There also is a substantial amount of work on methods exploiting particular independence assumptions. In particular, transition and observation independent Dec-MDPs (Becker et al., 2004b; Wu & Durfee, 2006) and Dec-POMDPs (Kim, Nair, Varakantham, Tambe, & Yokoo, 2006; Varakantham et al., 2007) have received quite some attention. These models assume that each agent i has an individual state space $\mathcal{S}_i$ and that the actions of one agent do not influence the transitions between the local states of another agent. Although such models are easier to solve, the independence assumptions severely restrict their applicability. Other special cases that have been considered are, for instance, goal oriented Dec-POMDPs (Goldman & Zilberstein, 2004), event-driven Dec-MDPs (Becker, Zilberstein, & Lesser, 2004a), Dec-MDPs with time and resource constraints (Beynier & Mouaddib, 2005, 2006; Marecki & Tambe, 2007), Dec-MDPs with local interactions (Spaan & Melo, 2008) and factored Dec-POMDPs with additive rewards (Oliehoek, Spaan, Whiteson, & Vlassis, 2008).

A final body of related work which is beyond the scope of this article are models and techniques for explicit communication in Dec-POMDP settings (Ooi & Wornell, 1996; Pynadath & Tambe, 2002; Goldman & Zilberstein, 2003; Nair, Roth, & Yohoo, 2004; Becker, Lesser, & Zilberstein, 2005; Roth, Simmons, & Veloso, 2005; Oliehoek, Spaan, & Vlassis, 2007b; Roth, Simmons, & Veloso, 2007; Goldman, Allen, & Zilberstein, 2007). The Dec-POMDP model itself can model communication actions as regular actions, in which case the semantics of the communication actions becomes part of the optimization problem (Xuan, Lesser, & Zilberstein, 2001; Goldman & Zilberstein, 2003; Spaan, Gordon, & Vlassis, 2006).

In contrast, most approaches mentioned typically assume that communication happens outside the Dec-POMDP model and with pre-defined semantics. A typical assumption is that at every time step the agents communicate their individual observations before selecting an action. Pynadath and Tambe (2002) showed that, under assumptions of instantaneous and cost-free communication, sharing individual observations in such a way is optimal.

3.1 Brute Force Policy Evaluation

Because there exists an optimal pure joint policy for a finite-horizon Dec-POMDP, it is in theory possible to enumerate all different pure joint policies, evaluate them using equations (2.10) and (2.13) and choose the best one. The number of pure joint policies to be evaluated is:

$$O\left(|\mathcal{A}_*|^{\frac{n(|\mathcal{O}_*|^h-1)}{|\mathcal{O}_*|-1}}\right), \tag{3.1}$$

where $|\mathcal{A}_*|$ and $|\mathcal{O}_*|$ denote the largest individual action and observation sets. The cost of evaluating each policy is $O\left(|\mathcal{S}| \cdot |\mathcal{O}|^h\right)$. The resulting total cost of brute-force policy evaluation is

$$O\left(|\mathcal{A}_*|^{\frac{n(|\mathcal{O}_*|^h-1)}{|\mathcal{O}_*|-1}} \times |\mathcal{S}| \times |\mathcal{O}_*|^{nh}\right), \tag{3.2}$$

which is doubly exponential in the horizon h.

3.2 Alternating Maximization

Nair et al. (2003b) introduced *Joint Equilibrium based Search for Policies (JESP).* This method guarantees to find a locally optimal joint policy, more specifically, a *Nash equilibrium*: a tuple of policies such that for each agent i its policy π_i is a best response for the policies employed by the other agents $\pi_{\neq i}$. It relies on a process we refer to as *alternating maximization.* This is a procedure that computes a policy π_i for an agent i that maximizes the joint reward, while keeping the policies of the other agents fixed. Next, another agent is chosen to maximize the joint reward by finding its best-response to the fixed policies of the other agents. This process is repeated until the joint policy converges to a Nash equilibrium, which is a local optimum. The main idea of fixing some agents and having others improve their policy was presented before by Chades, Scherrer, and Charpillet (2002), but they used a heuristic approach for memory-less agents. The process of alternating maximization is also referred to as *hill-climbing* or *coordinate ascent.*

Nair et al. (2003b) describe two variants of JESP, the first of which, Exhaustive-JESP, implements the above idea in a very straightforward fashion: Starting from a random joint policy, the first agent is chosen. This agent then selects its best-response policy by evaluating the joint reward obtained for all of its individual policies when the other agents follow their fixed policy.

The second variant, DP-JESP, uses a dynamic programming approach to compute the best-response policy for a selected agent i. In essence, fixing the policies of all other agents allows for a reformulation of the problem as an augmented POMDP. In this augmented POMDP a state $\bar{s} = \langle s, \vec{o}_{\neq i} \rangle$ consists of a nominal state s and the observation histories of the other agents $\vec{o}_{\neq i}$. Given the fixed deterministic policies of other agents $\pi_{\neq i}$, such an augmented state $\bar{s}$ is a Markovian state, and all transition and observation probabilities can easily be derived from $\pi_{\neq i}$.

Like most methods proposed for Dec-POMDPs, JESP exploits the knowledge of the initial belief b^0 by only considering reachable beliefs $b(\bar{s})$ in the solution of the POMDP. However, in some cases the initial belief might not be available. As demonstrated by Varakantham, Nair, Tambe, and Yokoo (2006), JESP can be extended to plan for the entire space of initial beliefs, overcoming this problem.

3.3 MAA*

Szer et al. (2005) introduced a heuristically guided policy search method called *multiagent A* (MAA*)*. It performs a guided A*-like search over partially specified joint policies, pruning joint policies that are guaranteed to be worse than the best (fully specified) joint policy found so far by an admissible heuristic.

In particular MAA* considers joint policies that are partially specified with respect to time: a partial joint policy $\varphi^t = (\delta^0,\delta^1,\dots,\delta^{t-1})$ specifies the joint decision rules for the first t stages. For such a partial joint policy φ^t a heuristic value $\widehat{V}(\varphi^t)$ is calculated by taking $V^{0\dots t-1}(\varphi^t)$, the actual expected reward φ^t achieves over the first t stages, and adding $\widehat{V}^{t\dots h-1}$, a heuristic value for the remaining $h-t$ stages. Clearly when $\widehat{V}^{t\dots h-1}$ is an *admissible heuristic*—a guaranteed overestimation—so is $\widehat{V}(\varphi^t)$.

MAA* starts by placing the completely unspecified joint policy φ^0 in an open list. Then, it proceeds by selecting partial joint policies $\varphi^t = (\delta^0,\delta^1,\dots,\delta^{t-1})$ from the list and 'expanding' them: generating all $\varphi^{t+1} = (\delta^0,\delta^1,\dots,\delta^{t-1},\delta^t)$ by appending all possible joint decision rules δ^t for next time step (t). The left side of Figure (3) illustrates the expansion process. After expansion, all created children are heuristically valuated and placed in the open list, any partial joint policies φ^{t+1} with $\widehat{V}(\varphi^{t+1})$ less than the expected value $V(\pi)$ of some earlier found (fully specified) joint policy π, can be pruned. The search ends when the list becomes empty, at which point we have found an optimal fully specified joint policy.

3.4 Dynamic Programming for Dec-POMDPs

MAA* incrementally builds policies from the first stage $t = 0$ to the last $t = h-1$. Prior to this work, Hansen et al. (2004) introduced dynamic programming (DP) for Dec-POMDPs, which constructs policies the other way around: starting with a set of '1-step policies' (actions) that can be executed at the last stage, they construct a set of 2-step policies to be executed at $h-2$, etc.

It should be stressed that the policies maintained are quite different from those used by MAA*. In particular a partial policy in MAA* has the form $\varphi^t = (\delta^0,\delta^1,\dots,\delta^{t-1})$. The policies maintained by DP do not have such a correspondence to decision rules. We define the *time-to-go* τ at stage t as

$$\tau = h - t. \tag{3.3}$$

Now $q_i^{\tau=k}$ denotes a k-steps-to-go *sub-tree policy* for agent i. That is, $q_i^{\tau=k}$ is a policy tree that has the same form as a full policy for the horizon-k problem. Within the original horizon-h problem $q_i^{\tau=k}$ is a candidate for execution starting at stage $t = h-k$. The set of k-steps-to-go sub-tree policies maintained for agent i is denoted $\mathcal{Q}_i^{\tau=k}$. Dynamic programming

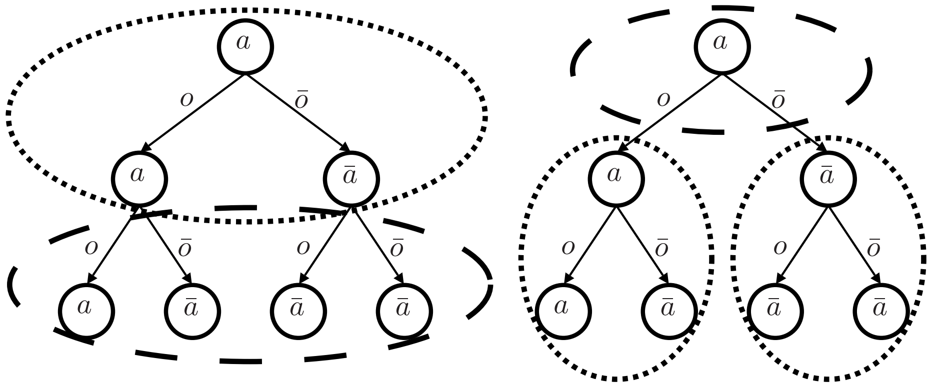


Figure 3: Difference between policy construction in MAA* (left) and dynamic programming (right) for an agent with actions a,$\bar{a}$ and observations o,$\bar{o}$. The dashed components are newly generated, dotted components result from the previous iteration. MAA* 'expands' a partial policy from the leaves, while dynamic programming backs up a set of 'sub-tree policies' forming new ones.

for Dec-POMDPs is based on backup operations: constructing $\mathcal{Q}_i^{\tau=k+1}$ a set of sub-tree policies $q_i^{\tau=k+1}$ from a set $\mathcal{Q}_i^{\tau=k}$. For instance, the right side of Figure 3 shows how $q_i^{\tau=3}$, a 3-steps-to-go sub-tree policy, is constructed from two $q_i^{\tau=2} \in \mathcal{Q}_i^{\tau=2}$. Also illustrated is the difference between this process and MAA* expansion (on the left side).

Dynamic programming consecutively constructs $\mathcal{Q}_i^{\tau=1}$,$\mathcal{Q}_i^{\tau=2}$,$\ldots$,$\mathcal{Q}_i^{\tau=h}$ for all agents i. However, the size of the set $\mathcal{Q}_i^{\tau=k+1}$ is given by

$$|\mathcal{Q}_i^{\tau=k+1}| = |\mathcal{A}_i| \, |\mathcal{Q}_i^{\tau=k}|^{|\mathcal{O}_i|},$$

and as a result the sizes of the maintained sets grow doubly exponential with k. To counter this source of intractability, Hansen et al. (2004) propose to eliminate dominated sub-tree policies. The expected reward of a particular sub-tree policy $q_i^{\tau=k}$ depends on the probability over states when $q_i^{\tau=k}$ is started (at stage $t = h - k$) as well as the probability with which the other agents $j \neq i$ select their sub-tree policies $q_j^{\tau=k} \in \mathcal{Q}_j^{\tau=k}$. If we let $q_{\neq i}^{\tau=k}$ denote a sub-tree profile for all agents but i, and $\mathcal{Q}_{\neq i}^{\tau=k}$ the set of such profiles, we can say that $q_i^{\tau=k}$ is dominated if it is not maximizing at any point in the *multiagent belief* space: the simplex over $S \times \mathcal{Q}_{\neq i}^{\tau=k}$. Hansen et al. test for dominance over the entire multiagent belief space by linear programming. Removal of a dominated sub-tree policy $q_i^{\tau=k}$ of an agent i may cause a sub-tree policy $q_j^{\tau=k}$ of an other agent j to become dominated. Therefore Hansen et al. propose to iterate over agents until no further pruning is possible, a procedure known as *iterated elimination of dominated policies* (Osborne & Rubinstein, 1994).

Finally, when the last backup step is completed the optimal policy can be found by evaluating all joint policies $\pi \in \mathcal{Q}_1^{\tau=h} \times \cdots \times \mathcal{Q}_n^{\tau=h}$ for the initial belief b^0.

3.5 Extensions on DP for Dec-POMDPs

In the last few years several extensions to the dynamic programming algorithm for Dec-POMDPs have been proposed. The first of these extensions is due to Szer and Charpillet (2006). Rather than testing for dominance over the entire multiagent belief space, Szer and Charpillet propose to perform point-based dynamic programming (PBDP). In order to prune the set of sub-tree policies $\mathcal{Q}_i^{\tau=k}$, the set of all the belief points $\mathcal{B}_{i,\text{reachable}} \subset \mathcal{P}(S \times \mathcal{Q}_{\neq i}^{\tau=k})$ that can possibly be reached by deterministic joint policies are generated. Only the sub-tree policies $q_i^{\tau=k}$ that maximize the value at some $b_i \in \mathcal{B}_{i,\text{reachable}}$ are kept. The proposed algorithm is optimal, but intractable because it needs to generate all the multiagent belief points that are reachable through all joint policies. To overcome this bottleneck, Szer and Charpillet propose to randomly sample one or more joint policies and use those to generate $\mathcal{B}_{i,\text{reachable}}$.

Seuken and Zilberstein (2007b) also proposed a point-based extension of the DP algorithm, called memory-bounded dynamic programming (MBDP). Rather than using a randomly selected policy to generate the belief points, they propose to use heuristic policies. A more important difference, however, lies in the pruning step. Rather than pruning dominated sub-tree policies $q_i^{\tau=k}$, MBDP prunes all sub-tree policies except a few in each iteration. More specifically, for each agent *maxTrees* sub-tree policies are retained, which is a parameter of the planning method. As a result, MBDP has only linear space and time complexity with respect to the horizon. The MBDP algorithm still depends on the exhaustive generation of the sets $\mathcal{Q}_i^{\tau=k+1}$ which now contain $|\mathcal{A}_i|\, maxTrees^{|\mathcal{O}_i|}$ sub-tree policies. Moreover, in each iteration all $\left(|\mathcal{A}_*|\, maxTrees^{|\mathcal{O}_*|}\right)^n$ joint sub-tree policies have to be evaluated for each of the sampled belief points. To counter this growth, Seuken and Zilberstein (2007a) proposed an extension that limits the considered observations during the backup step to the *maxObs* most likely observations.

Finally, a further extension of the DP for Dec-POMDPs algorithm is given by Amato, Carlin, and Zilberstein (2007b). Their approach, bounded DP (BDP), establishes a bound not on the used memory, but on the quality of approximation. In particular, BDP uses ϵ-pruning in each iteration. That is, a $q_i^{\tau=k}$ that is maximizing in some region of the multiagent belief space, but improves the value in this region by at most ϵ, is also pruned. Because iterated elimination using ϵ- pruning can still lead to an unbounded reduction in value, Amato et al. propose to perform one iteration of ϵ-pruning, followed by iterated elimination using normal pruning.

3.6 Other Approaches for Finite-Horizon Dec-POMDPs

There are a few other approaches for finite-horizon Dec-POMDPs, which we will only briefly describe here. Aras, Dutech, and Charpillet (2007) proposed a mixed integer linear programming formulation for the optimal solution of finite-horizon Dec-POMDPs. Their approach is based on representing the set of possible policies for each agent in *sequence form* (Romanovskii, 1962; Koller, Megiddo, & von Stengel, 1994; Koller & Pfeffer, 1997). In sequence form, a single policy for an agent i is represented as a subset of the set of 'sequences' (roughly corresponding to action-observation histories) for the agent. As such the problem can be interpreted as a combinatorial optimization problem, which Aras et al. propose to solve with a mixed integer linear program.

Oliehoek, Kooij, and Vlassis (2007a) also recognize that finding a solution for Dec-POMDPs in essence is a combinatorial optimization problem and propose to apply the Cross-Entropy method (de Boer, Kroese, Mannor, & Rubinstein, 2005), a method for combinatorial optimization that recently has become popular because of its ability to find near-optimal solutions in large optimization problems. The resulting algorithm performs a sampling-based policy search for approximately solving Dec-POMDPs. It operates by sampling pure policies from an appropriately parameterized stochastic policy, and then evaluates these policies either exactly or approximately in order to define the next stochastic policy to sample from, and so on until convergence.

Finally, Emery-Montemerlo et al. (2004, 2005) proposed to approximate Dec-POMDPs through series of Bayesian games. Since our work in this article is based on the same representation, we defer a detailed explanation to the next section. We do mention here that while Emery-Montemerlo et al. assume that the algorithm is run on-line (interleaving planning and execution), no such assumption is necessary. Rather we will apply the same framework during a off-line planning phase, just like the other algorithms covered in this overview.

4. Optimal Q-value Functions

In this section we will show how a Dec-POMDP can be modeled as a series of *Bayesian games (BGs)*. A BG is a game-theoretic model that can deal with uncertainty (Osborne & Rubinstein, 1994). Bayesian games are similar to the more well-known *normal form*, or *matrix games,* but allow to model agents that have some private information. This section will introduce Bayesian games and show how a Dec-POMDP can be modeled as a series of Bayesian games (BGs). This idea of using a series of BGs to find policies for a Dec-POMDP has been proposed in an approximate setting by Emery-Montemerlo et al. (2004). In particular, they showed that using series of BGs and an approximate payoff function, they were able to obtain approximate solutions on the Dec-Tiger problem, comparable to results for JESP (see Section 3.2).

The main result of this section is that an optimal Dec-POMDP policy can be computed from the solution of a sequence of Bayesian games, if the payoff function of those games coincides with the Q-value function of an optimal policy π^*, i.e., with the optimal Q-value function Q^*. Thus, we extend the results of Emery-Montemerlo et al. (2004) to include the optimal setting. Also, we conjecture that this form of Q^* can not be computed without already knowing an optimal policy π^*. By transferring the game-theoretic concept of sequential rationality to Dec-POMDPs, we find a description of Q^* that is computable without knowing π^* up front.

4.1 Game-Theoretic Background

Before we can explain how Dec-POMDPs can be modeled using Bayesian games, we will first introduce them together with some other necessary game theoretic background.

	D	C
D	$-1,-1$	$+2,0$
C	$0,+2$	$+1,+1$

	A	B
A	$+2$	0
B	0	$+2$

Figure 4: Left: The game 'Chicken'. Both players have the option to (D)rive on or (C)hicken out. Right: The meeting location problem. Because the game has identical payoffs, each entry contains just one number.

4.1.1 Strategic Form Games and Nash Equilibria

At the basis of the concept of a Bayesian game lies a simpler form of game: the *strategic*- or *normal form game*. A strategic game consists of a set of agents or players, each of which has a set of actions (or strategies). The combination of selected actions specifies a particular outcome. When a strategic game consists of two agents, it can be visualized as a matrix as shown in Figure 4. The first game shown is called 'Chicken' and involves two teenagers who are driving head on. Both have the option to drive on or chicken out. Each teenager's payoff is maximal (+2) when he drives on and his opponent chickens out. However, if both drive on, a collision follows giving both a payoff of -1. The second game is the meeting location problem. Both agents want to meet in location A or B. They have no preference over which location, as long as both pick the same location. This game is fully cooperative, which is modeled by the fact that the agents receive identical payoffs.

Definition 4.1. Formally, a *strategic game* is a tuple $\langle n, \mathcal{A}, u \rangle$, where n is the number of agents, $\mathcal{A} = \times_i \mathcal{A}_i$ is the set of joint actions, and $u = \langle u_1, \ldots, u_n \rangle$ with $u_i : \mathcal{A} \to \mathbb{R}$ is the payoff function of agent i.

Game theory tries to specify for each agent how to play. That is, a game-theoretic solution should suggest a policy for each agent. In a strategic game we write α_i to denote a policy for agent i and α for a joint policy. A policy for agent i is simply one of its actions $\alpha_i = a_i \in \mathcal{A}_i$ (i.e., a pure policy), or a probability distribution over its actions $\alpha_i \in \mathcal{P}(\mathcal{A}_i)$ (i.e., a mixed policy). Also, the policy suggested to each agent should be rational given the policies suggested to the other agent; it would be undesirable to suggest a particular policy to an agent, if it can get a better payoff by switching to another policy. Rather, the suggested policies should form an equilibrium, meaning that it is not profitable for an agent to unilaterally deviate from its suggested policy. This notion is formalized by the concept of Nash equilibrium.

Definition 4.2. A pure policy profile $\alpha = \langle \alpha_1, \ldots, \alpha_i, \ldots, \alpha_n \rangle$ specifying a pure policy for each agent is a *Nash Equilibrium (NE)* if and only if

$$u_i(\langle \alpha_1, \ldots, \alpha_i, \ldots, \alpha_n \rangle) \geq u_i(\langle \alpha_1, \ldots, \alpha_i', \ldots, \alpha_n \rangle), \quad \forall_{i:1 \leq i \leq n}, \forall_{\alpha_i' \in \mathcal{A}_i}. \tag{4.1}$$

This definition can be easily extended to incorporate mixed policies by defining

$$u_i(\langle \alpha_1, \ldots, \alpha_n \rangle) = \sum_{\langle a_1, \ldots, a_n \rangle} u_i(\langle a_1, \ldots, a_n \rangle) \prod_{i=1}^{n} P_{\alpha_i}(a_i).$$

Nash (1950) proved that when allowing mixed policies, every (finite) strategic game contains at least one NE, making it a proper solution for a game. However, it is unclear how such a NE should be found. In particular, there may be multiple NEs in a game, making it unclear which one to select. In order to make some discrimination between Nash equilibria, we can consider NEs such that there is no other NE that is better for everyone.

Definition 4.3. A Nash Equilibrium $\alpha = \langle \alpha_1, \ldots, \alpha_i, \ldots, \alpha_n \rangle$ is referred to as *Pareto Optimal (PO)* when there is no other NE α' that specifies at least the same payoff for all agents and a higher payoff for at least one agent:

$$\nexists_{\alpha'} \quad \left(\forall_i \, u_i(\alpha') \geq u_i(\alpha) \;\wedge\; \exists_i \, u_i(\alpha') > u_i(\alpha) \right).$$

In the case when multiple Pareto optimal Nash equilibria exist, the agents can agree beforehand on a particular ordering, to ensure the same NE is chosen.

