Local Multiagent Coordination in Decentralized MDPs with Sparse Interactions

Francisco S. Melo and Manuela Veloso

July 7, 2010
CMU-CS-10-133

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

This research was partially sponsored by the Portuguese Fundação para a Ciência e a Tecnologia under the Carnegie Mellon-Portugal Program and the Information and Communications Technologies Institute (ICTI), www.icti.cmu.edu. The views and conclusions contained in this document are those of the author only.

Keywords: decentralized Markov decision process, sparse interaction, multiagent planning

Abstract

Creating coordinated multiagent policies in environments with uncertainty is a challenging problem, which can be greatly simplified if the coordination needs are known to be limited to specific parts of the state-space. In this work, we explore how such local interactions can simplify coordination in multiagent systems. We focus on problems in which the interaction between the agents is sparse, exploiting this property to minimize the coupling of the decision processes for the different agents. We contribute a new decision-theoretic model for multiagent systems, Dec-SIMDPs, that explicitly distinguishes the situations in which the agents in the team must coordinate from those in which they can act independently. We relate our new model to other existing models from the literature, such as MMDPs and Dec-MDPs. We then propose a solution method that takes advantage of the particular structure of Dec-SIMDPs and provide theoretical error bounds on the quality of the obtained solution. Finally, we illustrate the performance of our method in several simulated navigation problems.

1 Introduction

Decision-theoretic models such as Dec-MDPs and Dec-POMDPs provide a rich framework to tackle decentralized decision-making problems. However, using these models to create coordinated multiagent policies in environments with uncertainty is a challenging problem, even more so if the decision-makers must tackle issues of partial observability. As such, solving Dec-POMDPs is a NEXP-complete problem and thus computationally too demanding to solve except for the simplest scenarios.

Recent years have witnessed a profusion of work on Dec-(PO)MDP-related models that aim at capturing some of the fundamental features of this class of problems such as partial observability without incurring in the associated computational cost. In this paper, we contribute to this area of research, and propose a new model for cooperative multiagent decision-making in the presence of partial observability. Our model is motivated by the observation that, in many real-world scenarios, the tasks of the different agents in a multiagent system are not coupled at every decision-step but only in relatively infrequent situations. We dub such problems as having *sparse interaction*.

Multi-robot systems provide our primary motivation and constitute natural examples for the class of problems considered herein. In multi-robot systems, the interaction among the different robots is naturally limited by each robot's physical boundaries, such as workspace or communication range, and limited perception capabilities. Therefore, when programming a multi-robot system to perform some task, one natural approach is to subdivide this task into smaller tasks that each robot can then execute autonomously or as part of a smaller group (see, for example, Fig. 1).

Other examples include problems of sequential resource allocation, in which groups of agents must interact only to the extent that they need to share some common resource. In this context, several methods have been proposed that leverage sparse interactions by decomposing the "global" problem into several smaller "local" problems that can be solved more efficiently, and then combining the obtained solutions [23, 28]. Such approaches, however, are not particularly concerned with partial observability issues.

Several previous works have exploited simplified models of interaction in multiagent settings. For example, learning tasks involving multiple agents can be partitioned in a state-wise manner, allowing different agents to independently learn the resulting "smaller tasks" [31]. Similarly, a hierarchical learning algorithm can be used that considers only interaction between the different agents at a higher control level, while allowing the agents to learn lower level tasks independently [10]. Other works use coordination graphs to compactly represent dependences between the actions of different agents, thus capturing the local interaction between them [14, 17]. Local interactions have also been exploited to minimize communication during policy execution [25] and in the game-theoretic literature to attain compact game representations. Examples include graphical games [15] and action-graph games [35].

In this paper we consider Dec-MDPs with sparse interactions, henceforth Dec-SIMDPs. Dec-SIMDPs leverage the independence between agents to decouple the decision process in significant portions of the joint state-space. On those situations in which the agents interact – the *interaction areas*, – Dec-SIMDPs rely on communication to bring down the computational complexity of the joint decision process. Dec-SIMDPs "balance" the independence assumptions with observability: in any given state, the agents are either independent or can share state information (*e.g.*, by com-

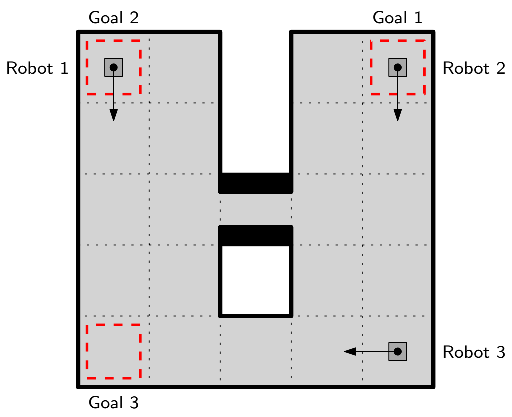


Figure 1: Example of a simple navigation task. Goals are marked with red, dashed lines. While Robot 3 can navigate to its goal, disregarding the remaining robots, Robots 1 and 2 need to coordinate so as not to cross the narrow doorway simultaneously. However, this coordination needs only occur around the doorway.

municating).[1] A related model has recently been proposed under the designation of distributed POMDPs with coordination locales [32].

The contributions of this paper are two-fold. On one hand we provide a precise formalization of the Dec-SIMDP model and discuss in some detail the relation with well-established decision-theoretic models such as Dec-MDPs, MMDPs and MDPs. On the other hand, we contribute two new algorithms that exhibit significant computational savings when compared to existing algorithms for Dec-SIMDPs, and illustrate their application in several simple navigation tasks.

2 Decision Theoretic Models for Multiagent Systems

We now review several standard decision theoretic models, pinpointing the main differences between these. We start with single agent models, namely Markov decision problems (MDPs) and their partially observable counterparts (POMDPs) before moving to multiagent models such as multiagent MDPs (MMDPs) and their partially observable counterparts (Dec-MDPs). The purpose of this introductory section is to establish the notation used throughout the paper and review some fundamental concepts and results that will play a fundamental role in the development to come.

2.1 Markov Decision Processes

A *Markov decision problem* (MDP) describes a sequential decision problem in which a single agent must choose the sequence of actions that maximizes some reward-based optimization criterion. Formally, an MDP is a tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, where $\mathcal{X}$ represents the finite state-space, $\mathcal{A}$ represents the finite action-space, $\mathsf{P}(x, a, y)$ represents the transition probability from state x to

[1]Both independence assumptions and communication can significantly bring down the computational complexity in Dec-(PO)MDP related models [3, 12].

state y when action a is taken and $r(x, a)$ represents the expected reward for taking action a in state x. The scalar γ is a discount factor.

A Markov *policy* is a mapping $\pi : \mathcal{X} \times \mathcal{A} \longrightarrow [0, 1]$ such that, for all $x \in \mathcal{X}$,

$$\sum_{a \in \mathcal{A}} \pi(x, a) = 1.$$

The purpose of the agent is to determine a policy π so as to maximize, for all $x \in \mathcal{X}$,

$$V^\pi(x) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r\big(X(t), A(t)\big) \mid X(0) = x \right],$$

where $X(t)$ denotes the state at time t, $A(t)$ denotes the action taken at that time instant such that

$$\mathbb{P}\left[A(t) = a \mid H(t) = h\right] = \mathbb{P}\left[A(t) = a \mid X(t) = x\right] = \pi(x, a),$$

where $H(t) = \{X(0), A(0), \ldots, X(t-1), A(t-1), X(t)\}$ is the random variable corresponding to the history of the process up to time t, and h denotes a particular realization of $H(t)$ such that $X(t) = x$. We define the Q-function associated with a policy π as

$$Q^\pi(x, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r\big(X(t), A(t)\big) \mid X(0) = x, A(0) = a \right].$$

where, again, $A(t) \sim \pi$ for all $t > 0$.

For any finite MDP, there is at least one *optimal policy* π^* such that

$$V^{\pi^*}(x) \geq V^\pi(x)$$

for any π and every $x \in \mathcal{X}$. The corresponding value function is denoted by V^* and verifies the Bellman optimality equation,

$$V^*(x) = \max_{a \in \mathcal{A}} \left[r(x, a) + \gamma \sum_{y \in \mathcal{X}} \mathsf{P}(x, a, y) V^*(y) \right]. \tag{1}$$

The associated Q-function in turn verifies

$$Q^*(x, a) = r(x, a) + \gamma \sum_{y \in \mathcal{X}} \mathsf{P}(x, a, y) \max_{u \in \mathcal{A}} Q^*(y, u). \tag{2}$$

The optimal policy can be recovered directly from Q^* by setting $\pi^*(x, a) > 0$ only if $a \in \arg\max_u Q^*(x, u)$. As such, the solution for any given MDP can be obtained by computing the corresponding optimal Q-function, Q^*.

For future reference, given a functions q defined over $\mathcal{X} \times \mathcal{A}$, we define the *Bellman operator* $\mathbf{H}$ as

$$(\mathbf{H}q)(x, a) = r(x, a) + \gamma \sum_{y \in \mathcal{X}} \mathsf{P}(x, a, y) \max_{u \in \mathcal{A}} q(y, u). \tag{3}$$

The function Q^* in (2) is the fixed-point of $\mathbf{H}$ and thus can be computed, *e.g.*, by iteratively applying $\mathbf{H}$ to some initial estimate $Q^{(0)}$, a dynamic programming (DP) method known as *value iteration*.

2.2 Partially Observable Markov Decision Processes

Partially observable MDPs describe problems essentially similar to MDPs, in which an agent must choose a sequence of actions to maximize a reward-based criterion. However, unlike MDPs, in a POMDP the agent has only access to the underlying state of the process, $X(t)$, by means of indirect observations. Formally, a POMDP is a tuple $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$, where $\mathcal{X}$ is the finite state-space, $\mathcal{A}$ is the finite action-space and $\mathcal{Z}$ is the finite observation space. As before, $\mathsf{P}(x, a, y)$ represents the transition probability from state x to state y when action a is taken, and now $\mathsf{O}(x, a, z)$ represents the probability of observation z given that the state is x and action a was taken, *i.e.*,

$$\mathsf{O}(x, a, z) = \mathbb{P}\left[Z(t+1) = z \mid X(t+1) = x, A(t) = a\right].$$

Finally, $r(x, a)$ again represents the expected reward for taking action a in state x and γ is a discount factor.

A non-Markov policy is a mapping $\pi : \mathcal{H} \longrightarrow [0, 1]$ such that, for all $h \in \mathcal{H}$,

$$\sum_{a \in \mathcal{A}} \pi(h, a) = 1,$$

where $h = \{a(0), z(1), a(1), \ldots, a(t-1), z(t)\}$ is a *finite history*, *i.e.*, a finite sequence of action-observation pairs. As before, the history observed up to time t is a random variable, denoted as $H(t)$, and denote by $\mathcal{H}$ the set of all possible finite histories for the POMDP. The purpose of the agent is now to determine a policy π so as to maximize, for all probability vectors $\mathbf{b}$,

$$V^\pi(\mathbf{b}) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r\big(X(t), A(t)\big) \mid X(0) \sim \mathbf{b}\right],$$

where $X(0) \sim \mathbf{b}$ denotes the fact that $X(0)$ is distributed according to $\mathbf{b}$. Similarly, we define the Q-function associated with a policy π as

$$Q^\pi(\mathbf{b}, a) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r\big(X(t), A(t)\big) \mid X(0) \sim \mathbf{b}, A(0) = a\right].$$

We refer to the probability vector $\mathbf{b}$ in the expressions above as the *initial belief state* or just the *initial belief*. It translates the "belief" that the agent has at time $t = 0$ regarding its state at that time. From the initial belief and given the history up to time t, $H(t)$, we can construct a sequence $\{\mathbf{b}(t)\}$ of probability vectors recursively as

$$\begin{aligned}\mathbf{b}_y(t+1) &= \mathsf{Bel}(b(t), A(t), Z(t+1)) \\ &\triangleq \eta \sum_x \mathbf{b}_x(t)\mathsf{P}(x, A(t), y)\mathsf{O}(y, A(t), Z(t+1)),\end{aligned}$$

where Bel is the belief update operator, $\mathbf{b}_x(t)$ denotes the x component of $\mathbf{b}(t)$ and η is a normalization factor. We generally refer to the vector $\mathbf{b}(t)$ as the *belief* at time t. It corresponds to a distribution over the unknown state at time t such that

$$\mathbf{b}_x(t) = \mathbb{P}\left[X(t) = x \mid H(t)\right].$$

Given the POMDP model parameters P and O, every finite history $h \in \mathcal{H}$ can be mapped to a belief $\mathbf{b}$. Moreover, for any given policy, two histories leading to the same belief will have the same value. As such, beliefs can be used as compact representations of histories, and we can define policies in terms of beliefs instead of histories [29]. In fact, it is possible to reinterpret a POMDP as an infinite MDP in which the state-space corresponds to the set of all possible beliefs. Therefore, for any finite POMDP, there is at least one *optimal policy* π^* such that

$$V^{\pi^*}(\mathbf{b}) \geq V^{\pi}(\mathbf{b})$$

for any π and every initial belief $\mathbf{b}$. The corresponding value function is denoted by V^* and also verifies the Bellman optimality equation,

$$V^*(\mathbf{b}) = \max_{a \in \mathcal{A}} \sum_x \mathbf{b}_x \left[r(x,a) + \gamma \sum_{z,y} \mathsf{P}(x,a,y) \mathsf{O}(y,a,z) V^*(\mathsf{Bel}(\mathbf{b},a,z)) \right],$$

where x, y take values in $\mathcal{X}$, z takes values in $\mathcal{Z}$ and $\mathsf{Bel}(\mathbf{b}, a, z)$ corresponds to the updated belief after taking action a and observing z. The associated Q-function in turn verifies

$$Q^*(\mathbf{b},a) = \sum_x \mathbf{b}_x \left[r(x,a) + \gamma \sum_{z,y} \mathsf{P}(x,a,y) \mathsf{O}(y,a,z) \max_{u \in \mathcal{A}} Q^*(\mathsf{Bel}(\mathbf{b},a,z),u) \right].$$

In spite of its representative power, POMDPs have been shown to be undecidable in the worst case for infinite horizon settings such as those considered herein [19]. As such, exact solutions can be computed only in very specific instances, and most approaches in the literature resort to approximate or heuristic methods. Good surveys on POMDP solution methods can be found, *e.g.*, in [1, 9].

