Decentralized Planning in Stochastic Environments with Submodular Rewards

Rajiv Ranjan Kumar, Pradeep Varakantham and Akshat Kumar
School of Information Systems
Singapore Management University
{rajivk,pradeepv,akshatkumar}@smu.edu.sg

Abstract

Decentralized Markov Decision Process (Dec-MDP) provides a rich framework to represent cooperative decentralized and stochastic planning problems under transition uncertainty. However, solving a Dec-MDP to generate coordinated yet decentralized policies is NEXP-Hard. Researchers have made significant progress in providing approximate approaches to improve scalability with respect to number of agents. However, there has been little or no research devoted to finding guarantees on solution quality for approximate approaches considering multiple (more than 2 agents) agents. We have a similar situation with respect to the competitive decentralized planning problem and the Stochastic Game (SG) model. To address this, we identify models in the cooperative and competitive case that rely on submodular rewards, where we show that existing approximate approaches can provide strong quality guarantees (*a priori*, and for cooperative case also *posteriori* guarantees). We then provide solution approaches and demonstrate improved online guarantees on benchmark problems from the literature for the cooperative case.

1 Introduction

Decentralized stochastic planning for a team of agents is required in a wide variety of problems such as target tracking by a team of sensors (Nair et al. 2005; Kumar and Zilberstein 2011; Chapman and Varakantham 2014), securing targets from unknown attackers using a team of defenders (Shieh et al. 2014), rescuing of victims by a team of robots during disaster (Melo and Veloso 2011; Varakantham et al. 2009) and analysing underwater samples using a team of underwater vehicles (Yin and Tambe 2011). Existing literature has focussed on Decentralized Markov Decision Processes (Dec-MDPs) or Stochastic Games to represent such problems of interest. Unfortunately, solving Dec-MDPs and Stochastic Games is computationally challenging, which precludes solving problems with many ($>> 2$) agents. While approximate approaches have been proposed to solve multiple agent problems (Velagapudi et al. 2011; Varakantham et al. 2009; Kumar, Zilberstein, and Toussaint 2015), there is little or no research in approximation methods that provide strong guarantees on solution quality in such decentralized settings.

Copyright © 2017, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

With the emphasis on quality guarantees, we focus specifically on multi-agent sequential planning models with joint submodular reward functions. Previous work has considered submodularity in the context of sequential decision making for multi-agent problems. Kumar *et al.* (2009) considered submodularity in rewards across time steps in the context of Multi-agent MDPs (MMDPs) in weather tracking satellites. Detecting storms early is of key interest and this reward function for detecting storms is submodular over time steps. Satsangi *et al.* (2015) considered submodularity in multi-agent sensor selection problem modelled as a POMDP with submodular belief based reward function. Our contributions differ from this line of work due to the focus on decentralized planning models (and not on centralized planning models) and our algorithms rely on showing submodularity of the value function.

Specifically we consider Transition Independent Dec-MDPs (TI-Dec-MDPs), Transition Independent Dec-POMDPs (TI-Dec-POMDPs) and Transition Independent Stochastic Games (TI-SGs). There has been existing work in solving some of these models (Dibangoye, Amato, and Doniec 2012; Mostafa and Lesser 2009), and in some instances providing *posteriori* guarantees (using lower and upper bounds). In this paper, we specifically focus on submodular joint rewards and hence are able to provide strong *a priori* and *posteriori* guarantees.

The main contributions of this paper are:

- In the cooperative case, for Transition Independent Dec-MDPs and Transition, Observation Independent Dec-POMDPs with monotone submodular reward functions, we show that greedy solution provides an *a priori* guarantee of at least 50% from optimal. We also provide a method to compute *posteriori* guarantees which are much tighter than the worst case of bound of 50%.
- We provide a lazy version of the well-known greedy algorithm and enhance a recent approach for exploiting anonymity and homogeneity to solve the problem models identified above.
- In the competitive case, we show that for TI-SGs with submodular social rewards and private rewards that are greater than Vickrey rewards, any Nash Equilibrium solution is at least 50% from the social welfare optimal. That is to say, price of anarchy is 2.

- Finally, we provide experimental results on two benchmarks from the Dec-MDP literature, where we show that the *posteriori* quality guarantees are significantly better (in some cases close to 90% of optimal) than the a priori guarantee of at least 50% from optimal.

2 Background

2.1 Transition Independent Dec-MDP

Transition Independent Decentralized Markov Decision Process (TI-Dec-MDP) model (Becker et al. 2004) is characterized by the tuple:

$$\Big\langle Ag, S, A, \{T_i\}_{i \in Ag}, R, H, \alpha \Big\rangle$$

- Ag is the set of agents.
- S is the factored joint state space. $S = S_1 \times S_2 \ldots S_{|Ag|}$, where S_i is the state space corresponding to each individual agent i. We can also have a global unaffected state feature, S_u. It is a trivial extension and not provided here.
- A is the joint action space. $A = \times_{i \in Ag} A_i$, where A_i is the action space corresponding to each individual agent i.
- T_i is the individual agent transition function. $T_i(s_i, a_i, s_i')$ indicates the transition probability of moving from s_i to s_i' on taking action a_i.
- R is the submodular joint reward, with $R(s, a)$ representing the reward for taking joint action a in joint state s.
- H is the time horizon and α is the starting state distribution.

The goal is to obtain a joint policy $\vec{\pi} = \langle \vec{\pi}_1, \vec{\pi}_2, \ldots, \rangle$ (with one policy, $\vec{\pi}_i$ for each agent i) that maximizes expected reward or value defined as follows:

$$V(\vec{\pi}) = \sum_s \alpha(s) \cdot V^H(s, \vec{\pi}) \tag{1}$$

$$V^t(s, \vec{\pi}) = R\Big(s, \Big\langle \pi_1^t(s_1), \ldots, \pi_{|Ag|}^t(s_{|Ag|}) \Big\rangle\Big) + \sum_{s'} \Big[\prod_{i \in Ag} T_i\Big(s_i, \pi_i^t(s_i), s_i'\Big) \Big] \cdot V^{t-1}(s', \vec{\pi}) \tag{2}$$

2.2 Monotone Submodularity and Matroids

We now describe submodular functions and matroids.

