Interaction-Driven Markov Games for Decentralized Multiagent Planning under Uncertainty

Matthijs T.J. Spaan
Institute for Systems and Robotics
Instituto Superior Técnico
Av. Rovisco Pais, 1, 1049-001
Lisbon, Portugal
mtjspaan@isr.ist.utl.pt

Francisco S. Melo*
Institute for Systems and Robotics
Instituto Superior Técnico
Av. Rovisco Pais, 1, 1049-001
Lisbon, Portugal
fmelo@isr.ist.utl.pt

ABSTRACT

In this paper we propose *interaction-driven Markov games* (IDMGs), a new model for multiagent decision making under uncertainty. IDMGs aim at describing multiagent decision problems in which interaction among agents is a *local phenomenon*. To this purpose, we explicitly distinguish between situations in which agents should interact and situations in which they can afford to act independently. The agents are coupled through the joint rewards and joint transitions in the states in which they interact. The model combines several fundamental properties from transition-independent Dec-MDPs and weakly coupled MDPs while allowing to address, in several aspects, more general problems. We introduce a fast approximate solution method for planning in IDMGs, exploiting their particular structure, and we illustrate its successful application on several large multiagent tasks.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Theory

Keywords

Planning under uncertainty, cooperative multiagent systems, team Markov games.

1. INTRODUCTION

Decision making under uncertainty is an important skill for any intelligent agent. In this paper we consider situations in which a group of independent intelligent agents co-exist in a common environment. Each agent has been assigned its own task which it must fulfill independently of the remaining agents' tasks. However, as the agents inhabit the same environment, it may happen that two or more agents should *interact* in order for each of them to complete its task. For example, multiple agents might need to share some common resource, such as space. When this interaction is necessary, we assume it to be *local*. A set of robots which all have to navigate in the same physical environment form a prime example of the type of problems we are considering. Although each robot has its own destination which it can reach independently, it is beneficial for them to reason about interactions such as potential collisions in tight corridors. We are interested in addressing these types of decision problems while assuming minimum communication.

Markov decision processes (MDPs) [4] are a popular framework for sequential decision making for single agents, when an agent's actions have a stochastic effect on the state of the environment. For the cooperative multiagent case, team Markov games (also known as multiagent MDPs [5]) arise as a natural model to address such sequential decision problems, and several methods have been proposed to address decision making for this model. However, the vast majority of such methods takes advantage of the underlying assumption of *continuous implicit communication*, as each agent is aware of the complete state as well as the actions played by all agents at all time steps. This is, indeed, an appealing feature of these models, which however is seldom verified in practice. Many actual problems require richer models that include *partial observability* in both state and action perception.

When considering partial state observability and lack of joint-action observability, the literature provides an abundance of models that include partially observable stochastic games [11], decentralized MDPs and POMDPs [3], or interactive POMDPs [8]. However, when considering partial observability of states and actions, the optimal decision-making problem significantly hardens. As shown in [3], even in the case where all agents have the same payoff function (Dec-MDPs), the problem of acting optimally is provably undecidable. As such, to address this class of problems simplified models such transition independent Dec-MDPs [1] as well as approximate algorithms [7, 19] have been proposed.

In this paper we propose a new model for decentralized decision-making problems that takes advantage of local interactions between the agents. Our model is dubbed *interaction-driven Markov game* (IDMG) and its main feature is the distinction between those states where interaction *occurs* and those where interaction can be ignored. In the former family of states, we make explicit use of communication be-

*Since January 2008, Francisco S. Melo is with the School of Computer Science, Carnegie Mellon University.

Cite as: Interaction-Driven Markov Games for Decentralized Multiagent Planning under Uncertainty, Spaan and Melo, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

tween agents. In the remaining states, we take each agent as independent, reverting to a single-agent decision problem.

IDMGs bring several important advantages over standard Markov games and other multiagent planning approaches such as Dec-MDPs. Namely, IDMGs alleviate assumptions on state and joint-action observability and continuous communication usually adopted (either implicitly or explicitly) in Markov game models, without suffering from the large computational burden of Dec-MDPs. Furthermore, IDMGs provide more realistic models for many practical multiagent problems as they explicitly consider local interaction and communication; they thus capture important features inherent to many practical multiagent decision problems. We introduce a fast approximate solution method for planning in IDMGs, exploiting their particular structure, and we illustrate its successful application on several multiagent tasks.

2. BACKGROUND

We start by introducing MDPs as a model for single-agent sequential decision making, and Markov games for multiagent decision making.

2.1 Markov decision processes

Let $\{X(t)\}$ a $\mathcal{X}$-valued controlled Markov chain, where $\mathcal{X}$ is some finite set. The transition probabilities are given by

$$\mathbb{P}\left[X(t+1) = y \mid X(t) = x, A(t) = a\right] = \mathsf{P}^a(x, y).\ \forall x, y \in \mathcal{X} \tag{2.1}$$

The $\mathcal{A}$-valued process $\{A(t)\}$ represents the control process: $A(t)$ is the control action at time instant t and $\mathcal{A}$ is the finite set of possible actions. A decision maker must determine the control process $\{A(t)\}$ so as to maximize the functional

$$V(\{A(t)\}, x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R\big(X(t), A(t)\big) \mid X(0) = x\right], \tag{2.2}$$

where $0 \leq \gamma < 1$ is a discount factor and $R(x, a)$ represents a random "reward" received for taking action $a \in \mathcal{A}$ in state $x \in \mathcal{X}$. We assume throughout this paper that there is a deterministic real function r defined on $\mathcal{X} \times \mathcal{A} \times \mathcal{X}$ assigning a reward $r(x, a, y)$ every time a transition from x to y occurs after taking action a and that $\mathbb{E}\left[R(x, a)\right] = \sum_{y \in \mathcal{X}} \mathsf{P}^a(x, y) r(x, a, y)$. We refer to the tuple $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$ as a *Markov decision process* [4].