We describe an MDP-based heuristic solution that will prove of later use in the paper. This method is known as Q_{MDP} as it makes use of the optimal Q-function for the underlying MDP as an estimate for the optimal Q-function for the POMDP. Since the optimal solution for the underlying MDP can be computed in a straightforward manner, this method is very simple and fast to implement and attains good performance in many practical situations [9, 18].

Let $\mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$ be a POMDP with finite state, action and observation spaces. Associated with this POMDP there is an underlying MDP $\bar{\mathcal{M}} = (\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$. Let $\bar{Q}^*$ be the optimal Q-function for $\bar{\mathcal{M}}$. Q_{MDP} uses an estimate for the optimal Q-function for the POMDP given by

$$\hat{Q}(\mathbf{b},a) = \sum_x \mathbf{b}_x \bar{Q}^*(x,a).$$

When using Q_{MDP}, the agent acts under the implicit assumption that state uncertainty only affects the immediate decision, *i.e.*, after one decision the agent will act on the underlying MDP. This sometimes leads to poor performance and several works have proposed further improvements on Q_{MDP} to address this issue [9, 21], but we do not pursue such discussion here.

2.3 Multiagent MDPs

Multiagent Markov decision processes (MMDPs) generalize MDPs to multiagent cooperative scenarios. MMDPs describe sequential decision tasks in which multiple agents must each choose a sequence of individual actions that jointly maximize some common reward-based optimization criterion. Formally, an MMDP is a tuple $\mathcal{M} = (N, \mathcal{X}, (\mathcal{A}_k), \mathsf{P}, r, \gamma)$, where N is the number of agents, $\mathcal{X}$ represents the finite state-space, $\mathcal{A}_k$ is the finite *individual* action space of agent k. As in MDPs, $\mathsf{P}(x, a, y)$ represents the transition probability from state x to state y when the *joint action* $a = (a_1, \ldots, a_N)$ is taken; $r(x, a)$ represents the expected reward received by *all* agents for taking the joint action a in state x. In an MMDP all agents receive the same reward, which implies that MMDPs represent *fully cooperative* multiagent tasks.

As noted above, a joint action a is a tuple $a = (a_1, \ldots, a_N)$ and we denote by $\mathcal{A} = \times_{k=1}^{N} \mathcal{A}_k$ the set of all possible joint actions – the joint action space. For $k = 1, \ldots, N$, let

$$\mathcal{A}_{-k} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_{k-1} \times \mathcal{A}_{k+1} \times \ldots \times \mathcal{A}_N.$$

We write a_{-k} to denote a general element of $\mathcal{A}_{-k}, k = 1, \ldots, N$ and refer to any such action as a *reduced joint action* or simply a reduced action. We write $a = (a_{-k}, a_k)$ to denote the fact that the joint action a is composed by the reduced action a_{-k} and the individual action a_k for agent k.

In this work, we also assume that the state-space $\mathcal{X}$ can be factored as $\mathcal{X} = \mathcal{X}_0 \times \mathcal{X}_1 \times \ldots \times \mathcal{X}_N$. As such, each element $x \in \mathcal{X}$ is a tuple $x = (x_0, \ldots, x_N)$, with $x_k \in \mathcal{X}_k, k = 0, \ldots, N$. For any $x \in \mathcal{X}$, we refer to the pair (x_0, x_k) as the local state of agent k, and generally denote it as $\bar{x}_k$. Note that this factorization implies no loss of generality, as any set can be factorized as indicated above by considering singleton sets $\mathcal{X}_1, \ldots, \mathcal{X}_N$.[2]

An *individual* Markov policy for agent k is a mapping $\pi_k : \mathcal{X} \times \mathcal{A}_k \longrightarrow [0, 1]$ such that, for all $x \in \mathcal{X}$,

$$\sum_{a_k \in \mathcal{A}_k} \pi_k(x, a_k) = 1.$$

Similarly, a *joint* policy is a mapping $\pi : \mathcal{X} \times \mathcal{A} \longrightarrow [0, 1]$ that we take as the combination of N individual policies, *i.e.*,

$$\pi(x, a) = \prod_{k=1}^{N} \pi_k(x, a_k),$$

where $a = (a_1, \ldots, a_N)$. As we do with actions, we write π_{-k} to denote a *reduced policy* and $\pi = (\pi_{-k}, \pi_k)$ to denote the fact that the joint policy π is composed by the reduced policy π_{-k} and the individual policy π_k for agent k.

In an MMDP, the purpose of *all* agents is to determine a joint policy π so as to maximize, for all $x \in \mathcal{X}$,

$$V^{\pi}(x) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r\big(X(t), A(t)\big) \mid X(0) = x\right],$$

[2]Not all multiagent problems can effectively leverage this factorization of the state-space to simplify the decision process. However, there are problems in which this factorization can significantly simplify the decision process and bring significant computational savings.

where $X(t)$ denotes the state at time t and $A(t)$ denotes the joint action taken at that time instant. The Q-function associated with a joint policy π is defined from V^π as its single-agent counterpart.

We conclude by noting that, for the purposes of planning, *i.e.*, computing the optimal policy, an MMDP is indistinguishable from an ordinary MDP. It is only at *execution time* that an MMDP differs from an MDP, since the process of decision making is not centralized, but distributed. This poses severe difficulties when we move to partially observable settings.

2.4 Dec-MDPs

Decentralized MDPs (Dec-MDPs) are partially observable generalizations of MMDPs. As in MMDPs, the agents in a Dec-MDP must each choose a sequence of individual actions that jointly maximize some common reward-based optimization criterion. Unlike MMDPs, however, the agents can only access the global state of the process by means of local indirect observations. Formally, a Dec-MDP can be represented a tuple

$$\mathcal{M} = (N, (\mathcal{X}_k), (\mathcal{A}_k), (\mathcal{Z}_k), \mathsf{P}, (\mathsf{O}_k), r, \gamma),$$

where N is the number of agents, $\mathcal{X} = \times_{k=0}^{N} \mathcal{X}_k$ is the joint state-space, $\mathcal{A} = \times_{k=1}^{N} \mathcal{A}_k$ is the set of joint actions, each $\mathcal{Z}_k$ represents the set of possible local observation for agent k, $\mathsf{P}(x, a, y)$ represents the transition probabilities from joint state x to joint state y when the joint action a is taken, each $\mathsf{O}_k(x, a, z_k)$ represents the probability of agent k making the local observation z_k when the joint state is x and the last action taken was a, and $r(x, a)$ represents the expected reward received by all agents for taking the joint action a in joint state x. The scalar γ is a discount factor. In a Dec-MDP, for every joint observation $z \in \mathcal{Z}$, there is a state $x \in \mathcal{X}$ such that

$$\mathbb{P}\left[X(t) = x \mid Z(t) = z\right] = 1.$$

This means that, in a Dec-MDP, the agents have *joint full observability*: if all agents share their observations, they can recover the state of the Dec-MDP unambiguously.

Throughout this work, we consider only Dec-MDPs with *local full observability*, meaning that each agent can infer from its local observations the corresponding local state unambiguously. Formally this can be translated into the following condition: for every local observation $z_k \in \mathcal{Z}_k$ there is a local state $\bar{x}_k \in \mathcal{X}_0 \times \mathcal{X}_k$ such that

$$\mathbb{P}\left[\bar{X}_k(t) = \bar{x}_k \mid Z_k(t) = z_k\right] = 1.$$

Although more general Dec-MDP models are possible, we adhere to this simplified version, as this is sufficient for our purposes and makes the presentation both clearer and simpler. We refer to [6] for a more general formulation.

For future reference, define the set

$$\mathcal{X}_{-k} = \mathcal{X}_0 \times \ldots \times \mathcal{X}_{k-1} \times \mathcal{X}_{k+1} \times \ldots \times \mathcal{X}_N.$$

and denote by x_{-k} a general element of $\mathcal{X}_{-k}$. As with actions, we write $x = (x_{-k}, x_k)$ to denote the fact that the kth component of x takes the value x_k.

In this partially observable multiagent setting, an individual non-Markov policy for agent k is a mapping $\pi_k : \mathcal{H}_k \longrightarrow [0, 1]$ such that, for all $h_k \in \mathcal{H}_k$,

$$\sum_{a_k \in \mathcal{A}_k} \pi(h_k, a_k) = 1,$$

where $h_k = \{a_k(0), z_k(1), \ldots, a_k(t-1), z_k(t)\}$ is an *individual history* for agent k, *i.e.*, a sequence of individual action-observation pairs. $\mathcal{H}_k$ is the set of all possible finite histories for agent k and we denote by $H_k(t)$ the random variable that represents the history of agent k at time t.

Like in POMDPs, in a Dec-MDP each agent has only partial perception of the global state. Therefore, from the agent's perspective, its local state is *non-Markovian* – the current local state and action are not sufficient to uniquely determine its next local state. It is also noteworthy that, in the general multiagent setting, there is no compact representation of histories that plays the role of beliefs in POMDPs. This implies, in particular, that the passage from MMDP to its partially observable counterpart is fundamentally different from the same passage in single-agent scenarios.[3] However, if communication between the agents is instantaneous, free and error-free, then a Dec-MDP reduces to a MMDP, and partial observability is no longer an issue.

In a Dec-MDP, the purpose of all agents is to determine a joint policy π so as to maximize the total sum of discounted rewards. In order to write this in terms of a function, we consider a distinguished initial state, $x^0 \in \mathcal{X}$, that is assumed common knowledge among all agents.[4] The purpose of the agents is then to maximize

$$V^\pi = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r\big(X(t), A(t)\big) \mid X(0) = x^0 \right].$$

Transition-independent Dec-MDPs constitute a particular subclass of Dec-MDPs in which, for all $(x, a) \in \mathcal{X} \times \mathcal{A}$,

$$\mathbb{P}\left[X_0(t+1) = y_0 \mid X(t) = x, A(t) = a\right] = \mathbb{P}\left[X_0(t+1) = y_0 \mid X_0(t) = x_0\right] \tag{4a}$$

$$\mathbb{P}\left[X_k(t+1) = y_k \mid X(t) = x, A(t) = a\right] = \mathbb{P}\left[X_k(t+1) = y_k \mid \bar{X}_k(t) = \bar{x}_k, A_k(t) = a_k\right]. \tag{4b}$$

The transition probabilities can thus be factorized as

$$\mathsf{P}(x, a, y) = \mathsf{P}_0(x_0, y_0) \prod_{k=1}^{N} \mathsf{P}_k(\bar{x}_k, a_k, y_k), \tag{5}$$

where

$$\mathsf{P}_0(x_0, y_0) = \mathbb{P}\left[X_0(t+1) = y_0 \mid X_0(t) = x_0\right]$$
$$\mathsf{P}_k(\bar{x}_k, a_k, y_k) = \mathbb{P}\left[X_k(t+1) = y_k \mid \bar{X}_k(t) = \bar{x}_k, A_k(t) = a_k\right].$$

[3]This fact can also be observed by considering the worst-case computational complexity of each of the different models. In finite horizon settings, POMDPs are PSPACE-complete, versus the P-completeness of fully observable MDPs [24]. In multiagent settings, however, Dec-MDPs are NEXP-complete [6] even in the "benign" 2-agent case, versus the P-completeness of MMDPs.

[4]In fact, the Dec-MDP definition should explicitly include the initial state x^0. However, in order to avoid cluttering the notation, we omit the explicit reference to this initial state in the Dec-MDP tuple, with the understanding that one such state is implicit.

This particular class of Dec-MDPs was introduced in [5] and seeks to exploit a particular form of independence to somehow bring down the computational complexity required to solve such models. In this class of problems, the local state of each agent constitutes a sufficient statistic for its history, and the optimal policy for each agent can thus be computed in terms of this individual state [12]. This particular class of Dec-MDPs has been shown to be NP-complete in finite-horizon settings, versus the NEXP-completeness of general Dec-MDPs [12].

Similarly, *reward independent Dec-MDPs* correspond to a subclass of Dec-MDPs in which, for all x, a,

$$r(x, a) = f(r_k(\bar{x}_k, a_k), k = 1, \ldots, N), \tag{6}$$

i.e., the global reward function r can be obtained from local reward functions $r_k, k = 1, \ldots, N$. To ensure consistency of the decision process, we also require that

$$f(r_{-k}(x_{-k}, a_{-k}), r_k(\bar{x}_k, a_k)) \geq f(r_{-k}(x_{-k}, a_{-k}), r_k(\bar{x}_k, u_k))$$

if and only if

$$r_k(\bar{x}_k, a_k) \geq r_k(\bar{x}_k, u_k).$$

One typical example is

$$r(x, a) = \sum_{k=1}^{N} r_k(\bar{x}_k, a_k), \tag{7}$$

where $x = (x_0, \ldots, x_N)$ and $a = (a_1, \ldots, a_N)$. Interestingly, it was recently shown [2, 3] that reward independent Dec-MDPs retain NEXP-complete complexity. However, when associated with transition independence, reward independence implies that a Dec-MDP can be decomposed into N independent MDPs, each of which can be solved separately. The complexity of this class of problems thus reduces to that of standard MDPs (P-complete).

3 Decentralized Sparse-Interaction MDPs (Dec-SIMDPs)

We now depart from the transition-independent Dec-MDP and introduce a new model for multi-agent decision problems that is, at the same time, more general and more specific than transition independent Dec-MDPs. In the previous section we discussed how different degrees of independence between the agents in a Dec-MDP translate in terms of reduced worst-case complexity. This discussion is summarized in the diagram in Fig. 2.[5]

The goal of this paper is to exploit sparse interactions among the different agents in a Dec-MDP. In particular, we are interested in Dec-MDPs in which there is some level of both transition and reward dependency, but this dependency is limited to specific regions of the state-space. Therefore, in the diagram of Fig. 2, our model would correspond to the open blue circle.

[5]In the diagram we have included a "complexity gap" between the P and NP classes. Formally, that such a gap actually exists must yet be proved. However, for illustration purposes, we have included it in the diagram, translating the common belief that P$\neq$NP.