Definition 1. *Given a finite set, Π, a **submodular** function is a set function, $g : 2^{\Pi} \rightarrow \mathbb{R}$, where 2^{Π} is the power set corresponding to Π. More importantly, $\forall X, Y \subseteq \Pi$ with $X \subseteq Y$ and for every $i \in \Pi \setminus Y$, we have:*

$$g(X \cup i) - g(X) \geq g(Y \cup i) - g(Y)$$

*A submodular function g is **monotone** if $g(Y) \geq g(X)$ for $X \subseteq Y$.*

Monotone submodular functions are interesting because maximizing a submodular function to pick a fixed number of elements (say k) from the finite set (Π) while difficult can be approximated efficiently with a strong quality guarantee. Specifically, a greedy algorithm that incrementally generates the solution set by maximizing marginal utility provides solutions that are at least 63% ($1 - \frac{1}{e}$) of the optimal solution.

In this paper, we are also interested in maximizing a submodular function, however, under a specific constraint on the finite set (Π) and the elements that are picked. Specifically, the constraint is specified using a partition matroid. We provide the formal definitions below:

Definition 2. *For a finite ground set, Π, let $\mathcal{P}$ be a non-empty collection of subsets of Π. The system $\Gamma = (\Pi, \mathcal{P})$ is a matroid if it satisfies the following two properties:*

- *The hereditary property: $\mathcal{P}_1 \in \mathcal{P} \wedge \mathcal{P}_2 \subset \mathcal{P}_1 \implies \mathcal{P}_2 \in \mathcal{P}$. In other words, all the subsets of $\mathcal{P}_1$ must be in $\mathcal{P}$.*
- *The exchange property: $\forall \mathcal{P}_1, \mathcal{P}_2 \in \mathcal{P} : |\mathcal{P}_1| < |\mathcal{P}_2| \implies \exists x \in \mathcal{P}_2 \setminus \mathcal{P}_1; \mathcal{P}_1 \cup x \in \mathcal{P}$.*

We are specifically interested in a ground set that is partitioned as $\Pi = \Pi_1 \cup \Pi_2 \cup \ldots \cup \Pi_k$. The family of subsets, $\mathcal{P} = \{P \subseteq \Pi : \forall i, |P \cap \Pi_i| \leq 1\}$ forms a matroid called a partition matroid. This family of subsets denotes that any solution can include at most one element from each ground set partition. This is relevant in this paper, as ground set partitions represent policy space of each agent and we need to pick one policy for each agent.

3 Submodular TI-Dec-MDP

In this section, we describe important properties of TI-Dec-MDPs with monotone submodular rewards. First, we provide a concrete example of a joint reward function employed to coordinate defender teams in security domains (Shieh et al. 2014) that is both monotonically increasing and submodular. It is defined as follows:

$$R(s, a) = \sum_b y_b \cdot f_b(\sigma(s, a, b)) \tag{3}$$

y_b indicates value of target b and hence is a non-negative number. $f_b(.)$ is a monotone submodular function referred to as the effectiveness of patrolling a target b. Effectiveness of patrols at a target b depends on the number of agents patrolling the target. $\sigma(s, a, b)$ counts the number of agents at target b if the current joint state is s and joint action is a. The usual definition of $f(.)$ for effectiveness parameter ϵ ($0 < \epsilon \leq 1$) is as follows:

$$f(k) = 1 - (1 - \epsilon)^k.$$

We now formally state our definition of the submodular joint-reward function. We define our ground set $\mathcal{SA}$ as the set of all possible state-action pairs for all the agents

$$\mathcal{SA} = \cup_{i=1}^{|Ag|} \{(s_i, a_i)\ \forall s_i \in S_i, a_i \in A_i\}.$$

The joint-reward function R is defined for all the possible subsets of this ground set. As per definition 1, we assume that this set-based reward function R is submodular and monotone. Notice that explicitly defining such a reward function for all the possible subsets of $\mathcal{SA}$ would be intractable. Therefore, one often uses a *functional* definition of such reward function R similar to the security domain case describe above. We also note that the reward function R needs to be properly defined even when two different actions are selected for the same state for an agent. That is, when we have $\{(s_i, a_i), (s_i, a_i')\}$ part of the subset for which R needs

to be defined. As such, this is not a valid state-action combination because for the given state s_i, we have two different actions a_i and a'_i mapped to s_i. In many cases, we can still define a reward function R for this invalid case by assuming that agent i repeated, and inconsistent state-action pairs are assigned properly to different repeated agents. Note that when computing final policies, this case will never occur as we will optimize policies subject to a partition matroid constraint that enforces valid policies, as described in section 4.

For TI-Dec-MDPs with monotone submodular rewards, expected reward, $V(\vec{\pi})$ is monotone submodular. We show this in three key steps:

Step 1: There exists a finite ground set, Π such that expected reward can be defined over its power set. In our case, the ground set is the set of all individual agent policies, i.e.,

$$\Pi = \cup_i \Pi_i$$

Step 2: Value function is defined over the power set for Π. We define $V : 2^{\Pi} \rightarrow \mathbb{R}$ as a mapping from all subsets of Π to a real value. This entails that the value function will be defined for not only correct joint policies (i.e., one policy for each agent) but also for incorrect joint policies (i.e., more than one policy for certain agents). We can directly employ the definition provided earlier in Equations 1 and 2. When we have an incorrect policy with an agent i having multiple policies, then we assume that there are multiple versions of agent i, each executing a different policy.

Step 3: Finally, we show that the $V^H(s, \vec{\pi})$ is monotone submodular in the proofs of Proposition 1 and Proposition 2. Then, $V(\vec{\pi})$ is a linear combination of submodular functions and hence is also monotone submodular.