4.1.2 Bayesian Games

A Bayesian game (Osborne & Rubinstein, 1994) is an augmented normal form game in which the players hold some private information. This private information defines the *type* of the agent, i.e., a particular type $\theta_i \in \Theta_i$ of an agent i corresponds to that agent knowing some particular information. The payoff the agents receive now no longer only depends on their actions, but also on their private information. Formally, a BG is defined as follows:

Definition 4.4. A *Bayesian game (BG)* is a tuple $\langle n, \mathcal{A}, \Theta, P(\Theta), \langle u_1, ... u_n \rangle \rangle$, where n is the number of agents, $\mathcal{A}$ is the set of joint actions, $\Theta = \times_i \Theta_i$ is the set of joint types over which a probability function $P(\Theta)$ is specified, and $u_i : \Theta \times \mathcal{A} \rightarrow \mathbb{R}$ is the payoff function of agent i.

In a normal form game the agents select an action. Now, in a BG the agents can condition their action on their private information. This means that in BGs the agents use a different type of policies. For a BG, we denote a joint policy $\beta = \langle \beta_1, ..., \beta_n \rangle$, where the individual policies are mappings from types to actions: $\beta_i : \Theta_i \rightarrow \mathcal{A}_i$. In the case of identical payoffs for the agents, the solution of a BG is given by the following theorem:

Theorem 4.1. *For a BG with identical payoffs, i.e., $\forall_{i,j} \forall_\theta \forall_a \; u_i(\theta,a) = u_j(\theta,a)$, the solution is given by:*

$$\beta^* = \arg\max_{\beta} \sum_{\theta \in \Theta} P(\theta) u(\theta, \beta(\theta)), \tag{4.2}$$

where $\beta(\theta) = \langle \beta_1(\theta_1), ..., \beta_n(\theta_n) \rangle$ is the joint action specified by β for joint type θ. This solution constitutes a Pareto optimal Nash equilibrium.

Proof. The proof consists of two parts: the first shows that β^* is a Nash equilibrium, the second shows it is Pareto optimal.

Nash equilibrium proof. It is clear that β^* satisfying 4.2 is a Nash equilibrium by rewriting from the perspective of an arbitrary agent i as follows:

$$\begin{aligned}
\beta_i^* &= \arg\max_{\beta_i} \left[\max_{\beta_{\neq i}} \sum_{\theta \in \Theta} P(\theta) u(\theta, \beta(\theta)) \right], \\
&= \arg\max_{\beta_i} \left[\max_{\beta_{\neq i}} \sum_{\theta_i} \sum_{\theta_{\neq i}} P(\theta_{\neq i}|\theta_i) \underbrace{\left[\sum_{\theta_{\neq i}} P(\langle \theta_i, \theta_{\neq i} \rangle) \right]}_{P(\theta_i)} u(\theta, \beta(\theta)) \right], \\
&= \arg\max_{\beta_i} \left[\max_{\beta_{\neq i}} \sum_{\theta_i} P(\theta_i) \sum_{\theta_{\neq i}} P(\theta_{\neq i}|\theta_i) u(\theta, \beta(\theta)) \right], \\
&= \arg\max_{\beta_i} \sum_{\theta_i} P(\theta_i) \sum_{\theta_{\neq i}} P(\theta_{\neq i}|\theta_i) u(\langle \theta_i, \theta_{\neq i} \rangle, \langle \beta_i(\theta_i), \beta^*_{\neq i}(\theta_{\neq i}) \rangle),
\end{aligned}$$

which means that β_i^* is a best response for $\beta^*_{\neq i}$. Since no special assumptions were made on i, it follows that β^* is a Nash equilibrium.

Pareto optimality proof. Let us write $V_{\theta_i}(a_i, \beta_{\neq i})$ for the payoff agent i expects for θ_i when performing a_i when the other agents use policy profile $\beta_{\neq i}$. We have that

$$V_{\theta_i}(a_i, \beta_{\neq i}) = \sum_{\theta_{\neq i}} P(\theta_{\neq i}|\theta_i) u(\langle \theta_i, \theta_{\neq i} \rangle, \langle a_i, \beta_{\neq i}(\theta_{\neq i}) \rangle).$$

Now, a joint policy β^* satisfying (4.2) is not Pareto optimal if and only if there is another Nash equilibrium β' that attains at least the same payoff for all agents i and for all types θ_i and strictly more for at least one agent and type. Formally β^* is not Pareto optimal when $\exists \beta'$ such that:

$$\forall_i \forall_{\theta_i} \quad V_{\theta_i}(\beta_i^*(\theta_i), \beta^*_{\neq i}) \leq V_{\theta_i}({\beta_i}'(\theta_i), {\beta_{\neq i}}') \ \wedge \ \exists_i \exists_{\theta_i} V_{\theta_i}(\beta_i^*(\theta_i), \beta^*_{\neq i}) < V_{\theta_i}({\beta_i}'(\theta_i), {\beta_{\neq i}}'). \quad (4.3)$$

We prove that no such β' can exist by contradiction. Suppose that $\beta' = \langle {\beta_i}', \beta'_{\neq i} \rangle$ is a NE such that (4.3) holds (and thus β^* is not Pareto optimal). Because β^* satisfies (4.2) we know that:

$$\sum_{\theta \in \Theta} P(\theta) u(\theta, \beta^*(\theta)) \geq \sum_{\theta \in \Theta} P(\theta) u(\theta, \beta'(\theta)), \quad (4.4)$$

and therefore, for all agents i

$$\begin{aligned}
P(\theta_{i,1}) V_{\theta_{i,1}}(\beta_i^*(\theta_{i,1}), \beta^*_{\neq i}) + \ldots + P(\theta_{i,|\Theta_i|}) V_{\theta_{i,|\Theta_i|}}(\beta_i^*(\theta_{i,|\Theta_i|}), \beta^*_{\neq i}) \geq \\
P(\theta_{i,1}) V_{\theta_{i,1}}(\beta'_i(\theta_{i,1}), \beta'_{\neq i}) + \ldots + P(\theta_{i,|\Theta_i|}) V_{\theta_{i,|\Theta_i|}}(\beta'_i(\theta_{i,|\Theta_i|}), \beta'_{\neq i})
\end{aligned}$$

holds. However, by assumption that β' satisfies (4.3) we get that

$$\exists_j \quad V_{\theta_{i,j}}(\beta_i^*(\theta_{i,j}), \beta^*_{\neq i}) < V_{\theta_{i,j}}(\beta'_i(\theta_{i,j}), \beta'_{\neq i}).$$

Therefore it must be that

$$\sum_{k \neq j} P(\theta_{i,k}) V_{\theta_{i,k}}(\beta_i^*(\theta_{i,k}), \beta^*_{\neq i}) > \sum_{k \neq j} P(\theta_{i,k}) V_{\theta_{i,k}}(\beta'_i(\theta_{i,k}), \beta'_{\neq i}),$$

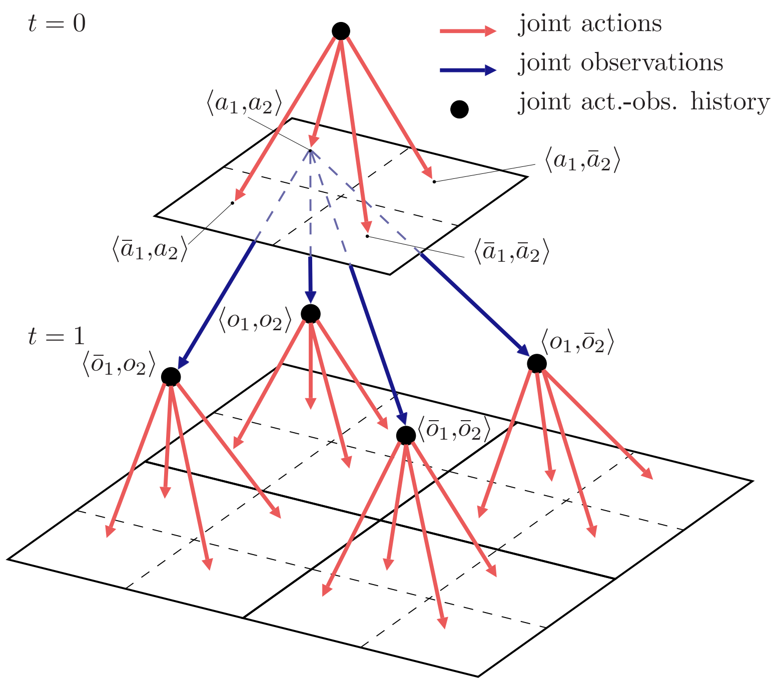


Figure 5: A Dec-POMDP can be seen as a tree of joint actions and observations. The indicated planes correspond with the Bayesian games for the first two stages.

and thus that

$$\exists_k \quad V_{\theta_{i,k}}(\beta^*_i(\theta_{i,k}),\beta^*_{\neq i}) > V_{\theta_{i,k}}(\beta'_i(\theta_{i,k}),\beta'_{\neq i}),$$

contradicting the assumption that β' satisfies (4.3). □

4.2 Modeling Dec-POMDPs with Series of Bayesian Games

Now we will discuss how Bayesian games can be used to model Dec-POMDPs. Essentially, a Dec-POMDP can be seen as a tree where nodes are joint action-observation histories and edges represent joint actions and observations, as illustrated in Figure 5. At a specific stage t in a Dec-POMDP, the main difficulty in coordinating action selection is presented by the fact that each agent has its own individual action-observation history. That is, there is no global signal that the agents can use to coordinate their actions. This situation can be conveniently modeled by a Bayesian game as we will now discuss.

At a time step t, one can directly associate the primitives of a Dec-POMDP with those of a BG with identical payoffs: the actions of the agents are the same in both cases, and the types of agent i correspond to its action-observation histories $\Theta_i \equiv \vec{\Theta}^t_i$. Figure 6 shows the Bayesian games for $t = 0$ and $t = 1$ for a fictitious Dec-POMDP with 2 agents.

We denote the payoff function of the BG that models a stage of a Dec-POMDP by $Q(\vec{\theta}^{\,t},a)$. This payoff function should be naturally defined in accordance with the value function of the planning task. For instance, Emery-Montemerlo et al. (2004) define $Q(\vec{\theta}^{\,t},a)$

as the Q_{MDP}-value of the underlying MDP. We will more extensively discuss the payoff function in Section 4.3.

The probability $P(\theta)$ is equal to the probability of the joint action-observation history to which θ corresponds and depends on the past joint policy $\varphi^t = (\delta^0, \ldots, \delta^{t-1})$ and the initial state distribution. It can be calculated as the marginal of (2.7):

$$P(\theta) = P(\vec{\theta}^t|\varphi^t,b^0) = \sum_{s^t \in \mathcal{S}} P(s^t,\vec{\theta}^t|\varphi^t,b^0). \tag{4.5}$$

When only considering pure joint policies φ^t, the action probability component $P_\varphi(a|\vec{\theta})$ in (2.7) is 1 for joint action-observation histories $\vec{\theta}^t$ that are 'consistent' with the past joint policy φ^t and 0 otherwise. We say that an action-observation $\vec{\theta}_i$ history is consistent with a pure policy π_i if it can occur when executing π_i, i.e., when the actions in $\vec{\theta}_i$ would be selected by π_i. Let us more formally define this consistency as follows.

Definition 4.5 (Consistency)**.** Let us write $\vec{\theta}_i^{t'}$ for the restriction of $\vec{\theta}_i^t$ to stage $0, \ldots, t'$ (with $0 \leq t' < t$). An action-observation history $\vec{\theta}_i^t$ of agent i is *consistent* with a pure policy π_i if and only if at each time step t' with $0 \leq t' < t$

$$\pi_i(\vec{\theta}_i^{t'}) = \pi_i(\vec{o}_i^{t'}) = a_i^{t'}$$

is the $(t'+1)$-th action in $\vec{\theta}_i^t$. A joint action-observation history $\vec{\theta}^t = \langle \vec{\theta}_1^t, \ldots, \vec{\theta}_n^t \rangle$ is consistent with a pure joint policy $\pi = \langle \pi_1, \ldots, \pi_n \rangle$ if each individual $\vec{\theta}_i^t$ is consistent with the corresponding individual policy π_i. C is the indicator function for consistency. For instance $C(\vec{\theta}^t,\pi)$ 'filters out' the action-observation histories $\vec{\theta}^t$ that are inconsistent with a joint pure policy π:

$$C(\vec{\theta}^t,\pi) = \begin{cases} 1 & , \vec{\theta}^t = \big(o^0,\pi(o^0),o^1,\pi(o^0,o^1),\ldots\big) \,) \\ 0 & , \text{otherwise.} \end{cases} \tag{4.6}$$

We will also write $\vec{\Theta}_\pi^t \equiv \{\vec{\theta}^t \mid C(\vec{\theta}^t,\pi) = 1\}$ for the set of $\vec{\theta}^t$ consistent with π.

This definition allows us to write

$$P(\vec{\theta}^t|\varphi^t,b^0) = C(\vec{\theta}^t,\varphi^t) \sum_{s^t \in \mathcal{S}} P(s^t,\vec{\theta}^t|b^0) \tag{4.7}$$

with

$$P(s^t,\vec{\theta}^t|b^0) = \sum_{s^{t-1} \in \mathcal{S}} P(o^t|a^{t-1},s^t) P(s^t|s^{t-1},a^{t-1}) P(s^{t-1},\vec{\theta}^{t-1}|b^0). \tag{4.8}$$

Figure 6 illustrates how the indicator function 'filters out' policies, when $\pi^{t=0}(\vec{\theta}^{t=0}) = \langle a_1,a_2 \rangle$, only the non-shaded part of the BG for $t = 1$ 'can be reached' (has positive probability).

$\vec{\theta}_1^{t=0}$ \ $\vec{\theta}_2^{t=0}$		()	
		a_2	$\bar{a}_2$
()	a_1	+2.75	−4.1
	$\bar{a}_1$	−0.9	+0.3

$\vec{\theta}_1^{t=1}$ \ $\vec{\theta}_2^{t=1}$		(a_2,o_2)		$(a_2,\bar{o}_2)$		...
		a_2	$\bar{a}_2$	a_2	$\bar{a}_2$	
(a_1,o_1)	a_1	−0.3	+0.6	−0.6	+4.0	...
	$\bar{a}_1$	−0.6	+2.0	−1.3	+3.6	...
$(a_1,\bar{o}_1)$	a_1	+3.1	+4.4	−1.9	+1.0	...
	$\bar{a}_1$	+1.1	−2.9	+2.0	−0.4	
$(\bar{a}_1,o_1)$	a_1	−0.4	−0.9	−0.5	−1.0	...
	$\bar{a}_1$	−0.9	−4.5	−1.0	+3.5	...
$(\bar{a}_1,\bar{o}_1)$	...	...	...	...	...	...

Figure 6: The Bayesian game for the first and second time step (top: $t = 0$, bottom: $t = 1$). The entries $\vec{\theta}^t$, a^t are given by the payoff function $Q(\vec{\theta}^t,a^t)$. Light shaded entries indicate the solutions. Dark entries will not be realized given $\langle a_1,a_2\rangle$ the solution of the BG for $t = 0$.

4.3 The Q-value Function of an Optimal Joint Policy

Given the perspective of a Dec-POMDP interpreted as a series of BGs as outlined in the previous section, the solution of the BG for stage t is a joint decision rule δ^t. If the payoff function for the BG is chosen well, the quality of δ^t should be high. Emery-Montemerlo et al. (2004) try to find a good joint policy $\pi = (\delta^0, \ldots ,\delta^{h-1})$ by a procedure we refer to as *forward-sweep policy computation (FSPC)*: in one sweep forward through time, the BG for each stage $t = 0,1, \ldots ,h-1$ is consecutively solved. As such, the payoff function for the BGs constitute what we call a Q-value function for the Dec-POMDP.

Here, we show that there is an optimal Q-value function Q^*: when using this Q^* as the payoff functions for the BGs, forward-sweep policy computation will lead to an optimal joint policy $\pi^* = (\delta^{0,*}, \ldots ,\delta^{h-1,*})$. We first give a derivation of this Q^*. Next, we will discuss that Q^* can indeed be used to calculate π^*, but computing Q^* seems impractical without already knowing an optimal joint policy π^*. This issue will be further addressed in Section 4.4.

4.3.1 Existence of Q^*

We now state a theorem identifying a normative description of Q^* as the Q-value function for an optimal joint policy.

Theorem 4.2. *The expected cumulative reward over stages $t, \ldots ,h-1$ induced by π^*, an optimal joint policy for a Dec-POMDP, is given by:*

$$V^t(\pi^*) = \sum_{\vec{\theta}^t \in \vec{\Theta}^t_{\pi^*}} P(\vec{\theta}^t|b^0)Q^*(\vec{\theta}^t,\pi^*(\vec{\theta}^t)), \tag{4.9}$$

where $\vec{\theta}^t = \langle \vec{o}^t, \vec{a}^t \rangle$, *where* $\pi^*(\vec{\theta}^t) = \pi^*(\vec{o}^t)$ *denotes the joint action that pure joint policy* π^* *specifies for* $\vec{o}^t$, *and where*

$$Q^*(\vec{\theta}^t, a) = R(\vec{\theta}^t, a) + \sum_{o^{t+1} \in \mathcal{O}} P(o^{t+1}|\vec{\theta}^t, a) Q^*(\vec{\theta}^{t+1}, \pi^*(\vec{\theta}^{t+1})) \tag{4.10}$$

is the Q-value function for π^*, *which gives the expected cumulative future reward when taking joint action* a *at* $\vec{\theta}^t$ *given that an optimal joint policy* π^* *is followed hereafter.*

Proof. By filling out (2.11) for an optimal pure joint policy π^*, we obtain its expected cumulative reward as the summation of $E\left[R(s^t, a^t) \middle| \pi^*\right]$ the expected rewards it yields for each time step:

$$V(\pi^*) = \sum_{t=0}^{h-1} E\left[R(s^t, a^t) \middle| \pi^*\right] = \sum_{t=0}^{h-1} \sum_{\vec{\theta}^t \in \vec{\Theta}^t} P(\vec{\theta}^t | \pi^*, b^0) R(\vec{\theta}^t, \pi^*(\vec{\theta}^t)). \tag{4.11}$$

In this equation, $P(\vec{\theta}^t | \pi^*, b^0)$ is given by (4.7). As a result, the influence of π^* on $P(\vec{\theta}^t | \pi^*, b^0)$ is only through C. I.e., π^* is only used to 'filter out' inconsistent histories. Therefore we can write:

$$E\left[R(s^t, a^t) \middle| \pi^*\right] = \sum_{\vec{\theta}^t \in \vec{\Theta}^t_{\pi^*}} P(\vec{\theta}^t | b^0) R(\vec{\theta}^t, \pi^*(\vec{\theta}^t)), \tag{4.12}$$

where $P(\vec{\theta}^t | b^0)$ is given by directly taking the marginal of (4.8). Now, let us define the value starting from time step t:

$$V^t(\pi^*) = E\left[R(s^t, a^t) \middle| \pi^*\right] + V^{t+1}(\pi^*) = \sum_{\vec{\theta}^t \in \vec{\Theta}^t_{\pi^*}} P(\vec{\theta}^t | b^0) R(\vec{\theta}^t, \pi^*(\vec{\theta}^t)) + V^{t+1}(\pi^*). \tag{4.13}$$

For the last time step $h-1$ there is no expected future reward, so we get:

$$V^{h-1}(\pi^*) = \sum_{\vec{\theta}^{h-1} \in \vec{\Theta}^{h-1}_{\pi^*}} P(\vec{\theta}^{h-1} | b^0) \underbrace{R(\vec{\theta}^{h-1}, \pi^*(\vec{\theta}^{h-1}))}_{Q^*(\vec{\theta}^{h-1}, \pi^*(\vec{\theta}^{h-1}))}. \tag{4.14}$$

For time step $h-2$ this becomes:

$$\begin{aligned} V^{h-2}(\pi^*) \equiv\ & E\left[R(s^{h-2}, a^{h-2}) \middle| \pi^*\right] + V^{h-1}(\pi^*) = \\ & \sum_{\vec{\theta}^{h-2} \in \vec{\Theta}^{h-2}_{\pi^*}} P(\vec{\theta}^{h-2} | b^0) R(\vec{\theta}^{h-2}, \pi^*(\vec{\theta}^{h-2})) + \sum_{\vec{\theta}^{h-1} \in \vec{\Theta}^{h-1}_{\pi^*}} P(\vec{\theta}^{h-1} | b^0) Q^*(\vec{\theta}^{h-1}, \pi^*(\vec{\theta}^{h-1})). \end{aligned} \tag{4.15}$$

Because $P(\vec{\theta}^{h-1}) = P(\vec{\theta}^{h-2}) P(o^{h-1} | \vec{\theta}^{h-2}, \pi^*(\vec{\theta}^{h-2}))$, (4.15) can be rewritten to:

$$V^{h-2}(\pi^*) = \sum_{\vec{\theta}^{h-2} \in \vec{\Theta}^{h-2}_{\pi^*}} P(\vec{\theta}^{h-2} | b^0) Q^*(\vec{\theta}^{h-2}, \pi^*(\vec{\theta}^{h-2})), \tag{4.16}$$

with

$$Q^*(\vec{\theta}^{h-2},\pi^*(\vec{\theta}^{h-2})) = R(\vec{\theta}^{h-2},\pi^*(\vec{\theta}^{h-2}))+ \sum_{o^{h-1}} P(o^{h-1}|\vec{\theta}^{h-2},\pi^*(\vec{\theta}^{h-2}))Q^*(\vec{\theta}^{h-1},\pi^*(\vec{\theta}^{h-1})). \quad (4.17)$$

Reasoning in the same way we see that (4.9) and (4.10) constitute a generic expression for the expected cumulative future reward starting from time step t. □

Note that in the above derivation, we explicitly included b^0 as one of the given arguments. In the rest of this text, we will always assume b^0 is given and therefore omit it, unless necessary.