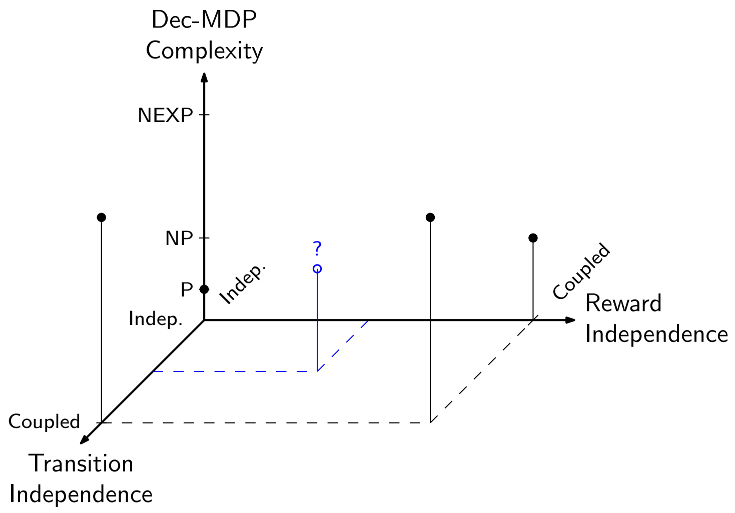


Figure 2: Currently known complexity results for different sub-classes of Dec-MDPs (see discussion in the main text). We refer to [2] for formal proofs of most such results.

We start with a couple of definitions that will be used to refine the notions of transition and reward independence described in Section 2.4. Let $K = \{k_1, \ldots, k_m\}$ be a subset of agents in a Dec-MDP. We denote by

$$\mathcal{X}_K = \mathcal{X}_0 \times \mathcal{X}_{k_1} \times \ldots \times \mathcal{X}_{k_m}$$

the joint state-space of all agents in K. Extending the notation introduced in Section 2, we write $\mathcal{X}_{-K}$ to denote the joint state-space of the agents *not* in K. We write x_K to denote a general element of $\mathcal{X}_K$ and x_{-K} to denote a general element of $\mathcal{X}_{-K}$. We write $x = (x_{-K}, x_K)$ to distinguish the components of x corresponding to agents in K and those corresponding to agents not in K.

Also we decompose the reward function for a Dec-MDP as

$$r(x, a) = \sum_{k=1}^{N} r_k(\bar{x}_k, a_k) + \sum_{i=1}^{M} r_i^I(x_{K_i}, a_{K_i}), \tag{8}$$

where each r_k corresponds to an individual component of the reward function that depends only on agent k and there are M sets, $K_i, i = 1, \ldots, M$, and M reward components, r_i^I (the interaction components), each depending on all the agents in K_i and only on these.

The decomposition in (8) can be performed at no loss of generality, since any reward r can be trivially written in that form by setting $M = 1$, $r_k \equiv 0$, $K_1 = \{1, \ldots, N\}$, and $r_1^I = r$. The scenarios that we are interested in, however, are those in which the support of $\sum_{i=1}^{M} r_i^I$ – the subset of $\mathcal{X} \times \mathcal{A}$ in which this sum is non-zero – is small when compared with $\mathcal{X} \times \mathcal{A}$.

Definition 3.1. *In a Dec-MDP $\mathcal{M} = (N, (\mathcal{X}_k), (\mathcal{A}_k), (\mathcal{Z}_k), \mathsf{P}, (\mathsf{O}_k), r, \gamma)$, an agent k_0 is* independent *of agent k_1 in a state $x \in \mathcal{X}$ if the following conditions both hold at state x:*

- *The transition probabilities for the individual state of agent k_0 at x do not depend on the* *state/action of agent k_1, i.e.,*

$$\begin{aligned}&\mathbb{P}\left[X_{k_0}(t+1) = y_{k_0} \mid X(t) = x, A(t) = a\right]\\&= \mathbb{P}\left[X_{k_0}(t+1) = y_{k_0} \mid X_{-k_1}(t) = x_{-k_1}, A_{-k_1}(t) = a_{-k_1}\right].\end{aligned}$$

- *It is possible to decompose the global reward function $r(x, a)$ as in* (8) *in such a way that no agent set K_i contains both k_0 and k_1.*

When any of the above conditions does not hold, agent k_0 is said to depend *on agent k_1 in x. Similarly, agent k_0 is independent of a set of agents $K = \{k_1, \ldots, k_m\}$ at state x if the above conditions hold simultaneously for all $k \in K$ in state x, and dependent otherwise.*

Roughly speaking, an agent k_0 depends on another agent k_1 in a particular state if either the reward function or the transition probabilities or both can not be decomposed so as to decouple the influence of each agent's individual state and action. Note however that the dependence relation is not symmetrical: even if agent k_0 depends on agent k_1, the reverse need not hold. In fact, it is possible that the two agents are reward independent and that the transition probabilities from state x_{k_0} depend on the individual state/action of agent k_1 without the reverse holding.

Definition 3.2. *In a Dec-MDP $\mathcal{M} = (N, (\mathcal{X}_k), (\mathcal{A}_k), (\mathcal{Z}_k), \mathsf{P}, (\mathsf{O}_k), r, \gamma)$ a set of agents K* interact *at state $x \in \mathcal{X}$ if the following conditions simultaneously hold*

- *If $k_0 \in K$ and agent k_0 depends on agent k_1 in state x, then $k_1 \in K$.*
- *If $k_1 \in K$ and there is an agent k_0 that depends on agent k_1 in state x, then $k_0 \in K$.*
- *There is no strict subset $K' \subset K$ such that the above conditions hold for K'.*

If the agents in a set K interact in a state x, then we refer to x_K as an interaction state *for the agents in K.*

The concept of interaction introduced above captures a local dependence between a set of agents in a Dec-MDP. Note that, if x_K is an interaction state for the agents in K, this does not mean that each agent k in K depends on all other agents in that state x. Instead, it means that there is at least one agent in K that either depends on k or k depends on it. Note also that the definition above is transitive in the following sense: if agents k_0 and k_1 interact at some state x and agents k_1 and k_2 interact at that same state x, then agents k_0 and k_2 interact at x.

Definition 3.3 (Interaction Area). *Let $\mathcal{M} = (N, (\mathcal{X}_k), (\mathcal{A}_k), (\mathcal{Z}_k), \mathsf{P}, (\mathsf{O}_k), r, \gamma)$ be a general N-agent Dec-MDP. We define an* interaction area *$\mathcal{X}^I$ as follows:*

(i) $\mathcal{X}^I \subset \mathcal{X}_K$ for some set of agents K;

(ii) There is at least one state $x^* \in \mathcal{X}^I$ such that x^* is an interaction state for the agents in K;

(iii) For any $x \in \mathcal{X}^I$, $a_K \in \mathcal{A}_K$ and $y \notin \mathcal{X}^I$,

$$\mathbb{P}\left[X_K(t+1) = y \mid X_K(t) = x, A_K(t) = a_K\right] = \mathsf{P}_0(x_0, y_0) \prod_{k \in K} \mathsf{P}(x_k, a_k, y_k);$$

(iv) The set $\mathcal{X}^I$ is connected.[6]

An agent k is involved in an interaction at time t if there is one interaction area X^I involving a set of agents K such that $k \in K$ and $X(t) = x$, with $x = (x_K, x_{-K})$ and $x_K \in \mathcal{X}^I$.

Let us briefly review conditions i through iv. The first condition states that each interaction area will involve a subset K of the agents in a Dec-MDP. The second condition ensures that in every interaction area there is at least one interaction state involving all agents in K. This seeks to minimize the number of agents involved in each interaction. Condition iii defines interaction areas as regions of the state-space "around" an interaction state. This is perhaps most easily visualized in a robot navigation task. Returning to the scenario in Fig. 1, one interaction state could be $x_{-3}^{\text{narrow}} = (x_1^{\text{narrow}}, x_2^{\text{narrow}})$, corresponding to the situation in which both agents 1 and 2 are in the narrow doorway. In this case, an interaction area should include the state x_{-3}^{narrow} and at least those states in $\mathcal{X}_1 \times \mathcal{X}_2$ immediately adjacent to it. In this navigation example, in order for two agents to avoid crashing in the narrow doorway, they must coordinate *before* actually reaching the doorway and hence the inclusion of neighboring states in the interaction area associated with this interaction state. Finally, condition iv merely states that interaction areas are sets of "adjacent" states.

The purpose of defining/identifying the interaction areas in a Dec-MDP is to single out those situations in which the actions of one agent depend on other agents. An agent that is not involved in any interaction should be able to choose its individual actions somewhat independently of the other agents and thus be unaffected by partial joint state observability. In contrast, when in an interaction area, the agent should use state information from the other agents in the interaction area when choosing its actions. It is important to note that, unlike interaction states, interaction areas are not disjoint: an agent can be simultaneously involved in an interaction at two different interaction areas.

In this paper we are interested in those problems for which all agents involved in an interaction in a particular interaction area $\mathcal{X}^I \subset \mathcal{X}_K$ at time t have full access to the state $X_K(t)$. We henceforth refer to such a Dec-MDP as having *observable interactions*, a concept that we formalize in the next definition.

Definition 3.4. *A Dec-MDP has* observable interactions *if for any interaction area $\mathcal{X}^I$ involving a set K of agents it holds that for each $k \in K$ there is a set of local observations $\mathcal{Z}_k^I \subset \mathcal{Z}_k$ such that for every $x \in \mathcal{X}^I$*

$$\mathbb{P}\left[Z_k(t) \in \mathcal{Z}_k^I \mid X_K(t) \in \mathcal{X}^I\right] = 1$$

and, for every $z_k \in \mathcal{Z}_k^I$ there is a local state $x_K \in \mathcal{X}^I$ such that

$$\mathbb{P}\left[X_K(t) = x_K \mid Z_k(t) = z_k\right] = 1.$$

Our focus on Dec-MDPs with observable interactions, although apparently restrictive, actually translates a property often observed in real-world scenarios: when involved in an interaction, agents are often able to observe/communicate information that is relevant for coordination. In a sense,

[6]In this context we say that a set $U \subset \mathcal{X}$ is *connected* if, for any pair of states $x, y \in U$, there is a sequence of actions that, with positive probability, yields a trajectory $\{x(0), \ldots, x(T)\}$ such that $x(t) \in U, t = 0, \ldots, T$, and either $x(0) = x$ and $x(T) = y$ or vice-versa.

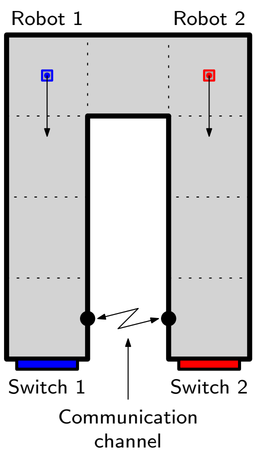


Figure 3: Example of a simple navigation task. The two robots, 1 and 2, must simultaneously activate the corresponding switch, marked with the same number and color. In this scenario, coordination needs only to occur when the robots are in the switch cells. For the purpose of coordination, the two switch cells have been equipped with a communication channel that allows the two robots to coordinate.

interaction areas encapsulate the need for information sharing in a general multiagent decision problem.

Generally, one may interpret interaction areas in one of two ways:

- As arising naturally from the sensory information available to the agents. The example in Fig. 1 provides an illustration of one such situation: the interaction between Robots 1 and 2 occurs only when both robots stand in opposite sides of the narrow doorway. In this situation, spacial proximity of the agents will typically allow the robots to perceive each other's state and/or communicate, allowing the agents to use joint state information in the decision process.

- As arising specifically to address inter-agent dependences. In this class of scenarios, communication capabilities are built into the agents specifically to allow for explicit handling of interactions among agents. An example of one such situation is depicted in Fig. 3: the interaction between Robots 1 and 2 occurs only at the switch locations. In these locations, a communication channel has been setup that can be used for the two robots to coordinate.

The main distinction between the two situations depicted: in the former situation, interaction is localized in such a way that it allows the agents to mutually perceive the state of the other; in the latter, however, interaction arises from a coupling that is part of the task definition. As such, full state observability should be explicitly ensured – in this case by the communication channel. In this paper, we are not concerned with the distinction between the two above situations.

We say that a Dec-MDP $\mathcal{M} = (N, (\mathcal{X}_k), (\mathcal{A}_k), (\mathcal{Z}_k), \mathsf{P}, (\mathsf{O}_k), r, \gamma)$ has *sparse interactions* if all agents are independent except in a set of M interaction areas, $\{X_1^I, \ldots, X_M^I\}$, with $X_i^I \subset \mathcal{X}_{K_i}$ for some set of agents K_i, and such that $|\mathcal{X}_i^I| \ll |\mathcal{X}_{K_i}|$. We refer to a Dec-MDP with sparse, observable interactions as a Dec-SIMDP (decentralized sparse-interaction MDP). For all agents outside interaction areas, the joint transition probabilities and reward function for a Dec-SIMDP can be factorized as in (5) and (7), and it is possible to model these agents using “individual MDPs”. On the other hand, the agents involved in an interaction can be modeled using a “local” MMDP.

This leads to the central definition in this section.

Definition 3.5 (Dec-SIMDP)**.** *Let $\mathcal{M} = (N, (\mathcal{X}_k), (\mathcal{A}_k), (\mathcal{Z}_k), \mathsf{P}, (\mathsf{O}_k), r, \gamma)$ be a Dec-MDP with sparse observable interactions, encapsulated in a set of M interaction areas, $\{\mathcal{X}_1^I, \ldots, \mathcal{X}_M^I\}$, as described above. We represent such a* decentralized sparse-interaction MDP *(Dec-SIMDP) as a tuple*

$$\Gamma = \big(\{\mathcal{M}_k, k = 1, \ldots, N\}, \{(\mathcal{X}_i^I, \mathcal{M}_i^I), i = 1, \ldots, M\}\big),$$

where

- *Each $\mathcal{M}_k$ is an MDP $\mathcal{M}_k = \big(\mathcal{X}_0 \times \mathcal{X}_k, \mathcal{A}_k, \mathsf{P}_k, r_k, \gamma\big)$ that individually models agent k in the absence of other agents, where r_k is the component of the joint reward function associated with agent k in the decomposition in* (7)*;*

- *Each $\mathcal{M}_i^I$ is an MMDP that captures a* local interaction *between K_i agents in the states in $\mathcal{X}_i^I$ and is given by $\mathcal{M}_i^I = (K_i, \mathcal{X}_{K_i}, (\mathcal{A}_k), \mathsf{P}_i^I, r_i^I, \gamma)$, with $\mathcal{X}_i^I \subset \mathcal{X}_{K_i}$.*

Each MMDP $\mathcal{M}_i^I$ describes the interaction between a subset K_i of the N agents, and the corresponding state-space $\mathcal{X}_{K_i}$ is a superset of an interaction area as defined above.