In the following proposition and proof, we will employ the following notation. For a set of agents Z, joint state for all agents in Z is given by s_Z and joint policy for agents in Z is given by $\vec{\pi}_Z = \langle \vec{\pi}_1, \ldots, \vec{\pi}_{|Z|} \rangle$. We employ the following short forms in the proof that follows:

$$\mathcal{R}^t_Z = R(s_Z, \pi^t_Z(s_Z))$$
$$\mathcal{T}^t_i = T_i(s_i, \pi^t_i(s_i), s'_i)$$
$$\mathcal{V}^t_Z = V^t(s_Z, \vec{\pi}_Z)$$
$$\mathcal{V'}^t_Z = V^t(s'_Z, \vec{\pi}_Z)$$

Proposition 1. *For a TI-Dec-MDP, $V^H(s, \vec{\pi})$ is submodular if joint reward, R is submodular.*

Proof. Let us consider two agent sets M ($\in Ag$) and N ($\in Ag$), with $M \subseteq N$. s_M and s_N denote joint states of agents in sets M and N respectively. a_M and a_N denote joint action of agents in sets M and N respectively. Each agent has one policy[1]. In order to show submodularity of the value function, $V^H()$, we employ mathematical induction on the time horizon, H.

Base case: $H = 1$

$$\mathcal{V}^1_M = \mathcal{R}^1_M \ ; \ \mathcal{V}^1_{M\cup i} = \mathcal{R}^1_{M\cup i} \ ; \ \mathcal{V}^1_N = \mathcal{R}^1_N \ ; \ \mathcal{V}^1_{N\cup i} = \mathcal{R}^1_{N\cup i}$$
$$\implies \mathcal{V}^1_{M\cup i} - \mathcal{V}^1_M = \mathcal{R}^1_{M\cup i} - \mathcal{R}^1_M \ ; \ \mathcal{V}^1_{N\cup i} - \mathcal{V}^1_N = \mathcal{R}^1_{N\cup i} - \mathcal{R}^1_N$$

[1]It should be noted that we can also represent the case of incorrect joint policies by assuming the same agent is repeated (with different policies) multiple times in sets M and N

Since R is submodular: $\mathcal{R}^1_{M\cup i} - \mathcal{R}^1_M \geq \mathcal{R}^1_{N\cup i} - \mathcal{R}^1_N$
Hence, $\mathcal{V}^1_{M\cup i} - \mathcal{V}^1_M \geq \mathcal{V}^1_{N\cup i} - \mathcal{V}^1_N$

Let us assume the proposition holds for $H = \tau$, i.e:.

$$\mathcal{V}^\tau_{M\cup i} - \mathcal{V}^\tau_M \geq \mathcal{V}^\tau_{N\cup i} - \mathcal{V}^\tau_N \tag{4}$$

For $H = \tau + 1$: Using value function definition from Equation 2:

$$\mathcal{V}^{\tau+1}_{N\cup i} - \mathcal{V}^{\tau+1}_N = \mathcal{R}^{\tau+1}_{N\cup i} - \mathcal{R}^{\tau+1}_N + \sum_{s'_N = (s'_1, \ldots, s'_{|N|})} \left[\prod_{j\in N} \mathcal{T}^{\tau+1}_j \right] \cdot \cdot \left[\sum_{s'_i} \mathcal{T}^{\tau+1}_i \mathcal{V'}^{\tau}_{N\cup i} - \mathcal{V'}^{\tau}_N \right]$$

Since $\sum_{s'_i} T_i(s_i, \pi^{\tau+1}_i(s_i), s'_i) = 1$ or $\sum_{s'_i} \mathcal{T}^{\tau+1}_i = 1$, we have

$$= \mathcal{R}^{\tau+1}_{N\cup i} - \mathcal{R}^{\tau+1}_N + \sum_{s'_N = (s'_1, \ldots, s'_{|N|})} \Big[\prod_{j\in N} \mathcal{T}^{\tau+1}_j \Big] \cdot \cdot \left[\sum_{s'_i} \mathcal{T}^{\tau+1}_i \cdot \Big[\mathcal{V'}^{\tau}_{N\cup i} - \mathcal{V'}^{\tau}_N \Big] \right] \tag{5}$$

From Equation 4 and submodularity of reward function, we have:

$$\leq \mathcal{R}^{\tau+1}_{M\cup i} - \mathcal{R}^{\tau+1}_M + \sum_{s'_N = (s'_1, \ldots, s'_{|N|})} \Big[\prod_{j\in N} \mathcal{T}^{\tau+1}_j \Big] \cdot \cdot \Big[\sum_{s'_i} \mathcal{T}^{\tau+1}_i \big[\mathcal{V'}^{\tau}_{M\cup i} - \mathcal{V'}^{\tau}_M \big] \Big]$$

$$\leq \mathcal{R}^{\tau+1}_{M\cup i} - \mathcal{R}^{\tau+1}_M + \sum_{s'_M = (s'_1, \ldots, s'_{|M|})} \Big[\prod_{j\in M} \mathcal{T}^{\tau+1}_j \Big] \cdot \cdot \sum_{s'_{N\setminus M}} \prod_{j\in N\setminus M} \mathcal{T}^{\tau+1}_j \Big[\sum_{s'_i} \mathcal{T}^{\tau+1}_i \big[\mathcal{V'}^{\tau}_{M\cup i} - \mathcal{V'}^{\tau}_M \big] \Big]$$

$$\leq \mathcal{R}^{\tau+1}_{M\cup i} - \mathcal{R}^{\tau+1}_M + \sum_{s'_{M\cup i}} \Big[\prod_{j\in M\cup i} \mathcal{T}^{\tau+1}_j \Big] \cdot \big[\mathcal{V'}^{\tau}_{M\cup i} - \mathcal{V'}^{\tau}_M \big] \cdot \sum_{s'_{N\setminus M}} \prod_{j\in N\setminus M} \mathcal{T}^{\tau+1}_j$$

Since $\sum_{s'_{N\setminus M}} \prod_{j\in N\setminus M} \mathcal{T}^{\tau+1}_j = 1$, we conclude

$$\mathcal{V}^{\tau+1}_{N\cup i} - \mathcal{V}^{\tau+1}_N \leq \mathcal{V}^{\tau+1}_{M\cup i} - \mathcal{V}^{\tau+1}_M \qquad \blacksquare$$