4.3.2 Deriving an Optimal Joint Policy from Q^*

At this point we have derived Q^*, a Q-value function for an optimal joint policy. Now, we extend the results of Emery-Montemerlo et al. (2004) into the exact setting:

Theorem 4.3. *Applying forward-sweep policy computation using Q^* as defined by* (4.10) *yields an optimal joint policy.*

Proof. Note that, per definition, the optimal Dec-POMDP policy π^* maximizes the expected future reward $V^t(\pi^*)$ specified by (4.9). Therefore $\delta^{t,*}$, the optimal decision rule for stage t, is identical to an optimal joint policy $\beta^{t,*}$ for the Bayesian game for time step t, if the payoff function of the BG is given by Q^*, that is:

$$\delta^{t,*} \equiv \beta^{t,*} = \arg\max_{\beta^t} \sum_{\vec{\theta}^t \in \vec{\Theta}^t_{\pi^*}} P(\vec{\theta}^t)Q^*(\vec{\theta}^t,\beta^t(\vec{\theta}^t)). \quad (4.18)$$

Equation (4.18) tells us that $\delta^{t,*} \equiv \beta^{t,*}$. This means that it is possible to construct the complete optimal Dec-POMDP policy $\pi^* = (\delta^{0,*}, \ldots, \delta^{h-1,*})$, by computing $\delta^{t,*}$ for all t. □

A subtlety in the calculation of π^* is that (4.18) itself is dependent on an optimal joint policy, as the summation is over all $\vec{\theta}^t \in \vec{\Theta}^t_{\pi^*} \equiv \{\vec{\theta}^t \mid C(\vec{\theta}^t,\pi^*) = 1\}$. This is resolved by realizing that only the past actions influence which action-observation histories can be reached at time step t. Formally, let $\varphi^t = (\delta^{0,*}, \ldots, \delta^{t-1,*})$ denote the past joint policy, which is a partial joint policy π specified for stages 0,...,$t-1$. If we denote the optimal past joint policy by $\varphi^{t,*}$, we have that $\vec{\Theta}^t_{\pi^*} = \vec{\Theta}^t_{\varphi^{t,*}}$, and therefore that:

$$\beta^{t,*} = \arg\max_{\beta^t} \sum_{\vec{\theta}^t \in \vec{\Theta}^t_{\varphi^{t,*}}} P(\vec{\theta}^t)Q^*(\vec{\theta}^t,\beta^t(\vec{\theta}^t)). \quad (4.19)$$

This can be solved in a forward manner for time steps $t = 0,1,2,...,h-1$, because at every time step $\varphi^{t,*} = (\delta^{0,*}, \ldots, \delta^{t-1,*})$ will be available: it is specified by $(\beta^{0,*},...,\beta^{t-1,*})$ the solutions of the previously solved BGs.

4.3.3 Computing Q^*

So far we discussed that Q^* can be used to find an optimal joint policy π^*. Unfortunately, when an optimal joint policy π^* is not known, computing Q^* itself is impractical, as we will discuss here. This is in contrast with the (fully observable) single-agent case where the optimal Q-values can be found relatively easily in a single sweep backward through time.

For MDPs and POMDPs we can compute the Q-values for time step t from those for $t+1$ by applying a backup operator. This is possible because there is a single agent that perceives a Markovian signal. This allows the agent to (1) select the optimal action (policy) for the next time step and (2) determine the expected future reward given the optimal action (policy) found in step 1. For instance, the backup operator for a POMDP is given by:

$$Q^*(b^t,a) = R(b^t,a) + \sum_o P(o|b^t,a) \max_a Q^*(b^{t+1},a),$$

which can be rewritten as a 2-step procedure:

1. $\pi^{t+1,*}(b^{t+1}) = \arg\max_{a'} Q^*(b^{t+1},a')$
2. $Q^*(b^t,a) = R(b^t,a) + \sum_o P(o|b^t,a) Q^*(b^{t+1},\pi^{t+1,*}(b^{t+1}))$.

In the case of Dec-POMDPs, step 2 would correspond to calculating Q^* using (4.10) and thus depends on $\pi^{t+1,*}$ an optimal joint policy at the next stage. However, step 1 that calculates $\pi^{t+1,*}$, corresponds to (4.19) and therefore is dependent on $\varphi^{t+1,*}$ (an optimal joint policy for time steps 0,...,t). So to calculate the $Q^{t,*}$ the optimal Q-value function as specified by (4.10) for stage t, an optimal joint policy up to and including stage t is needed. Effectively, there is a dependence on both the future and the past optimal policy, rather than only on the future optimal policies as in the single agent case. The only clear solution seems to be evaluation for all possible past policies, as detailed next. We conjecture that the problem encountered here is inherent to all decentralized decision making with imperfect information. For example, we can also observe this in exact point-based dynamic programming for Dec-POMDPs, as described in Section 3.5, where it is necessary to to generate all (multiagent belief points generated by all) possible past policies.

4.4 Sequential Rationality for Dec-POMDPs

We conjectured that computing Q^* as introduced in Section 4.3 seems impractical without knowing π^*. Here we will relate this to concepts from game theory. In particular, we discuss a different formulation of Q^* based on the principle of sequential rationality, i.e., also considering joint action-observation histories that are not realized given an optimal joint policy. This formulation of Q^* is computable without knowing an optimal joint policy in advance, and we present a dynamic programming algorithm to perform this computation.

4.4.1 Sub-game Perfect and Sequential Equilibria

The problem we are facing is very much related to the notion of *sub-game perfect equilibria* from game theory. A sub-game perfect Nash equilibrium $\pi = \langle \pi_1, \ldots, \pi_n \rangle$ has the characteristic that the contained policies π_i specify an optimal action for *all* possible situations—even situations that can not occur when following π. A commonly given rationale behind this concept is that, by a mistake of one of the agents during execution, situations that should not occur according to π, can occur, and also in these situations the agents should act optimally. A different rationale is given by Binmore (1992), who remarks that it is "tempting to shrug one's shoulders at these difficulties [because] rational players will not stray from the equilibrium path", but that would clearly be a mistake, because the agents "remain on the equilibrium path because of what they anticipate *would* happen if they *were* to deviate". This implies that agents can decide upon a Nash equilibrium by analyzing what the expected outcome would be by following other policies: That is, when acting optimally from other situations. We will perform a similar reasoning here for Dec-POMDPs, which—in a similar fashion—will result in a description that allows to deduce an optimal Q-value function and thus joint policy.

A Dec-POMDP can be modeled as an extensive form game of imperfect information (Oliehoek & Vlassis, 2006). For such games, the notion of sub-game perfect equilibria is inadequate; because this type of games often do not contain proper sub-games, every Nash equilibrium is trivially sub-game perfect.[4] To overcome this problem different refinements of the Nash equilibrium concept have been defined, of which we will mention the *assessment equilibrium* (Binmore, 1992) and the closely related, but stronger *sequential equilibrium* (Osborne & Rubinstein, 1994). Both these equilibria are based on the concept of an assessment, which is a pair $\langle\pi,\mathbf{b}\rangle$ consisting of a joint policy π and a *belief system* $\mathbf{b}$. The belief system maps each possible situation, or information set, of an agent—also the ones that are not reachable given π—to a probability distribution over possible joint histories. Roughly speaking, an assessment equilibrium requires *sequential rationality* and *belief consistency*.[5] The former entails that the joint policy π specifies optimal actions for each information set given $\mathbf{b}$. Belief consistency means that all the beliefs that are assigned by $\mathbf{b}$ are Bayes rational given the specified joint policy π. For instance, in the context of Dec-POMDPs $\mathbf{b}$ would prescribe, for a particular $\vec{\theta}_i^t$ of agent i, a belief over joint histories $P(\vec{\theta}^t|\vec{\theta}_i^t)$. If all beliefs prescribed by belief system $\mathbf{b}$ are Bayes-rational (i.e., computed as the appropriate conditionals of (4.5)), $\mathbf{b}$ is called belief consistent.[6]

4.4.2 Sequential Rationality and the Optimal Q-value Function

The dependence of sequential rationality on a belief system $\mathbf{b}$ indicates that the optimal action at a particular point is dependent on the probability distribution over histories. In Section 4.3.3 we encountered a similar dependence on the history as specified by $\varphi^{t+1,*}$. Here we will make this dependence more exact.

At a particular stage t, a policy is optimal or, in game-theoretic terms, rational if it maximizes the expected return from that point on. In Section 4.3.1, we were able to express this expected return as $Q^*(\vec{\theta}^t,a)$ *assuming an optimal joint policy* π^* *is followed up*

4. The extensive form of a Dec-POMDP indeed does not contain proper sub-games, because agent can never discriminate between the other agents' observations.
5. Osborne and Rubinstein (1994) refer to this second requirement as simply 'consistency'. In order to avoid any confusion with definition 4.5 we will use the term 'belief consistency'.
6. A sequential equilibrium includes a more technical part in the definition of belief consistency that addresses what beliefs should be held for information sets that are not reached according to π. For more information we refer to Osborne and Rubinstein (1994).

to the current stage t. However, when no such previous policy is assumed, the maximal expected return is not defined.

Proposition 4.1. *For a pair $(\vec{\theta}^t, a^t)$ with $t < h-1$ the optimal value $Q^*(\vec{\theta}^t, a^t)$ cannot be defined without assuming some (possibly randomized) past policy $\varphi^{t+1} = (\delta^0, \ldots, \delta^t)$. Only for the last stage $t = h-1$ such expected reward is defined as*

$$Q^*(\vec{\theta}^{h-1}, a^{h-1}) \equiv R(\vec{\theta}^{h-1}, a^{h-1})$$

without assuming a past policy.

Proof. Let us try to deduce $Q^*(\vec{\theta}^t, a^t)$ the optimal value for a particular $\vec{\theta}^t$ assuming the Q^*-values for the next time step $t+1$ are known. The $Q^*(\vec{\theta}^t, a^t)$-values for each of the possible joint actions can be evaluated as follows

$$\forall_a \quad Q^*(\vec{\theta}^t, a^t) = R(\vec{\theta}^t, a^t) + \sum_{o^{t+1}} P(o^{t+1}|\vec{\theta}^t, a^t) Q^*(\vec{\theta}^{t+1}, \delta^{t+1,*}(\vec{\theta}^{t+1})).$$

where $\delta^{t+1,*}$ is an optimal decision rule for the next stage. But what should $\delta^{t+1,*}$ be? If we assume that up to stage $t+1$ we followed a particular (possibly randomized) φ^{t+1},

$$\delta_{\varphi}^{t+1,*} = \arg\max_{\beta^{t+1}} \sum_{\vec{\theta}^{t+1} \in \vec{\Theta}^{t+1}} P(\vec{\theta}^{t+1}|\varphi^{t+1}, b^0) Q^*(\vec{\theta}^{t+1}, \beta^{t+1}(\vec{\theta}^{t+1})).$$

is optimal. However, there are many pure and infinite randomized past policies φ^{t+1} that are consistent with $\vec{\theta}^t, a^t$, leading to many $\delta_{\varphi}^{t+1,*}$ that might be optimal. The conclusion we can draw is that $Q^*(\vec{\theta}^t, a^t)$ is ill-defined without $P(\vec{\theta}^{t+1}|\varphi^{t+1}, b^0)$, the probability distribution (belief) over joint action-observation histories, which is induced by φ^{t+1}, the policy followed for stages $0, \ldots, t$. □

Let us illustrate this by reviewing the optimal Q-value function as defined in Section 4.3.1. Consider $\pi^*(\vec{\theta}^{t+1})$ in (4.10). This optimal policy is a mapping from observation histories to actions $\pi^* : \vec{\mathcal{O}} \to \mathcal{A}$ induced by the individual policies and observation histories. This means that for two joint action-observation histories with the same joint observation history π^* results in the same joint action. That is $\forall \vec{a}, \vec{o}, \vec{a}'\ \pi^*(\langle \vec{a}, \vec{o} \rangle) = \pi^*(\langle \vec{a}', \vec{o} \rangle)$. Effectively this means that when we reach some $\vec{\theta}^t \notin \vec{\Theta}^t_{\pi^*}$, say through a mistake[7], π^* continues to specify actions as if no mistake ever happened: That is, *still* assuming that π^* has been followed up to this stage t. In fact, $\pi^*(\vec{\theta}^t)$ *might not even be optimal* if $\vec{\theta}^t \notin \vec{\Theta}^t_{\pi^*}$. Which in turn means that $Q^*(\vec{\theta}^{t-1}, a)$, the Q-values for predecessors of $\vec{\theta}^t$, might not be the optimal expected reward.

We demonstrated that the optimal Q-value function for a Dec-POMDP is not well-defined without assuming a past joint policy. We propose a new definition of Q^* that explicitly incorporates φ^{t+1}.

7. The question as to how the mistake of one agent should be detected by another agent is a different matter altogether and beyond the scope of this text.

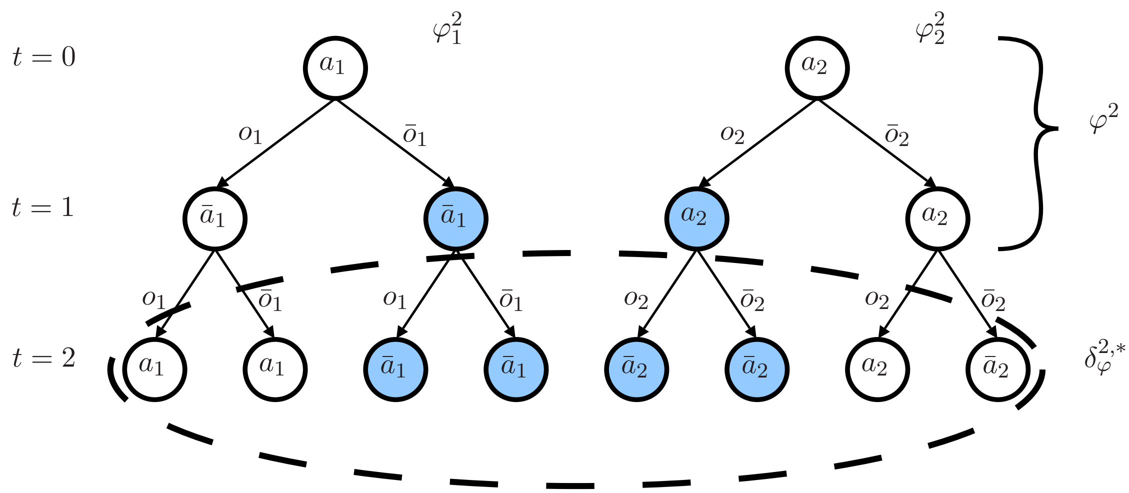


Figure 7: Computation of sequential rational Q^*. $\delta_\varphi^{2,*}$ is the optimal decision rule for stage $t = 2$, given that φ^2 is followed for the first two stages. $Q^*(\vec{\theta}^1,\varphi^2)$ entries are computed by propagating relevant Q^*-values of the next stage. For instance, for the highlighted joint history $\vec{\theta}^1 = \langle(a_1,\bar{o}_1),(a_2,o_2)\rangle$, the Q^*-value under φ^2 is computed by propagating the values of the four successor joint histories, as per (4.20).

Theorem 4.4 (Sequentially rational Q^*)**.** *The optimal Q-value function is properly defined as a function of joint action-observation histories and past joint policies, $Q^*(\vec{\theta}^t,\varphi^{t+1})$. This Q^* specifies the optimal value given for all $(\vec{\theta}^t,\varphi^{t+1})$, even for $\vec{\theta}^t$ that are not reached by execution of an optimal policy π^*, and therefore is referred to as sequentially rational.*

Proof. For all $\vec{\theta}^t,\varphi^{t+1}$, the optimal expected return is given by

$$Q^*(\vec{\theta}^t,\varphi^{t+1}) = \begin{cases} R(\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t)), & t = h-1 \\ R(\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t)) + \displaystyle\sum_{o^{t+1}} P(o^{t+1}|\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t))Q^*(\vec{\theta}^{t+1},\varphi^{t+2,*}), & 0 \le t < h-1 \end{cases} \tag{4.20}$$

where $\varphi^{t+2,*} = \left(\varphi^{t+1},\delta_\varphi^{t+1,*}\right)$ and

$$\delta_\varphi^{t+1,*} = \arg\max_{\beta^{t+1}} \sum_{\vec{\theta}^{t+1}\in\vec{\Theta}^{t+1}} P(\vec{\theta}^{t+1}|\varphi^{t+1},b^0)Q^*(\vec{\theta}^{t+1},\left(\varphi^{t+1},\beta^{t+1}\right)). \tag{4.21}$$

which is well-defined. $\square$

The above equations constitute a dynamic program. When assuming that only pure joint past policies φ can be used, (4.21) transforms to

$$\delta_\varphi^{t+1,*} = \arg\max_{\beta^{t+1}} \sum_{\vec{\theta}^{t+1}\in\vec{\Theta}_\varphi^{t+1}} P(\vec{\theta}^{t+1})Q^*(\vec{\theta}^{t+1},\left(\varphi^{t+1},\beta^{t+1}\right)) \tag{4.22}$$

and for all $(\vec{\theta},\varphi)$ such that $\vec{\theta}$ is consistent with φ the dynamic program can be evaluated from the end $(t = h-1)$ to the begin $(t = 0)$. Figure 7 illustrates the computation of Q^*. When arriving at stage 0, the φ^1 reduce to joint actions and it is possible to select

$$\delta^{0,*} = \arg\max_a Q^*(\vec{\theta}_\emptyset,a) = \arg\max_{\varphi^1} Q^*(\vec{\theta}^0,\varphi^1).$$

Then given $\varphi^1 = \delta^{0,*}$ we can determine $\delta^{1,*} = \delta^{1,*}_\varphi$ using (4.22), etc. This essentially is the forward-sweep policy computation using the optimal Q-value function $Q^*(\vec{\theta}^t,\varphi^{t+1})$ as defined by (4.20).

The computation of Q^* is also closely related to point-based dynamic programming for Dec-POMDPs as discussed in section 3.5. Suppose that $t = 2$ in Figure 7 is the last stage (i.e., $h = 3$). When the $\delta^{2,*}$ for all φ^2 have been computed, it is easy to construct the sets of non-dominated action for each agent: every action a_i of agent i that is specified by some $\delta_i^{2,*}$ is non-dominated. Once we have computed the values for all $(\vec{\theta}^1,\varphi^2)$ at $t = 1$, each φ^2 has an associated optimal future policy $\delta^{2,*}_\varphi$. This means that each individual history $\vec{\theta}^1_i$ has an associated sub-tree policy $q_i^{\tau=2}$ for each φ^2 and as such each $(\vec{\theta}^1,\varphi^2)$-pair has an associated joint sub-tree policy (e.g, the shaded trees in Figure 7). Clearly, $Q^*(\vec{\theta}^1,\varphi^2)$ corresponds to expected value of this associated joint sub-tree policy. Rather than keeping track of these sub-trees policies, however, the algorithm presented here keeps track of the values.

The advantage of the description of Q^* using (4.21) rather than (4.10) is twofold. First the description treated here describes the way to actually compute the values which can then be used to construct π^*, while the latter only gives a normative description and needs π^* in order to compute the Q-values.

Second, this $Q^*(\vec{\theta}^t,\varphi^{t+1})$ describes sequential rationality for Dec-POMDPs. For *any* past policy (and corresponding consistent belief system) the optimal future policy can be computed. A variation of this might even be applied on-line. Suppose agent i makes a mistake at stage t, executing an action not prescribed by π^*_i, assuming the other agents execute their policy $\pi_{\neq i}$ without mistakes, agent i knows the actually executed previous policy φ^{t+1}. Therefore it can compute a new individual policy by

$$\delta^{t+1,*}_{i,\varphi^{t+1}} = \arg\max_{\beta_i{}^{t+1}} \sum_{\vec{\theta}^{t+1}\in\vec{\Theta}^{t+1}_\varphi} P(\vec{\theta}^{t+1})Q^*(\vec{\theta}^{t+1},\Big(\varphi^{t+1},\Big\langle\beta_i{}^{t+1},\delta^{t+1}_{\neq i}\Big\rangle\Big)).$$

4.4.3 The Complexity of Computing a Sequentially Rational Q^*

Although we have now found a way to compute Q^*, this computation is intractable for all but the smallest problems, as we will now show. At stage $t-1$ there are $\sum_{t'=0}^{t-1}|\mathcal{O}_i|^{t'} = \frac{|\mathcal{O}_i|^t-1}{|\mathcal{O}_i|-1}$ observation histories for agent i, leading to

$$|\mathcal{A}_*|^{\frac{n\left(|\mathcal{O}_*|^t-1\right)}{|\mathcal{O}_*|-1}}$$

pure joint past policies φ^t. For each of these there are $|\vec{\mathcal{O}}^t| = |\mathcal{O}|^{t-1}$ consistent joint action-observation histories (for each observation history $\vec{o}^{t-1}$, φ^t specifies the actions forming $\vec{\theta}^{t-1}$). This means that for stage $h-2$ (for $h-1$, the Q-values are easily calculated), the number of entries to be computed is the number of joint past policies φ^{h-1} times the number of joint histories

$$O\left(|\mathcal{A}_*|^{\frac{n\left(|\mathcal{O}_*|^{h-1}-1\right)}{|\mathcal{O}_*|-1}} \cdot |\mathcal{O}|^{h-2}\right),$$

indicating that computation of this function is doubly exponential, just as brute force policy evaluation. Also, for each joint past policy φ^{h-1}, we need to compute $\varphi^h = (\varphi^{h-1},\delta_\varphi^{h-1,*})$ by solving the next-stage BG:

$$\delta_\varphi^{h-1,*} = \arg\max_{\beta^{h-1}} \sum_{\vec{\theta}^{h-1} \in \vec{\Theta}_\varphi^{h-1}} P(\vec{\theta}^{h-1}) Q^*(\vec{\theta}^{h-1}, \left(\varphi^{h-1},\beta^{h-1}\right)).$$

To the authors' knowledge, the only method to optimally solve these BGs is evaluation of all

$$O\left(|\mathcal{A}_*|^{n|\mathcal{O}_*|^{h-1}}\right)$$

joint BG-polices, which is also doubly exponential in the horizon.

5. Approximate Q-value Functions

As indicated in the previous section, although an optimal Q-value function Q^* exists, it is costly to compute and thus impractical. In this section, we review some other Q-value functions, $\widehat{Q}$, that can be used as an approximation for Q^*. We will discuss underlying assumptions, computation, computational complexity and other properties, thereby providing a taxonomy of approximate Q-value functions for Dec-POMDPs. In particular we will treat two well-known approximate Q-value functions, $\mathrm{Q}_{\mathrm{MDP}}$ and $\mathrm{Q}_{\mathrm{POMDP}}$, and Q_{BG} recently introduced by Oliehoek and Vlassis (2007).