A Dec-SIMDP is an alternative way of representing a Dec-MDP with observable interactions. In the states of each interaction area in a Dec-SIMDP, and only in these, the agents involved in the associated MMDP are *able to communicate freely*. In these areas the agents can thus use communication to overcome local state perception and decide jointly on their action. Outside these areas, the agents have only a local perception of the state and should, therefore, choose the actions independently of the other agents. A simple albeit innacurate way of thinking of a a Dec-SIMDP is as a Dec-MDP in which each agent has access, at each time step, to all state-information required to predict its next local state and reward. In the remainder of this section and throughout Section 4, we assume interaction areas to be known in advance, *i.e.*, they are provided as part of the model specification. For a discussion on how these areas can be determined, we refer to [22].

It is interesting to explore the relation between the Dec-SIMDP model and the MDP, MMDP, and Dec-MDP models. First of all, as expected, in the absence of any interaction areas, the Dec-SIMDP reduces to a set of independent MDPs that can be solved separately. This captures the situation in which the agents are completely independent. On the other hand, given an Dec-SIMDP Γ, it possible to construct an associated MMDP $\mathcal{M}$ whose optimal policies provide a performance upper bound on the Dec-SIMDP solution. Our algorithm for Dec-SIMDPs arises precisely from the consideration of this associated MMDP and is described in greater detail in the following subsection.

Finally, as discussed above, the Dec-SIMDP model is essentially a Dec-MDP model with joint state observability in the interaction areas. In those situations in which all agents interact in all states, as assumed in the general Dec-MDP model, the whole state-space is an interaction area and, as such, our assumption of observable interactions renders our model equivalent to an MMDP.

Nevertheless, the appeal of the Dec-SIMDP model is that many practical situations do not fall in either of the two extreme cases *i.e.*, independent MDPs vs. fully observable MMDP. It is in these situations that the Dec-SIMDP model may bring an advantage over more general but potentially intractable models.

4 Planning in Dec-SIMDPs

In this section we address the problem of planning in Dec-SIMDPs, *i.e.*, estimating the optimal policy for each agent in a Dec-SIMDP when the model is fully specified – this including the interaction areas. We start by introducing a general heuristic approach to the problem of planning in a Dec-SIMDP that relies on the solution for an associated POMDP. This leads to the two general algorithms dubbed MPSI and LAPSI. We then introduce the concept of *generalized α-vectors* for Dec-SIMDPs and describe instances of both MPSI and LAPSI that use generalized alpha-vectors. We discuss some of the appealing features of these methods as well as some interesting issues raised by our solution method.

To minimize the disruption of the main text, we collected the proofs of all results in this Section in Appendix A and provide only brief overviews as needed for the presentation.

4.1 Heuristic Planning in Dec-SIMDP

Let us start by considering a Dec-SIMDP in which all except one of the agents have full state observability. Let this agent be agent k and let us further suppose that the remaining agents (those with full state observability) follow some fixed known policy, π_{-k}. Agent k can thus be modeled as a POMDP and the other agents can be collectively regarded as part of the environment. In this particular situation, any POMDP solution method can be used to compute the policy for agent k.

Our heuristic departs from this simplified setting and computes a policy for each agent k as if all other agents indeed had full observability and followed some fixed known policy π_{-k}. This hypothesized policy π_{-k} will allow each agent k to approximately "track" the other agents and choose its actions accordingly. The closer π_{-k} is to the actual policy of the other agents, the better agent k will be able to track them, and the better he will decide.

This idea can be used in general Dec-(PO)MDPs. However, as the hypothesized policy π_{-k} will seldom correspond to the actual policy followed by the other agents, it is only natural that this method will not allow each agent k to properly "track" the other agents and decide accordingly, this leading to poor results in general Dec-(PO)MDPs. The particular structure of Dec-SIMDPs, however, renders this approach more appealing for two reasons: on one hand, outside interaction areas the policy of agent k ideally exhibits minimum dependence on the state/policy of the other agents. As such, poor tracking in these areas has little impact on the policy of agent k. In interaction areas, on the other hand, local full observability allows agent k to perfectly track the other agents involved in the interaction and choose its actions accordingly.

The two proposed algorithms, dubbed MPSI (Myopic Planning for Sparse Interactions) and LAPSI (Look-Ahead Planning for Sparse Interactions), share the underlying idea described above but arise from considering different hypothetical policies for the other agents. In MPSI, agent k

considers each of the other agents as completely self-centered and oblivious to the interactions. Agent k thus acts as if each agent j, $j \neq k$, acts according to a policy π_j – the optimal policy for the corresponding MDP $\mathcal{M}_j$ in the Dec-SIMDP. In environments with almost no interaction, the MPSI heuristic actually provides a good approximation to the policy of the other agents outside the interaction areas.

In contrast, in LAPSI, agent k considers that all other agents jointly adopt the optimal policy for the underlying MMDP.[7] LAPSI is, in a sense, the counterpart to MPSI, as it provides a good approximation to the policy of the other agents in scenarios where the interactions are not so sparse.

Using the corresponding hypothesized policies for the remaining agents, MPSI and LAPSI can now leverage any POMDP solution methods to obtain a policy for each agent k. In the continuation we introduce the concept of *generalized α-vectors*, that will later be used to construct particular instances of both MPSI and LAPSI.

4.2 Generalized α-vectors for Dec-SIMDPs

The two methods proposed in the previous subsection, MPSI and LAPSI, are described in terms of general POMDP solvers and allow efficiently computing an individual policy for each agent in a Dec-SIMDP. As seen in the previous section, this arises from the observation that, if all remaining agents have full-state observability and follow some known policy π_{-k}, then agent k can be modeled as a POMDP. In this section we follow on this idea and propose particular instances of both MPSI and LAPSI that further exploit the structure of the Dec-SIMDPs model.

Using the POMDP model obtained by adopting the assumption above, agent k computes an individual policy π_k that maps *beliefs* to actions. However, that agent k has full local state observability, implying that kth component of the state is always unambiguously determined. Furthermore, given our assumption of observable interactions (see Definition 3.4), at each time step only those state-components corresponding to agents not interacting with agent k will be unobservable. However, by definition, the evolution of these state-components does not depend on the state/action of agent k and depends only on π_{-k}. In the continuation, and to avoid unnecessarily complicating the presentation, we focus on a 2-agent scenario, remarking however that the development presented extends trivially to more than two agents at the cost of more cumbersome expressions.

Recovering the POMDP model for agent k, we have from Section 2,

$$Q^*(\mathbf{b}, a) = \sum_x \mathbf{b}_x \left[r(x, a) + \gamma \sum_{z,y} \mathsf{P}(x, a, y)\mathsf{O}(y, a, z) \max_u Q^*(\mathsf{Bel}(\mathbf{b}, a, z), u) \right].$$

Taking into consideration that agent k has full local observability, the belief $\mathbf{b}$ concerns only the state component of the other agent. We write this explicitly as

$$\begin{aligned} &Q^*(\bar{x}_k, \mathbf{b}_{-k}, a) \\ &\qquad = \sum_{x_{-k}} \mathbf{b}_{x_{-k}} \left[r(x, a) + \gamma \sum_{z,y} \mathsf{P}(x, a, y)\mathsf{O}(y, a, z) \max_u Q^*(\bar{y}_k, \mathsf{Bel}(\mathbf{b}, a, z), u) \right]. \end{aligned}$$

[7] We recall that a Dec-SIMDP is a particular class of Dec-MDP. The MMDP associated with the Dec-SIMDP is thus the MMDP obtained by endowing all agents with full-state observability at all times.

Noting now that the other agent is assumed to follow a fixed policy that depends only on the current state, we can eliminate the explicit dependence on its action and write

$$\begin{aligned}&Q^*(\bar{x}_k, \mathbf{b}_{-k}, a_k)\\&\quad= \sum_{x_{-k}} \mathbf{b}_{x_{-k}} \left[r_{\pi_{-k}}(x, a_k) + \gamma \sum_{z,y} \mathsf{P}_{\pi_{-k}}(x, a_k, y)\mathsf{O}(y, a, z) \max_{u_k} Q^*(\bar{y}_k, \mathsf{Bel}(\mathbf{b}, a_k, z), u_k) \right].\end{aligned}$$

where

$$\begin{aligned}r_{\pi_{-k}}(x, a_k) &= \sum_{a_{-k}} \pi_{-k}(x, a_{-k}) r\big(x, (a_{-k}, a_k)\big)\\ \mathsf{P}_{\pi_{-k}}(x, a_k, y) &= \sum_{a_{-k}} \pi_{-k}(x, a_{-k}) \mathsf{P}\big(x, (a_{-k}, a_k), y\big).\end{aligned}$$

Now, every time step t that agent k is in an interaction area, implying that so is the other agent, it can unambiguously perceive their joint state and hence $Z_k(t) = X(t)$. In all remaining time steps, $Z_k(t) = \bar{X}_k(t)$, meaning that the agent observes only its local state. Denoting by $\mathcal{X}_I$ the set of *all* joint states in any interaction area, *i.e.*, $\mathcal{X}_I = \bigcup_{i=1}^{M} \mathcal{X}_i^I$, we have

$$\begin{aligned}&Q^*(\bar{x}_k, \mathbf{b}_{-k}, a_k)\\&\quad= \sum_{x_{-k}} \mathbf{b}_{x_{-k}} \Bigg[r_{\pi_{-k}}(x, a_k) + \gamma \sum_{y \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} Q^*(\bar{y}_k, y_{-k}, u_k)\\&\qquad\qquad + \gamma \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} Q^*(\bar{y}_k, \mathsf{Bel}(\mathbf{b}, a_k, \bar{y}_k), u_k) \Bigg].\end{aligned} \tag{9}$$

Let us now focus explicitly on two elements in the above expression, namely, the updated belief $\mathsf{Bel}(\cdot)$ and the optimal Q-function for the states in the interaction areas. Recall the general belief-update expression from Section 2.2,

$$\mathsf{Bel}_y(\mathbf{b}, a_k, Z_k(t+1)) = \eta \sum_x \mathbf{b}_x(t)\mathsf{P}(x, A(t), y)\mathsf{O}(y, A(t), Z_k(t+1)).$$

We note that, in the particular setting considered here, the belief concerns only the distribution over states of the other agent. As such, if $X_k(t) = x_k$, we have

$$\mathsf{Bel}_{y_{-k}}(\mathbf{b}, a_k, Z_k(t+1)) = \eta \sum_{x_{-k}} \mathbf{b}_{x_{-k}}(t)\mathsf{P}_{\pi_{-k}}(x, A_k(t), y_{-k})\mathsf{O}(y_{-k}, A_k(t), Z_k(t+1)),$$

where $x = (x_{-k}, \bar{x}_k)$ and

$$\mathsf{P}_{\pi_{-k}}(x, a_k, y_{-k}) = \sum_{a_{-k}} \pi_{-k}(x, a_{-k})\mathsf{P}_{-k}\big(x_{-k}, (a_{-k}, a), y_{-k}\big).$$

If the agents are not in an interaction area, the transitions of the other agent does not depend on the actions of agent k and hence the above expression simplifies to

$$\mathsf{Bel}_{y_{-k}}(\mathbf{b}, a_k, Z_k(t+1)) = \eta \sum_{x_{-k}} \mathbf{b}_{x_{-k}}(t)\mathsf{P}_{\pi_{-k}}(x, y_{-k})\mathsf{O}(y_{-k}, A_k(t), Z_k(t+1)).$$

If $\mathcal{X}_k(t+1) = \bar{y}_k$ and $y = (y_{-k}, \bar{y}_k)$ is in an interaction area, then agent k can observe the state of the other agent and, as such,

$$\mathsf{Bel}_{y_{-k}}(\mathbf{b}, a_k, y) = \mathbf{e}_{y_{-k}}.$$

For the general situation in which $y \notin \mathcal{X}_I$,

$$\mathsf{Bel}_{y_{-k}}(\mathbf{b}, a_k, \bar{y}_k) = \eta \sum_{x_{-k}} \mathbf{b}_{x_{-k}}(t)\mathsf{P}_{\pi_{-k}}(x, A_k(t), y_{-k})\mathbb{I}_{\mathcal{X}_I^c}(y), \tag{10}$$

where, for a general set $U \subset \mathcal{X}$, $\mathbb{I}_U$ is the indicator function for U and U^c denotes the complement of U in $\mathcal{X}$.

Let us now consider the expression for $Q^*(\bar{x}_k, \mathbf{b}_{-k}, a_k)$ when the agents are in an interaction area. In this case, $\mathbf{b}_{-k} = \mathbf{e}_{x_{-k}}$ for some x_{-k}, and we have

$$\begin{aligned} &Q^*(\bar{x}_k, x_{-k}, a_k) \\ &\quad = r_{\pi_{-k}}(x, a_k) + \gamma \sum_{y \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} Q^*(\bar{y}_k, y_{-k}, u_k) \\ &\qquad\qquad + \gamma \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} Q^*(\bar{y}_k, \mathsf{Bel}(b, a_k, \bar{y}_k), u_k). \end{aligned} \tag{11}$$

Noting the similarity between the right-hand side of (11) and the term in square brackets in the right-hand side of (9), we define a *generalized α-vector* for agent k, $\boldsymbol{\alpha}_k$, recursively as follows:

$$\begin{aligned} \boldsymbol{\alpha}_k(x, a_k) &= r_{\pi_{-k}}(x, a_k) + \gamma \sum_{y} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} \boldsymbol{\alpha}_k(y, u_k)\mathbb{I}_{\mathcal{X}_I}(y) \\ &\quad + \gamma \sum_{y_k} \mathsf{P}(\bar{x}_k, a_k, \bar{y}_k) \max_{u_k} \sum_{y_{-k}} \mathsf{P}_{\pi_{-k}}(x_{-k}, y_{-k})\boldsymbol{\alpha}_k(y, u_k)\mathbb{I}_{\mathcal{X}_I^c}(y). \end{aligned} \tag{12}$$

Theorem 4.1. *Given a two-agent Dec-SIMDP Γ, the generalized α-vectors associated with agent k when the other agent follows a fixed known policy π_{-k} are well-defined, i.e., they always exist and are unique.*

Proof. In order to minimize the disruption of the text, we include the complete proof of the theorem in Appendix A, of which we present here only a brief sketch.