Proposition 2. *For a TI-Dec-MDP, $V^H()$ is monotonically increasing if R is monotonically increasing.*

Proof Sketch. Again, we employ mathematical induction on time horizon H. Since R is monotonically increasing, base case of H = 1 is trivially satisfied. We can make the assumption that V^H is monotonically increasing for $H = \tau$. For $H = \tau + 1$, we have from Equation (5):

$$\mathcal{V}^{\tau+1}_{N\cup i} - \mathcal{V}^{\tau+1}_N = \mathcal{R}^{\tau+1}_{N\cup i} - \mathcal{R}^{\tau+1}_N + \sum_{s'_N = (s'_1, \ldots, s'_{|N|})} \left[\prod_{j\in N} \mathcal{T}^{\tau+1}_j \right] \cdot \cdot \left[\sum_{s'_i} \mathcal{T}^{\tau+1}_i \Big[\mathcal{V'}^{\tau}_{N\cup i} - \mathcal{V'}^{\tau}_N \Big] \right]$$

From the proposition statement: $\mathcal{R}^{\tau+1}_{N\cup i} - \mathcal{R}^{\tau+1}_{N} \geq 0$. Furthermore, from assumption for $H = \tau$, $\mathcal{V'}^{\tau}_{N\cup i} - \mathcal{V'}^{\tau}_{N} \geq 0$. Therefore,

$$\mathcal{V}^{\tau+1}_{N\cup i} - \mathcal{V}^{\tau+1}_{N} \geq 0 \qquad \blacksquare$$

3.1 Submodular TOI-Dec-POMDPs

Another important model of relevance in decentralized stochastic planning, where submodularity of reward plays a significant role is TOI-Dec-POMDP. In Transition Observation Independent Dec-POMDPs, for each agent i we have a set of observations, Ω_i and an individual observation function O_i in addition to the TI-Dec-MDP tuple. $O_i(s'_i, a_i, \omega_i)$ refers to the probability of receiving observation ω_i on taking action a_i to arrive in state s'_i. Value function for this case is defined as follows:

$$V(\vec{\pi}) = \sum_s b(s) \cdot V^t(s, \vec{\pi}, \vec{\omega})$$

$$V^t(s, \vec{\pi}, \vec{\omega}) = R\Big(s, \langle \pi^t_1(\vec{\omega}_1), \ldots, \pi^t_{|Ag|}(\vec{\omega}_{|Ag|})\rangle\Big) + \sum_{s',\omega'} \prod_{i\in Ag} T_i\Big(s_i, \pi^t_i(s_i), s'_i\Big) \cdot \prod_{i\in Ag} O_i\Big(s'_i, \pi^t_i(\vec{\omega}_i), \omega'_i\Big) \cdot V^{t-1}(s', \vec{\pi}, \vec{\omega}')$$

ND-POMDP (Nair et al. 2005) is a subcategory of the TOI-Dec-POMDP model. In ND-POMDPs, apart from transition and observation independence, there is also a fixed network structure of interactions among agents.

Proposition 3. *For a TOI-Dec-POMDP, $V^H()$ is monotonically increasing and submodular if R is monotonically increasing and submodular.*

Proof Sketch. We employ mathematical induction similar to the proofs for TI-Dec-MDP. ■

3.2 Submodular TI-SG

Both TOI-Dec-POMDPs and TI-Dec-MDPs are cooperative models for decentralized stochastic planning. We now consider Transition Independent Stochastic Games (TI-SGs) (Parthasarathy and Sinha 1989; Eitan Altman and Miller 2005) model, a competitive model that is a sub-class of the well-known Stochastic Games model (Shapley 1953). The tuple for TI-SG is given by:

$$\Big\langle Ag, S, A, \{T_i\}_{i\in Ag}, \{R_i\}_{i\in Ag}, H\Big\rangle$$

Except for the reward function, the elements in this tuple carry the same meaning and representation as in TI-Dec-MDPs. Unlike in TI-Dec-MDPs, reward function is defined for each agent i separately and is dependent on actions of all other agents. Formally, private reward function for agent i, $R_i(s, a)$ is defined over joint state and dependent on the joint action $a = \langle a_1, a_2, \ldots, a_{|Ag|}\rangle$.

Private value of an agent i for a joint policy $\pi = \langle \pi_1, \pi_2, \ldots\rangle$ is defined as follows:

$$V^t_i(s, \vec{\pi}) = R_i(s, \Big\langle \pi^t_1(s_1), \ldots, \pi^t_{|Ag|}(s_{Ag})\Big\rangle) + \sum_{s'} \Big[\prod_{k\in Ag} T_k\Big(s_k, \pi^t_k(s_k), s'_k\Big)\Big] \cdot V^{t-1}_i(s', \vec{\pi})$$

Social value of all agents for a joint policy π is the sum of private values of all agents: $V^H(s, \vec{\pi}) = \sum_i V^H_i(s, \vec{\pi})$

Given the selfish nature of individual agents, the solution concept of interest is no longer optimality. Instead, we are interested in Nash equilibrium solutions. The key result for Nash Equilibrium solutions in TI-SGs is provided In Proposition 6. We show that any Nash equilibrium solution is at least 50% of the social welfare optimal solution (optimal social value policy). Proof for Proposition 6 requires two key major steps: (i) showing social value is submodular; (ii) showing private value for any agent is as much as its *Vickrey* value. We use the following short forms in proofs:

$$\mathcal{R}^t_i = R_i\left(s, \Big\langle \pi^t_1(s_1), \ldots, \pi^t_{|Ag|}(s_{Ag})\Big\rangle\right)$$

$$\mathcal{T}^t_i = T_i(s_i, \pi^t_i(s_i), s'_i)$$

Proposition 4. *In TI-SGs, the social value function is monotone submodular given sum of individual agent rewards is monotone submodular.*

Proof Sketch. As with proofs for Proposition 1 and Proposition 2, we will employ mathematical induction. ■

Before we describe the next proposition. *Vickrey* reward for an agent is defined as the marginal contribution of that agent, i.e., difference in joint reward with and without the agent. In the following proposition we show that if the private reward for an agent is greater than the *Vickrey* reward with respect to that agent, then the private value for that agent is also greater than the *Vickrey* value with respect to that agent.