5.1 $\mathrm{Q}_{\mathrm{MDP}}$

$\mathrm{Q}_{\mathrm{MDP}}$ was originally proposed to approximately solve POMDPs by Littman, Cassandra, and Kaelbling (1995), but has also been applied to Dec-POMDPs (Emery-Montemerlo et al., 2004; Szer et al., 2005). The idea is that Q^* can be approximated using the state-action values $Q_{\mathrm{M}}(s,a)$ found when solving the 'underlying MDP' of a Dec-POMDP. This 'underlying MDP' is the horizon-h MDP defined by a single agent that takes joint actions $a \in \mathcal{A}$ and observes the nominal state s that has the same transition model T and reward model R as the original Dec-POMDP. Solving this underlying MDP can be efficiently done using dynamic programming techniques (Puterman, 1994), resulting in the optimal non-stationary MDP Q-value function:

$$Q_{\mathrm{M}}^{t,*}(s^t,a) = R(s^t,a) + \sum_{s^{t+1} \in \mathcal{S}} P(s^{t+1}|s^t,a) \max_a Q_{\mathrm{M}}^{t+1,*}(s^{t+1},a). \tag{5.1}$$

In this equation, the maximization is an implicit selection of $\pi_{\mathrm{M}}^{t+1,*}$, the optimal MDP policy at the next time step, as explained in Section 4.3.3. Note that $Q_{\mathrm{M}}^{t,*}$ also is an optimal Q-value function, but in the MDP setting. In this article Q^* will always denote the optimal value function for the (original) Dec-POMDP. In order to transform the $Q_{\mathrm{M}}^{t,*}(s^t,a)$-values to approximate $\widehat{Q}_{\mathrm{M}}(\vec{\theta}^t,a)$-values to be used the original Dec-POMDP, we compute:

$$\widehat{Q}_{\mathrm{M}}(\vec{\theta}^{t},a) = \sum_{s \in \mathcal{S}} Q_{\mathrm{M}}^{t,*}(s,a) P(s|\vec{\theta}^{t}), \tag{5.2}$$

where $P(s|\vec{\theta}^{t})$ can be computed from (4.8). Combining (5.1) and (5.2) and making the selection of $\pi_{\mathrm{M}}^{t+1,*}$ explicit we get:

$$\widehat{Q}_{\mathrm{M}}(\vec{\theta}^{t},a) = R(\vec{\theta}^{t},a) + \sum_{s^{t+1} \in \mathcal{S}} P(s^{t+1}|\vec{\theta}^{t},a) \max_{\pi_{\mathrm{M}}^{t+1}(s^{t+1})} Q_{\mathrm{M}}^{t+1,*}(s^{t+1},\pi_{\mathrm{M}}^{t+1}(s^{t+1})), \tag{5.3}$$

which defines the approximate Q-value function that can be used as payoff function for the various BGs of the Dec-POMDP. Note that $\widehat{Q}_{\mathrm{M}}$ is consistent with the established definition of Q-value functions since it is defined as the expected immediate reward of performing (joint) action a plus the value of following an optimal joint policy (in this case the optimal MDP-policy) thereafter.

Because calculation of the $Q_{\mathrm{M}}^{t}(s,a)$-values by dynamic programming (which has a cost of $O(|\mathcal{S}| \times h)$ can be performed in a separate phase, the cost of computation of $\mathrm{Q_{MDP}}$ is only dependent on the cost of evaluation of (5.3), which is $O(|\mathcal{S}|)$. When we want to evaluate $\mathrm{Q_{MDP}}$ for all $\sum_{t=0}^{h-1} (|\mathcal{A}|\,|\mathcal{O}|)^{t} = \frac{(|\mathcal{A}||\mathcal{O}|)^{h}-1}{(|\mathcal{A}||\mathcal{O}|)-1}$ joint action-observation histories is, the total computational cost becomes:

$$O\left(\frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h} - 1}{(|\mathcal{A}|\,|\mathcal{O}|) - 1} |\mathcal{A}||\mathcal{S}| \right). \tag{5.4}$$

However, when applying $\mathrm{Q_{MDP}}$ in forward-sweep policy computation, we do not have to consider *all* action-observation histories, but only those that are consistent with the policy found for earlier stages. Effectively we only have to evaluate (5.3) for all observation histories and joint actions, leading to:

$$O\left(\frac{(|\mathcal{O}|)^{h} - 1}{(|\mathcal{O}|) - 1} |\mathcal{A}||\mathcal{S}| \right). \tag{5.5}$$

When used in the context of Dec-POMDPs, $\mathrm{Q_{MDP}}$ solutions are known to undervalue actions that gain information (Fernández, Sanz, Simmons, & Diéguez, 2006). This is explained by realizing that the $\mathrm{Q_{MDP}}$ solution assumes that the state will be fully observable in the next time step. Therefore actions that provide information about the state, and thus can lead to a high future reward (but might have a low immediate reward), will be undervalued. When applying $\mathrm{Q_{MDP}}$ in the Dec-POMDP setting, this effect can also be expected. Another consequence of the simplifying assumption is that the $\mathrm{Q_{MDP}}$-value function is an upper bound to the optimal value function when used to approximate a POMDP (Hauskrecht, 2000), as a consequence it is also an upper bound to the optimal value function of a Dec-POMDP. This is intuitively clear, as a Dec-POMDP is a POMDP but with the additional difficulty of decentralization. A formal argument will be presented in Section 5.4.

5.2 $\mathrm{Q_{POMDP}}$

Similar to the 'underlying MDP', one can define the 'underlying POMDP' of a Dec-POMDP as the POMDP with the same T, O and R, but in which there is only a single agent that takes joint actions $a \in \mathcal{A}$ and receives joint observations $o \in \mathcal{O}$. $\mathrm{Q}_{\mathrm{POMDP}}$ approximates Q^* using the solution of the underlying POMDP (Szer et al., 2005; Roth et al., 2005).

In particular, the optimal $\mathrm{Q}_{\mathrm{POMDP}}$ value function for an underlying POMDP satisfies:

$$Q^*_{\mathrm{P}}(b^{\vec{\theta}^t},a) = R(b^{\vec{\theta}^t},a) + \sum_{o^{t+1}\in\mathcal{O}} P(o^{t+1}|b^{\vec{\theta}^t},a) \max_{\pi_{\mathrm{P}}^{t+1}(b^{\vec{\theta}^{t+1}})} Q^*_{\mathrm{P}}(b^{\vec{\theta}^{t+1}},\pi_{\mathrm{P}}^{t+1}(b^{\vec{\theta}^{t+1}})), \tag{5.6}$$

where $b^{\vec{\theta}^t}$ is the *joint belief* of the single agent that selects joint actions and receives joint observations at time step t, where

$$R(b^{\vec{\theta}^t},a) = \sum_{s\in\mathcal{S}} R(s,a) b^{\vec{\theta}^t}(s) \tag{5.7}$$

is the immediate reward, and where $b^{\vec{\theta}^{t+1}}$ is the joint belief resulting from $b^{\vec{\theta}^t}$ by action a and joint observation o^{t+1}, calculated by

$$\forall_{s'} \quad b^{\vec{\theta}^{t+1}}(s') = \frac{P(o|a,s')\sum_{s\in\mathcal{S}} P(s'|s,a) b^{\vec{\theta}^t}(s)}{\sum_{s'\in\mathcal{S}} P(o|a,s')\sum_{s\in\mathcal{S}} P(s'|s,a) b^{\vec{\theta}^t}(s)}. \tag{5.8}$$

For each $\vec{\theta}^t$ there is one joint belief $b^{\vec{\theta}^t}$, which corresponds to $P(s|\vec{\theta}^t)$ as can be derived from (4.8). Therefore it is possible to directly use the computed $\mathrm{Q}_{\mathrm{POMDP}}$ values as payoffs for the BGs of the Dec-POMDP, that is, we define:

$$\widehat{Q}_{\mathrm{P}}(\vec{\theta}^t,a) \equiv Q^*_{\mathrm{P}}(b^{\vec{\theta}^t},a). \tag{5.9}$$

The maximization in (5.6) is stated in its explicit form: a maximization over time step $t+1$ POMDP policies. However, it should be clear that this maximization effectively is one over joint actions, as it is conditional on the received joint observation o^{t+1} and thus the resulting belief $b^{\vec{\theta}^{t+1}}$.

For a finite horizon, Q^*_{P} can be computed by generating all possible joint beliefs and solving the 'belief MDP'. Generating all possible beliefs is easy: starting with b^0 corresponding to the empty joint action-observation history $\vec{\theta}^{t=0}$, for each a and o we calculate the resulting $\vec{\theta}^{t=1}$ and corresponding belief $b^{\vec{\theta}^1}$ and continue recursively. Solving the belief MDP amounts to recursively applying (5.6).

In the computation of $\mathrm{Q}_{\mathrm{MDP}}$ we could restrict our attention to only those $(\vec{\theta}^t,a)$-pairs that were specified by forward-sweep policy computation, because the $\widehat{Q}_{\mathrm{M}}(\vec{\theta}^t,a)$-values do not depend on the values of successor-histories $\widehat{Q}_{\mathrm{M}}(\vec{\theta}^{t+1},a)$. For $\mathrm{Q}_{\mathrm{POMDP}}$, however, there is such a dependence, meaning that it is necessary to evaluate for all $\vec{\theta}^t,a$. In particular, the cost of calculating $\mathrm{Q}_{\mathrm{POMDP}}$ can be divided in the cost of calculating the expected immediate reward for all $\vec{\theta}^t,a$, and the cost of evaluating future reward for all $\vec{\theta}^t,a$, with $t = 0,...,h-2$. The former operation is given by (5.7) and has cost $O(|\mathcal{S}|)$ per $\vec{\theta}^t,a$ and thus a total cost equal to (5.4). The latter requires selecting the maximizing joint action for each joint observation for all $\vec{\theta}^t,a$ with $t = 0,...,h-2$, leading to

$$O\left(\frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h-1}-1}{(|\mathcal{A}|\,|\mathcal{O}|)-1}|\mathcal{A}|\,(|\mathcal{A}|\,|\mathcal{O}|)\right). \tag{5.10}$$

$\vec{\theta}_1^{t=0}$ \ $\vec{\theta}_2^{t=0}$		()	
		a_2	$\bar{a}_2$
()	a_1	$+3.1$	-4.1
	$\bar{a}_1$	-0.9	$+0.3$

$\vec{\theta}_1^{t=1}$ \ $\vec{\theta}_2^{t=1}$		(a_2,o_2)		$(a_2,\bar{o}_2)$		...
		a_2	$\bar{a}_2$	a_2	$\bar{a}_2$	
(a_1,o_1)	a_1	-0.3	$+0.6$	-0.6	$+4.0$	...
	$\bar{a}_1$	-0.6	$+2.0$	-1.3	$+3.6$	...
$(a_1,\bar{o}_1)$	a_1	$+3.1$	$+4.4$	-1.9	$+1.0$	...
	$\bar{a}_1$	$+1.1$	-2.9	$+2.0$	-0.4	
$(\bar{a}_1,o_1)$	a_1	-0.4	-0.9	-0.5	-1.0	...
	$\bar{a}_1$	-0.9	-4.5	-1.0	$+3.5$	...
$(\bar{a}_1,\bar{o}_1)$	...	...	...	...	...	...

Figure 8: Backward calculation of Q_{POMDP}-values. Note that the solutions (the highlighted entries) are different from those in Figure 6: Q_{POMDP} assumes that the actions can be conditioned on the joint action-observation history. The highlighted '+3.1' entry for the Bayesian game for $t = 0$ is calculated as the expected immediate reward ($= 0$) plus a weighted sum of the maximizing entry (joint action) per next joint observation history. When assuming a uniform distribution over joint observations given $\langle a_1,a_2\rangle$ the future reward is given by: $+3.1 = 0 + 0.25 \times 2.0 + 0.25 \times 4.0 + 0.25 \times 4.4 + 0.25 \times 2.0$.

Therefore the total complexity of computing Q_{POMDP} becomes

$$O\left(\frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h-1}-1}{(|\mathcal{A}|\,|\mathcal{O}|)-1}|\mathcal{A}|\,(|\mathcal{A}|\,|\mathcal{O}|)+\frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h}-1}{(|\mathcal{A}|\,|\mathcal{O}|)-1}|\mathcal{A}||\mathcal{S}|\right). \tag{5.11}$$

Evaluating (5.6) for all joint action-observation histories $\vec{\theta}^t \in \vec{\Theta}^t$ can be done in a single backward sweep through time, as we mentioned in Section 4.3.3. This can also be visualized in Bayesian games as illustrated in Figure 8; the expected future reward is calculated as a maximizing weighted sum of the entries of the next time step BG.

Nevertheless, solving a POMDP optimally is also known as an intractable problem. As a result, POMDP research in the last decade has focused on approximate solutions for POMDPs. In particular, it is known that the value function of a POMDP is *piecewise-linear and convex (PWLC)* over the (joint) belief space (Sondik, 1971). This property is exploited by many approximate POMDP solution methods (Pineau, Gordon, & Thrun, 2003; Spaan & Vlassis, 2005). Clearly such methods can also be used to calculate an approximate Q_{POMDP}-value function for use with Dec-POMDPs.

It is intuitively clear that Q_{POMDP} is also an admissible heuristic for Dec-POMDPs, as it still assumes that more information is available than actually is the case (again a formal proof will be given in Section 5.4). Also it should be clear that, as fewer assumptions are made, Q_{POMDP} should yield less of an over-estimation than Q_{MDP}. I.e., the Q_{POMDP}-values should lie between the Q_{MDP} and optimal Q^*-values.

In contrast to Q_{MDP}, Q_{POMDP} does not assume full observability of nominal states. As a result the latter does not share the drawback of undervaluing actions that will gain information regarding the nominal state. When applied in a Dec-POMDP setting, however, Q_{POMDP} does share the assumption of centralized control. This assumption might also cause a relative undervaluation: there might be situations where some action might gain information regarding the joint (i.e., each other's) observation history. Under Q_{POMDP} this will be considered redundant, while in decentralized execution this might be very beneficial, as it allows for better coordination.

5.3 $\mathbf{Q_{BG}}$

Q_{MDP} approximates Q^* by assuming that the state becomes fully observable in the next time step, while Q_{POMDP} assumes that at every time step t the agents know the joint action-observation history $\vec{\theta}^t$. Here we present a new approximate Q-value function, called Q_{BG}, that relaxes the assumptions further: it assumes that the agents know $\vec{\theta}^{t-1}$, the joint action-observation history up to time step $t-1$, and the joint action a^{t-1} that was taken at the previous time step. This means that the agents are uncertain regarding each other's last observation, which effectively defines a BG for each $\vec{\theta}^{t-1}$,a. Note, that these BGs are different from the BGs used in Section 4.2: the BGs here have types that correspond to single observations, whereas the BGs in 4.2 have types that correspond to complete action-observation histories. Hence, the BGs of Q_{BG} are much smaller in size and thus easier to solve. Formally Q_{BG} is defined as:

$$Q^*_{\text{B}}(\vec{\theta}^t,a) = R(\vec{\theta}^t,a) + \max_{\beta} \sum_{o^{t+1}\in\mathcal{O}} P(o^{t+1}|\vec{\theta}^t,a) Q^*_{\text{B}}(\vec{\theta}^{t+1},\beta(o^{t+1})), \tag{5.12}$$

where $\beta = \langle \beta_1(o_1^{t+1}),...,\beta_n(o_n^{t+1})\rangle$ is a tuple of individual policies $\beta_i : \mathcal{O}_i \rightarrow \mathcal{A}_i$ for the BG constructed for $\vec{\theta}^t$,a.

Note that the only difference between (5.12) and (5.6) is the position and argument of the maximization operator: (5.12) maximizes over a (conditional) BG-policy, while the maximization in (5.6) is effectively over unconditional joint actions.

The BG representation of the fictitious Dec-POMDP in Figure 6 illustrates the computation of Q_{BG}.[8] The probability distribution $P(\vec{\Theta}^1_{\langle a_1,a_2\rangle})$ over joint action-observation histories that can be reached given $\langle a_1,a_2\rangle$ at $t=0$ is uniform and the immediate reward for $\langle a_1,a_2\rangle$ is 0. Therefore, we have that $2.75 = 0.25\cdot 2.0 + 0.25\cdot 3.6 + 0.25\cdot 4.4 + 0.25\cdot 1.0$.

The cost of computing Q_{BG} for all $\vec{\theta}^t$,a can be split up in the cost of computing the immediate reward (see (5.4)) and the cost of computing the future reward (solving a BG over the last received observation), which is

$$O\left(\frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h-1}-1}{(|\mathcal{A}|\,|\mathcal{O}|)-1}|\mathcal{A}|\cdot|\mathcal{A}_*|^{n|\mathcal{O}_*|}\right),$$

leading to a total cost of:

$$O\left(\frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h-1}-1}{(|\mathcal{A}|\,|\mathcal{O}|)-1}|\mathcal{A}|\cdot|\mathcal{A}_*|^{n|\mathcal{O}_*|} + \frac{(|\mathcal{A}|\,|\mathcal{O}|)^{h}-1}{(|\mathcal{A}|\,|\mathcal{O}|)-1}|\mathcal{A}||\mathcal{S}|\right). \tag{5.13}$$

Comparing to the cost of computing Q_{POMDP}, this contains an additional exponential term, but this term does not depend on the horizon of the problem.

8. Because the BG representing $t=1$ of a Dec-POMDP also involves observation histories of length 1, the illustration of such a BG corresponds to the BGs as considered in Q_{BG}. For other stages this is not the case.

As mentioned in Section 5.2, Q_{POMDP} can be approximated by exploiting the PWLC-property of the value function. It turns out that the Q_{BG}-value function corresponds to an optimal value function for the situation where the agents can communicate freely with a one-step delay (Oliehoek et al., 2007b). Hsu and Marcus (1982) showed how a complex dynamic program can be constructed for such settings and that the resulting value function also preserves the PWLC property. Not surprisingly, the Q_{BG}-value function also is piecewise-linear and convex over the joint belief space and, as a result, approximation methods for POMDPs can be transferred to the computation of Q_{BG} (Oliehoek et al., 2007b).

5.4 Generalized Q_{BG} and Bounds

We can think of an extension of the Q_{BG}-value function framework to the case of k-steps delayed communication, where each agent perceives the joint action-observation history with k stages delay. That is, at stage t, each agent i knows $\vec{\theta}^{t-k}$ the joint action-observation history of k stages before in addition to its own current action-observation history $\vec{\theta}_i^t$. Similar k-step delayed observation models for decentralized control have been previously proposed by Aicardi, Davoli, and Minciardi (1987) and Ooi and Wornell (1996). In particular Aicardi et al. consider the Dec-MDP setting in which agent i's observations are local states s_i and where a joint observation identifies the state $s = \langle s_1, \ldots, s_n \rangle$. Ooi and Wornell examine the decentralized control of a broadcast channel over an infinite horizon, where they allow the local observations to be arbitrary, but still require the joint state to be observed with a k-steps delay. Our assumption is less strong, as we only require observation of $\vec{\theta}^{t-k}$ and because we assume the general Dec-POMDP (not Dec-MDP) setting.

Such a k-step delayed communication model for the Dec-POMDP setting allows expressing the different Q-value functions defined in this article as optimal value functions of appropriate k-step delay models. More importantly, by resorting to such a k-step delay model we can prove a hierarchy of bounds that hold over the various Q-functions defined in this article:

Theorem 5.1 (Hierarchy of upper bounds)**.** *The approximate Q-value functions Q_{BG} and Q_{POMDP} correspond to the optimal Q-value functions of appropriately defined k-step delayed communication models. Moreover these Q-value functions form a hierarchy of upper bounds to the optimal Q^* of the Dec-POMDP:*

$$Q^* \leq Q_{BG} \leq Q_{POMDP} \leq Q_{MDP}. \tag{5.14}$$

Proof. See appendix. □

The idea is that a POMDP corresponds to a system with no (0-steps) delayed communication, while the Q_{BG}-setting corresponds to a 1-step delayed communication system. The appendix shows that the Q-value function of a system with k steps delay forms an upper bound to that of a decentralized system with $k+1$ steps delay. We note that the last inequality of (5.14) is a well-known result (Hauskrecht, 2000).

6. Generalized Value-Based Policy Search

The hierarchy of approximate Q-value functions implies that all of these Q-value functions can be used as *admissible heuristics* in MAA* policy search, treated in Section 3.3. In

Algorithm 1 GMAA*

$\underline{v}^\star \leftarrow -\infty$
$\mathrm{P} \leftarrow \{\varphi^0 = ()\}$
repeat
  $\varphi^t \leftarrow \texttt{Select}(\mathrm{P})$
  $\Phi_{\texttt{Next}} \leftarrow \texttt{Next}(\varphi^t)$
  if $\Phi_{\texttt{Next}}$ contains a subset of full policies $\Pi_{\texttt{Next}} \subseteq \Phi_{\texttt{Next}}$ **then**
    $\pi' \leftarrow \arg\max_{\pi \in \Pi_{\texttt{Next}}} V(\pi)$
    if $V(\pi') > \underline{v}^\star$ **then**
      $\underline{v}^\star \leftarrow V(\pi')$
      $\pi^\star \leftarrow \pi'$
      $\mathrm{P} \leftarrow \left\{ \varphi \in \mathrm{P} \mid \widehat{V}(\varphi) > \underline{v}^\star \right\}$ {prune the policy pool}
    end if
    $\Phi_{\texttt{Next}} \leftarrow \Phi_{\texttt{Next}} \setminus \Pi_{\texttt{Next}}$ {remove full policies}
  end if
  $\mathrm{P} \leftarrow (\mathrm{P} \setminus \varphi^t) \cup \left\{ \varphi \in \Phi_{\texttt{Next}} \mid \widehat{V}(\varphi) > \underline{v}^\star \right\}$ {remove processed/add new partial policies}
until P is empty

this section we will present a more general heuristic policy search framework which we will call Generalized MAA* (GMAA*), and show how it unifies some of the solution methods proposed for Dec-POMDPs.

GMAA* generalizes MAA* (Szer et al., 2005) by making explicit different procedures that are implicit in MAA*: (1) iterating over a pool of partial joint policies, pruning this pool whenever possible, (2) selecting a partial joint policy from the policy pool, and (3) finding some new partial and/or full joint policies given the selected policy. The first procedure is the core of GMAA* and is fixed, while the other two procedures can be performed in many ways.