To establish this result, we introduce a dynamic-programming operator $\mathbf{T}_k$ such that

$$\boldsymbol{\alpha}_k = \mathbf{T}_k \boldsymbol{\alpha}_k$$

and show this operator to be a contraction in the sup-norm. The statement in the theorem follows from Banach's fixed-point theorem. □

It is worth noting that the operator $\mathbf{T}_k$ introduced in the proof of Theorem 4.1 can actually be used to iteratively compute the generalized α-vectors, in a way very similar to the value-iteration algorithm used to compute the optimal Q-function for an MDP. And, in fact, the next result establishes that the generalized α-vectors associated with a Dec-SIMDP can be computed efficiently.

Theorem 4.2. *The generalized α-vectors for a 2-agent Dec-SIMDP Γ verifying the conditions of Theorem 4.1 can be computed in polynomial time.*

Proof. The result follows from noting that the generalized α-vectors can be computed by solving an associated MDP. We again refer to Appendix A for a complete proof. $\square$

Actually, an MDP is a particular case of a Dec-SIMDP in which there is a single agent. For this particular Dec-SIMDP, the generalized α-vectors indeed correspond to the optimal Q-values. It then follows from the above result that computing the generalized α-vectors for a Dec-SIMDP is actually a P-complete problem.

We conclude by noting that all results extend naturally to scenarios with more than two agents. For example, the definition in (12) takes the more general form

$$\boldsymbol{\alpha}_k(x, a_k) = r_{\pi_{-k}}(x, a_k) + \gamma \sum_{y_O} \mathsf{P}_{\pi_{-k}}(x_O, a_k, y_O) \max_{u_k} \sum_{y_{-O}} \mathsf{P}_{\pi_{-k}}(x_{-O}, y_{-O}) \boldsymbol{\alpha}_k(y, u_k)$$

where we write a state $y \in \mathcal{X}$ as $y = (y_O, y_{-O})$. The components y_O correspond to the observable components of y – those that belong to agents involved in an interaction with agent k, – and y_{-O} correspond to the remaining components.

4.3 Generalized α-vectors in LAPSI and MPSI

We now propose using the generalized α-vectors and use as estimates $\hat{Q}(\mathbf{b}, a_k)$ for the optimal Q-function for agent k,

$$\hat{Q}(\bar{x}_k, \mathbf{b}_{-k}, a_k) = \sum_{x_{-k}} \mathbf{b}_{x_{-k}} \boldsymbol{\alpha}_k(x, a_k). \tag{13}$$

We point out that the approximation above shares several features with the Q_{MDP} algorithm reviewed in Section 2.2. In fact, as seen in our previous discussion, computing the generalized α-vectors for a Dec-SIMDP can be done by solving and associated MDP, for which the above approximation actually corresponds to the Q_{MDP} algorithm.

We now derive error bounds for the approximation in (13) that depend only on the *dispersion* of the maximum values of the generalized α-vectors outside the interaction areas. This result can be extended to general POMDPs, providing error bounds for the Q_{MDP} algorithm that depend only on the optimal Q-function for the underlying MDP.

Given an α-vector $\boldsymbol{\alpha}_k(x, a_k)$, let x_O denote the *observable* components of x – those corresponding to agents interacting with k at $\bar{x}_k$, – and x_{-O} the remaining components. We define the *dispersion* of a set of α-vectors $\boldsymbol{\alpha}_k(x, a_k)$, with $x \in \mathcal{X}$ and $a_k \in \mathcal{A}_k$, as

$$\boldsymbol{\delta}_k = \max_{x_O} \left| \max_{u_k} \sum_{x_{-O}} \boldsymbol{\alpha}_k((x_O, x_{-O}), u_k) - \sum_{x_{-O}} \max_{u_k} \boldsymbol{\alpha}_k((x_O, x_{-O}), u_k) \right| \tag{14}$$

The dispersion of a set of α-vectors measures how the maximum value of $\boldsymbol{\alpha}_k$ taken over all actions differs from the corresponding average in the non-observable components of the state. In other words, the dispersion quantifies how the lack of knowledge of agent k on the state of the other agents can impact the action choice of agent k in terms of value. This leads to the following result.

Theorem 4.3. *Let $\mathcal{M} = \big(\{\mathcal{M}_k, k = 1, \ldots, N\}, \{(\mathcal{X}_i^I, \mathcal{M}_i^I), i = 1, \ldots, M\}\big)$ be a Dec-SIMDP and let π_k denote the policy for agent k obtained from the approximation* (13) *when the policy π_{-k} for the other agents is fixed and known. Then,*

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma^2}{1-\gamma}\boldsymbol{\delta}_k, \tag{15}$$

where $\boldsymbol{\delta}_k$ represents the dispersion of the alpha-vectors associated with π_{-k}.

Proof. See Appendix A. □

Theorem 4.3 translates well-known bounds for approximate policies to the particular setting considered herein. Nevertheless, it prompts several interesting observations. First of all, as expected, the bound in (15) is proportional to the total dispersion of the generalized α-vectors. As noted before, the dispersion of the generalized α-vectors somehow quantifies the fundamental trade-off being made in the approximation (13): it measures how much state uncertainty outside the interaction areas can impact the choice of the maximizing action.

Secondly, it is clear that the bound in (15) is zero if one of two things happens, to know

- The whole state-space is an interaction area, *i.e.*, $\mathcal{X}_I = \mathcal{X}$. In this case, we recover the MMDP version of the problem, as discussed in Section 3.
- There are no interaction states, *i.e.*, $\mathcal{X}_I = \emptyset$. In this case, the definition of the generalized α-vectors for agent k does not depend on the other agents, implying that $\boldsymbol{\delta}_k = 0$.

Using the generalized α-vectors in LAPSI and MPSI is now straightforward. Essentially, both methods use the estimate in (13) to chooose the action for agent k. The difference between the two methods thus lies on the policy π_{-k} hypothesized for the other agents, needed to both track the belief $\mathbf{b}_{-k}$ and to compute the α-vectors. In MPSI, π_{-k} is taken as the reduced policy obtained from the individual policies pi_j, $j \neq k$, the optimal policy for the corresponding MDP $\mathcal{M}_j$ in the Dec-SIMDP. In contrast, in LAPSI, π_{-k} is obtained from the optimal policy for the underlying MMDP by ignoring component k.

For illustration purposes, in the following section we apply both these methods to several problems of different dimension, using (13) as our estimate for the POMDP optimal Q-function. Our results indicate that, even using such a sub-optimal POMDP solver, LAPSI is able to attain a performance that is close to optimal in all test scenarios while incurring in a computational cost similar to that of solving the underlying MMDP. MPSI, on the other hand, while computationally more efficient, seems to lead to agents that are "too cautious". We also compare our algorithms with a previous algorithm for Dec-SIMDPs introduced in [30] and henceforth referred as the IDMG algorithm. Our results indicate that LAPSI is able to attain similar performance to that of IDMG while providing significant computational savings. We also show that the IDMG algorithm is, by design, unable to consider future interactions when planning outside the interaction areas, this potentially leading to poor performance. This limitation is not present in LAPSI.

Table 1: Dimension of the different test scenarios.

Environment	**# States**
Map 1	441
Map 2	$1,296$
Map 3	400
Map 4	$65,536$
CIT	$4,900$
CMU	$17,689$
ISR	$1,849$
MIT	$2,401$
PENTAGON	$2,704$
SUNY	$5,476$

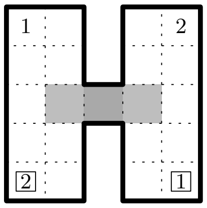
4.4 Results

In this section we describe the results obtained from applying both MPSI and LAPSI to a range of problems of different dimensions, and analyze the performance of our methods in each of the test scenarios. To gain a better understanding on the applicability and general properties of our methods, we compared the performance of both MPSI and LAPSI to that of the optimal fully observable MMDP policy and that of the IDMG algorithm from [30]. In the IDMG algorithm, each agent k in a Dec-SIMDP $\big(\{\mathcal{M}_k, k = 1, \ldots, N\}, \{(\mathcal{X}_i^I, \mathcal{M}_i^I), i = 1, \ldots, M\}\big)$ follows the optimal individual policy π_k for the MDP $\mathcal{M}_k$ outside the interaction areas. In the interaction areas, the agents engage in a sequence of local matrix games in which they jointly adopt the equilibrium policy.

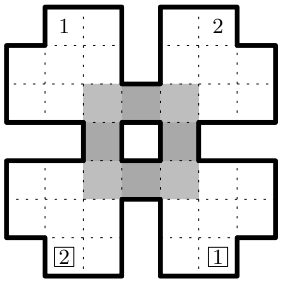
The different scenarios used to test our algorithm are depicted in Fig. 4, and the dimension of the state-space for the corresponding Dec-MDP is summarized in Table 1. The reason for using navigation scenarios is that the Dec-SIMDP model appears particularly appealing for modeling multi-robot problems. Furthermore, in this class of problems, the results can be easily visualized and interpreted. In each of the test scenarios, each robot in a set of two/four robots must reach one specific state. In the smaller environments (Maps 1 through 4), the goal state is marked with a number, corresponding to the number of the robot. The cells with a boxed number correspond to the initial states for the robots. In the larger environments, the goal for each robot is marked with a cross, $\times$, and the robots each depart from the other robot's goal state, in an attempt to increase the possibility of interaction. Each robot has 4 possible actions that move the robot in the corresponding direction with probability 0.8 and fail with probability 0.2. The shaded regions correspond to interaction areas, inside of which the darker cells correspond to interaction states, in which the robots get a penalty of -20 if they stand in the same cell simultaneously. Also, in these interaction states, the rate of action failure is increased to 0.4.[8] Upon reaching the corresponding goal, each agent receives a reward of $+1$ and its position is reset to the initial state.

For each of the different scenarios in Fig. 4, we ran the four algorithms above and then tested

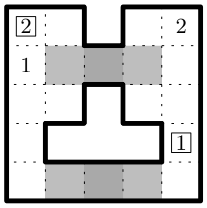
[8]Both the penalty and the increased action failure rate imply that there is both reward and transition dependence in the interaction areas.



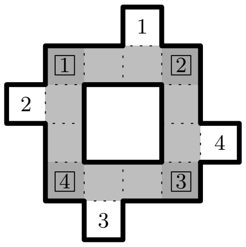
(a) Map 01.



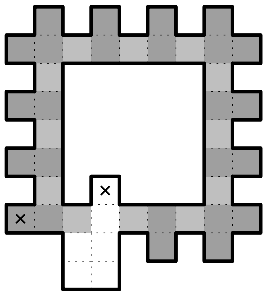
(b) Map 02.



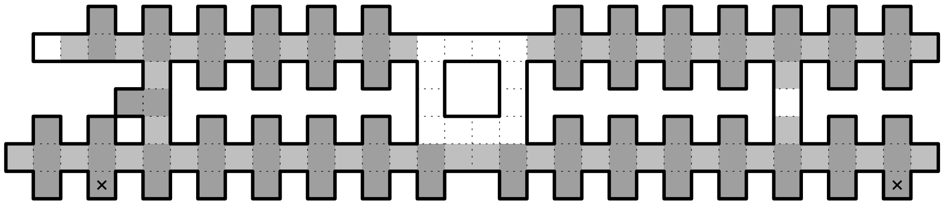
(c) Map 03.



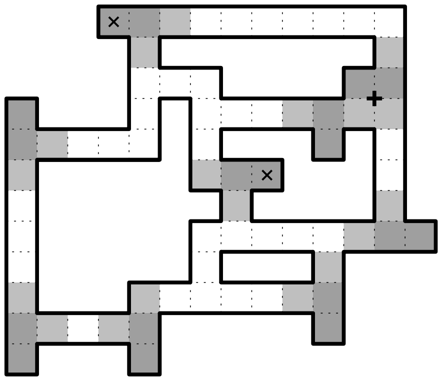
(d) Map 04.



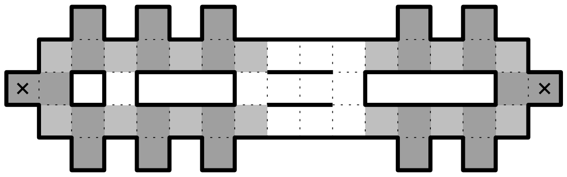
(e) ISR.



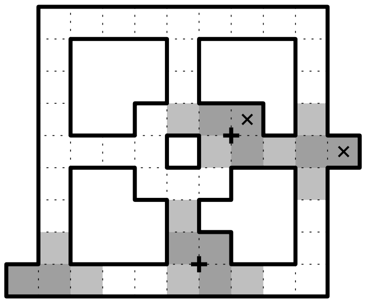
(f) CMU.



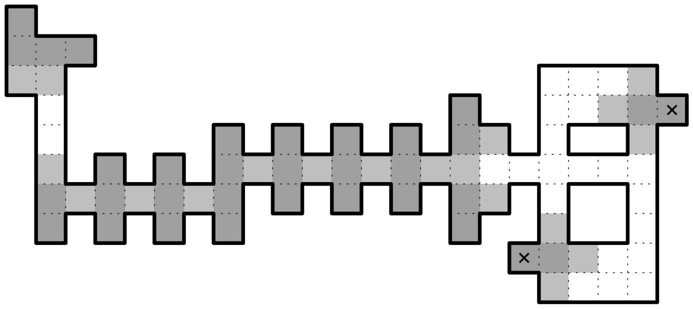
(g) CIT.



(h) MIT.



(i) PENTAGON.



(j) SUNY.

Figure 4: Environments used in the experiments.

Table 2: Total discounted reward for each of the four different algorithms in each of the test-scenarios. The results are averaged over $1,000$ independent Monte-Carlo runs. Bold entries correspond to optimal values (differences are not statistically significant). Also, italic entries correspond to values whose differences are not statistically different.