Given an MDP $(\mathcal{X}, \mathcal{A}, \mathsf{P}, r, \gamma)$, the *optimal value function* V^* is defined for each state $x \in \mathcal{X}$ as

$$V^*(x) = \max_{\{A(t)\}} \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^t R\big(X(t), A(t)\big) \mid X(0) = x\right] \tag{2.3}$$

and verifies the Bellman optimality equation [4]. The optimal Q-values $Q^*(x, a)$ are defined for each state-action pair $(x, a) \in \mathcal{X} \times \mathcal{A}$ as

$$Q^*(x, a) = \sum_{y \in \mathcal{X}} \mathsf{P}^a(x, y)\big[r(x, a, y) + \gamma V^*(y)\big]. \tag{2.4}$$

From Q^* we can define the *optimal policy* for the MDP as $\pi^*(x) = \text{argmax}_a Q^*(x, a)$. Dynamic programming techniques such as value iteration [4] can be readily applied to compute Q^*, and hence π^*.

In this section, we review the framework of Markov games, a cornerstone for the developments in this paper.

2.2 Markov games

An N-agent Markov game or stochastic game is a tuple $\big(N, \mathcal{X}, (\mathcal{A}_k), \mathsf{P}, (r_k), \gamma\big)$, where $\mathcal{X}$ is the state space, $\mathcal{A} = \times_{k=1}^{N} \mathcal{A}_k$ is the finite set of joint actions, P denotes the transition probabilities, r_k is the reward function for agent $k, k = 1, \ldots, N$, and γ is the discount factor. Notice that the differences between a Markov game and an MDP lie, first of all, on the fact that the action space $\mathcal{A}$ of the former is the Cartesian product of the N individual action spaces $\mathcal{A}_k$, and corresponds to the *joint-action space*. This means that, in a Markov game, the transition probabilities depend on the actions of *all* agents. Furthermore, unlike MDPs, the reward function in Markov games may differ from one agent to the other. We say that a Markov game is *fully cooperative* if all agents share the same reward function. In this case, the model is equivalent to a multiagent MDP (MMDP) [5].

The purpose of decision maker k is to determine the individual control process $\{A_k(t)\}$ maximizing the functional

$$V_k(\{A(t)\}, x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_k\big(X(t), A(t)\big) \mid X(0) = x\right], \tag{2.5}$$

where $0 \leq \gamma < 1$ and $R_k(x, a)$ represents the random reward received by agent k when all agents take action $a \in \mathcal{A}$ in state $x \in \mathcal{X}$. It is important to emphasize that an *individual policy* in a Markov game is a mapping $\pi_k(t)$ defined over $\mathcal{X} \times \mathcal{A}_k$ for each t and such that the process $\{A_k(t)\}$ generated by $\pi_k(t)$ verifies $\mathbb{P}\left[A_k(t) = a_k \mid X(t) = x\right] = \pi_k(t; x, a^k)$, with $a^k \in \mathcal{A}^k$. An individual policy $\pi_k(t)$ is a *pure policy* if, for each t and each $x \in \mathcal{X}$, $\pi_k(t; x, a_k) = 1$ for some action in $\mathcal{A}_k$ and is a *mixed policy* otherwise. A *stationary* individual policy is a policy that does not depend on t. In the remainder of the paper, we only consider stationary policies. A joint policy is a vector $\pi = (\pi_1, \ldots, \pi_N)$ of individual policies, and is stationary if each π_k is stationary. We refer to π_{-k} as a *reduced policy*, obtained from π by removing the individual policy of agent k. Finally, we write $V_k^{\pi}(x)$ instead of $V_k(\{A(t)\}, x)$ if $\{A(t)\}$ is generated from policy π.

An individual policy π_k^* is a *best response* of agent k to a reduced policy π_{-k} if agent k cannot improve its expected value by using any other individual policy π_k, *i.e.*,

$$V_k^{(\pi_{-k}, \pi_k^*)}(x) \geq V_k^{(\pi_{-k}, \pi_k)}(x),\ \forall x \in \mathcal{X}. \tag{2.6}$$

A *Nash equilibrium* is a policy profile $\pi^* = (\pi_1^*, \ldots, \pi_N^*)$ in which each individual policy π_k^* is a best response to the reduced policy π_{-k}^*. Every finite Markov game has at least one Nash equilibrium. If the Markov game is fully cooperative, *i.e.*, $r_1 = \ldots = r_N$, there is always a *pure* Nash equilibrium that yields maximum payoff for all agents (a *Pareto optimal* equilibrium). We refer to a joint policy as being *optimal* if it is a Pareto optimal Nash equilibrium.

3. COOPERATIVE INTERACTION

As discussed in the introduction, we are interested in addressing multiagent decision problems in which the interaction between the different agents exhibits a strong *locality*. In this section, we introduce and motivate the IDMG framework for sequential decision making. This model differs from standard Markov game models in a fundamental way: no joint-state or joint-action observability is assumed. We do admit *local* state observability, which makes our model closer to Dec-MDPs than to more general models such as

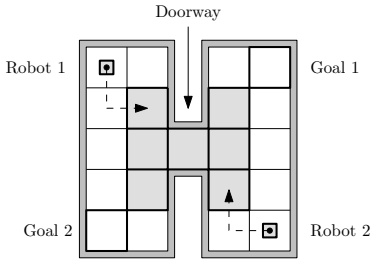

Figure 1: Illustration of the H-shaped example environment. The bottom-left state is the goal location for robot 2, and the top-right state is robot 1's goal. The shaded states correspond to those in which interaction may be required.

Dec-POMDPs or I-POMDPs. Throughout the paper we illustrate our ideas with a simple problem that we introduce next.