The second procedure, `Select`, chooses which policy to process next and thus determines the type of search (e.g., depth-first, breadth-first, A*-like) (Russell & Norvig, 2003; Bertsekas, 2005). The third procedure, which we will refer to as `Next`, determines how the set of next (partial) joint policies are constructed, given a previous partial joint policy. The original MAA* can be seen as an instance of the generalized case with a particular `Next`-operator, namely that shown in algorithm 2.

6.1 The GMAA* Algorithm

In GMAA* we refer to a 'policy pool' P rather than an open list, as it is a more neutral word which does not imply any ordering. This policy pool P is initialized with a completely unspecified joint policy $\varphi^0 = ()$ and the maximum lower bound (found so far) $\underline{v}^\star$ is set to $-\infty$. $\pi^\star$ denotes the best joint policy found so far.

At this point GMAA* starts. First, the selection operator, `Select`, selects a partial joint policy φ from P. We will assume that, in accordance with MAA*, the partial policy with the highest heuristic value is selected. In general, however, any kind of selection algorithm may be used. Next, the selected policy is processed by the policy search operator `Next`,

Algorithm 2 $\texttt{Next}(\varphi^t)$ — MAA*

1: $\Phi^{t+1} \leftarrow \left\{ \varphi^{t+1} = \left\langle \varphi_1^{t+1}, \ldots, \varphi_n^{t+1} \right\rangle \mid \varphi_i^{t+1} = \left(\varphi_i^t, \delta_i^t \right), \delta_i^t : \vec{\mathcal{O}}_i^t \rightarrow \mathcal{A}_i \right\}$
2: $\forall_{\varphi^{t+1} \in \Phi^{t+1}} \quad \widehat{V}(\varphi^{t+1}) \leftarrow V^{0\ldots t-1}(\varphi^t) + E\left[R(s^t, a) \middle| \varphi^{t+1} \right] + \widehat{V}^{(t+1)\ldots h}(\varphi^{t+1})$
3: **return** Φ^{t+1}

which returns a set of (partial) joint policies $\Phi_{\texttt{Next}}$ and their heuristic values. When $\texttt{Next}$ returns one or more full policies $\pi \in \Phi_{\texttt{Next}}$, the provided values $\widehat{V}(\pi) = V(\pi)$ are a lower bound for an optimal joint policy, which can be used to prune the search space. Any found partial joint policies $\varphi \in \Phi_{\texttt{Next}}$ with a heuristic value $\widehat{V}(\varphi) > \underline{v}^\star$ are added to P. The process is repeated until the policy pool is empty.

6.2 The $\texttt{Next}$ Operator

Here we describe some different choices for the $\texttt{Next}$-operator and how they correspond to existing Dec-POMDP solution methods.

6.2.1 MAA*

GMAA* reduces to standard MAA* by using the $\texttt{Next}$-operator described by Algorithm 2. Line 1 expands φ^t forming Φ^{t+1} the set of partial joint policies for one extra stage. Line 2 valuates all these child policies, where

$$V^{0\ldots t}(\varphi^{t+1}) = V^{0\ldots t-1}(\varphi^t) + E\left[R(s^t, a) \middle| \varphi^{t+1} \right]$$

gives the true expected reward over the first $t+1$ stages. $\widehat{V}^{(t+1)\ldots h}(\varphi^{t+1})$ is the heuristic value over stages $(t+1)\ldots h$ given that φ^{t+1} has been followed the first $t+1$ stages.

When using an admissible heuristic, GMAA* will never prune a partial policy that can be expanded into an optimal policy. When combining this with the fact that the MAA*-$\texttt{Next}$ operator returns all possible φ^{t+1} for a φ^t, it is clear that when P becomes empty an optimal policy has been found.

6.2.2 Forward-Sweep Policy Computation

Forward-sweep policy computation, as introduced in Section 4.3.1, is described by algorithms 1 and 3 jointly. Given a partial joint policy φ^t, the $\texttt{Next}$ operator now constructs and solves a BG for time step t. Because $\texttt{Next}$ in algorithm 3 only returns the best-ranked policy, P will never contain more than 1 joint policy and the whole search process reduces to solving BGs for time steps $0,\ldots,h-1$.

The approach of Emery-Montemerlo et al. (2004) is identical to forward-sweep policy computation, except that 1) smaller BGs are created by discarding or clustering low probability action-observation histories, and 2) the BGs are approximately solved by alternating maximization. Therefore this approach can also be incorporated in the GMAA* policy search framework by making the appropriate modifications in Algorithm 3.

Algorithm 3 $\texttt{Next}(\varphi^t)$ — Forward-sweep policy computation

$BG \leftarrow \left\langle \mathcal{A},\vec{\Theta}^t_{\varphi^t},P(\vec{\Theta}^t_{\varphi^t}),\widehat{Q}^t \right\rangle$
for all $\beta = \langle\beta_1,...,\beta_n\rangle$ s.t. $\beta_i : \vec{\mathcal{O}}^t_i \to \mathcal{A}_i$ **do**
  $\widehat{V}^t(\beta) \leftarrow \sum_{\vec{\theta}^t\in\vec{\Theta}^t_{\varphi^t}} P(\vec{\theta}^t)\widehat{Q}^t(\vec{\theta}^t,\beta(\vec{\theta}^t))$
  $\varphi^{t+1} \leftarrow \left(\varphi^t,\beta\right)$
  $\widehat{V}(\varphi^{t+1}) \leftarrow V^{0...t-1}(\varphi^t) + \widehat{V}^t(\beta)$
end for
return $\arg\max_{\varphi^{t+1}} \widehat{V}(\varphi^{t+1})$

6.2.3 Unification

Here we will give a unified perspective of the MAA* and forward-sweep policy computation by examining the relation between the corresponding Next-operators. In particular we show that, when using any of the approximate Q-value functions described in Section 5 as a heuristic, the sole difference between the two is that FSPC returns only the joint policy with the highest heuristic value.

Proposition 6.1. *If a heuristic $\widehat{Q}$ has the following form*

$$\widehat{Q}^t(\vec{\theta}^t,a) = R(\vec{\theta}^t,a) + \sum_{o^{t+1}} P(o^{t+1}|\vec{\theta}^t,a)\widehat{V}^{t+1}(\vec{\theta}^{t+1}), \tag{6.1}$$

then for a partial policy $\varphi^{t+1} = \left(\varphi^t,\beta^t\right)$

$$\sum_{\vec{\theta}^t\in\vec{\Theta}^t_\varphi} P(\vec{\theta}^t)\widehat{Q}^t(\vec{\theta}^t,\beta(\vec{\theta}^t)) = E\left[R(s^t,a)\middle|\varphi^{t+1}\right] + \widehat{V}^{(t+1)...h}(\varphi^{t+1}) \tag{6.2}$$

holds.

Proof. The expectation of R^t given φ^{t+1} can be written as

$$E\left[R(s^t,a)\middle|\varphi^{t+1}\right] = \sum_{\vec{\theta}^t\in\vec{\Theta}^t_\varphi} P(\vec{\theta}^t)\sum_{s\in\mathcal{S}} R(s,\varphi^{t+1}(\vec{\theta}^t))P(s|\vec{\theta}^t) = \sum_{\vec{\theta}^t\in\vec{\Theta}^t_\varphi} P(\vec{\theta}^t)R(\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t)).$$

Also, we can rewrite $\widehat{V}^{(t+1)...h}(\varphi^{t+1})$ as

$$\widehat{V}^{(t+1)...h}(\varphi^{t+1}) = \sum_{\vec{\theta}^t\in\vec{\Theta}^t_\varphi} P(\vec{\theta}^t)\sum_{o^{t+1}} P(o^{t+1}|\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t))\widehat{V}^{(t+1)...h}(\vec{\theta}^{t+1}),$$

such that

$$E\left[R(s^t,a)\middle|\varphi^{t+1}\right] + \widehat{V}^{(t+1)...h}(\varphi^{t+1}) = \sum_{\vec{\theta}^t\in\vec{\Theta}^t_\varphi} P(\vec{\theta}^t)$$
$$\left[R(\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t)) + \sum_{o^{t+1}} P(o^{t+1}|\vec{\theta}^t,\varphi^{t+1}(\vec{\theta}^t))\widehat{V}^{(t+1)...h}(\vec{\theta}^{t+1})\right] \tag{6.3}$$

Therefore, assuming (6.1) yields (6.2). □

$\vec{o}_\emptyset \rightarrow$ go house 3	$\vec{o}_\emptyset \rightarrow$ go house 2
flames $\rightarrow$ go house 3	flames $\rightarrow$ go house 2
no flames $\rightarrow$ go house 1	no flames $\rightarrow$ go house 2
flames, flames $\rightarrow$ go house 1	flames, flames $\rightarrow$ go house 1
flames, no flames $\rightarrow$ go house 1	flames, no flames $\rightarrow$ go house 1
no flames, flames $\rightarrow$ go house 2	no flames, flames $\rightarrow$ go house 1
no flames, no flames $\rightarrow$ go house 2	no flames, no flames $\rightarrow$ go house 1

Figure 9: Optimal policy for FireFighting $\langle n_h = 3, n_f = 3 \rangle$, horizon 3. On the left the policy for the first agent, on the right the second agent's policy.

This means that if a heuristic satisfies (6.1), which is the case for all the Q-value functions we discussed in this paper, the `Next` operators of algorithms 2 and 3 evaluate the expanded policies the same. I.e., algorithms 2 and 3 calculate identical heuristic values for the same next time step joint policies. Also the expanded policies φ^{t+1} are formed in the same way: by considering all possible δ^t respectively β^t to extend φ^t. Therefore, the sole difference in this case is that the latter returns only the joint policy with the highest heuristic value.

Clearly there is a computation time/quality trade-off between MAA* and FSPC: MAA* is guaranteed to find an optimal policy (given an admissible heuristic), while FSPC is guaranteed to finish in one forward sweep. We propose a generalization, that returns the k-best ranked policies. We refer to this as the 'k-best joint BG policies' GMAA* variant, or k-GMAA*. In this way, k-GMAA* reduces to forward-sweep policy computation for $k = 1$ and to MAA* for $k = \infty$.

7. Experiments

In order to compare the different approximate Q-value functions discussed in this work, as well as to show the flexibility of the GMAA* algorithm, we have performed several experiments. We use Q_{MDP}, Q_{POMDP} and Q_{BG} as heuristic estimates of Q^*. We will provide some qualitative insight in the different Q-value functions we considered, as well as results on computing optimal policies using MAA*, and on the performance of forward-sweep policy computation. First we will describe our problem domains, some of which are standard test problems, while others are introduced in this work.

7.1 Problem Domains

In Section 2.2 we discussed the decentralized tiger (Dec-Tiger) problem as introduced by Nair et al. (2003b). Apart from the standard Dec-Tiger domain, we consider a modified version, called Skewed Dec-Tiger, in which the start distribution is not uniform. Instead, initially the tiger is located on the left with probability 0.8. We also include results from the BroadcastChannel problem, introduced by Hansen et al. (2004), which models two nodes that have to cooperate to maximize the throughput of a shared communication channel. Furthermore, a test problem called "Meeting on a Grid" is provided by Bernstein et al. (2005), in which two robots navigate on a two-by-two grid. We consider the version with 2 observations per agent (Amato et al., 2006).

We introduce a new benchmark problem, which models a team of n fire fighters that have to extinguish fires in a row of n_h houses. Each house is characterized by an integer parameter f, or fire level. It indicates to what degree a house is burning, and it can have n_f different values, $0 \leq f < n_f$. Its minimum value is 0, indicating the house is not burning. At every time step, the agents receive a reward of $-f$ for each house and each agent can choose to move to any of the houses to fight fires at that location. If a house is burning ($f > 0$) and no fire fighting agent is present, its fire level will increase by one point with probability 0.8 if any of its neighboring houses are burning, and with probability 0.4 if none of its neighbors are on fire. A house that is not burning can only catch fire with probability 0.8 if one of its neighbors is on fire. When two agents are in the same house, they will extinguish any present fire completely, setting the house's fire level to 0. A single agent present at a house will lower the fire level by one point with probability 1 if no neighbors are burning, and with probability 0.6 otherwise. Each agent can only observe whether there are flames or not at its location. Flames are observed with probability 0.2 if $f = 0$, with probability 0.5 if $f = 1$, and with probability 0.8 otherwise. Initially, the agents start outside any of the houses, and the fire level f of each house is drawn from a uniform distribution.

We will test different variations of this problems, where the number of agents is always 2, but which differ in the number of houses and fire levels. In particular, we will consider $\langle n_h = 3, n_f = 3\rangle$ and $\langle n_h = 4, n_f = 3\rangle$. Figure 9 shows an optimal joint policy for horizon 3 of the former variation. One agent initially moves to the middle house to fight fires there, which helps prevent fire from spreading to its two neighbors. The other agent moves to house 3, and stays there if it observes fire, and moves to house 1 if it does not observe flames. As well as being optimal, such a joint policy makes sense intuitively speaking.

7.2 Comparing Q-value Functions

Before providing a comparison of performance of some of the approximate Q-value functions described in this work, we will first give some more insights in their actual values. For the $h = 4$ Dec-Tiger problem, we generated all possible $\vec{\theta}^t$ and the corresponding $P(s_l|\vec{\theta}^t)$, according to (4.8). For each of these, the maximal $Q(\vec{\theta}^t, a)$-value is plotted in Figure 10. Apart from the three approximate Q-value functions, we also plotted the optimal value for each joint action-observation history $\vec{\theta}^t$ that can be realized when using π^*. Note that different $\vec{\theta}^t$ can have different optimal values, but induce the same $P(s_l|\vec{\theta}^t)$, as demonstrated in the figure: there are multiple Q^*-values plotted for some $P(s_l|\vec{\theta}^t)$. For the horizon 3 Meeting on a Grid problem we also collected all $\vec{\theta}^t$ that can be visited by the optimal policy, and in Figure 11 we again plotted maximal $Q(\vec{\theta}^t, a)$-values. Because this problem has many states, a representation as in Figure 10 is not possible. Instead, we ordered the $\vec{\theta}$ according to their optimal value. We can see that the bounds are tight for some $\vec{\theta}$, while for others they can be quite loose. However, when used in the GMAA* framework, their actual performance as a heuristic also depends on their valuation of $\vec{\theta} \in \vec{\Theta}$ not shown by Figure 11, namely those that will *not* be visited by an optimal policy: especially when these are overestimated, GMAA* will first examine a sub-optimal branch of the search tree. A tighter upper bound can speed up computation to a very large extent, as it allows the algorithm to prune the policy pool more, reducing the number of Bayesian games that need

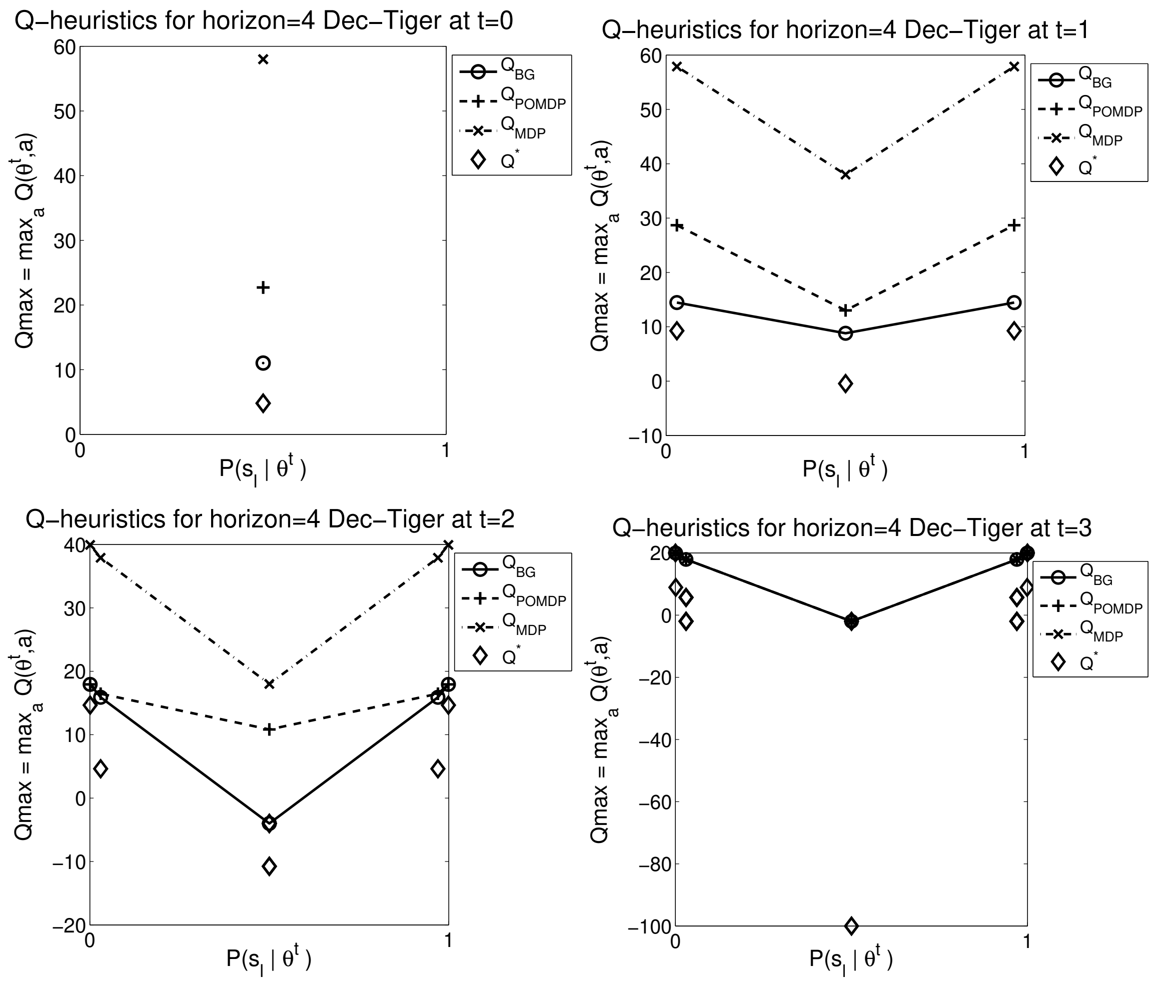


Figure 10: Q-values for horizon 4 Dec-Tiger. For each $\vec{\theta}^t$, corresponding to some $P(s_l|\vec{\theta}^t)$, the maximal $Q(\vec{\theta}^t,a)$-value is plotted.

to be solved. Both figures clearly illustrate the main property of the upper bounds we discussed, namely that $Q^* \leq \mathrm{Q_{BG}} \leq \mathrm{Q_{POMDP}} \leq \mathrm{Q_{MDP}}$ (see Theorem 5.1).

7.3 Computing Optimal Policies

As shown above, the hierarchy of upper bounds $Q^* \leq \mathrm{Q_{BG}} \leq \mathrm{Q_{POMDP}} \leq \mathrm{Q_{MDP}}$ is not just a theoretical construct, but the differences in value specified can be significant for particular problems. In order to evaluate what the impact is of the differences between the approximate Q-value functions, we performed several experiments. Here we describe our evaluation of MAA* on a number of test problems using $\mathrm{Q_{BG}}$, $\mathrm{Q_{POMDP}}$ and $\mathrm{Q_{MDP}}$ as heuristic. All timing results in this paper are CPU times with a resolution of 0.01s, and were obtained on 3.4GHz Intel Xeon processors.

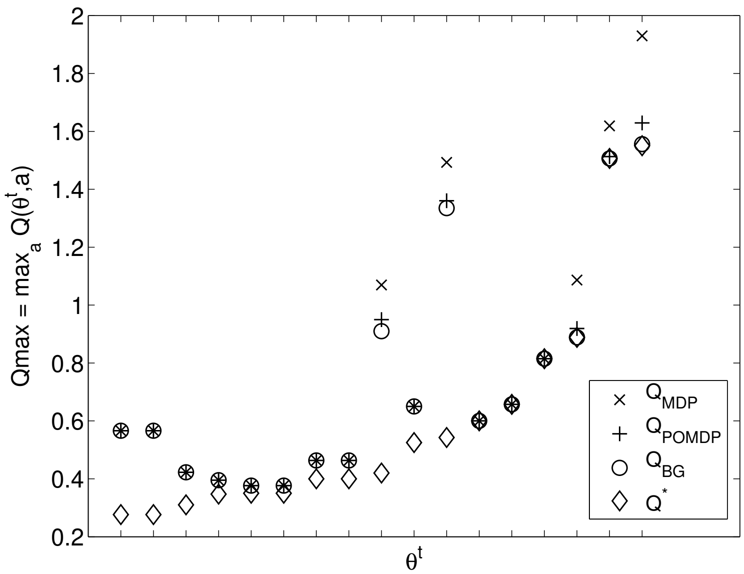


Figure 11: Comparison of maximal $Q(\vec{\theta}^t, a)$-values for Meeting on a Grid. We plot the value of all θ^t that can be reached by an optimal policy, ordered according their optimal value.

h	V^*		n_φ	T_{GMAA^*}	T_Q
3	5.1908	Q_{MDP}	105,228	0.31 s	0 s
		Q_{POMDP}	6,651	0.02 s	0 s
		Q_{BG}	6,651	0.02 s	0.02 s
4	4.8028	Q_{MDP}	37,536,938,118	431,776 s	0 s
		Q_{POMDP}	559,653,390	5,961 s	0.13 s
		Q_{BG}	301,333,698	3,208 s	0.94 s

Table 1: MAA* results for Dec-Tiger.

h	V^*		n_φ	T_{GMAA^*}	T_Q
3	5.8402	Q_{MDP}	151,236	0.46 s	0 s
		Q_{POMDP}	19,854	0.06 s	0.01 s
		Q_{BG}	13,212	0.04 s	0.03 s
4	11.1908	Q_{MDP}	33,921,256,149	388,894 s	0 s
		Q_{POMDP}	774,880,515	8,908 s	0.13 s
		Q_{BG}	86,106,735	919 s	0.92 s

Table 2: MAA* results for Skewed Dec-Tiger.

h	V^*		n_φ	T_{GMAA^*}	T_Q
4	3.8900	$\mathrm{Q}_{\mathrm{MDP}}$	328,212	3.54 s	0 s
		$\mathrm{Q}_{\mathrm{POMDP}}$	531	0 s	0.01 s
		Q_{BG}	531	0 s	0.03 s
5	4.7900	$\mathrm{Q}_{\mathrm{MDP}}$	N/A	$>$ 4.32e5 s	0 s
		$\mathrm{Q}_{\mathrm{POMDP}}$	196,883	5.30 s	0.20 s
		Q_{BG}	196,883	5.15 s	0.53 s

Table 3: MAA* results for BroadcastChannel.

h	V^*		n_φ	T_{GMAA^*}	T_Q
2	0.9100	$\mathrm{Q}_{\mathrm{MDP}}$	1,275	0 s	0 s
		$\mathrm{Q}_{\mathrm{POMDP}}$	1,275	0 s	0 s
		Q_{BG}	194	0 s	0 s
3	1.5504	$\mathrm{Q}_{\mathrm{MDP}}$	29,688,775	81.93 s	0 s
		$\mathrm{Q}_{\mathrm{POMDP}}$	3,907,525	10.80 s	0.15 s
		Q_{BG}	1,563,775	4.44 s	1.37 s

Table 4: MAA* results for Meeting on a Grid.