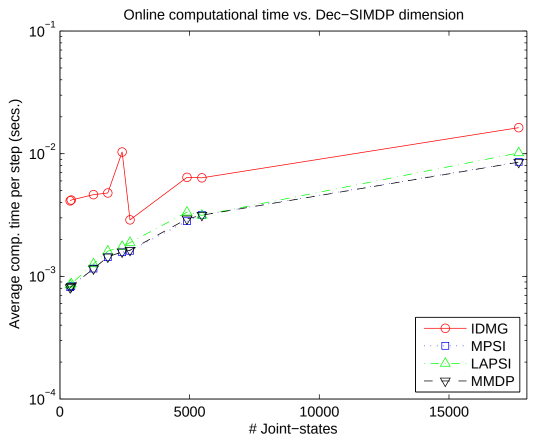
Environment	**IDMG**	**MPSI**	**LAPSI**	**Opt.**
Map 1	*12.035*	11.130	*11.992*	**12.588**
Map 2	10.672	10.159	10.947	**11.069**
Map 3	*13.722*	13.249	*13.701*	**14.380**
Map 4	–	15.384	15.564	**16.447**
CIT	**11.178**	**11.105**	**11.126**	**11.151**
CMU	**2.839**	2.688	**2.824**	**2.906**
ISR	14.168	*13.937*	*13.997*	14.335
MIT	**6.663**	**6.641**	**6.648**	**6.681**
PENTAGON	*16.031*	15.162	*15.976*	**16.312**
SUNY	**11.161**	**11.130**	**11.139**	**11.110**

Table 3: Total number of steps until the two robots reach the corresponding goals for each of the four different algorithms in each of the test-scenarios. The results are averaged over $1,000$ independent Monte-Carlo runs. Bold entries correspond to optimal values (differences are not statistically significant). Also, italic entries correspond to values whose differences are not statistically different.

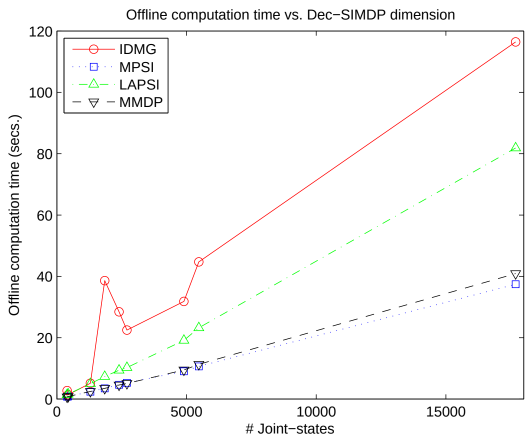
Environment	**IDMG**	**MPSI**	**LAPSI**	**Opt.**
Map 1	*11.021*	12.752	*11.091*	**10.090**
Map 2	13.368	14.433	12.828	**12.511**
Map 3	*8.450*	9.282	*8.477*	**7.477**
Map 4	–	*6.088*	*6.071*	**5.001**
CIT	**12.422**	**12.552**	**12.514**	**12.466**
CMU	*39.338*	49.341	*39.444*	**38.850**
ISR	*7.993*	8.986	*8.012*	**7.504**
MIT	**22.578**	24.507	**22.618**	**22.523**
PENTAGON	*5.348*	19.684	*5.416*	**5.006**
SUNY	**12.448**	**12.500**	**12.487**	**12.539**

the computed policy for $1,000$ independent trials of 100 steps each, in the smaller environments, and 250 time-steps each, in the larger environments. The obtained results can be found in Tables 2 and 3.

First of all, that the LAPSI algorithm performed very close to the optimal MMDP policy in all environments, in spite of the significant difference in terms of state information available to both methods. Also, in most scenarios, LAPSI and IDMG performed similarly, both in terms



(a) Online



(b) Offline

Figure 5: Computation time for the different algorithms as a function of the problem dimension.

of total discounted reward and in terms of time-to-goal. The only exceptions are Map 2, where LAPSI outperformed IDMG, and ISR, where IDMG outperformed LAPSI. Interestingly, however, the difference in terms of time-to-goal in the ISR environment is not significant. In any case, our results agree with previous ones that showed that IDMG attained close-to-optimal performance in most such scenarios [30].

Another interesting observation is that MPSI typically performed worse than the other methods. As pointed out before, since an agent in MPSI considers the other agents to be selfish and disregard the consequences of mis-coordinations (each is focused only on its individual goal), it is expected that the agent following MPSI is more "cautious" and hence the observed longer time to the goal.

We note in our results that the difference in performance between LAPSI/IDMG and the optimal MMDP policy occurs both in terms of total discounted reward and also in terms of *time-to-goal*. And indeed, given the discount factor γ, the latter in part explains the former: if the agents take longer to reach their goal, the corresponding reward will be further discounted. The above results thus seem to indicate that both our algorithms and the IDMG algorithm require more time to reach the goal configuration than that needed by MMDP solution, and this time must be spent in avoiding the penalties.

The choice of interaction areas greatly influences the ability of the algorithms to avoid penalties without incurring in delays in reaching the goal. This phenomenon was also reported in [30] concerning the IDMG algorithm.

It is also worth noting at this point that, since the IDMG method requires the computation of several equilibria both in the off-line planning phase and in the on-line running phase, the computational complexity of the IDMG algorithm may quickly become prohibitive, in scenarios with large action spaces and/or with many interaction areas. To assess whether this is indeed so, we compared the computational effort of our methods with that of IDMG, both in terms of the average off-line computation time and the on-line computation time. These results are summarized in Fig. 5.

Both MPSI and LAPSI are significantly more computation efficient than the IDMG algorithm

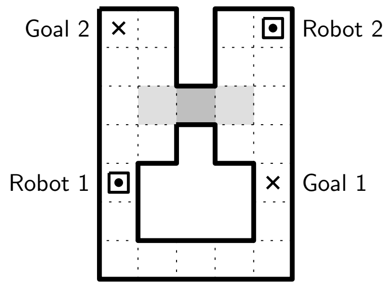


Figure 6: Example scenario where avoiding the interaction may be beneficial.

according to any of the two performance metrics. It is also interesting to note how the average computation times evolve with the dimension of the problem.

Finally, the IDMG method is, by construction, unable to consider future interactions when planning for the action in a non-interaction area. In this sense, the IDMG algorithm is "myopic" to such interactions and only handles these as it reaches an interaction area. This can have a negative impact on the performance of the method, as seen in the following final example.

Consider the scenario depicted in Fig. 6. Once again, the two robots must reach the marked states while avoiding simultaneously crossing the narrow pathways. We model this problem using a Dec-SIMDP: the two sets of shaded cells represent two interaction areas in which the robots only get a non-zero penalty by standing simultaneously in the darker state. In this environment, and ignoring the interaction, Robot 1 can reach its goal by using either of the narrow pathways, since both trajectories have the same length. However, Robot 2 should use the upper pathway, since it is significantly faster than using the lower pathway.

By using the IDMG algorithm, Robot 2 goes for the upper pathway while Robot 1 chooses randomly between the two. For concreteness, let's suppose that Robot 1 chooses to go for the upper pathway. In this case, according to the IDMG algorithm, both robots reach the interaction area simultaneously and Robot 1 must move out of the way for Robot 2 to go on. This means that, in total, the two robots take a mean time of 9 steps to reach the goal. If, instead, Robot 1 takes the lower pathway, the two robots will reach their goal states in 8 steps. Since the IDMG algorithm chooses between the two randomly – or, at least, has no way to differentiate between the two – the average time to the goal is 8.5 time-steps. We ran $1,000$ independent trials using the IDMG algorithm in this scenario and, indeed, obtained an average of 8.485 steps to goal, with a standard deviation of 0.5. Clearly, it seems possible to do better in this scenario by simply considering that it may be more convenient to use the lower pathway.

For comparison purposes, we also ran $1,000$ independent trials using the LAPSI algorithm in this same scenario. Out of $1,000$ trials, Robot 1 *always* picked the lower pathway. As expected, the group had an average time-to-goal of 8 time-steps with a variance of 0. Notice that this difference could be made arbitrarily large by increasing the "narrow doorway" to an arbitrary number of states, thus causing an arbitrarily large delay.

5 Related Work

In the past decades, a wide range of models have been proposed to formalize decision-making problems in multiagent systems. These include general models such as stochastic games [26] as well as more specific models such as multiagent MDPs [8], Dec-(PO)MDPs [6], I-POMDPs [11] among others. This paper follows on the extensive literature on Dec-(PO)MDP-related approaches.

The Dec-MDP and Dec-POMDP models were originally proposed by Bernstein et al. [6]. The paper introduces both models and proceeds by analyzing the computational complexity of finite-horizon Dec-MDPs and Dec-POMDPs. The fundamental result is that finite-horizon Dec-MDPs are NEXP-complete even for the "benign" 2-agent scenarios. Complexity results are even worst in non-cooperative settings: non-cooperative partially observable stochastic games are complete for the $\mathrm{NEXP}^{\mathrm{NP}}$ class [13].

These disencouraging complexity results have led to a significant amount of research on (i) approximate methods for Dec-(PO)MDPs that, in a sense, trade-off optimality for computability; and (ii) models that, while less general than Dec-POMDPs, still manage to capture fundamental features of this class of problems such as decentralized control and partial state observability. These models, in a sense, trade-off representability for computability.

Several of these models assume, to some degree, that the interaction between the different agents can be simplified. For example in *transition-independent Dec-MDPs* [4, 5], the transition probabilities for each agent depend solely on its own actions and local states (see Section 2.4). This class of problems can be solved using the Coverage Set algorithm [5] and has been shown to be NP-complete, in contrast with the NEXP-completeness of general Dec-MDPs.

As seen in Section 3, the Dec-SIMDP model proposed in this paper allows the transition probabilities for an agent to depend on the actions of other agents in the interaction areas. In this sense, it is more general than transition independent Dec-MDPs. On the other hand, to deal with these dependences we assume joint state observability in these areas – or, equivalently, free instantaneous communication. It remains an open question what is the worst-case complexity associated with the Dec-SIMDP model.

Local interactions have also been exploited in other multiagent scenarios. For example, several works propose the use of hierarchical approaches that allow the agents to learn at different levels of abstraction [10, 20, 31]. The idea in these approaches is to subdivide the overall task in a hierarchy of subtasks, each restricted to the states and actions relevant to that particular subtask. This task decomposition allows the subtasks to be conducted *individually*, without requiring the agents to know the state of others or communicate their own state. Going up in the hierarchy thus corresponds to moving from low-level "local" tasks to higher-level "global" tasks, in which coordination is necessary and must be accounted for explicitly. However, since execution at the highest level actually corresponds to several low-level time-steps, this means that communication needs are minimized.

Coordination graphs [14] capture local dependences between the different agents in an MMDP. Coordination graphs thus allow the overall Q-function to be decomposed into "local" Q-functions, each of which can be optimized individually. Coordination graphs have been used for efficient planning [14] and learning [16] in large multiagent MDPs.

Both hierarchical approaches and coordination-graph-based approaches exploit local interactions between the agents and can thus accommodate some level of partial state observablity, even if not originally designed to. The interactions between the agents captured by coordination graphs are closely related to the notion of *interaction* introduced in Section 3. Also, as with the interaction areas in the Dec-SIMDP model, both aforementioned works assume the coordination graph or subtask hierarchy to be known in advance. A posterior work [17] introduces the concept of "utile coordination", proposing a method to actually *learn* the coordination graph structure. Unlike the previous approaches [14, 16], this work does not assume that the coordination graph is known but, instead, it is learned from experience. Although using a fundamentally different approach, it shares the same underlying idea as [22]: both methods infer where joint-state information can improve the performance of the agents.

In the game-theoretic literature, several works have explored local dependences between the players in large games. For example, *graphical games* [15] represent n-player matrix games using a graph where players correspond to nodes in an undirected graph and the payoff for that player depends only on the actions of its direct neighbors in the graph. This concept is further exploited in *action-graph* games, that generalize graphical games in several aspects. Multiple algorithns have been proposed to compute Nash equilibria in this class of games, mostly relying on the so-called "continuation methods" [7]. In these methods a known solution for a "simple" game is gradually perturbed toward the desired game until a solution is obtained. These methods take an amount of time that is exponential in the *in-degree* of the action-graph, and not in the *number of nodes* – typically considered as the dimension of the game. Therefore, games with many context-specific independences yield sparsely connected action-graphs, leading to exponential savings in computational time. Posterior works [33, 34] further exploit the structure of action-graph games for some classes of problems to allow for more efficient computation of equilibria. They also provide a worst-case complexity analysis on the general computation of equilibria in these problems. A good overview of action-graph games and associated methods can be found in [35].

Our Dec-SIMDP model was originally proposed under the designation of *interaction-driven Markov games* [30]. This work also introduced the IDMG algorithm mentioned in Section 4.4, which differs from IDMG and LAPSI in two fundamental aspects. On one hand, outside the interaction areas each agent k uses the optimal policy for the corresponding MDP $\mathcal{M}_k$ in the Dec-SIMDP model. This makes the agents unable to act taking into account the impact that interaction areas visited in the future can have in the current decision. On the other hand, when at interaction area i, each agent k combines the optimal Q-function associated with its individual MDP $\mathcal{M}_k$ with the optimal Q-function associated with the MMDP $\mathcal{M}_i^I$. All agents interacting in that area share the corresponding Q-functions thus obtained and play a matrix game in that state, adopting an equilibrium strategy for that game.

Put simply, both the MPSI and LAPSI algorithms compute the optimal Q-function for a full MMDP associated with the original Dec-SIMDP and use the corresponding policy in the interaction areas. Although the dimension of the full MMDP is typically much larger than that of the MMDPs solved by IDMG, the fact that the latter algorithm must compute several Nash equilibria brings a computational advantage to our method that larger in problems with many states in interaction areas. As an example, the environment in Fig. 4a requires the computation of about 90 Nash equilibria.[9]

Another closely related model is that of distributed POMDPs with coordination locales [32]. In DCPLs, each agent is assumed independent of all other agents except on previously specified *coordination locales*. This work also proposes the TREMOR algorithm for DCPLs: much as in the IDMG algorithm, TREMOR models each agent k using a POMDP model that is solved to yield a policy π_k for that agent. Coordination locales are handled by modifying the POMDP model for each agent taking the policies of the other agents into account.

Finally, several works have analyzed the worst-case complexity of known models for decentralized multiagent systems, including transition-independent Dec-MDPs and reward-independent Dec-MDPs [3, 12]. We summarized several of these results in Fig. 2. Although the worst-case complexity for the proposed Dec-SIMDP model remains an open question, our conjecture is that may be possible to replicate the reasoning in Section 3 and relate the worst-case complexity of Dec-SIMDPs with that of single-agent partially observable models.