3.1 Local interaction in multiagent settings

Consider a problem in which two mobile robots have to navigate in a common environment, each trying to reach its own goal state, as depicted in Fig. 1. Each robot has 4 possible actions (namely "Move North", "Move South", "Move East" and "Move West") that we represent succinctly as N, S, E and W. Each action moves the robot in the corresponding direction with a 0.8 probability and, with a 0.2 probability, it has an unexpected outcome and moves the robot in any of the other directions. The actions of one robot do not affect the movement of the other.

It is possible to "separate" this problem in two distinct problems, in each of which one robot must reach its own goal state and completely disregards the existence of the other robot. This separation is possible since the task of one robot can be completed "independently" of the task of the other. Each robot can thus be modeled independently as an MDP and there is no need to consider interaction between the two robots. As such, referring the two robots as robot 1 and robot 2, we could model our multiagent problem as a pair $(\mathcal{M}_1, \mathcal{M}_2)$. Each $\mathcal{M}_k$ is an MDP $\mathcal{M}_k = (\mathcal{X}_k, \mathcal{A}_k, \mathsf{P}_k, r_k, \gamma)$ where

- $\mathcal{X}_k = \{1, \ldots, 21\}$ represents the possible states for robot k;
- $\mathcal{A}_k = \{N, S, E, W\}$ represents the possible actions for robot k;
- P_k represents the transition probabilities for robot k, conditioned on the robot's action as described above.
- r_k represents the reward function for robot k. For example, we could set $r_k(x, a, y) = 10$ if $y = \text{GOAL}_i$ and 0 otherwise;
- Finally, γ is a discount factor.[1]

Each such MDP can be solved separately as mentioned in Section 2.1, resulting in a $Q_k^*(x_k, a_k)$ function for each robot.

[1]For consistency, we assume both robots to assign the same "importance" to the future rewards and hence the common discount factor.

However, there may be situations where interaction is mandatory. This is the case either if the transition probabilities or the rewards (or both) for at least one agent depend on the actions of the other agents. In this situation there is an implicit interaction between the agents, since the action choice of each agent may influence the dynamics and/or rewards of all other agents. A typical example is when multiple agents attempt to simultaneously use one common resource such as space, tools, raw materials or shared communication channels, which requires them to interact.

In our work, we are interested in considering situations in which this interaction is *local*. This means that each agent can generally choose its actions irrespectively of the other agents' states/actions but, in some specific situations, the agents will mutually influence each others' transitions and/or rewards. For example, considering once again the scenario above, it may be undesirable that both robots cross the doorway simultaneously, as the robots may crash and get damaged. As such, a negative reward may be issued to both robots if they both simultaneously end up in the doorway. This means that the robots should coordinate when both are close to the doorway (darker states in Fig. 1).

We will now formalize this model and describe some of its important aspects.

3.2 Interaction-driven Markov games

For simplicity of presentation, we consider a 2-agent scenario, remarking however that the formalism easily generalizes to any number of agents. As before, let $\mathcal{M}_1 = (\mathcal{X}_1, \mathcal{A}_1, \mathsf{P}_1, r_1, \gamma)$ be an MDP modeling agent 1 and its individual task. Similarly, let $\mathcal{M}_2 = (\mathcal{X}_2, \mathcal{A}_2, \mathsf{P}_2, r_2, \gamma)$ describe agent 2 and its individual task. Let $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2$.

Definition 1. *We define an* interaction-driven Markov game *(IDMG) as a tuple* $(\mathcal{M}_1, \mathcal{M}_2, \{{}^i\mathcal{M}_I, i = 1, \ldots, n\})$, *where* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *are as defined above and each* ${}^i\mathcal{M}_I$ *is an* interaction game*, i.e., a two-agent, fully cooperative Markov game* ${}^i\mathcal{M}_I = ({}^i\mathcal{X}_I, \mathcal{A}, {}^i\mathsf{P}_I, {}^ir_I)$ *where*

- ${}^i\mathcal{X}_I$ *is a set of* interaction states. *This is a subset of* $\mathcal{X}$ *and consists of a set of* adjacent joint states *(in terms of transition probabilities) where interaction should occur (in the form of coordination);*
- $\mathcal{A}$ *is the set of joint actions, i.e.,* $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$*;*
- ${}^i\mathsf{P}_I$ *is a transition probability matrix where*

$$ {}^i\mathsf{P}_I^a(x, y) = \mathbb{P}\left[X(t+1) = y \mid X(t) = x, A(t) = a\right], $$

for $x \in {}^i\mathcal{X}_I$*,* $y \in \mathcal{X}$ *and* $a \in \mathcal{A}$*;*

- ${}^ir_I : \mathcal{X}_I^i \times \mathcal{A} \times \mathcal{X} \to \mathbb{R}$ *is the* joint interaction reward *that the agents receive at each interaction state for each joint action.*

We note that each interaction game ${}^i\mathcal{M}$ describes a "situation" where the agents should coordinate. In the corresponding interaction states, ${}^i\mathcal{X}_I$, each agent explicitly communicates all information useful to the decision process, and we assume that communication is unlimited and noise-free. Also, at these states, the rewards depend on the *joint state* and on the *joint action* of the agents, as prescribed by the functions ${}^ir_I$. As such, interaction states are defined as parts of the joint state space where interaction may lead to an *improvement of the joint overall performance.* For example, in

the navigation example of Fig. 1, there would be a single $\mathcal{M}_I$ component, whose states would correspond to the joint states in which *both robots* are in the shaded area. In general, there will be several distinct ${}^i\mathcal{M}_I$ components.