Table 1 shows the results MAA* obtained on the original Dec-Tiger problem for horizon 3 and 4. It shows for each heuristic the number of partial joint policies evaluated n_φ, CPU time spent on the GMAA* phase T_{GMAA^*}, and CPU time spent on calculating the heuristic T_Q. As Q_{BG}, $\mathrm{Q}_{\mathrm{POMDP}}$ and $\mathrm{Q}_{\mathrm{MDP}}$ are upper bounds to Q^*, MAA* is guaranteed to find the optimal policy when using them as heuristic, however the timing results may differ.

For $h = 3$ we see that using $\mathrm{Q}_{\mathrm{POMDP}}$ and Q_{BG} only a fraction of the number of policies are evaluated when compared to $\mathrm{Q}_{\mathrm{MDP}}$ which reflects proportionally in the time spent on GMAA*. For this horizon $\mathrm{Q}_{\mathrm{POMDP}}$ and Q_{BG} perform the same, but the time needed to compute the Q_{BG} heuristic is as long as the GMAA*-phase, therefore $\mathrm{Q}_{\mathrm{POMDP}}$ outperforms Q_{BG} here. For $h = 4$, the impact of using tighter heuristics becomes even more pronounced. In this case the computation time of the heuristic is negligible, and Q_{BG} outperforms both, as it is able to prune much more partial joint policies from the policy pool. Table 2 shows results for Skewed Dec-Tiger. For this problem the $\mathrm{Q}_{\mathrm{MDP}}$ and Q_{BG} results are roughly the same as the original Dec-Tiger problem; for $h = 3$ the timings are a bit slower, and for $h = 4$ they are faster. For $\mathrm{Q}_{\mathrm{POMDP}}$, however, we see that for $h = 4$ the results are slower as well and that Q_{BG} outperforms $\mathrm{Q}_{\mathrm{POMDP}}$ by an order of magnitude.

Results for the Broadcast Channel (Table 3), Meeting on a Grid (Table 4) and a Fire fighting problem (Table 5) are similar. The N/A entry in Table 3 indicates the $\mathrm{Q}_{\mathrm{MDP}}$ was not able to compute a solution within 5 days. For these problems we also see that the performance of $\mathrm{Q}_{\mathrm{POMDP}}$ and Q_{BG} is roughly equal. For the Meeting on a Grid problem, Q_{BG} yields a significant speedup over $\mathrm{Q}_{\mathrm{POMDP}}$.

h	V^*		n_φ	T_{GMAA^*}	T_Q
3	-5.7370	Q_{MDP}	446,724	1.58 s	0.56 s
		Q_{POMDP}	26,577	0.08 s	0.21 s
		Q_{BG}	26,577	0.08 s	0.33 s
4	-6.5788	Q_{MDP}	25,656,607,368	309,235 s	0.85 s
		Q_{POMDP}	516,587,229	5,730 s	7.22 s
		Q_{BG}	516,587,229	5,499 s	11.72 s

Table 5: MAA* results for Fire Fighting $\langle n_h = 3, n_f = 3 \rangle$.

$\vec{o}_\emptyset \to a_{\text{Li}}$
$o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HR}} \to a_{\text{Li}}$
$\boxed{o_{\text{HL}}, o_{\text{HL}} \to a_{\text{OR}}}$
$o_{\text{HL}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HL}} \to a_{\text{Li}}$
$\boxed{o_{\text{HR}}, o_{\text{HR}} \to a_{\text{OL}}}$
$o_{\text{HL}}, o_{\text{HL}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HL}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HR}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HR}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HL}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HL}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HR}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HR}}, o_{\text{HR}} \to a_{\text{Li}}$

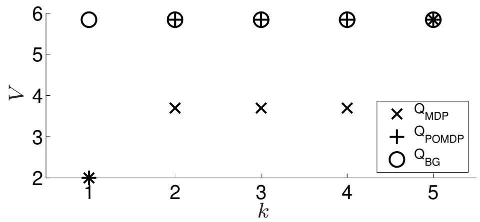
$\vec{o}_\emptyset \to a_{\text{Li}}$
$o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HR}} \to a_{\text{Li}}$
$\boxed{o_{\text{HL}}, o_{\text{HL}}, o_{\text{HL}} \to a_{\text{OR}}}$
$o_{\text{HL}}, o_{\text{HL}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HR}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HL}}, o_{\text{HR}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HL}}, o_{\text{HL}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HL}}, o_{\text{HR}} \to a_{\text{Li}}$
$o_{\text{HR}}, o_{\text{HR}}, o_{\text{HL}} \to a_{\text{Li}}$
$\boxed{o_{\text{HR}}, o_{\text{HR}}, o_{\text{HR}} \to a_{\text{OL}}}$

Figure 12: Policies found using forward-sweep policy computation (i.e., $k = 1$) for the $h = 4$ Dec-Tiger problem. Left: the policy resulting from Q_{MDP}. Right: the optimal policy as calculated by Q_{POMDP} and Q_{BG}. The framed entries highlight the crucial differences.

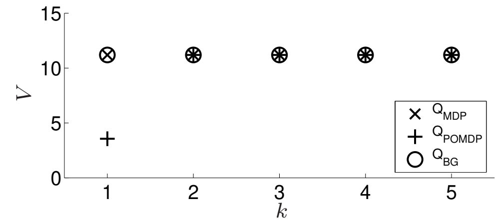
7.4 Forward-Sweep Policy Computation

The MAA* results described above indicate that the use of a tighter heuristic can yield substantial time savings. In this section, the approximate Q-value functions are used in forward-sweep policy computation. We would expect that when using a Q-value function that more closely resembles Q^*, the quality of the resulting policy will be higher. We also tested whether k-GMAA* with $k > 1$ improved the quality of the computed policies. In particular, we tested $k = 1,2,\ldots,5$.

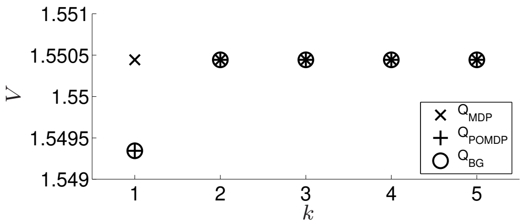
For the Dec-Tiger problem, k-GMAA* with $k = 1$ (and thus also $2 \leq k \leq 5$) found the optimal policy (with $V(\pi^*) = 5.19$) for horizon 3 using all approximate Q-value functions. For horizon $h = 4$, also all different values of k produced the same result for each approximate Q-value function. In this case, however, Q_{MDP} found a policy with expected return of 3.19. Q_{POMDP} and Q_{BG} did find the optimal policy ($V(\pi^*) = 4.80$). Figure 12 illustrates the optimal policy (right) and the one found by Q_{MDP} (left). It shows that Q_{MDP} overestimates the value for opening the door in stage $t = 2$.



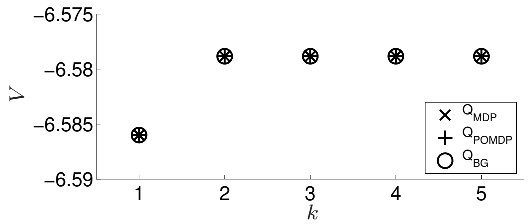
(a) Skewed Dec-Tiger, $h = 3$.



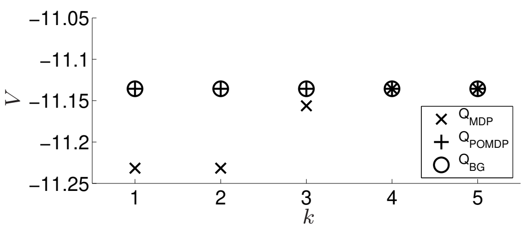
(b) Skewed Dec-Tiger, $h = 4$.



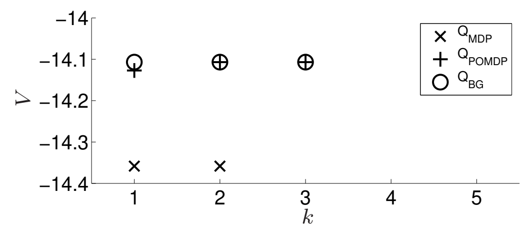
(c) GridSmall, $h = 3$.



(d) FireFighting with $\langle n_h = 3, n_f = 3 \rangle$, $h = 4$.



(e) FireFighting with $\langle n_h = 4, n_f = 3 \rangle$, $h = 3$.



(f) FireFighting with $\langle n_h = 4, n_f = 3 \rangle$, $h = 4$.

Figure 13: k-GMAA* results for different problems and horizons. The y-axis indicates value of the initial joint belief, while the x-axis denotes k.

For the Skewed Dec-Tiger problem, different values of k did produce different results. In particular, for $h = 3$ only Q_{BG} finds the optimal policy (and thus attains the optimal value) for all values of k, as shown in Figure 13(a). Q_{POMDP} does find it starting from $k = 2$, and Q_{MDP} only from $k = 5$. Figure 13(b) shows a somewhat unexpected result for $h = 4$: here for $k = 1$ Q_{MDP} and Q_{BG} find the optimal policy, but Q_{POMDP} doesn't. This clearly illustrates that a tighter approximate Q-value function is not a guarantee for a better joint policy, which is also illustrated by the results for GridSmall in Figure 13(c).

We also performed the same experiment for two settings of the FireFighting problem. For $\langle n_h = 3, n_f = 3 \rangle$ and $h = 3$ all Q-value functions found the optimal policy (with value -5.7370) for all k, and horizon 4 is shown in Figure 13(d). Figures 13(e) and 13(f) show the results for $\langle n_h = 4, n_f = 3 \rangle$. For $h = 4$, Q_{MDP} did not finish for $k \geq 3$ within 5 days.

It is encouraging that for all experiments k-GMAA* using Q_{BG} and Q_{POMDP} with $k \leq 2$ found the optimal policy. Using Q_{MDP} the optimal policy was also always found with $k \leq 5$, except in horizon 4 Dec-Tiger and the $\langle n_h = 4, n_f = 3 \rangle$ FireFighting problem. These results seem to indicate that this type of approximation might be likely to produce (near-) optimal results for other domains as well.

8. Conclusions

A large body of work in single-agent decision-theoretic planning is based on value functions, but such theory has been lacking thus far for Dec-POMDPs. Given the large impact of value functions on single-agent planning under uncertainty, we expect that a thorough study of value functions for Dec-POMDPs can greatly benefit multiagent planning under certainty. In this work, we presented a framework of Q-value functions for Dec-POMDPs, providing a significant contribution to fill this gap in Dec-POMDP theory. Our theoretical contributions have lead to new insights, which we applied to improve and extend solution methods.

We have shown how an optimal joint policy π^* induces an optimal Q-value function $Q^*(\vec{\theta}^t,a)$, and how it is possible to construct the optimal policy π^* using forward-sweep policy computation. This entails solving Bayesian games for time steps $t = 0,...,h-1$ which use $Q^*(\vec{\theta}^t,a)$ as the payoff function. Because there is no clear way to compute $Q^*(\vec{\theta}^t,a)$, we introduced a different description of the optimal Q-value function $Q^*(\vec{\theta}^t,\varphi^{t+1})$ that is based on sequential rationality. This new description of Q^* can be computed using dynamic programming and can then be used to construct π^*.

Because calculating Q^* is computationally expensive, we examined approximate Q-value functions that can be calculated more efficiently and we discussed how they relate to Q^*. We covered $\mathrm{Q}_{\mathrm{MDP}}$, $\mathrm{Q}_{\mathrm{POMDP}}$, and Q_{BG}, a recently proposed approximate Q-value function. Also, we established that decreasing communication delays in decentralized systems cannot decrease the expected value and thus that $Q^* \leq \mathrm{Q}_{\mathrm{BG}} \leq \mathrm{Q}_{\mathrm{POMDP}} \leq \mathrm{Q}_{\mathrm{MDP}}$. Experimental evaluation indicated that these upper bounds are not just of theoretical interest, but that significant differences exist in the tightness of the various approximate Q-value functions.

Additionally we showed how the approximate Q-value functions can be used as heuristics in a generalized policy search method GMAA*, thereby unifying forward-sweep policy computation and the recent Dec-POMDP solution techniques of Emery-Montemerlo et al. (2004) and Szer et al. (2005). Finally, we performed an empirical evaluation of GMAA* showing significant reductions in computation time when using tighter heuristics to calculate optimal policies. Also Q_{BG} generally found better approximate solutions in forward-sweep policy computation and the 'k-best joint BG policies' GMAA* variant, or k-GMAA*.

There are quite a few directions for future research. One is to try to extend the results of this paper to partially observable stochastic games (POSGs) (Hansen et al., 2004), which are Dec-POMDPs with an individual reward function for each agent. Since the dynamics of the POSG model are identical to those of a Dec-POMDP, a similar modeling via Bayesian games is possible. An interesting question is whether also in this case, an optimal (i.e., rational) joint policy can be found by forward-sweep policy computation.

Staying within the context of Dec-POMDPs, a research direction could be to further generalize GMAA*, by defining other `Next` or `Select` operators, with the hope that the resulting algorithms will be able to scale to larger problems. Also it is important to establish bounds on the performance and learning curves of GMAA* in combination with different `Next` operators and heuristics. A different direction is to experimentally evaluate the use of even tighter heuristics such as Q-value functions for the case of observations delayed by multiple time steps. This research should be paired with methods to efficiently find such Q-value functions. Finally, future research should further examine Bayesian games. In particular, the work of Emery-Montemerlo et al. (2005) could be used as a starting point for further research to approximately modeling Dec-POMDPs using BGs. Finally, there is a need for efficient approximate methods for solving the Bayesian games.

Acknowledgments

We thank the anonymous reviewers for their useful comments. The research reported here is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant nr: BSIK03024. This work was supported by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) through the POS_Conhecimento Program that includes FEDER funds, and through grant PTDC/EEA-ACR/73266/2006.

Appendix A. Proofs

A.1 There is At Least One Optimal Pure Joint Policy

Proposition (2.1)**.** *A Dec-POMDP has at least one optimal pure joint policy.*

Proof. This proof follows a proof by Schoute (1978). It is possible to convert a Dec-POMDP to an extensive game and thus to a strategic game, in which the actions are pure policies for the Dec-POMDP (Oliehoek & Vlassis, 2006). In this strategic game, there is at least one maximizing entry corresponding to a pure joint policy which we denote $\pi_{\max}$. Now, assume that there is a joint stochastic policy $\varsigma = \langle \varsigma_1, \ldots, \varsigma_n \rangle$ that attains a higher payoff. Kuhn (1953) showed that for each stochastic ς_i policy, there is a corresponding mixed policy μ_i. Therefore ς corresponds to a joint mixed policy $\mu = \langle \mu_1, \ldots, \mu_n \rangle$. Let us write Π_{i,μ_i} for the support of μ_i. μ now induces a probability distribution P_μ over the set of joint policies $\Pi_\mu = \Pi_{1,\mu_1} \times \cdots \times \Pi_{n,\mu_n} \subseteq \Pi$ which is a subset of the set of all joint policies. The expected payoff can now be written as

$$V(\varsigma) = E_{P_\mu}(V(\pi) | \pi \in \Pi_\mu) \leq \max_{\pi \in \Pi} V(\pi) = V(\pi_{\max}),$$

contradicting that ς is a joint stochastic policy that attains a higher payoff. $\square$

A.2 Hierarchy of Q-value Functions

This section lists the proof of theorem 5.1. It is ordered as follows. First, Section A.2.1 presents a model and resulting value functions for Dec-POMDPs with k-steps delayed communication. Next, Section A.2.2 shows that $\mathrm{Q}_{\mathrm{POMDP}}$, Q_{BG} and Q^* correspond with the case that k is respectively 0, 1 and h. Finally, Section A.2.3 shows that when the communication delay k increases, the optimal expected return cannot decrease, thereby proving theorem 5.1.

A.2.1 Modeling Dec-POMDPs with k-Steps Delayed Communication

Here we present an augmented MDP that can be used to find the optimal solution for Dec-POMDPs with k steps delayed communication. This is a reformulation of the work by Aicardi et al. (1987) and Ooi and Wornell (1996), extended to the Dec-POMDP setting.

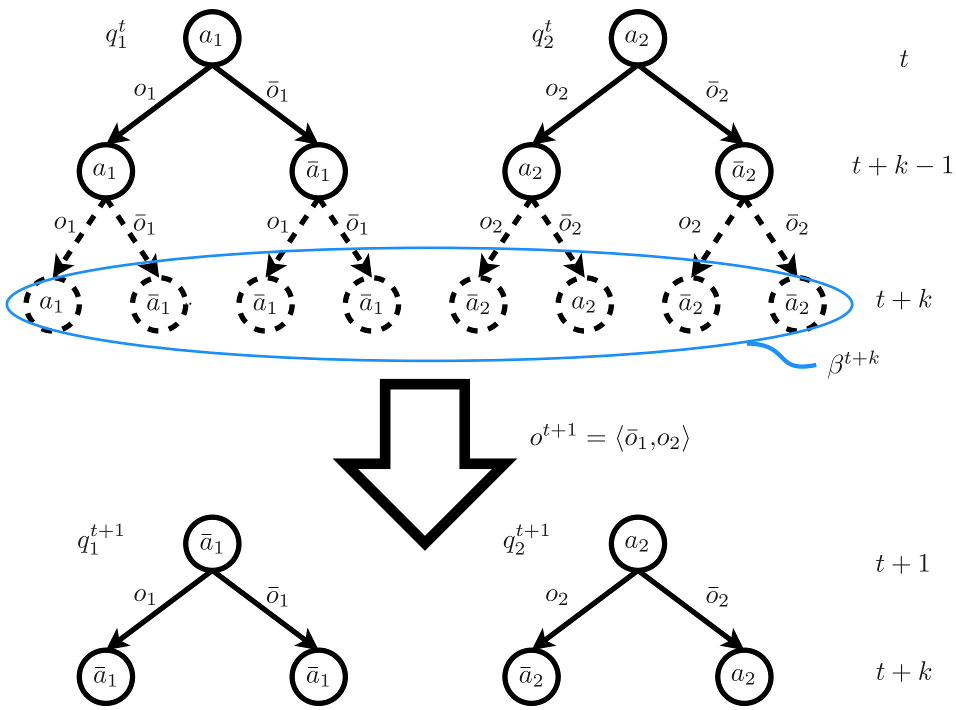


Figure 14: Policies specified by the states of the augmented MDP for $k = 2$. Top: policies for $\hat{s}^t$. The policy extended by augmented MDP action $\hat{a}^t = \beta$ is shown dashed. Bottom: The resulting policies after for joint observation $\langle \bar{o}_1, o_2 \rangle$.

We define this augmented MDP as $\hat{M} = \langle \hat{\mathcal{S}}, \hat{\mathcal{A}}, \hat{T}, \hat{R} \rangle$, where the augmented MDP stages are indicated $\hat{t}$.

The state space is $\hat{\mathcal{S}} = (\hat{\mathcal{S}}^{\hat{t}=0}, \ldots, \hat{\mathcal{S}}^{\hat{t}=h-1})$. An augmented state is composed of a joint action-observation history, and a joint policy tree q^t.

$$\hat{s}^{\hat{t}=t} = \begin{cases} \langle \vec{\theta}^{\,t}, q^t \rangle & , 0 \leq t \leq h-k-1 \\ \langle \vec{\theta}^{\,t}, q^{\tau=h-t,t} \rangle & , h-k \leq t \leq h-1 \end{cases}.$$

The contained q^t is a joint depth-k (specifying actions for k stages) joint policy tree $q^t = \left\langle q_1^t, \ldots, q_n^t \right\rangle$, to be used starting at stage t. For the last k stages, the contained joint policy $q^{\tau=h-t,t}$ specifies $\tau = h - t \leq k$ stages.

$\hat{\mathcal{A}}$ is the set of augmented actions. For $0 \leq \hat{t} \leq h-k-1$, an action $\hat{a}^{\hat{t}} \in \hat{\mathcal{A}}$ is a joint policy $\hat{a}^{\hat{t}=t} = \beta^{t+k} = \langle \beta_1^{t+k} \ldots \beta_n^{t+k} \rangle$ implicitly mapping length-k observation histories to joint actions to be taken at stage $t+k$. I.e., $\beta_i^{t+k} : \vec{\mathcal{O}}_i^k \to \mathcal{A}_i^{t+k}$. For the last k stages $h-k \leq \hat{t} \leq h-1$ there only is one empty action $a_\emptyset$ that has no influence whatsoever.