More recently, an information-theoretic measure of inter-agent influence has been proposed under the designation of *influence gap* [2]. The influence gap, in a sense, indicates how much the actions of one agent determines the actions of the other agents in the optimal policy, in an attempt to quantify the level of dependence between the different agents. As expected, larger influence gaps – corresponding to smaller inter-agent influence, – typically translate into less computational complexity. This opens an interesting question to be addressed in future work. While the influence gap described above is a measure of *global* inter-agent influence, our interaction areas capture *local* interactions between the agents. However, larger or numerous interaction areas typically lead to "harder" problems. It would be interesting to translate this intuition in terms of the proposed influence gap, as it would certainly provide a more direct way of assessing the general applicability of the Dec-SIMDP model.

6 Conclusion

Both instances of LAPSI and MPSI used in the experimental results rest on having each agent track the other agents in the environment using a belief vector that is then used to choose the actions. The difference between the two algorithms lies in the assumed policy for the other agents. MPSI assumes each of the other agents to be completely absorbed by their "individual goals", thus discarding whatever interaction there may be. In the cases where these interactions are negative, this causes the MPSI agents to act more cautiously. In contrast, the LAPSI agent assumes that the other agents are "team-players", in that they choose their actions for the common goal of the group. This allows the agent to adopt a policy that is closer to the actual optimal fully-observable policy, which indicates that the LAPSI algorithm successfully leverages the particular independences between the different agents to attain efficient and yet near-optimal performance. In MPSI and LAPSI, these "modeling strategies" are used to abstract the decision process of each agent into a single-agent

[9]More precisely, the IDMG method requires the computation of one equilibrium for every state inside or adjacent to the interaction area and one additional equilibrium at every state in the interaction area. In the environment of Fig. 4a this corresponds to having 9×9 equilibria for the states inside and adjacent to the interaction area plus the aditional 3×3 equilibria for the states in the interaction area, leading to a total of 90 equilibria.

decision process – namely, a POMDP. Although we illustrated our methods using a Q_{MDP}-like approach, the same principle can be used with any other POMDP solver.

Also, the ability that both MPSI and LAPSI have to track the other agent allows the planning process to take into consideration the possibility of future interaction. This, as seen in the example in Fig. 6, is an important property of the method that overcomes one important limitation of the IDMG algorithm.

We further note that the differences between MPSI and LAPSI may provide additional information in defining the interaction areas. While MPSI relies on the optimal policies for the individual MDPs in the Dec-SIMDP model, LAPSI relies on the joint policy for the underlying MMDP. Since outside interaction areas we expect the actions of the different agents to be approximately independent, the interactions areas should be those in which the estimated policies using the individual MDPs and the joint MMDP disagree. This provides one recipe for choosing the interaction states as those in which individual state-information is not sufficient to determine the best action. In [25] a similar approach was used to implement decentralized execution of a jointly optimal policy.

Finally, given the similarity between our methods and Q_{MDP}, one would expect our methods to suffer in states of great uncertainty, much like Q_{MDP} does. In these states, action selection may be "conservative" but sparse interactions will hopefully minimize the effects of this situation.

We conclude by noting that it would be interesting to extend the ideas in this paper to more general models such as Dec-POMDPs, alleviating the requirement of full local state observability. In this case, POMDP models could replace the individual MDP models used in this paper, similarly to the approach in [32].

Acknowledgements

The authors would like to acknowledge the numerous helpful discussions, comments and suggestions by Michael Bowling. Also, some of the material in this report has previously been published in the Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems and benefitted from contributions by Matthijs Spaan.

A Proofs

In this appendix we present the proofs of the several theorems along the text.

A.1 Proof of Theorem 4.1

We establish this statement by showing that the generalized α-vectors can be computed using a convergent dynamic-programming like approach, by iterating over the recurrent expression in (12).

We start by noting that, in Dec-SIMDP verifying the conditions of the theorem, a generalized α-vector $\boldsymbol{\alpha}_k$ is actually a $|\mathcal{X}| \times |\mathcal{A}_k|$ matrix with component (x, a_k) given by $\boldsymbol{\alpha}_k(x, a_k)$. Let us define, for a general matrix $|\mathcal{X}| \times |\mathcal{A}_k|$ matrix W, the operator $\mathbf{T}_k$ as follows

$$\begin{aligned}(\mathbf{T}_k W)(x, a_k) = r_{\pi_{-k}}(x, a_k) + \gamma \sum_{y \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} W(y, u_k) \\ + \gamma \max_{u_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) W(y, u_k),\end{aligned}$$

where $(\mathbf{T}_k W)(x, a_k)$ denotes the element (x, a_k) of the matrix $\mathbf{T}_k W$. We now establish the assertion of the theorem by showing $\mathbf{T}_k$ to be a contraction in the sup-norm. In fact, we have

$$\begin{aligned}\|\mathbf{T}_k W_1 - \mathbf{T}_k W_2\|_\infty &= \max_{x, a_k} |(\mathbf{T}_k W_1)(x, a_k) - (\mathbf{T}_k W_2)(x, a_k)| \\ &\leq \gamma \max_{x, a_k} \sum_y \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} |W_1(y, u_k) - W_2(y, u_k)|,\end{aligned}$$

where the last inequality follows from Jensen's inequality. This immediately implies that

$$\begin{aligned}\|\mathbf{T}_k W_1 - \mathbf{T}_k W_2\|_\infty &\leq \gamma \max_{x, a_k} |W_1(x, a_k) - W_2(x, a_k)| \\ &= \gamma \|W_1 - W_2\|_\infty.\end{aligned}$$

We have thus shown that $\mathbf{T}_k$ is a contraction in the sup-norm, which implies that

- It has a unique fixed-point, corresponding to the generalized α-vectors. This establishes the claim of the theorem.
- It can be used to compute the generalized α-vectors in a dynamic-programming-like fashion, using the update rule
$$\boldsymbol{\alpha}_k^{(n+1)}(x, a_k) = (\mathbf{T}_k \boldsymbol{\alpha}_k^{(n)})(x, a_k),$$
where $\boldsymbol{\alpha}_k^{(n)}$ denotes the nth estimate of $\boldsymbol{\alpha}_k(x, a_k)$. □

A.2 Proof of Theorem 4.2

To establish the claim of the theorem, we show that the problem of computing the generalized α-vectors for a Dec-SIMDP verifying the conditions of the theorem is equivalent (in terms of complexity) to that of solving an MDP whose dimension depends polynomially on the dimension of the original Dec-SIMDP. In particular, we show that computing the generalized α-vectors for such Dec-SIMDP is equivalent to computing the optimal Q-function for an MDP. Since MDPs are known to be P-complete [24], this establishes the desired result.

We start by noting that (12) can be rewritten as

$$\begin{aligned}\boldsymbol{\alpha}_k(x, a_k) = r_{\pi_{-k}}(x, a_k) + \gamma \sum_{y \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} \boldsymbol{\alpha}_k(y, u_k) \\ + \gamma \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \max_{u_k} \eta_{xa_k} \sum_{z \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, z) \boldsymbol{\alpha}_k(z, u_k)\end{aligned} \tag{16}$$

where

$$\eta_{xa_k} = \frac{1}{\sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(x, a_k, y)}.$$

We now construct an MDP $\hat{\mathcal{M}} = (\hat{\mathcal{X}}, \mathcal{A}_k, \hat{\mathsf{P}}, \hat{r})$, where $\hat{\mathcal{X}} = \mathcal{X} \cup \mathcal{X} \times \mathcal{A}_k$. We define the transition probabilities for this MDP as

$$\hat{\mathsf{P}}(\hat{x}, a_k, \hat{y}) = \begin{cases} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, \hat{y}) & \text{if } \hat{x} \in \mathcal{X} \text{ and } \hat{y} \in \mathcal{X}_I \\ 1/\eta_{\hat{x}a_k} & \text{if } \hat{x} \in \mathcal{X} \text{ and } \hat{y} = (\hat{x}, a_k) \\ \eta_{zu_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(z, u_k, y) \mathsf{P}_{\pi_{-k}}(y, a_k, \hat{y}) & \text{if } \hat{x} = (z, u_k) \text{ and } \hat{y} \in \mathcal{X}_I \\ \eta_{zu_k} \mathsf{P}_{\pi_{-k}}(z, u_k, y)/\eta_{ya_k} & \text{if } \hat{x} = (z, u_k),\ \hat{y} = (y, a_k), \\ & \qquad \text{and } y \notin \mathcal{X}_I \\ 0 & \text{otherwise.} \end{cases}$$

These probabilities are well-defined since, for $\hat{x} \in \mathcal{X}$,

$$\begin{aligned} \sum_{\hat{y}} \hat{\mathsf{P}}(\hat{x}, a_k, \hat{y}) &= \sum_{\hat{y} \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, \hat{y}) + 1/\eta_{\hat{x}a_k} \\ &= \sum_{\hat{y} \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, \hat{y}) + \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, y) = 1 \end{aligned}$$

and, for $\hat{x} = (z, u_k)$

$$\begin{aligned} \sum_{\hat{y}} \hat{\mathsf{P}}(\hat{x}, a_k, \hat{y}) &= \eta_{zu_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(z, u_k, y) \Big(\sum_{\hat{y} \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(y, a_k, \hat{y}) + 1/\eta_{ya_k} \Big) \\ &= \eta_{zu_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(z, u_k, y) = 1. \end{aligned}$$

Similarly, we define the reward function for this MDP as

$$\hat{r}(\hat{x}, a_k) = \begin{cases} r_{\pi_{-k}}(\hat{x}, a_k) & \text{if } \hat{x} \in \mathcal{X} \\ \eta_{zu_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(z, u_k, y) r_{\pi_{-k}}(y, a_k) & \text{if } \hat{x} = (z, u_k). \end{cases}$$

The optimal Q-function for this MDP verifies the recursive relation (2), that we repeat here for commodity:

$$Q^*(\hat{x}, a_k) = \hat{r}(\hat{x}, a_k) + \gamma \sum_{\hat{y} \in \hat{\mathcal{X}}} \hat{\mathsf{P}}(\hat{x}, a_k, \hat{y}) \max_{u_k} Q^*(\hat{y}, u_k).$$

Now, for $\hat{x} \in \mathcal{X}$ and replacing the definitions of $\hat{\mathsf{P}}$ and $\hat{r}$, we have

$$\begin{aligned} Q^*(\hat{x}, a_k) = r_{\pi_{-k}}(\hat{x}, a_k) &+ \gamma \sum_{\hat{y} \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, \hat{y}) \max_{u_k} Q^*(\hat{y}, u_k) \\ &+ \frac{\gamma}{\eta_{\hat{x}a_k}} \max_{u_k} Q^*((\hat{x}, a_k), u_k). \end{aligned}$$

Similarly, for $\hat{x} = (z, u_k)$, we have

$$\begin{aligned} Q^*(\hat{x}, a_k) &= \eta_{zu_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(z, u_k, y) \Bigg[r_{\pi_{-k}}(y, a_k) + \gamma \sum_{\hat{y} \in \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(y, a_k, \hat{y}) \max_{u_k} Q^*(\hat{y}, u_k) \\ &\qquad\qquad + \frac{\gamma}{\eta_{ya_k}} \max_{u_k} Q^*((y, a_k), u_k) \Bigg] \\ &= \eta_{zu_k} \sum_{y \notin \mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(z, u_k, y) Q^*(y, a_k). \end{aligned}$$

Replacing in the previous expression finally yields

$$\begin{aligned} Q^*(\hat{x}, a_k) = r_{\pi_{-k}}(\hat{x}, a_k) + \gamma \sum_{\hat{y}\in\mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, \hat{y}) \max_{u_k} Q^*(\hat{y}, u_k) \\ + \frac{\gamma}{\eta_{\hat{x}a_k}} \max_{u_k} \eta_{\hat{x}a_k} \sum_{y\notin\mathcal{X}_I} \mathsf{P}_{\pi_{-k}}(\hat{x}, a_k, y) Q^*(y, u_k), \end{aligned}$$

and this is (16). As such, in computing the optimal Q-function for the MDP $\hat{\mathcal{M}}$, we compute the generalized α-vectors for the original Dec-SIMDP as

$$\boldsymbol{\alpha}_k(x, a_k) = Q^*(x, a_k).$$

Since the dimension of the new MDP grows polynomially (linearly, actually) with the dimension of the generalized α-vectors (and, hence, with the dimension of the corresponding Dec-SIMDP), the statement of the theorem follows. $\square$

A.3 Proof of Theorem 4.3

It is well-known that, for a general MDP $\mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, if $\hat{\pi}$ is the greedy policy with respect to a function $\hat{Q}$, *i.e.*, if

$$\hat{\pi}(x) = \arg\max_{a\in\mathcal{A}} \hat{Q}(x, a)$$

for all $x \in \mathcal{X}$, then

$$\left\| V^{\hat{\pi}} - V^* \right\|_\infty \leq \frac{2\gamma}{1-\gamma} \mathbf{BE}(\hat{Q}), \tag{17}$$

where $\mathbf{BE}(\hat{Q})$ is the Bellman error associated with the function $\hat{Q}$,

$$\mathbf{BE}(\hat{Q}) = \sup_{x,a} \left| r(x, a) + \gamma \sum_y \mathsf{P}(x, a, y) \max_u \hat{Q}(y, u) - \hat{Q}(x, a) \right|.$$

In our Dec-SIMDP setting, since we are assuming the policy π_{-k} to be fixed and known, the decision-process (from agent k's perspective) is a standard POMDP. Recalling that a POMDP can be recast as an equivalent belief-MDP, it follows that the relation (17) also holds for POMDPs. Writing down the Bellman error for a general POMDP $(\mathcal{X}, \mathcal{A}, \mathcal{Z}, \mathsf{P}, \mathsf{O}, r, \gamma)$ thus yields

$$\begin{aligned} &\mathbf{BE}(\hat{Q}) \\ &= \sup_{\mathbf{b},a} \left| \sum_x \mathbf{b}_x \left[r(x, a) + \gamma \sum_{z,y} \mathsf{P}(x, a, y) \mathsf{O}(y, a, z) \max_u \hat{Q}(\mathbf{b}'_{za}, u) \right] - \hat{Q}(\mathbf{b}, a_k) \right|. \end{aligned}$$

For simplicity of notation, we consider the *Bellman error at* $(\mathbf{b}, a)$ to be

$$\begin{aligned} &\mathbf{BE}(\hat{Q}, \mathbf{b}, a) \\ &= \left| \sum_x \mathbf{b}_x \left[r(x, a) + \gamma \sum_{z,y} \mathsf{P}(x, a, y) \mathsf{O}(y, a, z) \max_u \hat{Q}(\mathbf{b}'_{za}, u) \right] - \hat{Q}(\mathbf{b}, a) \right|. \end{aligned}$$