To clarify the evolution of an IDMG, suppose that, at time t, agent 1 is at state $X_1(t) = x_1$ and agent 2 is at state $X_2(t) = x_2$. Each agent k chooses an action $A_k(t)$ following some underlying decision rule. Then, given the joint action $A(t) = (A_1(t), A_2(t))$, if the joint state $x = (x_1, x_2)$ is an interaction state, *i.e.*, if $x \in {}^i\mathcal{X}_I$ for some i, the agents move at time $t+1$ to state $y = (y_1, y_2)$ according to the probabilities in ${}^i\mathsf{P}_I^{A(t)}(x, y)$. Otherwise, the agents move to state y according to the probabilities

$$\mathsf{P}^{A(t)}(x, y) = \mathsf{P}_1^{A_1(t)}(x_1, y_1)\mathsf{P}_2^{A_2(t)}(x_2, y_2).$$

Each agent k then receives a reward $R_k = r_k(x_k, a_k, y_k) + {}^ir_I(x, a, y)$, where ${}^ir_I(x, a, y) = 0$ if $y \notin {}^i\mathcal{X}_I$. The process repeats with $X_1(t+1) = y_1$ and $X_2(t+1) = y_2$.

When $N > 2$ agents are considered, the structure of the IDMG is similar, but should now be augmented to accommodate the individual MDPs for each agent, $\mathcal{M}_1, \ldots, \mathcal{M}_N$. In this more general situation, the interaction states in each interaction game ${}^i\mathcal{M}_I$ will consist of a subset of $\mathcal{X}_{k_1} \times \ldots \times \mathcal{X}_{k_p}$ for some $k_1, \ldots, k_p$. In particular, each interaction game describes a situation where p agents interact. We will stick to the two-agent scenario for ease of exposition.

In many situations it is quite reasonable to assume communication only when the agents need to interact. In robotic tasks, for example, two robots equipped with wireless communication devices might only be able to communicate when they are spatially close. As the distance between them increases, their interaction capabilities will in general decrease and they will act more and more independently. This behavior is captured naturally by the IDMG model. Moreover, disregarding agent interaction in many of the states greatly simplifies the decision-making process, since the number of possible situations to consider is greatly reduced.

3.3 Related models

Several methods addressing related multiagent sequential decision problems have been proposed. Independent agents have been considered in [17]. Although focusing on a completely different application, the authors resort to the framework of *non-interacting Markov games* to address problems where a group of agents independently fulfill different tasks in a common environment. Other works [12, 21] address the problem of *weakly coupled MDPs*. These works tackle problems modeled using large MDPs that can be decomposed into smaller, independent MDPs. Given the solutions for the smaller MDPs, a method is proposed to merge these solutions into an optimal or near-optimal solution for the composite MDP. Our setting is somewhat more general than that considered in the latter works. We allow some degree of dependence both in terms of rewards and transitions. However, we do admit that this dependence is *local*, in that it does not "occur" over the complete joint state space. Our setting is also different from those above since we admit that there are multiple independent decision makers.

The work presented in this paper is also essentially distinct from most Dec-(PO)MDP approaches, both general ones (see, for example, [2, 14] and references therein) as well as techniques designed to exploit locality of interaction [13, 15]. The latter model the interactions between agents in a graph, which is either stationary [13] or can develop over time [15]. Although these techniques exploit factored models, due to the complexity of solving Dec-POMDPs, their scalability is limited.

Unlike Dec-(PO)MDPs, we assume that in most situations the multiagent decision problem can be decomposed into simpler independent single-agent problems. In this respect, our model bears some resemblances with transition-independent Dec-MDPs [1]. On the other hand, it is somewhat more general in that, in some situations, transitions *may depend* on the actions of several/all agents. A second fundamental difference is that we assume that when interaction is required, the agents are able to communicate.

Note that in the extreme case where $\mathcal{X}_I = \mathcal{X}^1 \times \mathcal{X}^2$, the IDMG model reduces to a standard Markov game (with the corresponding assumption of continuous communication). On the other hand, if $\mathcal{X}_I = \emptyset$, we are back to the situation of independent MDPs. In the former scenario, the complexity of solving an IDMG is still a great deal inferior to that of solving Dec-MDPs, at the cost of continuous communication [18]. In the latter scenario, the complexity of solving an IDMG is similar to that of single-agent problems.

4. PLANNING IN IDMG MODELS

We now describe an approximate solution method for IDMGs that takes advantage of the particular structure of this model. We start by deriving an equivalent Markov game model that will provide the ideas leading to our solution method. We then discuss several important features of this method, namely its effective use of the particular IDMG structure. We also argue that, although encompassing some approximations, our method is actually able to produce good performance (as seen in the experimental results in the next section).

4.1 An equivalent Markov game model

Given an IDMG $\Gamma = (\mathcal{M}_1, \mathcal{M}_2, \{{}^i\mathcal{M}_I, i = 1, \ldots, n\})$, we define an equivalent Markov game model as follows:

- $\bar{\mathcal{X}} = \mathcal{X}_1 \times \mathcal{X}_2$;
- $\bar{\mathcal{A}} = \mathcal{A}_1 \times \mathcal{A}_2$;
- For any $a \in \bar{\mathcal{A}}$ and $x, y \in \bar{\mathcal{X}}$, set

$$\bar{\mathsf{P}}^a(x, y) = \begin{cases} {}^i\mathsf{P}_I^a(x, y) & \text{if } x \in {}^i\mathcal{X}_I \text{ for some } i; \\ \mathsf{P}_1^{a_1}(x_1, y_1)\mathsf{P}_2^{a_2}(x_2, y_2) & \text{otherwise;} \end{cases}$$

- For any $a \in \bar{\mathcal{A}}$ and any $x, y \in \bar{\mathcal{X}}$, set

$$\bar{r}_k(x, a, y) = r_k(x_k, a_k, y_k) + \sum_i {}^ir_I(x, a, y). \quad (4.1)$$