Proposition 5. *In TI-SGs,*

$$R_i(s_{Ag}, a_{Ag}) \geq \sum_{k\in Ag} R_k(s_{Ag}, a_{Ag}) - \sum_{k\in Ag\setminus i} R_k(s_{Ag\setminus i}, a_{Ag\setminus i})$$

$$\implies V^H_i(s_{Ag}, \vec{\pi}_{Ag}) \geq \sum_{k\in Ag} V^H_k(s_{Ag}, \vec{\pi}_{Ag}) - \sum_{k\in Ag\setminus i} V^H_k(s_{Ag\setminus i}, \vec{\pi}_{Ag\setminus i})$$

Proof. We prove this using mathematical induction on time horizon, H. From the proposition statement, it naturally holds for the base case of $H = 1$. Let us assume it holds for $H = \tau$, i.e.,

$$V^\tau_i(s_{Ag}, \vec{\pi}_{Ag}) \geq \sum_{k\in Ag} V^\tau_k(s_{Ag}, \vec{\pi}_{Ag}) - \sum_{k\in Ag\setminus i} V^\tau_k(s_{Ag\setminus i}, \vec{\pi}_{Ag\setminus i})$$

For $H = \tau + 1$:

$$V^{\tau+1}_i(s_{Ag}, \vec{\pi}_{Ag}) = \mathcal{R}^{\tau+1}_i + \sum_{s'_{Ag}} \Big[\prod_{k\in Ag} \mathcal{T}^{\tau+1}_k\Big] \cdot V^\tau_i(s'_{Ag}, \vec{\pi}_{Ag})$$

By assumption for $H = \tau$ and proposition statement, we have:

$$\geq \sum_{k\in Ag} R_k(s_{Ag}, a_{Ag}) - \sum_{k\in Ag\setminus i} R_k(s_{Ag\setminus i}, a_{Ag\setminus i}) +$$

$$\sum_{s'_{Ag}} \left[\prod_k \mathcal{T}_k^t\right] \cdot \left[\sum_{k \in Ag} V_k^\tau(s_{Ag}, \vec{\pi}_{Ag}) - \sum_{k \in Ag \setminus i} V_k^\tau(s_{Ag \setminus i}, \vec{\pi}_{Ag \setminus i})\right]$$

Multiplying product of transition functions to the value functions and grouping terms gives us the final result

$$\geq \sum_{k \in Ag} V_k^{\tau+1}(s_{Ag}, \vec{\pi}_{Ag}) - \sum_{k \in Ag \setminus i} V_k^{\tau+1}(s_{Ag \setminus i}, \vec{\pi}_{Ag \setminus i}) \blacksquare$$

Definition 3. *Price of anarchy is defined as the ratio of social optimal (maximum social value for any joint policy) and any nash equilibrium solution.*

Proposition 6. *Price of anarchy (ratio of Social Optimal and any Nash Equilibrium) in TI-SGs of interest (monotone submodular social value function and private value greater than Vickrey utility for every agent) is upper bounded by 2.*

Proof Sketch. A basic utility/value system is characterized by three key properties:

- Social and private value functions are measured in the same unit: Both social and private values are based on the same reward unit in TI-SGs of interest.
- Social value function is monotone and submodular: Proposition 4 shows it for relevant TI-SGs.
- Private value function is as much as the *Vickrey* value: Proposition 5 demonstrates this for relevant TI-SGs.

Vetta *et al.* (Vetta 2002) have shown that in basic utility systems, the price of anarchy is upper bounded by 2. Given that TI-SGs of interest satisfy the three key properties of basic utility systems, price of anarchy is upper bounded by 2. ■

4 Approaches

In this section, we provide enhancements to existing approaches for solving TI-Dec-MDPs[2], so that we can provide and investigate *posteriori* guarantees.

The goal in TI-Dec-MDPs is to obtain a correct joint policy (i.e., a joint policy where there is exactly one policy for each agent) that maximizes the expected value. Formally, the goal is to maximize $V(\pi)$ given the partition matroid $\Gamma = (\Pi, I)$ where $I = \{X \subseteq \Pi : |X \cap \Pi_i| = 1\}$. Intuitively, the partition matroid enforces that we can only have one policy for each agent which results in a valid joint-policy. Partition matroid has also been used previously to compute the joint-policy in (Kumar and Zilberstein 2009). In (Kumar and Zilberstein 2009), partitions correspond to different states in the state-space. In our case, we have one partition for each agent i, and elements of this partition correspond to the set of all valid policies for the agent i.

Here we provide two types of approaches extending on existing work: (a) A lazy extension of the greedy algorithm that extends on the work by Shieh *et al.* (Shieh et al. 2014) to improve scalability; (b) An approach that exploits anonymity and submodularity by extending on the work by Varakantham *et al.* (2014).

[2]For TI-SGs, since well-known iterative best response and fictitious play approaches can be employed to obtain equilibrium solutions, we do not provide any new approaches. For TOI-Dec-MDPs, greedy approaches mentioned in this section are applicable. However, those approaches rely on solving single agent decision problem optimally so as to guarantee the *a priori* bound. This is a challenge in the case of POMDPs, unless specific size controllers are employed (Amato, Bonet, and Zilberstein 2010), in which case guarantee is with respect to optimal for that size controller.