For the POMDP as perceived by agent k in our Dec-SIMDP setting, we have

$$\begin{aligned}&\mathbf{BE}(\hat{Q}, \mathbf{b}, a_k)\\ &= \left|\sum_{x_{-k}} \mathbf{b}_{x_{-k}} \left[r_{\pi_{-k}}(x, a_k) + \gamma \sum_{z,y} \mathsf{P}_{\pi_{-k}}(x, a_k, y) \mathsf{O}(y, z) \max_u \hat{Q}(\mathbf{b}'_{za_k}, u) \right] - \hat{Q}(\mathbf{b}, a_k)\right|\end{aligned}$$

which, replacing the definitions of $\hat{Q}(\mathbf{b}, a_k)$ and the generalized α-vectors yields, after some shuffling,

$$\begin{aligned}&\mathbf{BE}(\hat{Q}, \mathbf{b}, a_k)\\ &= \gamma \Bigg| \sum_{x_O, y_O} \mathbf{b}_{x_O} \mathsf{P}_{\pi_{-k}}(x_O, a_k, y_O) \max_{u_k} \sum_{x_{-O}, y_{-O}} \mathbf{b}_{x_{-O}} \mathsf{P}_{\pi_{-k}}(x_{-O}, y_{-O}) \boldsymbol{\alpha}_k(y, u_k)\\ &\qquad - \sum_{x, y_O} b_{x_O} \mathbf{b}_{x_{-O}} \mathsf{P}_{\pi_{-k}}(x_O, a_k, y_O) \max_{u_k} \sum_{y_{-O}} \mathsf{P}_{\pi_{-k}}(x_{-O}, y_{-O}) \boldsymbol{\alpha}_k(y, u_k) \Bigg|,\end{aligned}$$

where we have used the notation introduced in Section 4.2. Letting

$$\Lambda_k(x_{-O}, y_O, u_k) = \sum_{y_{-O}} \mathsf{P}_{\pi_{-k}}(x_{-O}, y_{-O}) \boldsymbol{\alpha}_k(y, u_k),$$

we have

$$\mathbf{BE}(\hat{Q}, \mathbf{b}, a_k) \leq \gamma \max_{y_O} \left| \max_{u_k} \sum_{x_{-O}} \mathbf{b}_{x_{-O}} \Lambda(x_{-O}, y_O, u_k) - \sum_{x_{-O}} \mathbf{b}_{x_{-O}} \max_{u_k} \Lambda(x_{-O}, y_O, u_k) \right|.$$

In order to bound the right-hand side of the expression above, we need two auxiliary results that we promptly introduce. These results generalize some of the bounds in [27] to the case of functions defined over $\mathbb{R}^n$ and are of independent interest *per se*.

Lemma A.1. *Let $\{x_k, k = 1, \ldots, M\}$ be a set of points in $\mathbb{R}^n$, for some (finite) n, and $\{\beta_k, k = 1, \ldots, M\}$ a set of corresponding weights, verifying $0 \leq \beta_k \leq 1, k = 1, \ldots, M$ and $\sum_k \beta_k = 1$. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function. Then, it holds that*

$$\sum_k \beta_k f(x_k) - f\left(\sum_k \beta_k x_k\right) \leq \beta^* \left[\sum_k f(x_k) - Mf\left(\sum_k x_k/M\right)\right]. \tag{18}$$

where

$$\beta^* = \max_k \beta_k.$$

Proof. The proof essentially follows that of Lemma 1 in [27]. Let k^* be such that $\beta^* = \beta_{k^*}$. Eq. (18) can be rewritten as

$$\sum_{k \neq k^*} (\beta^* - \beta_k) f(x_k) + f\left(\sum_k \beta_k x_k\right) \geq \beta^* Mf\left(\sum_k x_k/M\right)$$

or, equivalently,

$$\sum_{k \neq k^*} \frac{\beta^* - \beta_k}{M\beta^*} f(x_k) + \frac{1}{M\beta^*} f\left(\sum_k \beta_k x_k\right) \geq f\left(\sum_k x_k/M\right).$$

Noting that

$$\sum_{k\neq k^*} \frac{\beta^* - \beta_k}{M\beta^*} + \frac{1}{M\beta^*} = 1,$$

we finally get

$$\begin{aligned}\sum_{k\neq k^*} \frac{\beta^* - \beta_k}{M\beta^*} f(x_k) + \frac{1}{M\beta^*} f\left(\sum_k \beta_k x_k\right) &\geq f\left(\sum_{k\neq k^*} \frac{\beta^* - \beta_k}{M\beta^*} x_k + \frac{1}{M\beta^*}\sum_k \beta_k x_k\right)\\ &= f\left(\sum_k x_k/M\right),\end{aligned}$$

where the first inequality follows from Jensen's inequality. This implies (18). □

Lemma A.1 implies the following corollary.

Corollary A.2. *Let $\{x_i, i = 1, \dots, N\}$ be a set of points in $\mathbb{R}^n$, for some (finite) n, and $\{p_i, i = 1, \dots, N\}$ a corresponding sequence of weights verifying $0 \leq p_i \leq 1$ and $\sum_i p_i = 1$. Let U denote a closed convex set that can be represented as the convex hull of a set of points $\{a_k, k = 1, \dots, M\}$ in $\mathbb{R}^n$. Let $f : U \to \mathbb{R}$ be a convex function. Then, it holds that*

$$\sum_i p_i f(x_i) - f\left(\sum_i p_i x_i\right) \leq \sum_k f(a_k) - Mf\left(\sum_k a_k/M\right). \tag{19}$$

Proof. We start by noting that each x_i can be written as

$$x_i = \sum_k \lambda_{ik} a_k,$$

with $0 \leq \lambda_{ik} \leq 1$ and $\sum_k \lambda_{ik} = 1, i = 1, \dots, n$. Then,

$$\begin{aligned}\sum_i p_i f(x_i) - f\left(\sum_i p_i x_i\right) &= \sum_i p_i f\left(\sum_k \lambda_{ik} a_k\right) - f\left(\sum_i p_i \sum_k \lambda_{ik} a_k\right)\\ &\leq \sum_k f(a_k) \sum_i p_i \lambda_{ik} - f\left(\sum_k a_k \sum_i p_i \lambda_{ik}\right).\end{aligned}$$

Letting $\beta_k = \sum_i p_i \lambda_{ik}$, we get

$$\sum_i p_i f(x_i) - f\left(\sum_i p_i x_i\right) \leq \sum_k \beta_k f(a_k) - f\left(\sum_k \beta_k a_k\right).$$

Finally, applying Lemma A.1,

$$\begin{aligned}\sum_i p_i f(x_i) - f\left(\sum_i p_i x_i\right) &\leq \sum_k \beta_k f(a_k) - f\left(\sum_k \beta_k a_k\right)\\ &\leq \beta^* \left[\sum_k f(a_k) - Mf\left(\sum_k a_k/M\right)\right]\\ &\leq \sum_k f(a_k) - Mf\left(\sum_k a_k/M\right)\end{aligned}$$

and the proof is complete. □

Note the difference between the two bounds: while (18) depends on the function f and the set of points $\{x_k, k = 1, \dots, M\}$, (19) depends only on the function f and on the set U.

We now have that, for any given $u_k^* \in \mathcal{A}_k$, $x_{-O}^* \in \mathcal{X}_{-O}$ and $y_O^* \in \mathcal{X}_O$, $\Lambda_k(x_{-O}^*, y_O^*, u_k^*)$ lies in the convex hull of the set of alpha-vectors $\boldsymbol{\alpha}_k(y, u_k^*)$ where $y_O = y_O^*$. Then, from Corollary A.2,

$$\mathbf{BE}(\hat{Q}, \mathbf{b}, a_k) \leq \gamma \max_{y_O} \left| \max_{u_k} \sum_{y_{-O}} \boldsymbol{\alpha}_k((y_O, y_{-O}), u_k) - \sum_{y_{-O}} \max_{u_k} \boldsymbol{\alpha}_k((y_O, y_{-O}), u_k) \right|.$$

This finally yields

$$\|V^* - V^{\pi_k}\|_\infty \leq \frac{2\gamma^2}{1-\gamma} \max_{y_O} \left| \max_{u_k} \sum_{y_{-O}} \boldsymbol{\alpha}_k((y_O, y_{-O}), u_k) - \sum_{y_{-O}} \max_{u_k} \boldsymbol{\alpha}_k((y_O, y_{-O}), u_k) \right|,$$

and the proof is complete. □

References

[1] D. Aberdeen. A (revised) survey of approximate methods for solving partially observable Markov decision processes. Technical report, National ICT Australia, 2003.

[2] M. Allen. *Interactions in decentralized environments*. PhD thesis, Univ. Massachusetts, Amherst, 2009.

[3] M. Allen and S. Zilberstein. Complexity of decentralized control: Special cases. In *Adv. Neural Information Proc. Systems*, volume 22, pages 19–27, 2009.

[4] R. Becker, S. Zilberstein, V. Lesser, and C. Goldman. Transition-independent decentralized Markov decision processes. In *Proc. 2nd Int. Conf. Autonomous Agents and Multiagent Systems*, pages 41–48, 2003.

[5] R. Becker, S. Zilberstein, V. Lesser, and C. Goldman. Solving transition independent decentralized Markov decision processes. *J. Artificial Intelligence Research*, 22:423–455, 2004.

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

[7] N. Bhat and K. Leyton-Brown. Computing Nash equilibria of action-graph games. In *Proc. 20th Conf. Uncertainty in Artificial Intelligence*, pages 35–42, 2004.

[8] C. Boutilier. Planning, learning and coordination in multiagent decision processes. In *Theoretical Aspects of Rationality and Knowledge*, 1996.

[9] A. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Dept. Computer Sciences, Brown Univ., 1998.

[10] M. Ghavamzadeh, S. Mahadevan, and R. Makar. Hierarchical multiagent reinforcement learning. *J. Autonomous Agents and Multiagent Systems*, 13(2):197–229, 2006.

[11] P. Gmytrasiewicz and P. Doshi. A framework for sequential planning in multiagent settings. *J. Artificial Intelligence Research*, 24:49–79, 2005.

[12] C. Goldman and S. Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *J. Artificial Intelligence Research*, 22:143–174, 2004.

[13] J. Goldsmith and M. Mundhenk. Competition adds complexity. In *Adv. Neural Information Proc. Systems*, volume 20, pages 561–568, 2008.

[14] C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored MDPs. In *Adv. Neural Information Proc. Systems*, volume 14, pages 1523–1530, 2001.

[15] M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In *Proc. 17th Conf. Uncertainty in Artificial Intelligence*, pages 253–260, 2001.

[16] J. Kok and N. Vlassis. Sparse cooperative Q-learning. In *Proc. 21st Int. Conf. Machine Learning*, pages 61–68, 2004.

[17] J. Kok, P. Hoen, B. Bakker, and N. Vlassis. Utile coordination: Learning interdependencies among cooperative agents. In *IEEE Symp. Computational Intelligence and Games*, pages 61–68, 2005.

[18] M. Littman, A. Cassandra, and L. Kaelbling. Learning policies for partially observable environments: Scaling up. In *Proc. 12th Int. Conf. Machine Learning*, pages 362–370, 1995.

[19] O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *Proc. 16th AAAI Conf. Artificial Intelligence*, 1999.

[20] R. Makar and S. Mahadevan. Hierarchical multiagent reinforcement learning. In *Proc. 5th Int. Conf. Autonomous agents*, pages 246–253, 2001.

[21] F. Melo and M.I. Ribeiro. Transition entropy in partially observable Markov decision processes. In *Proc. 9th Int. Conf. Intelligent Autonomous Systems*, pages 282–289, 2006.

[22] F. Melo and M. Veloso. Learning of coordination: Exploiting sparse interactions in multiagent systems. In *Proc. 8th Int. Conf. Autonomous Agents and Multiagent Systems*, pages 773–780, 2009.

[23] N. Meuleau, M. Hauskrecht, K. Kim, L. Peshkin, L. Kaelbling, T. Dean, and C. Boutilier. Solving very large weakly coupled Markov decision processes. In *Proc. 15th AAAI Conf. Artificial Intelligence*, pages 165–172, 1998.

[24] C. Papadimitriou and J. Tsitsiklis. The complexity of Markov decision processses. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[25] M. Roth, R. Simmons, and M. Veloso. Exploiting factored representations for decentralized execution in multiagent teams. In *Proc. 6th Int. Conf. Autonomous Agents and Multiagent Systems*, pages 469–475, 2007.

[26] L. Shapley. Stochastic games. *Proc. National Academy of Sciences*, 39:1095–1100, 1953.

[27] S. Simic. On a global upper bound for Jensen's inequality. *J. Mathematical Analysis and Applications*, 343:414–419, 2008.

[28] S. Singh and D. Cohn. How to dynamically merge Markov decision processes. In *Adv. Neural Information Proc. Systems*, volume 10, pages 1057–1063, 1998.

[29] R. Smallwood and E. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21(5):1071–1088, 1973.

[30] M. Spaan and F. Melo. Interaction-driven Markov games for decentralized multiagent planning under uncertainty. In *Proc. 7th Int. Conf. Autonomous Agents and Multiagent Systems*, pages 525–532, 2008.

[31] P. Stone and M. Veloso. Team-partitioned, opaque-transition reinforcement learning. In *Proc. RoboCup-98*, pages 206–212, 1998.

[32] P. Varakantham, J. Kwak, M. Taylor, J. Marecki, P. Scerri, and M. Tambe. Exploiting coordination locales in distributed POMDPs via social model shaping. In *Proc. 19th Int. Conf. Automated Planning and Scheduling*, pages 313–320, 2009.

[33] A. Xin Jiang and K. Leyton-Brown. A polynomial-time algorithm for action-graph games. In *Proc. 21st AAAI Conf. Artificial Intelligence*, pages 679–684, 2006.

[34] A. Xin Jiang and K. Leyton-Brown. Computing pure Nash equilibria in symmetric action-graph games. In *Proc. 22nd AAAI Conf. Artificial Intelligence*, pages 79–85, 2007.

[35] A. Xin Jiang, K. Leyton-Brown, and N. Bhat. Action-graph games. Technical Report TR-2008-13, Univ. British Columbia, 2008.