The tuple $\bar{\Gamma} = \left(2, \bar{\mathcal{X}}, (\bar{\mathcal{A}}_k), \bar{\mathsf{P}}, (\bar{r}_k), \gamma\right)$ is a Markov game which is equivalent to the IDMG defined in Section 3.2. Now given any joint policy π, it follows that

$$\bar{V}_k^\pi(x) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t \bar{R}_k(X(t), A(t)) \mid X(0) = x\right], \quad (4.2)$$

where we denoted by $\bar{R}_k(X(t), A(t))$ the random reward received by agent k when all agents take action $A(t)$ at the joint state $X(t)$ and is given by (4.1) upon the knowledge of $X(t+1)$. To simplify the notation, we henceforth consider

IDMGs $(\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_I)$ with a single interaction game $\mathcal{M}_I$, noting that everything trivially extends to the case of multiple such games. We also drop the explicit mention that the expectation is to be conditioned on $X(0) = x$.

Using (4.1), we can decompose (4.2) as

$$\begin{aligned}\bar{V}_k^{\pi}(x) &= \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t \bar{R}^k\big(X(t), A(t)\big)\right] \\ &= \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t R_k\big(X_k(t), A_k(t)\big) + R_I\big(X(t), A(t)\big)\right] \\ &= V_k^{\pi_k}(x_k) + V_I^{\pi}(x).\end{aligned}$$

Similarly, we can write the Q-function for a policy π as

$$\bar{Q}_k^{\pi}(x, a) = Q_k^{\pi_k}(x_k, a_k) + Q_I^{\pi}(x, a). \tag{4.3}$$

Notice that the above expression states that, for any policy, the corresponding Q-values can be decomposed into an individual component Q_k and an interaction component Q_I. For the particular case of a Nash equilibrium π^*, we have

$$\bar{Q}_k^{\pi^*}(x, a) = Q_k^{\pi_k^*}(x_k, a_k) + Q_I^{\pi^*}(x, a).$$

We emphasize that, although the above expression appears to decouple the individual and the interactive components of the Q-function, this decoupling is nothing more than apparent. In fact, the function $\bar{Q}_k^{\pi^*}$ still depends on the *joint policy* π^* and, as such, there is an implicit coupling (in terms of action choice) between each of the components $Q_k^{\pi_k^*}$ and $Q_I^{\pi^*}$.

As seen in [9], each Nash equilibrium π^* has an associated Q-function that verifies, for all k, the following recursion

$$\bar{Q}_k^{\pi^*}(x, a) = \sum_{y\in\mathcal{X}} \bar{\mathsf{P}}^a(x, y)\big[\bar{r}_k(x, a, y) \; + \; \gamma \; \mathrm{Nash}_y^k(\bar{Q}_1^{\pi^*}, \ldots, \bar{Q}_N^{\pi^*})\big], \tag{4.4}$$

where $\mathrm{Nash}_y^k(\bar{Q}_1^{\pi^*}, \ldots, \bar{Q}_N^{\pi^*})$ is a max-like operator denoting the Nash value for agent k at state y with respect to the game defined by the matrices $\bar{Q}_k^{\pi^*}$.

We now consider that $\mathcal{X}_I = \emptyset$. This means that, in (4.3),

$$\bar{Q}_k^{\pi}(x, a) = Q_k^{\pi_k}(x_k, a_k), \quad k = 1, \ldots, N$$

and each agent can decide independently of the other agents. Moreover, each agent has an optimal individual policy whose corresponding Q-function verifies

$$Q_k^*(x_k, a_k) = \sum_{y_k\in\mathcal{X}_k} \mathsf{P}_k^a(x, y)\big[r_k(x_k, a_k, y_k) \; + \; \gamma \max_{b_k\in\mathcal{A}_k} Q_k^*(y_k, b_k)\big]. \tag{4.5}$$

On the other hand, suppose that $r_k \equiv 0$ for all k. Then we get $\bar{Q}_k^{\pi}(x, a) = Q_I^{\pi}(x, a)$ and the game reduces to a fully cooperative Markov game. This means that there is an *optimal joint policy* whose corresponding Q-function verifies

$$Q_I^*(x, a) = \sum_{y\in\mathcal{X}} \bar{\mathsf{P}}^a(x, y)\big[r_I(x, a, y) + \gamma \max_{b\in\mathcal{A}} Q_I^*(y, b)\big]. \tag{4.6}$$

The above reasoning contains the main ideas in our solution method, which we will describe next.

4.2 An approximate solution method

We start by computing the functions Q_k^* and Q_I^* as defined in (4.5) and (4.6). The component Q_k^* accounts for the best that agent k can do if no interaction takes place, while Q_I^* accounts for the best that the several agents can do when acting as a team. We can now approximate each function $\bar{Q}_k^{\pi^*}$ as

$$\bar{Q}_k^{\pi^*}(x, a) \approx Q_k^*(x_k, a_k) + Q_I^*(x, a),$$

and, from the above functions, we define at state x a *matrix game* $\bar{\Gamma}_x = (2, (\mathcal{A}_k), \bar{q}_k)$ where

$$q_k(a) = \sum_{y\in\mathcal{X}} \bar{\mathsf{P}}_a(x, y)\big[\bar{r}_k(x, a, y) + \gamma \mathrm{Nash}_y^k(\bar{Q}_1^{\pi^*}, \ldots, \bar{Q}_N^{\pi^*})\big].$$

Notice that, in the above approximation, we are disregarding the coupling between the individual and interactive components of $Q_k^{\pi^*}$. Determining the Nash equilibrium for each such game provides each agent with a good approximation to the optimal policy for the Markov game $\bar{\Gamma}$.