Algorithm 1: GREEDY$\left(\left\langle Ag, S, A, \{T_i\}_{i \in Ag}, R, H, \alpha \right\rangle\right)$

1: $Z \leftarrow \emptyset$
2: $\vec{\pi}_i^* \leftarrow \emptyset, \forall i$
3: **repeat**
4: **for all** $i \in Ag \setminus Z$ **do**
5: $\vec{\pi}_i^* \leftarrow \max_{\vec{\pi}_i} V_i(\vec{\pi}_i, \alpha_i^0 | \vec{\pi}_Z^*)$
6: $\langle i^*, V_{i^*} \rangle \leftarrow \max_{i \in Ag \setminus Z} V_i(\vec{\pi}_i^*, \alpha_i^0 | \vec{\pi}_Z)$
7: $Z \leftarrow Z \cup \{i^*\}$
8: **until** $Ag \setminus Z = \emptyset$
9: **return** $\vec{\pi}^* \leftarrow \{\vec{\pi}_i^*\}_{i \in Ag}$

4.1 Greedy and Lazy Greedy

Greedy algorithm has previously been explored in the context of Dec-MDPs (Shieh et al. 2014; Agrawal, Varakantham, and Yeoh 2016) and Dec-POMDPs (Varakantham et al. 2012). Algorithm 1 provides the pseudo code for the greedy algorithm. It builds the solution set incrementally by adding policy for a different agent at each iteration. Initially, we start with an empty solution set (line 1). At every iteration, for each of the agents that have not yet been assigned a policy (line 4), we compute a policy with the highest marginal value given the current solution set (line 5). Among those highest marginal value policies, we pick the one with the highest value and add that to the solution set (lines 6-7). This process continues for $|Ag|$ iterations (line 8), when every agent is assigned exactly one policy.

Highest marginal value policy for an agent given the current solution set (line 5) is computed by constructing and solving an MDP. At an arbitrary iteration k, let the solution set contain policies, $\{\vec{\pi}_z\}_{z \in Z}$ where $|Z| = k$. We first compute the probability distribution, $Pr^t(s_Z)$ over the joint state s_Z at each time step, t corresponding to the current solution set policies, $\{\vec{\pi}_z\}_{z \in Z}$. The most efficient way to derive an accurate estimate of $Pr^t(s_Z)$ is by simulating the policies, $\{\vec{\pi}_z\}_{z \in Z}$ many times (tens of thousands of simulations). Once $Pr^t(s_Z)$ is computed, we identify the augmented MDP for agent i from the original TI-Dec-MDP as follows:

$$\langle S_i, A_i, T_i, \mathcal{R}_i \rangle$$

S_i, A_i, T_i are directly taken from the original TI-Dec-MDP tuple. As for the reward function, we have:

$$\mathcal{R}_i^t(s_i, a_i) = \sum_{s_Z} Pr^t(s_Z) \cdot R(\langle s_Z, s_i \rangle, \langle \pi_Z^t(s_Z), a_i \rangle)$$

We then optimally solve this MDP to obtain a policy with the highest marginal value for i.

Proposition 7. *(Fisher, Nemhauser, and Wolsey 1978): Greedy algorithm for maximizing a monotone submodular function subject to a partition matroid yields solutions that are at least 50% of the optimal solution.*

This provides an *a priori* guarantee on the quality of solutions obtained using the greedy algorithm. While the *a priori* guarantee bounds the worst case loss, in many cases, the solution provided by greedy is much better than 50% of optimal. In fact, we can provide much tighter *post hoc* quality guarantees after obtaining the final solution. Intuitively, by adding the marginal value of the best policy for every agent given the solution set obtained by greedy, we obtain an upper bound on the optimal solution. We provide the formal characterization of this online bound below:

Proposition 8. *For any joint policy, $\vec{\pi}$:*

$$V(\vec{\pi}^*) \leq V(\vec{\pi}) + \sum_{i \in Ag} \delta_i(\vec{\pi})$$

where $\delta_i(\vec{\pi}) = \max_{\vec{\pi}_i \in \Pi_i} V(\vec{\pi} \cup \pi_i) - V(\vec{\pi})$

Proof. The proof for the above proposition is a direct result of applying the greedy algorithm to a partition matroid (Goundan and Schulz 2007). For any monotone submodular function, $g : 2^{\Pi} \rightarrow \mathbb{R}$:

$$g(Z^*) \leq g(Z) + \sum_{e \in Z* \setminus Z} \delta_e(Z)$$

In the context of value function, this translates to

$$V(\vec{\pi}^*) \leq V(\vec{\pi}) + \sum_{\pi_k \in \vec{\pi}^* \setminus \vec{\pi}} \delta_{\pi_k}(\vec{\pi})$$

In the worst case, $\vec{\pi}$ may have completely different policies for every agent when compared to the individual policies in $\vec{\pi}^*$. While we do not know the composition of $\vec{\pi}^*$, by adding the individual policies that yield best marginal values for each agent, we get an upper bound on the value of the optimal policy. Hence,

$$V(\vec{\pi}^*) \leq V(\vec{\pi}) + \sum_{i \in Ag} \delta_i(\vec{\pi}) \blacksquare$$

Since greedy algorithm has to evaluate marginal value for every available agent in each and every iteration, scalability of greedy decreases significantly with increase in number of agents. Therefore, we pursue a lazy greedy method that has all the same properties of greedy, but is more efficient.

Unlike the original greedy algorithm, in lazy greedy algorithm (Minoux 1978) we do not need to test all available agents (line 5 of Algorithm 1) in order to find the optimal one. Submodularity property of value/objective function guarantees that the marginal gain for an agent is always equal to or lower than the previous iteration. Therefore, when marginal gain of one of the agents is evaluated and its marginal gain is higher than marginal gains of a certain subset of agents from the previous iteration, then those subset of agents do not have to calculate their marginal gain. In fact, to ensure we reduce marginal gain computation as much as possible, we maintain a sorted order of marginal values at each iteration and we compute marginal gain according to the descending order in the next iteration. Finally, we also exploit homogeneity in agent models by ensuring marginal value is evaluated for one agent of each type.