The augmented actions are used to expand the joint policy trees. When 'appending' a policy β^{t+k} to q^t we form a depth $k+1$ policy, which we denote $q^{\tau=k+1,t} = \langle q^t \circ \beta^{t+k} \rangle$. After execution of its initial joint action $q^{\tau=k+1,t}(\vec{o}_\emptyset)$ and receiving a particular joint observation o,

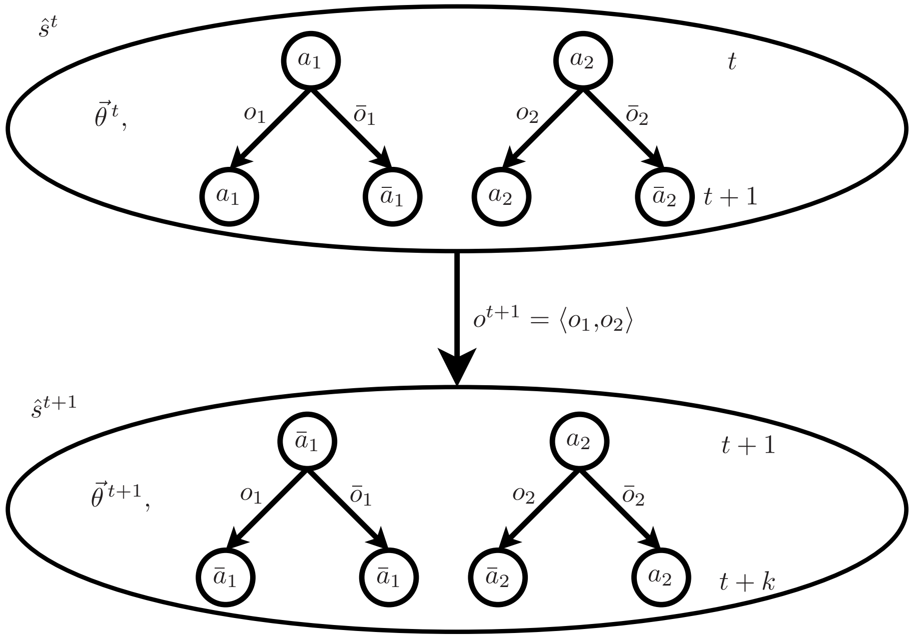


Figure 15: An illustration of the augmented MDP with $k = 2$, showing a transition from $\hat{s}^t$ to $\hat{s}^{t+1}$ by action $\hat{a} = \beta^t$. In this example $\vec{\theta}^{t-k+1} = (\vec{\theta}^{t-k+1}, \langle a_1,a_2 \rangle, \langle \bar{o}_1,o_2 \rangle)$. The actions specified for stage t are given by $\beta^t(\langle \bar{o}_1,o_2 \rangle)$ as depicted in Figure 14.

a $q^{\tau=k+1,t}$ reduces to its depth k sub-tree policy for that particular joint observation, denoted $q^{t-k+1} = q^{\tau=k+1,t}(o) = \langle q^{t-k} \circ \beta^t \rangle(o)$. This is illustrated in Figure 14.

$\hat{T}$ is the transition model. A probability $P(\hat{s}^{\hat{t}+1}|\hat{s}^{\hat{t}},\hat{a}^{\hat{t}})$ for stage $\hat{t} = t$ translates as follows for $0 \leq t \leq h-k-1$

$$P(\langle \vec{\theta}^{t+1},q^{t+1} \rangle | \langle \vec{\theta}^t,q^t \rangle,\beta^{t+k}) = \begin{cases} P(o^{t+1}|\vec{\theta}^t,q^t(\vec{o}_\emptyset)) & \text{if conditions hold,} \\ 0 & \text{otherwise,} \end{cases} \tag{A.1}$$

where the conditions are: 1) $q^{t+1} = \langle q^t \circ \beta^{t+k} \rangle(o^{t+1})$, and 2) $\vec{\theta}^{t+1} = (\vec{\theta}^t,a^t,o^{t+1})$. For $h-k \leq t \leq h-1$, β^{t+k} in (A.1) reduces to $a_\emptyset$. The probabilities are unaffected, but the first condition changes to $q^{\tau=h-t-1,t+1} = q^{\tau=h-t,t}(o^{t+1})$.

Finally, $\hat{R}$ is the reward model, which is specified as follows:

$$\forall_{0 \leq t \leq h-1} \quad \hat{R}(\hat{s}^{\hat{t}=t}) = \hat{R}(\langle \vec{\theta}^t,q^t \rangle) = R(\vec{\theta}^t,q^t(\vec{o}_\emptyset)), \tag{A.2}$$

where $q^t(\vec{o}_\emptyset)$ is the initial joint action specified by q^t. $R(\vec{\theta}^t,a)$ is defined as before in (2.12).

The resulting optimality equations $\hat{Q}^{\hat{t}}(\hat{s},\hat{a})$ for the augmented MDP are as follows. We will write Q_k for the optimal Q-value function for a k-steps delayed communication system. We will also refer to this as the k-$\mathrm{Q_{BG}}$ value function.

$$\forall_{0 \leq t \leq h-k-1} \quad Q_k(\vec{\theta}^t,q^t,\beta^{t+k}) = R(\vec{\theta}^t,q^t(\vec{o}_\emptyset)) + \sum_{o^{t+1} \in \mathcal{O}} P(o^{t+1}|\vec{\theta}^t,q^t(\vec{o}_\emptyset)) Q_k^*(\vec{\theta}^{t+1},q^{t+1}), \tag{A.3}$$

with $q^{t+1} = \langle q^t \circ \beta^{t+k} \rangle (o^{t+1})$ and where

$$Q_k^*(\vec{\theta}^t, q^t) \equiv \max_{\beta^{t+k}} Q_k(\vec{\theta}^t, q^t, \beta^{t+k}). \tag{A.4}$$

For the last k stages, $h - k \leq t \leq h - 1$, there are $\tau' = h - t$ stages to go and we get

$$Q_k^*(\vec{\theta}^t, q^{\tau=\tau',t}) = R(\vec{\theta}^t, q^{\tau=\tau',t}(\vec{o}_\emptyset)) + \sum_{o^{t+1}} P(o^{t+1} | \vec{\theta}^t, q^{\tau=\tau',t}(\vec{o}_\emptyset)) Q_k^*(\vec{\theta}^{t+1}, q^{\tau=\tau'-1,t+1}). \tag{A.5}$$

Note that (A.5) does not include any augmented actions $\hat{a}^{\hat{t}=t} = \beta^{t+k}$. Therefore, the last k stages should be interpreted as a Markov chain. Standard dynamic programming can be applied to calculate all $Q^*(\vec{\theta}^t, q^t)$-values.

A.2.2 Relation of k-$\mathrm{Q_{BG}}$ with Other Approximate Q-value Functions

Here we briefly show how k-$\mathrm{Q_{BG}}$ in fact reduces to some of the cases treated earlier.

For $k = 0$, k-$\mathrm{Q_{BG}}$ (A.3) reduces to $\mathrm{Q_{POMDP}}$. In the $k = 0$ case, q^{t-k} becomes a depth-0, i.e. empty, policy. Also, β^t becomes a mapping from length-0 observation histories to actions, i.e., it becomes a joint action. Substitution in (A.3) yields

$$Q_0(\left\langle \vec{\theta}^t, \emptyset \right\rangle, a^t) = R(\vec{\theta}^t, a^t) + \sum_{o^{t+1}} P(o^{t+1} | \vec{\theta}^t, a^t) \max_{a^{t+1}} Q_0(\left\langle \vec{\theta}^{t+1}, \emptyset \right\rangle, a^{t+1}).$$

Now, as $\widehat{Q}_{\mathrm{P}}(\vec{\theta}^t, a) \equiv Q_{\mathrm{P}}^*(b^{\vec{\theta}^t}, a)$, this clearly corresponds to the $\mathrm{Q_{POMDP}}$-value function (5.6).

1-$\mathrm{Q_{BG}}$ reduces to regular $\mathrm{Q_{BG}}$. Notice that for $k = 1$, $q^{\tau=k,t}$ reduces to a^t. Filling out yields:

$$Q_1(\left\langle \vec{\theta}^t, a^t \right\rangle, \beta^{t+1}) = R(\vec{\theta}^t, a^t) + \sum_{o^{t+1}} P(o^{t+1} | \vec{\theta}^t, a^t) \max_{\beta^{t+2}} Q_1(\left\langle \vec{\theta}^{t+1}, \beta^{t+1}(o^{t+1}) \right\rangle, \beta^{t+2}).$$

Now using (A.4) we obtain the $\mathrm{Q_{BG}}$-value function (5.12).

A Dec-POMDP is identical to an h-steps delayed communication system. Augmented states have the form $\hat{s}^{\hat{t}=0} = \langle \vec{\theta}_\emptyset, q^0 \rangle$, where $q^0 = \pi$ specifies a full length h joint policy. The first stage $t = 0$ in this augmented MDP, is also one of the last k $(= h)$ stages. Therefore, the applied Q-function is (A.5), which means that the Markov chain evaluation starts immediately. Effectively this boils down to evaluation of all joint policies (corresponding to all augmented start states). The maximizing one specifies the value function of an optimal joint policy Q^*.

A.2.3 Shorter Communication Delays cannot Decrease the Value

First, we introduce some notation. Let us write P_o for all the observation probabilities given $\vec{\theta}^t, q^t$ and the sequence of 'intermediate observations' $(o^{t+1}, \ldots, o^{t+l-1})$

$$P_o(o^{t+l}) \equiv P\left[o^{t+l} | (\vec{\theta}^t, q^t(\vec{o}_\emptyset), o^{t+1}, q^t(o^{t+1}), \ldots, o^{t+l-1}), q^t(o^{t+l-1}) \right], \qquad \forall l \leq k. \tag{A.6}$$

In order to avoid confusion, we write $\beta^t_{|k|}$ for a policy that implicitly maps k-length observation histories to actions, and $\beta^t_{|k+1|}$ for one that is a mapping from length $(k+1)$ observation-histories to actions.

Now we give a reformulation of Q_k. $Q^{t,*}_k(\vec{\theta}^t,q^t)$ specifies the expected return for $\vec{\theta}^t,q^t$ over stages $t,t+1,\ldots,h-1$. Here, we will split this

$$Q^*_k(\vec{\theta}^t,q^t) = K_k(\vec{\theta}^t,q^t) + F^{t,*}_k(\vec{\theta}^t,q^t) \tag{A.7}$$

in $K_k(\vec{\theta}^t,q^t)$, the *expected k-step reward*, i.e., the expected return over for stages $t,\ldots,t+k-1$ and $F^{t,*}_k(\vec{\theta}^t,q^t)$, the expected return over stages $t+k,t+k+1,\ldots,h-1$, referred to as the 'in k-steps' expected return.

The former is defined as

$$K_k(\vec{\theta}^t,q^t) \;\equiv\; E\left[\sum_{t'=t}^{t+k-1} R(\vec{\theta}^{t'},a^{t'}) \,\middle|\, \vec{\theta}^t,q^t\right]. \tag{A.8}$$

Let us define $K^{\tau=i}(\vec{\theta}^t,q^{\tau=i,t})$ as the expected reward for the next i stages, i.e.,

$$K_k(\vec{\theta}^t,q^t) = K^{\tau=k}(\vec{\theta}^t,q^t). \tag{A.9}$$

We then have $K^{\tau=1}(\vec{\theta}^t,a^t) = R(\vec{\theta}^t,a^t)$ and

$$\begin{aligned} K^{\tau=i}(\vec{\theta}^t,q^{\tau=i,t}) = R(\vec{\theta}^t,q^{\tau=i,t}(\vec{o}_\emptyset))+ \\ \sum_{o^{t+1}} P(o^{t+1}|\vec{\theta}^t,q^{\tau=i,t}(\vec{o}_\emptyset))K^{\tau=i-1}(\vec{\theta}^{t+1},q^{\tau=i-1,t+1}(o^{t+1})), \end{aligned} \tag{A.10}$$

where $q^{\tau=i-1,t+1}(o^{t+1})$ is the depth-$(i-1)$ joint policy that results from $q^{\tau=i,t}$ after observation of o^{t+1}.

If we define $F^{\tau=i,t}_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|})$ to be the expected reward for stages $t+i,t+i+1,\ldots,h-1$. That is, the time-to-go $\tau=i$ denotes how much time-to-go before we start accumulating expected reward. The 'in k-steps' expected return is then given by

$$F^t_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|}) = F^{\tau=k,t}_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|}).$$

The evaluation is then performed by

$$F^{\tau=0,t}_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|}) \;=\; Q_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|}) \tag{A.11}$$

$$F^{\tau=i,t}_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|}) \;=\; \sum_{o^{t+1}} P_o(o^{t+1})F^{\tau=i-1,t+1,*}_k(\vec{\theta}^{t+1},q^{t+1}), \tag{A.12}$$

where $q^{t+1} = \langle q^{t+1} \circ \beta^{t+k}_{|k|}\rangle(o^{t+1})$, and where

$$F^{\tau=i,t,*}_k(\vec{\theta}^t,q^t) = \max_{\beta^{t+k}_{|k|}} F^{\tau=i,t}_k(\vec{\theta}^t,q^t,\beta^{t+k}_{|k|}). \tag{A.13}$$

Theorem A.1 (Shorter communication delays cannot decrease the value)**.** *The optimal Q-value function Q_k of a finite horizon Dec-POMDP with k-steps delayed communication is an upper bound to Q_{k+1}, that of a $k+1$-steps delayed communication system. That is*

$$\forall_t \forall_{\vec{\theta}^t} \forall_{q^{\tau=k,t},\beta_{|k|}^{t+k}} \quad Q_k(\vec{\theta}^t, q^{\tau=k,t}, \beta_{|k|}^{t+k}) \geq \max_{\beta_{|k+1|}^{t+k+1}} Q_{k+1}(\vec{\theta}^t, \langle q^{\tau=k,t} \circ \beta_{|k|}^{t+k} \rangle, \beta_{|k+1|}^{t+k+1}). \tag{A.14}$$

Proof. The proof is by induction. The base case is that (A.14) holds for stages $h-(k+1) \leq t \leq h-1$, as shown by lemma A.1. The induction hypothesis states that, assuming (A.14) holds for some stage $t+k$, it also holds for stage t. The induction step is proven in lemma A.2. □

Lemma A.1 (Base case)**.** *For all $h-k-1 \leq t \leq h-1$, the expected cumulative future reward under k steps delay is equal to that under $k+1$ steps delay if the same policies are followed from that point. That is,*

$$\forall_{h-k \leq t \leq h-1} \forall_{\vec{\theta}^t} \forall_{q^{\tau=h-t,t}} \quad Q_k^*(\vec{\theta}^t, q^{\tau=h-t,t}) = Q_{k+1}^*(\vec{\theta}^t, q^{\tau=h-t,t}), \tag{A.15}$$

and $\forall_{\vec{\theta}^{h-k-1}} \forall_{q^{\tau=k,h-k-1},\beta_{|k|}^{h-1}}$

$$Q_k(\vec{\theta}^{h-k-1}, q^{\tau=k,h-k-1}, \beta_{|k|}^{h-1}) = Q_{k+1}(\vec{\theta}^{h-k-1}, \langle q^{\tau=k,h-k-1} \circ \beta_{|k|}^{h-1} \rangle). \tag{A.16}$$

Proof. For a particular stage $t = h - \tau'$ with $h-k \leq t \leq h-1$ and an arbitrary $\vec{\theta}^t, q^{\tau=\tau',t}$, we can write

$$Q_k^*(\vec{\theta}^t, q^{\tau=\tau',t}) = Q_{k+1}^*(\vec{\theta}^t, q^{\tau=\tau',t}),$$

because both are given by the evaluation of (A.5), and this evaluation involves no actions: Basically (A.5) has reduced to a Markov chain, and this Markov chain is the same for Q_k^* and Q_{k+1}^*. We can conclude that

$$\forall_{h-k \leq t \leq h-1} \forall_{\vec{\theta}^t, q^{\tau=\tau',t}} \quad Q_k^*(\vec{\theta}^t, q^{\tau=\tau',t}) = Q_{k+1}^*(\vec{\theta}^t, q^{\tau=\tau',t}).$$

Now we will prove (A.16). The left side of (A.16) is given by application of (A.3)

$$Q_k(\vec{\theta}^{h-k-1}, q^{\tau=k,h-k-1}, \beta_{|k|}^{h-1}) = R(\vec{\theta}^{h-k-1}, q^{\tau=k,h-k-1}(\vec{o}_\emptyset)) + \sum_{o^{h-k}} P(o^{h-k}|\vec{\theta}^{h-k-1}, q^{\tau=k,h-k-1}(\vec{o}_\emptyset)) Q_k^*(\vec{\theta}^{h-k}, q^{\tau=k,h-k}),$$

with $q^{\tau=k,h-k} = \langle q^{\tau=k,h-k-1} \circ \beta_{|k|}^{h-1} \rangle (o^{h-k})$. The right side is given by application of (A.5)

$$Q_{k+1}(\vec{\theta}^{h-k-1}, \langle q^{\tau=k,h-k-1} \circ \beta_{|k|}^{h-1} \rangle) = R(\vec{\theta}^{h-k-1}, q^{\tau=k,h-k-1}(\vec{o}_\emptyset)) + \sum_{o^{h-k}} P(o^{h-k}|\vec{\theta}^{h-k-1}, q^{\tau=k,h-k-1}(\vec{o}_\emptyset)) Q_{k+1}^*(\vec{\theta}^{h-k}, q^{\tau=k,h-k})$$

with $q^{\tau=k,h-k} = \langle q^{\tau=k,h-k-1} \circ \beta_{|k|}^{h-1} \rangle (o^{h-k})$. Now, because the policies $q^{\tau=k,h-k}$ are the same, we get

$$Q_k^*(\vec{\theta}^{h-k}, q^{\tau=k,h-k}) = Q_{k+1}^*(\vec{\theta}^{h-k}, q^{\tau=k,h-k})$$

and thus (A.16) holds. □

Lemma A.2 (Induction step)**.** *Given that*

$$\forall_{\vec{\theta}^{t'}} \forall_{q^{\tau=k,t'},\beta^{t'+k}_{|k|}} \quad Q_k(\vec{\theta}^{t'},q^{\tau=k,t'},\beta^{t'+k}_{|k|}) \geq \max_{\beta^{t'+k+1}_{|k+1|}} Q_{k+1}(\vec{\theta}^{t'},\langle q^{\tau=k,t'} \circ \beta^{t'+k}_{|k|}\rangle,\beta^{t'+k+1}_{|k+1|}) \quad \text{(A.17)}$$

holds for $t' = t + (k+1)$*, then*

$$\forall_{\vec{\theta}^{t}} \forall_{q^{\tau=k,t},\beta^{t+k}_{|k|}} \quad Q_k(\vec{\theta}^{t},q^{\tau=k,t},\beta^{t+k}_{|k|}) \geq \max_{\beta^{t+k+1}_{|k+1|}} Q_{k+1}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|}\rangle,\beta^{t+k+1}_{|k+1|}) \qquad \text{(A.18)}$$

holds for stage t*.*

Proof. For the k-steps delay Q-function, we can write

$$\begin{aligned} &Q_k(\vec{\theta}^{t},q^{\tau=k,t},\beta^{t+k}_{|k|}) = R(\vec{\theta}^{t},q^{\tau=k,t}(\vec{o}_\emptyset))+ \\ &\quad \sum_{o^{t+1}} P_o(o^{t+1}|\vec{\theta}^{t},q^{\tau=k,t}(\vec{o}_\emptyset)) \max_{\beta^{t+k+1}_{|k|}} \Big[K_k(\vec{\theta}^{t+1},q^{t+1}) + F^{t+1}_k(\vec{\theta}^{t+1},q^{\tau=k,t+1},\beta^{t+k+1}_{|k|})\Big] \qquad \text{(A.19)} \end{aligned}$$

where $q^{\tau=k,t+1} = \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|}\rangle(o^{t+1})$. Because K_k is independent of $\beta^{t+k+1}_{|k|}$, we can regroup the terms to get

$$\begin{aligned} Q_k(\vec{\theta}^{t},q^{\tau=k,t},\beta^{t+k}_{|k|}) = &\left[R(\vec{\theta}^{t},q^{\tau=k,t}(\vec{o}_\emptyset)) + \sum_{o^{t+1}} P_o(o^{t+1}) K_k(\vec{\theta}^{t+1},q^{\tau=k,t+1})\right] + \\ &\left[\sum_{o^{t+1}} P_o(o^{t+1}) \max_{\beta^{t+k+1}_{|k|}} F^{t+1}_k(\vec{\theta}^{t+1},q^{\tau=k,t+1},\beta^{t+k+1}_{|k|})\right]. \qquad \text{(A.20)} \end{aligned}$$

In the case of $k+1$-steps delay, we can write

$$Q_{k+1}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|}\rangle,\beta^{t+k}_{|k+1|}) = K_{k+1}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|}\rangle) + F^{t}_{k+1}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta^{t+k}_{|k+1|}\rangle,\beta^{t+k+1}_{|k+1|}) \qquad \text{(A.21)}$$

where, per definition (by (A.9) and (A.10))

$$\begin{aligned} K_{k+1}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|}\rangle) &= R(\vec{\theta}^{t},q^{\tau=k,t}(\vec{o}_\emptyset)) + \sum_{o^{t+1}} P_o(o^{t+1}) K^{\tau=k}(\vec{\theta}^{t+1},q^{\tau=k,t+1}), \\ &= R(\vec{\theta}^{t},q^{\tau=k,t}(\vec{o}_\emptyset)) + \sum_{o^{t+1}} P_o(o^{t+1}) K_k(\vec{\theta}^{t+1},q^{\tau=k,t+1}), \qquad \text{(A.22)} \end{aligned}$$

where $q^{\tau=k,t+1} = \langle q^{\tau=k,t} \circ \beta^{t+k}\rangle(o^{t+1})$.