We remark that the approximation above causes the policy thus obtained to disregard the long-term effects of the coupling between the individual and interactive components of $Q_k^{\pi^*}$. In this sense, the obtained policy is somewhat "shortsighted" with respect to the long-term effects of the interaction between the agents. However, as our experimental results illustrate, this effect is often negligible and the obtained policy will generally be a good approximation to the actual "optimal" joint policy.

There is, however, one inconvenience: the above computation can only be conducted in a Markov game, as it requires knowledge of the *joint state* of the game. In our IDMG framework, this knowledge is only available at the interaction states. Therefore, in the non-interaction states, a different process must be used to choose the actions. We want to avoid a complicated modeling of the effects of other agents' policies, which will be inherently uncertain due to the limited information available to each agent. Furthermore, in line with the assumption that each agent can successfully execute its task independently, we follow a simple heuristic approach that completely disregards the "existence" of other agents in the non-interaction states. This means that, in these states, each agent disregards the interaction component of $Q_k^{\pi^*}$ and chooses its individual actions at each state x_k as $a_k^* = \arg\max_{a_k\in\mathcal{A}_k} Q_k^*(x_k, a_k)$, where Q_k^* is defined in (4.5). In many situations it is actually possible to define the interaction games $\{{}^i\mathcal{M}_I\}$ so as to minimize the effect of the interactive component in the non-interaction states, when that interaction might incur a penalty. In this case, the action choice in the non-interaction states can be very close to the optimal. This point is illustrated in our results.

In the interaction states, we assume that communication is unlimited and noise-free. When in these states, each agent communicates its current individual state to the other agents and, if necessary, the corresponding individual Q-functions (if this knowledge is not available beforehand). At this point, all agents know the joint state of the team and the individual Q-functions for all agents. Therefore, they can compute the matrix-games $\bar{\Gamma}_x$ and corresponding Nash equilibria and choose their actions accordingly. Hence, our method needs to compute several Nash equilibria, but all with respect to *matrix games*. For this class of games, several computational methods are available. In our implementation, we resort to

the fast methods proposed in [16]. In the case of multiple equilibria, the agents choose the first one according to a shared lexicographical ordering.

4.3 Discussion

As discussed in Section 3.3, we emphasize that our model is different from Dec-POMDPs [3] and their transition-independent variations [1]. Our model resembles the one considered in [19], in which agents communicate their local information if they consider it could be useful to improve team performance. In [20] an algorithm for computing communication-based policies for factored Dec-MDP models is proposed. However, unlike the last two models, we do not assume that communication is always possible (even if desirable). Instead, we assume communication also to be a "local ability", and communication only occurs at the interaction states. Furthermore, the solution technique presented here is relatively fast, as we only need to compute an MDP solution for each agent according to (4.5) and solve an MDP for each ${}^{i}\mathcal{M}_I$, as seen in (4.6). This is in stark contrast with techniques for Dec-POMDPs, many of which rely on a solution of the centralized decision problem (*e.g.*, [14]), whose computational cost grows exponentially with the number of agents.

An approximate technique for Dec-POMDPs that also uses equilibria concepts has been proposed [7]. However, there Bayesian games are used with local histories as type, providing information over the joint state of the agents. The Bayesian game solutions are used to approximate the optimal policy, which requires solving a Bayesian game for each possible joint-history depth (an exponentially growing type space). Instead, we use local solutions which, in the absence of interaction, are optimal. Furthermore, we only need to solve a matrix game of constant size for each interaction state. Our proposed model and solution method offer more potential for scaling up, while still capturing relevant interactions between agents.

5. EXPERIMENTS

In our experiments we consider several different test problems for two agents, that we describe next. The first test problem is the navigation problem in the **H**-shaped environment as detailed in Section 3. Each robot has 4 available actions, move up, down, left and right. There is some noise in the transitions, causing random moves in a random direction. Each robot receives a reward of 0.5 for reaching its goal state. The interaction states correspond to the joint states where both robots are in the gray area in Fig. 1. Both robots get a penalty of -5 every time they are simultaneously in the doorway state.

In the second test problem we consider the problem of a travel agent that must assign clients to hotels trying to fit their preferences. A travel agent has available n_r rooms in n_h different hotels. At each time step there is a non-zero probability of a client arriving that wants to go to hotel i (Fig. 2). The agent should decide where to assign the client, knowing that it will get a reward of 10 for every properly assigned client, of -10 for every client assigned to a hotel that is already full, of 2 for every client assigned to a different hotel from the one it prefers and -1 for refusing a client. The agent also has the possibility of assigning a client to a private resort, which will grant the agent with a reward of 5; this resort has no limit on the number of people. For the purposes of our test, clients never leave hotels.

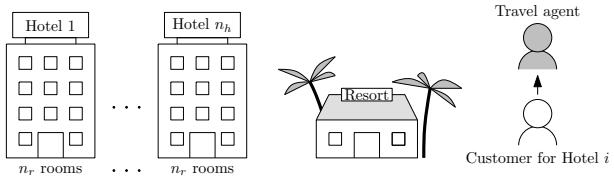

Figure 2: The single-agent version of the hotel problem.

	# Indiv. States	# Inter. States	# Inter. States Ext.	MG
H-shape	21	49	49	441
Hotel 1	4	3	15	16
Hotel 2	27	180	693	729
ISR	172	944	1,218	29,584
MIT	196	768	1,052	38,416
SUNY	296	1,296	1,668	87,616
PENTAGON	208	416	568	43,264
CIT	280	656	888	78,400
CMU	532	3,680	4,574	283,024

Table 1: Number of individual and interaction states in the different tests scenarios. In Hotel 1, we set $n_r = n_h = 1$ and in Hotel 2 we set $n_r = n_h = 2$. The third column corresponds to the tests with extended interaction states. In the last column, we provide the number of states of the equivalent Markov game model $\bar{\Gamma}$.