Algorithm 2: PWCSUBMOD()

1: $\max_x \sum_{s,t,k} y_k^t(s) V_k^t(s)$ **s.t**
2: $\sum_a x^t(s,a) - \sum_{s',a} x^{t-1}(s',a)\varphi(s',a,s) = \alpha^t(s) \quad \forall s,t$
3: $\sum_k y_k^t(s) = 1$
4: $|P| \cdot \sum_a x^t(s,a) \geq \sum_k y_k^t(s) \cdot \check{d}_k$
5: $|P| \cdot \sum_a x^t(s,a) \leq \sum_k y_k^t(s) \cdot \hat{d}_k$
6: $x^t(s,a) \in [0,1] \qquad \forall s,a,t$

4.2 Linear Optimization to Exploit Submodularity

Varakantham *et al.* (2014) introduced linear optimization formulations for approximately solving factored Dec-MDPs with homogeneous agents and anonymous interactions. Homogeneity refers to agents having similar models and anonymity in interactions refers to joint reward and/or transition functions being dependent on the number of agents and not on the identity of the agents. Anonymity and homogeneity were exploited in three ways to significantly improve scalability with respect to number of agents: (a) Employing reward and transition functions, that are parameterized on the number of agents; (b) Approximating expected reward in a state, action pair by using reward for expected number of agents in that state, action pair; (c) Extending dual formulation for solving MDPs to handle parametric reward and transition functions.

Building on the approach by Varakantham *et al.* (2014), we employ the formulation that considers piecewise constant reward functions, by approximating submodular reward using a piecewise constant reward function. In domains of interest (security games and sensor networks), due to submodularity, irrespective of the number of agents present in the system, reward function becomes constant after only a few agents (e.g., for efficiency of even 0.7 in security games, with 3 agents reward is close to 1, the maximum value). This reduces the number of PWC components required to represent the reward function.

We provide the formulation in Algorithm 2 where $x^t(s,a)$ represents the flow of agents, $y_k^t(s)$ represents whether component k is active at time t for state s, and $V_k^t(s)$ represents the value for component k at time t and state s. Submodularity is exploited in determining the intervals $[\check{d}_k, \hat{d}_k]$ for each component and also in reducing the number of intervals required.

5 Experimental Results

We focus primarily on the cooperative case with TI-Dec-MDPs to verify the online guarantees provided by our approaches.

5.1 Security Games

For this domain, we experiment with the problem domain provided by Shieh *et al.* (Shieh et al. 2014). In this domain, there are a set of targets on a rail network which have to be

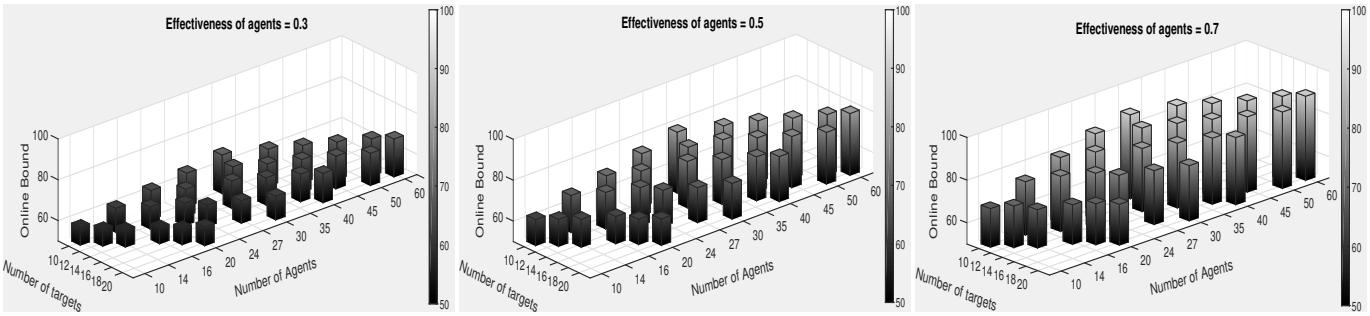

Figure 1: Online Bound of Lazy Greedy

defended by a set of decentralized (yet cooperative) defenders in the presence of transition uncertainty. We described the submodularity of the reward function in Section 3 around Equation 3. This domain also has anonymous interactions, as the reward is dependent on number of agents at a target (and not on the specific set of agents), hence our second approach PWCSUBMOD is also relevant. We experimented with the following values of the different parameters: (i) Number of targets = {10, 12, 14, 16, 18, 20}. Targets are different points in a network; (ii) Number of agents was varied from 10-60, depending on the number of targets; (iii) Effectiveness parameter, $\epsilon = \{0.3, 0.5, 0.7\}$.

Irrespective of the number of targets, agents or effectiveness parameter,PWCSUBMOD obtained the solution in a few seconds. We primarily experimented with lazy greedy, as greedy was unable to solve problems beyond the smallest ones and lazy greedy is more efficient without losing on solution quality. Lazy greedy took a few seconds for the smallest scenario (10 targets with 10 agents) to 12000 seconds for the largest scenario (20 targets and 60 agents).

Figure 1 provides the comparisons for the online (*posteriori*) guarantees obtained for the solutions generated by lazy greedy. We employ Proposition 8 to generate the online guarantee for any given solution. As can be seen the online guarantees are significantly better than the *a priori* guarantees (of 50% from optimal), with a best case of even 90% from optimal. To avoid clutter, we do not show the online guarantees provided by policies generated using PWCSUBMOD in the same graph. However, PWCSUBMOD fared slightly worse than lazy greedy in terms of guarantees. In the best case, PWCSUBMOD provided a guarantee that was 3% less and in the worst case, it was 9% less.

5.2 Sensor Network

We consider a sensor network domain that is a variant of the one employed in literature (Nair et al. 2005; Kumar and Zilberstein 2011), where the reward of tracking a target is dependent on number of sensors tracking the target. This reward function is submodular, as tracking accuracy is submodular in the number of tracking agents. More importantly, we consider n-ary interactions (any number of agents can track a target), as opposed to binary interactions (maximum 2 sensors track a target) employed in the literature (Nair et al. 2005; Kumar and Zilberstein 2011). Multiple sensors are fixed at various locations on a grid to track moving targets. All the targets move stochastically (according to some fixed distribution) in the grid . Movements of targets are independent of sensor actions. In this domain, there is a global unaffected state that represents the target locations. For instance, if there are 2 targets moving on fix paths of sizes 5 and 10 then there are 5*10= 50 possible global states.