Equation (A.22) is equal to the first part in (A.20). Therefore, for an arbitrary $\vec{\theta}^{t}$,$q^{\tau=k,t}$ and $\beta^{t+k}_{|k|}$, we know that (A.18) holds if and only if

$$\sum_{o^{t+1}} P_o(o^{t+1}) \max_{\beta^{t+k+1}_{|k|}} F^{t+1}_k(\vec{\theta}^{t+1},q^{\tau=k,t+1},\beta^{t+k+1}_{|k|}) \geq \max_{\beta^{t+k+1}_{|k+1|}} F^{t}_{k+1}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|}\rangle,\beta^{t+k+1}_{|k+1|}) \qquad \text{(A.23)}$$

where $q^{\tau=k,t+1} = \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1})$. When filling this out and expanding F^t_{k+1} using (A.12) we get

$$\sum_{o^{t+1}} P_o(o^{t+1}) \max_{\beta^{t+k+1}_{|k|}} F_k^{\tau=k,t+1}(\vec{\theta}^{t+1}, \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1}), \beta^{t+k+1}_{|k|}) \geq \max_{\beta^{t+k+1}_{|k+1|}} \sum_{o^{t+1}} P_o(o^{t+1}) F_{k+1}^{\tau=k,t+1,*}(\vec{\theta}^{t+1}, \langle \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle \circ \beta^{t+k+1}_{|k+1|} \rangle (o^{t+1})). \quad \text{(A.24)}$$

This clearly holds if

$$\sum_{o^{t+1}} P_o(o^{t+1}) \max_{\beta^{t+k+1}_{|k|}} F_k^{\tau=k,t+1}(\vec{\theta}^{t+1}, \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1}), \beta^{t+k+1}_{|k|}) \geq \sum_{o^{t+1}} P_o(o^{t+1}) \max_{\beta^{t+k+1}_{|k|}} F_{k+1}^{\tau=k,t+1,*}(\vec{\theta}^{t+1}, \langle \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1}) \circ \beta^{t+k+1}_{|k|} \rangle), \quad \text{(A.25)}$$

because the second part of (A.25) is an upper bound to the second part of (A.24). Therefore, the induction step is proved if we can show that

$$\forall_{o^{t+1}} \forall_{\beta^{t+k+1}_{|k|}} \quad F_k^{\tau=k,t+1}(\vec{\theta}^{t+1}, \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1}), \beta^{t+k+1}_{|k|}) \geq F_{k+1}^{\tau=k,t+1,*}(\vec{\theta}^{t+1}, \langle \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1}) \circ \beta^{t+k+1}_{|k|} \rangle). \quad \text{(A.26)}$$

which through (A.13) and $q^{\tau=k,t+1} = \langle q^{\tau=k,t} \circ \beta^{t+k}_{|k|} \rangle (o^{t+1})$ transforms to

$$\forall_{q^{\tau=k,t+1}} \forall_{\beta^{t+k+1}_{|k|}} \quad F_k^{\tau=k,t+1}(\vec{\theta}^{t+1}, q^{\tau=k,t+1}, \beta^{t+k+1}_{|k|}) \geq \max_{\beta^{t+k+2}_{|k+1|}} F_{k+1}^{\tau=k,t+1}(\vec{\theta}^{t+1}, \langle q^{\tau=k,t+1} \circ \beta^{t+k+1}_{|k|} \rangle, \beta^{t+k+2}_{|k+1|}). \quad \text{(A.27)}$$

Now, we apply (A.11) to the induction hypothesis (A.17) and yield

$$\forall_{\vec{\theta}^{t'}} \forall_{q^{\tau=k,t'}, \beta^{t'+k}_{|k|}} \quad F_k^{\tau=0,t'}(\vec{\theta}^{t'}, q^{\tau=k,t'}, \beta^{t'+k}_{|k|}) \geq \max_{\beta^{t'+k+1}_{|k+1|}} F_{k+1}^{\tau=0,t'}(\vec{\theta}^{t'}, \langle q^{\tau=k,t'} \circ \beta^{t'+k}_{|k|} \rangle, \beta^{t'+k+1}_{|k+1|}). \quad \text{(A.28)}$$

Application of lemma A.4 to this transformed induction hypothesis asserts (A.27) and thereby proves the lemma. □

Auxiliary Lemmas.

Lemma A.3. *If, at stage t, the 'in i-steps' expected return for a k-steps delayed system is higher than a $(k+1)$-steps delayed system, then at $t-1$ the 'in $(i+1)$-steps' expected return for a k-steps delayed system is higher than the $(k+1)$-steps delayed system. That is, if for a particular $q^{\tau=k,t} = \langle q^{\tau=k,t-1} \circ \beta^{t-1+k}_{|k|} \rangle (o^t)$*

$$\forall_{\beta^{t+k}_{|k|}} \forall_{o^t} \quad F_k^{\tau=i,t}(\vec{\theta}^t, q^{\tau=k,t}, \beta^{t+k}_{|k|}) = F_k^{\tau=i,t}(\vec{\theta}^t, \langle q^{\tau=k,t-1} \circ \beta^{t-1+k}_{|k|} \rangle (o^t), \beta^{t+k}_{|k|}) \geq \max_{\beta^{t+k+1}_{|k+1|}} F_{k+1}^{\tau=i,t}(\vec{\theta}^t, \langle \langle q^{\tau=k,t-1} \circ \beta^{t+k}_{|k|} \rangle (o^t) \circ \beta^{t+k}_{|k|} \rangle, \beta^{t+k+1}_{|k+1|}) \quad \text{(A.29)}$$

holds, then

$$F_k^{\tau=i+1,t-1}(\vec{\theta}^{t-1},q^{\tau=k,t-1},\beta_{|k|}^{t-1+k}) \geq \max_{\beta_{|k+1|}^{t+k}} F_{k+1}^{\tau=i+1,t-1}(\vec{\theta}^{t-1},\langle q^{\tau=k,t-1} \circ \beta_{|k|}^{t-1+k}\rangle,\beta_{|k+1|}^{t+k}). \tag{A.30}$$

Proof. The following derivation

$$\begin{aligned}
&F_k^{\tau=i+1,t-1}(\vec{\theta}^{t-1},q^{\tau=k,t-1},\beta_{|k|}^{t-1+k})\\
&=\sum_{o^t} P_o(o^t) \max_{\beta_{|k|}^{t+k}} \left[F_k^{\tau=i,t}(\vec{\theta}^{t},\langle q^{\tau=k,t-1} \circ \beta_{|k|}^{t-1+k}\rangle(o^t),\beta_{|k|}^{t+k})\right]\\
&\geq \sum_{o^t} P_o(o^t) \max_{\beta_{|k|}^{t+k}} \left[\max_{\beta_{|k+1|}^{t+k+1}} F_{k+1}^{\tau=i,t}(\vec{\theta}^{t},\langle\langle q^{\tau=k,t-1} \circ \beta_{|k|}^{t-1+k}\rangle(o^t) \circ \beta_{|k|}^{t+k}\rangle,\beta_{|k+1|}^{t+k+1})\right]\\
&\geq \max_{\beta_{|k+1|}^{t+k}} \sum_{o^t} P_o(o^t) \left[\max_{\beta_{|k+1|}^{t+k+1}} F_{k+1}^{\tau=i,t}(\vec{\theta}^{t},\langle\langle q^{\tau=k,t-1} \circ \beta_{|k|}^{t-1+k}\rangle \circ \beta_{|k+1|}^{t+k}\rangle(o^t),\beta_{|k+1|}^{t+k+1})\right]\\
&= \max_{\beta_{|k+1|}^{t+k}} F_{k+1}^{\tau=i+1,t-1}(\vec{\theta}^{t-1},\langle q^{\tau=k,t-1} \circ \beta_{|k|}^{t-1+k}\rangle,\beta_{|k+1|}^{t+k})
\end{aligned}$$

proves the lemma. □

Lemma A.4. *If, for some stage t*

$$\forall_{\vec{\theta}^{t},q^{\tau=k,t},\beta_{|k|}^{t+k}} \quad F_k^{\tau=0,t}(\vec{\theta}^{t},q^{\tau=k,t},\beta_{|k|}^{t+k}) \geq \max_{\beta_{|k+1|}^{t+k+1}} F_{k+1}^{\tau=0,t}(\vec{\theta}^{t},\langle q^{\tau=k,t} \circ \beta_{|k|}^{t+k}\rangle,\beta_{|k+1|}^{t+k+1}) \tag{A.31}$$

holds, then $\forall_i \forall_{\vec{\theta}^{t-i},q^{\tau=k,t-i},\beta_{|k|}^{t-i+k}}$

$$F_k^{\tau=i,t-i}(\vec{\theta}^{t-i},q^{\tau=k,t-i},\beta_{|k|}^{t-i+k}) \geq \max_{\beta_{|k+1|}^{t-i+k+1}} F_{k+1}^{\tau=i,t-i}(\vec{\theta}^{t-i},\langle q^{\tau=k,t-i} \circ \beta_{|k|}^{t-i+k}\rangle,\beta_{|k+1|}^{t-i+k+1}). \tag{A.32}$$

Proof. If (A.31) holds for all $\vec{\theta}^{t}$, $q^{\tau=k,t}$, $\beta_{|k|}^{t+k}$, then eq. (A.29) is satisfied for all $\vec{\theta}^{t}$, $q^{\tau=k,t}$, $\beta_{|k|}^{t+k}$, and lemma (A.3) yields $\forall_{\vec{\theta}^{t-1},q^{\tau=k,t-1},\beta^{t-1+k}}$

$$F_k^{\tau=1,t-1}(\vec{\theta}^{t-1},q^{\tau=k,t-1},\beta_{|k|}^{t-1+k}) \geq \max_{\beta_{|k+1|}^{t+k}} F_{k+1}^{\tau=1,t-1}(\vec{\theta}^{t-1},\langle q^{\tau=k,t-1} \circ \beta^{t+k-1}\rangle,\beta_{|k+1|}^{t+k}). \tag{A.33}$$

At this point we can apply the lemma again, etc. The i-th application of the lemma yields (A.32). □

References

Aicardi, M., Davoli, F., & Minciardi, R. (1987). Decentralized optimal control of Markov chains with a common past information set. *IEEE Transactions on Automatic Control*, *32*(11), 1028–1031.

Altman, E. (2002). Applications of Markov decision processes in communication networks. In Feinberg, E. A., & Shwartz, A. (Eds.), *Handbook of Markov Decision Processes: Methods and Applications*. Kluwer Academic Publishers.

Amato, C., Bernstein, D. S., & Zilberstein, S. (2006). Optimal fixed-size controllers for decentralized POMDPs. In *Proc. of the AAMAS Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains (MSDM)*.

Amato, C., Bernstein, D. S., & Zilberstein, S. (2007a). Optimizing memory-bounded controllers for decentralized POMDPs. In *Proc. of Uncertainty in Artificial Intelligence*.

Amato, C., Carlin, A., & Zilberstein, S. (2007b). Bounded dynamic programming for decentralized POMDPs. In *Proc. of the AAMAS Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains (MSDM)*.

Arai, T., Pagello, E., & Parker, L. (2002). Editorial: Advances in multirobot systems. *IEEE Transactions on Robotics and Automation*, *18*(5), 655–661.

Aras, R., Dutech, A., & Charpillet, F. (2007). Mixed integer linear programming for exact finite-horizon planning in decentralized POMDPs. In *Proc. of the International Conference on Automated Planning and Scheduling*.

Becker, R., Lesser, V., & Zilberstein, S. (2005). Analyzing myopic approaches for multi-agent communication. In *Proc. of the International Conference on Intelligent Agent Technology*, pp. 550–557.

Becker, R., Zilberstein, S., & Lesser, V. (2004a). Decentralized Markov decision processes with event-driven interactions. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 302–309.

Becker, R., Zilberstein, S., Lesser, V., & Goldman, C. V. (2004b). Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research*, *22*, 423–455.

Bernstein, D. S. (2005). *Complexity Analysis and Optimal Algorithms for Decentralized Decision Making*. Ph.D. thesis, University of Massachusets Amherst.

Bernstein, D. S., Givan, R., Immerman, N., & Zilberstein, S. (2002). The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, *27*(4), 819–840.

Bernstein, D. S., Hansen, E. A., & Zilberstein, S. (2005). Bounded policy iteration for decentralized POMDPs. In *Proc. of the International Joint Conference on Artificial Intelligence*, pp. 1287–1292.

Bertsekas, D. P. (2005). *Dynamic Programming and Optimal Control* (3rd edition)., Vol. I. Athena Scientific.

Beynier, A., & Mouaddib, A.-I. (2005). A polynomial algorithm for decentralized Markov decision processes with temporal constraints. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 963–969.

Beynier, A., & Mouaddib, A.-I. (2006). An iterative algorithm for solving constrained decentralized Markov decision processes. In *Proc. of the National Conference on Artificial Intelligence*.

Binmore, K. (1992). *Fun and Games*. D.C. Heath and Company.

de Boer, P.-T., Kroese, D. P., Mannor, S., & Rubinstein, R. Y. (2005). A tutorial on the cross-entropy method. *Annals of Operations Research*, *134*(1), 19–67.

Boutilier, C. (1996). Planning, learning and coordination in multiagent decision processes. In *Proc. of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, pp. 195–210.

Boutilier, C., Dean, T., & Hanks, S. (1999). Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, *11*, 1–94.

Chades, I., Scherrer, B., & Charpillet, F. (2002). A heuristic approach for solving decentralized-POMDP: assessment on the pursuit problem. In *Proc. of the 2002 ACM Symposium on Applied Computing*, pp. 57–62.

Cogill, R., Rotkowitz, M., Roy, B. V., & Lall, S. (2004). An approximate dynamic programming approach to decentralized control of stochastic systems. In *Proc. of the 2004 Allerton Conference on Communication, Control, and Computing*.

Emery-Montemerlo, R. (2005). *Game-Theoretic Control for Robot Teams*. Ph.D. thesis, Carnegie Mellon University.

Emery-Montemerlo, R., Gordon, G., Schneider, J., & Thrun, S. (2004). Approximate solutions for partially observable stochastic games with common payoffs. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 136–143.

Emery-Montemerlo, R., Gordon, G., Schneider, J., & Thrun, S. (2005). Game theoretic control for robot teams. In *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 1175–1181.

Fernández, J. L., Sanz, R., Simmons, R. G., & Diéguez, A. R. (2006). Heuristic anytime approaches to stochastic decision processes. *Journal of Heuristics*, *12*(3), 181–209.

Gmytrasiewicz, P. J., & Doshi, P. (2005). A framework for sequential planning in multiagent settings. *Journal of Artificial Intelligence Research*, *24*, 49–79.

Goldman, C. V., Allen, M., & Zilberstein, S. (2007). Learning to communicate in a decentralized environment. *Autonomous Agents and Multi-Agent Systems*, *15*(1), 47–90.

Goldman, C. V., & Zilberstein, S. (2003). Optimizing information exchange in cooperative multi-agent systems. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 137–144.

Goldman, C. V., & Zilberstein, S. (2004). Decentralized control of cooperative systems: Categorization and complexity analysis.. *Journal of Artificial Intelligence Research*, *22*, 143–174.

Guestrin, C., Koller, D., Parr, R., & Venkataraman, S. (2003). Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, *19*, 399–468.

Hansen, E. A., Bernstein, D. S., & Zilberstein, S. (2004). Dynamic programming for partially observable stochastic games. In *Proc. of the National Conference on Artificial Intelligence*, pp. 709–715.

Hauskrecht, M. (2000). Value-function approximations for partially observable Markov decision processes.. *Journal of Artificial Intelligence Research*, *13*, 33–94.

Hsu, K., & Marcus, S. (1982). Decentralized control of finite state Markov processes. *IEEE Transactions on Automatic Control*, *27*(2), 426–431.

Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, *101*(1-2), 99–134.

Kim, Y., Nair, R., Varakantham, P., Tambe, M., & Yokoo, M. (2006). Exploiting locality of interaction in networked distributed POMDPs. In *Proc. of the AAAI Spring Symposium on Distributed Plan and Schedule Management*.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., & Osawa, E. (1997). RoboCup: The robot world cup initiative. In *Proc. of the International Conference on Autonomous Agents*.

Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjoh, A., & Shimada, S. (1999). Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. In *Proc. of the International Conference on Systems, Man and Cybernetics*, pp. 739–743.

Koller, D., Megiddo, N., & von Stengel, B. (1994). Fast algorithms for finding randomized strategies in game trees. In *Proc. of the 26th ACM Symposium on Theory of Computing*, pp. 750–759.

Koller, D., & Pfeffer, A. (1997). Representations and solutions for game-theoretic problems. *Artificial Intelligence*, *94*(1-2), 167–215.

Kuhn, H. (1953). Extensive games and the problem of information. *Annals of Mathematics Studies*, *28*, 193–216.

Lesser, V., Ortiz Jr., C. L., & Tambe, M. (Eds.). (2003). *Distributed Sensor Networks: A Multiagent Perspective*, Vol. 9. Kluwer Academic Publishers.

Littman, M., Cassandra, A., & Kaelbling, L. (1995). Learning policies for partially observable environments: Scaling up. In *Proc. of the International Conference on Machine Learning*, pp. 362–370.

Marecki, J., & Tambe, M. (2007). On opportunistic techniques for solving decentralized Markov decision processes with temporal constraints. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 1–8.

Nair, R., Tambe, M., & Marsella, S. (2003). Team formation for reformation in multiagent domains like RoboCupRescue. In *Proc. of RoboCup-2002 International Symposium*.

Nair, R., Roth, M., & Yohoo, M. (2004). Communication for improving policy computation in distributed POMDPs. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 1098–1105.

Nair, R., Tambe, M., & Marsella, S. (2002). Team formation for reformation. In *Proc. of the AAAI Spring Symposium on Intelligent Distributed and Embedded Systems*.

Nair, R., Tambe, M., & Marsella, S. (2003a). Role allocation and reallocation in multiagent teams: towards a practical analysis. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 552–559.

Nair, R., Tambe, M., Yokoo, M., Pynadath, D. V., & Marsella, S. (2003b). Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Proc. of the International Joint Conference on Artificial Intelligence*, pp. 705–711.

Nair, R., Varakantham, P., Tambe, M., & Yokoo, M. (2005). Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *Proc. of the National Conference on Artificial Intelligence*, pp. 133–139.

Nash, J. F. (1950). Equilibrium points in N-person games. *Proc. of the National Academy of Sciences of the United States of America*, *36*, 48–49.

Oliehoek, F., & Vlassis, N. (2006). Dec-POMDPs and extensive form games: equivalence of models and algorithms. Ias technical report IAS-UVA-06-02, University of Amsterdam, Intelligent Systems Lab, Amsterdam, The Netherlands.

Oliehoek, F. A., Kooij, J. F., & Vlassis, N. (2007a). A cross-entropy approach to solving Dec-POMDPs. In *Proc. of the International Symposium on Intelligent and Distributed Computing*, pp. 145–154.

Oliehoek, F. A., Spaan, M. T. J., & Vlassis, N. (2007b). Dec-POMDPs with delayed communication. In *Proc. of the AAMAS Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains (MSDM)*.

Oliehoek, F. A., Spaan, M. T. J., Whiteson, S., & Vlassis, N. (2008). Exploiting locality of interaction in factored Dec-POMDPs. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*.

Oliehoek, F. A., & Visser, A. (2006). A hierarchical model for decentralized fighting of large scale urban fires. In *Proc. of the AAMAS'06 Workshop on Hierarchical Autonomous Agents and Multi-Agent Systems (H-AAMAS)*, pp. 14–21.

Oliehoek, F. A., & Vlassis, N. (2007). Q-value functions for decentralized POMDPs. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 833–840.

Ooi, J. M., & Wornell, G. W. (1996). Decentralized control of a multiple access broadcast channel: Performance bounds. In *Proc. of the 35th Conference on Decision and Control*, pp. 293–298.

Osborne, M. J., & Rubinstein, A. (1994). *A Course in Game Theory*. The MIT Press.

Papadimitriou, C. H., & Tsitsiklis, J. N. (1987). The complexity of Markov decision processes. *Mathematics of Operations Research*, *12*(3), 441–451.

Paquet, S., Tobin, L., & Chaib-draa, B. (2005). An online POMDP algorithm for complex multiagent environments. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*.

Peshkin, L. (2001). *Reinforcement Learning by Policy Search*. Ph.D. thesis, Brown University.

Peshkin, L., Kim, K.-E., Meuleau, N., & Kaelbling, L. P. (2000). Learning to cooperate via policy search. In *Proc. of Uncertainty in Artificial Intelligence*, pp. 307–314.

Pineau, J., Gordon, G., & Thrun, S. (2003). Point-based value iteration: An anytime algorithm for POMDPs. In *Proc. of the International Joint Conference on Artificial Intelligence*, pp. 1025–1032.

Puterman, M. L. (1994). *Markov Decision Processes—Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.

Pynadath, D. V., & Tambe, M. (2002). The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, *16*, 389–423.

Romanovskii, I. (1962). Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, *3*, 678–681.

Roth, M., Simmons, R., & Veloso, M. (2005). Reasoning about joint beliefs for execution-time communication decisions. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 786–793.

Roth, M., Simmons, R., & Veloso, M. (2007). Exploiting factored representations for decentralized execution in multi-agent teams. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 467–463.

Russell, S., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (2nd edition). Pearson Education.

Schoute, F. C. (1978). Symmetric team problems and multi access wire communication. *Automatica*, *14*, 255–269.

Seuken, S., & Zilberstein, S. (2007a). Improved memory-bounded dynamic programming for decentralized POMDPs. In *Proc. of Uncertainty in Artificial Intelligence*.

Seuken, S., & Zilberstein, S. (2007b). Memory-bounded dynamic programming for DEC-POMDPs.. In *Proc. of the International Joint Conference on Artificial Intelligence*, pp. 2009–2015.

Sondik, E. J. (1971). *The optimal control of partially observable Markov decision processes*. Ph.D. thesis, Stanford University.

Spaan, M. T. J., Gordon, G. J., & Vlassis, N. (2006). Decentralized planning under uncertainty for teams of communicating agents. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 249–256.

Spaan, M. T. J., & Melo, F. S. (2008). Interaction-driven Markov games for decentralized multiagent planning under uncertainty. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*.

Spaan, M. T. J., & Vlassis, N. (2005). Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, *24*, 195–220.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. The MIT Press.

Szer, D., & Charpillet, F. (2005). An optimal best-first search algorithm for solving infinite horizon DEC-POMDPs. In *Proc. of the European Conference on Machine Learning*, pp. 389–399.

Szer, D., & Charpillet, F. (2006). Point-based dynamic programming for DEC-POMDPs.. In *Proc. of the National Conference on Artificial Intelligence*.

Szer, D., Charpillet, F., & Zilberstein, S. (2005). MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *Proc. of Uncertainty in Artificial Intelligence*, pp. 576–583.

Tao, N., Baxter, J., & Weaver, L. (2001). A multi-agent policy-gradient approach to network routing. In *Proc. of the International Conference on Machine Learning*, pp. 553–560.

Varakantham, P., Marecki, J., Yabu, Y., Tambe, M., & Yokoo, M. (2007). Letting loose a SPIDER on a network of POMDPs: Generating quality guaranteed policies. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*.

Varakantham, P., Nair, R., Tambe, M., & Yokoo, M. (2006). Winning back the cup for distributed POMDPs: planning over continuous belief spaces. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 289–296.

Wu, J., & Durfee, E. H. (2006). Mixed-integer linear programming for transition-independent decentralized MDPs. In *Proc. of the International Joint Conference on Autonomous Agents and Multi Agent Systems*, pp. 1058–1060.

Xuan, P., Lesser, V., & Zilberstein, S. (2001). Communication decisions in multi-agent cooperation: Model and experiments. In *Proc. of the International Conference on Autonomous Agents*.