We considered a multiagent version of this problem, in which agent k has its own clients and n_r^k rooms available in each hotel. The interaction states are those in which agents would want to assign a client to the resort (*i.e.*, an agent has a client wanting to go to a full hotel, and the other one also has a client). If they both decide to assign a client to the resort at the same time, they both get a reward of -5. If we imagine the resort as a "temporary" solution for the client's lodging problem, assigning more than one client to the resort at the same time causes the resort to get too "crowded" and hence the received penalty.

Finally, in the last set of problems, we considered several larger and more elaborate navigation problems, pictured in Fig. 3 and originating from [6]. Each of two robots must navigate to one of the states marked with an "×", while starting from a location drawn uniformly at random. Each robot has 3 actions available: "Move forward", "Turn left" and "Turn right". Therefore, the robots can be in each of the square cells in one of 4 possible orientations. The total number of states for each problem is listed in Table 1, as well as the number of interaction states. The interaction states are those in dark, corresponding to those around the doors, and grant both agents with a reward of -100 every time they are both in the location (independent of orientation) in contiguous/coincident interaction states. Reaching the goal state is rewarded with 10, after which an agent is transferred to an absorbing state and can no longer obtain rewards. The interaction states correspond to the darkest cells in Fig. 3.

We used the solution method described in the previous section, resorting to the fast search algorithms in [16] to determine the necessary equilibria. We then compared the

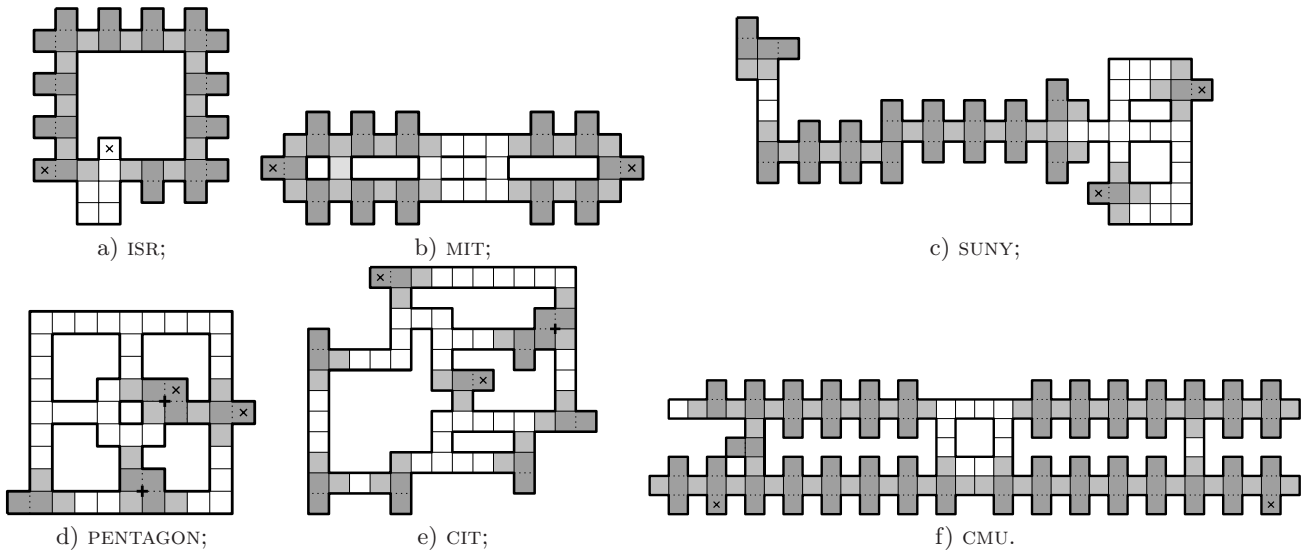

Figure 3: Large scenarios used in the experiments, from [6]. The dark shaded cells correspond to interaction states, while the lighter cells form part of the extended set of interaction states.

	Ind.	IDMG	MMDP	IDMG+IS
H-shape	−0.71	1.34	1.49	1.34
Hotel 1	58.92	74.03	87.63	75.87
Hotel 2	97.84	118.22	133.85	119.08
ISR	−8.03	−0.70	10.60	10.57
MIT	3.73	5.09	8.60	8.58
SUNY	−1.05	5.20	8.00	7.99
PENTAGON	6.03	6.69	11.63	11.57
CIT	5.47	6.70	8.93	8.91
CMU	−3.16	−2.21	5.84	5.81

Table 2: Results of our method vs. independent and centralized agents. The last column considers an extended set of interactive states.

performance of our approach with two other approaches. In particular, we also considered the case in which all agents act independently, *i.e.*, with no coordination and a case in which the control of the agents is *centralized*, corresponding to an MMDP solution [5]. We dub the three methods as "IDMG", "Ind." and "MMDP". For each experiment we ran 1000 independent trials, each trial consisting of either 15 time steps (in the **H**-shaped environment) or 50 time steps (in the others), and we used $\gamma = 0.95$. We also considered an extended set of interaction states.[2] These additional interaction states correspond to the cells immediately adjacent to the darkest cells in Fig. 3 and are also a little shaded. The results for this set of tests are dubbed as "IDMG+IS".

The results in Table 2 clearly illustrate the advantages of our method over independent agents. This is due to the fact that our method takes into consideration the local interactions occurring at the interaction states. The independent agents choose their actions according solely to the individual function Q_k^* (as defined in (4.5)). The centralized solution upper bounds the IDMG performance, as it assumes full state knowledge for each agent. Furthermore, its computational cost is much higher, as it solves an MMDP, *i.e.*, an MDP for joint actions and joint states (at polynomial cost). The last column of Table 1 lists the number of joint states for the MMDP. For example, to compute the IDMG solution for the CMU environment, which contains relatively many interaction states, we only need to estimate $N \cdot 531 \cdot |\mathcal{A}_k| + 3,680 \cdot |\mathcal{A}_k|^N = 36,312$ Q-values, vs. $238,024 \cdot |\mathcal{A}_k|^N = 2,547,216$ for the centralized solution.

Another important aspect of the results in Table 2 is related to the performance of our method in the larger scenarios. Even though our approach significantly outperforms the independent agents, it still falls behind the optimal centralized controller. This is due to the fact that the agents always receive a penalty of −100 in the interaction states: if their initial position is such that the only way to reach the goal is by potentially "meeting" in an interaction state, then the robots will do that and collect the corresponding penalty (in case the other agent is actually also present at the same location). In other words, our agents generally "choose to go" through the interaction states, even if this implies the possibility of collecting a penalty. On the other hand, if we extend the interaction states to minimize the effect of the interaction in the non-interaction states, *our method is able to attain near-optimal performance* in these navigation domains. The intuition behind this phenomenon is that by slightly extending the interaction states, the agents are now able to avoid the penalties that may arise in the interactive states and "safely cross" the interaction areas without collecting a penalty.

6. CONCLUSIONS

In this paper, we introduced the framework of *interaction-driven Markov games* (IDMGs) to address multiagent decision-making problems in which interaction between the agents is a local phenomenon. This framework considers

[2]Notice that, by extending the number of interactive states, the performance of the agents is expected to improve. In fact, as the number of interactive states increases, the solution of the IDMG approaches the optimal solution of the corresponding Markov game.

separately the situations in which the agents interact and those in which the agents can act independently. The agents are coupled through the joint rewards and joint transitions in the interaction states.

The fundamental idea behind IDMGs is to reduce the complexity of the multiagent decision problem to that of single-agent problems in the states where no interaction exists. On the other hand, in those states where interaction does exist, we consider communication explicitly as a means to optimize the exchange of information. Therefore, the IDMG model captures several properties of many actual systems in a natural way. In particular, the agents' ability to communicate is a local property in IDMGs and there is no joint state/action observability. Furthermore, an agent is generally unable to perceive the state of the other agents, except in situations where they are "close". The experimental performance of our algorithm also illustrates that the IDMG model is well suited to address large multiagent problems with the features described. Finally, the experimental results also show that our techniques allow for effective planning in domains which are too large to be solved by general methods for partially observable stochastic games.

As future work we would like to further investigate which classes of problems can adequately be addressed using the IDMG framework. Another interesting direction of research is to consider automatic learning of the interaction states. In this paper we consider the interaction states to be defined *a priori*, as part of the problem. However, the agents could *learn* the set of states where interaction is fundamental, for instance along the lines of [10].

Acknowledgments

The authors gratefully acknowledge the suggestions of Frans Oliehoek and the anonymous reviewers. This work was partially supported by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) under the POS_C Program that includes FEDER funds, and under the Carnegie Mellon-Portugal Program and the Information and Communications Technologies Institute (ICTI) (www.icti.cmu.edu). The first author also acknowledges FCT grant PTDC/EEA-ACR/73266/2006. The views and conclusions contained in this document are those of the authors only.

7. REFERENCES

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

[2] D. Bernstein, E. Hansen, and S. Zilberstein. Bounded policy iteration for decentralized POMDPs. In *Proc. Int. Joint Conf. Artificial Intelligence*, pages 1287–1292, 2005.

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

[4] D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

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

[6] A. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Brown University, 1998.

[7] R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Approximate solutions for partially observable stochastic games with common payoffs. In *Proc. Int. Joint Conf. Autonomous Agents and Multi Agent Systems*, volume 1, pages 136–143, 2004.

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

[9] J. Hu and M. Wellman. Nash Q-learning for general sum stochastic games. *J. Machine Learning Research*, 4:1039–1069, 2003.

[10] J. Kok, P. Hoen, B. Bakker, and N. Vlassis. Utile coordination: Learning interdependencies among cooperative agents. In *Proc. Symp. on Computational Intelligence and Games*, pages 29–36, 2005.

[11] H. Kuhn. Extensive games and the problem of information. *Annals of Mathematics Studies*, 28:193–216, 1953.

[12] 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. Nat. Conf. Artificial Intelligence*, pages 165–172, 1998.

[13] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *Proc. Nat. Conf. Artificial Intelligence*, pages 133–139, 2005.

[14] F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *J. Artificial Intelligence Research*, 2008. To appear.

[15] F. A. Oliehoek, M. T. J. Spaan, S. Whiteson, and N. Vlassis. Exploiting locality of interaction in factored Dec-POMDPs. In *Proc. Int. Joint Conf. Autonomous Agents and Multi Agent Systems*, 2008.

[16] R. Porter, E. Nudelman, and Y. Shoham. Simple search methods for finding a Nash equilibrium. *Games and Economic Behavior*, 2006 (in press).

[17] B. Price and C. Boutilier. Implicit imitation in multiagent reinforcement learning. In *Proc. Int. Conf. Machine Learning*, pages 325–334, 1999.

[18] D. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *J. Artificial Intelligence Research*, 16:389–423, 2002.

[19] M. Roth, R. Simmons, and M. Veloso. Decentralized communication strategies for coordinated multi-agent policies. In *Multi-Robot Systems: From Swarms to Intelligent Automata*, volume III, pages 93–106, 2005.

[20] M. Roth, R. Simmons, and M. Veloso. Exploiting factored representations for decentralized execution in multi-agent teams. In *Proc. Int. Joint Conf. Autonomous Agents and Multi Agent Systems*, 2007.

[21] S. Singh and D. Cohn. How to dynamically merge Markov decision processes. In M. Jordan, M. Kearns, and S. Solla, editors, *Adv. Neural Information Processing Systems 10*, pages 1057–1063, 1998.