Grid Size	Sensors, Targets, Global States	ϵ		
		0.7	0.5	0.3
2*2	2, 1, 4	68.9	63.8	58.1
2*2	4, 1, 4	76.1	68.0	61.6
3*3	4, 2, 5*5	64.7	62.3	58.3
3*3	3, 2, 5*5	68.5	63.9	58.3
5*5	5, 2, 6*6	71.4	63.3	57.9
5*5	5, 1, 10	65.6	64.7	58.4
5*5	4, 2, 6*6	68.2	62.2	57.7
10*5	6, 3, 14*10*10	63.6	61.0	57.2
10*5	6, 5, 6*5*5*5*5	64.1	61.3	57.0

Table 1: Online Bounds on Sensor Network Problems.

We experimented by varying grid size, number of Sensors, number of moving targets, global states (decided based on paths of the moving targets) and effectiveness parameter value, ϵ. Similar to the case of security games, even here, PWCSUBMOD[3] was significantly faster than lazy greedy. For lazy greedy, runtime linearly increased with the settings from being less than a second for the smallest setting to about 4800 seconds for the largest setting. PWCSUBMOD on the other hand was able to generate solutions in a few seconds and was even able to scale to much bigger problems (not shown here) with many more agents.

Unlike in the case of security games, PWCSUBMOD typically provided slightly better solutions than lazy greedy. However, in terms of online guarantees, they were typically the same as provided in Table 1. For different effectiveness parameter values, guarantees provided varied from 57% to 76%. While not as significant as with security games, online guarantees are still better than the *a priori* guarantees of 50% from optimal.

[3]It should be noted that for sensor networks, agents are not homogenous, so we consider an adaptation of the formulation in Algorithm 1 that considers multiple types.

6 Acknowledgements

This work was partially supported by the Singapore National Research Foundation through the Singapore-MIT Alliance for Research and Technology (SMART) Centre for Future Urban Mobility (FM).

References

Agrawal, P.; Varakantham, P.; and Yeoh, W. 2016. Scalable greedy algorithms for task/resource constrained multi-agent stochastic planning. In *International Joint Conference on Artificial Intelligence, IJCAI-2016*.

Amato, C.; Bonet, B.; and Zilberstein, S. 2010. Finite-state controllers based on mealy machines for centralized and decentralized pomdps. In *AAAI Conference on Artificial Intelligence, AAAI-2010*.

Becker, R.; Zilberstein, S.; Lesser, V.; and Goldman, C. V. 2004. Solving transition independent decentralized markov decision processes. *Journal of Artificial Intelligence Research* 22:423–455.

Chapman, A. C., and Varakantham, P. 2014. Marginal contribution stochastic games for dynamic resource allocation. In *International Conference on Principles and Practice of Multi-Agent Systems*, 333–340. Springer.

Dibangoye, J. S.; Amato, C.; and Doniec, A. 2012. Scaling up decentralized mdps through heuristic search. *arXiv preprint arXiv:1210.4865*.

Eitan Altman, Konstantin Avrachenkov, R. M., and Miller, G. 2005. Zero-sum constrained stochastic games with independent state processes. *Mathematical Methods of Operations Research* 62(3):375–386.

Fisher, M. L.; Nemhauser, G. L.; and Wolsey, L. A. 1978. An analysis of approximations for maxi- mizing submodular set functions - ii. *Math. Prog. Study* 8:73–87.

Goundan, P. R., and Schulz, A. S. 2007. Revisiting the greedy approach to submodular set function maximization. *Optimization online*.

Kumar, A., and Zilberstein, S. 2009. Event-detecting multi-agent mdps: Complexity and constant-factor approximation. In *International Joint Conference on Artificial Intelligence*, 201–207.

Kumar, A., and Zilberstein, S. 2011. Message-passing algorithms for large structured decentralized POMDPs. In *Proceedings of the Tenth International Conference on Autonomous Agents and Multiagent Systems*, 1087–1088.

Kumar, A.; Zilberstein, S.; and Toussaint, M. 2015. Probabilistic inference techniques for scalable multiagent decision making. *Journal of Artificial Intelligence Research* 53:223–270.

Melo, F. S., and Veloso, M. 2011. Decentralized mdps with sparse interactions. *Artificial Intelligence* 175(11):1757–1789.

Minoux, M. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*. Springer. 234–243.

Mostafa, H., and Lesser, V. 2009. Offline planning for communication by exploiting structured interactions in decentralized mdps. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, volume 2, 193–200. IET.

Nair, R.; Varakantham, P.; Tambe, M.; and Yokoo, M. 2005. Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In *AAAI*, volume 5, 133–139.

Parthasarathy, T., and Sinha, S. 1989. Existence of stationary equilibrium strategies in non-zero sum discounted stochastic games with uncountable state space and state-independent transitions. *International Journal of Game Theory* 18(2):189–194.

Shapley, L. S. 1953. Stochastic games. *Proceedings of the national academy of sciences* 39(10):1095–1100.

Shieh, E.; Jiang, A.; Yadav, A.; Varakantham, P.; and Tambe, M. 2014. Unleashing dec-mdps in security games: Enabling effective defender teamworks. In *ECAI'14*.

Varakantham, P.; Adulyasak, Y.; and Jaillet, P. 2014. Decentralized stochastic planning with anonymity in interactions. In *Proc. of the AAAI Conference on Artificial Intelligence*, 2505–2512.

Varakantham, P.; Kwak, J. Y.; Taylor, M.; Marecki, J.; Scerri, P.; and Tambe, M. 2009. Exploiting coordination locales in distributed POMDPs via social model shaping. In *Nineteenth International Conference on Automated Planning and Scheduling*, 313–320.

Varakantham, P.; Yeoh, W.; Velagapudi, P.; Sycara, K.; and Scerri, P. 2012. Prioritized shaping of models for solving dec-pomdps. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, 1269–1270. International Foundation for Autonomous Agents and Multiagent Systems.

Velagapudi, P.; Varakantham, P.; Sycara, K.; and Scerri, P. 2011. Distributed model shaping for scaling to decentralized pomdps with hundreds of agents. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, 955–962. International Foundation for Autonomous Agents and Multiagent Systems.

Vetta, A. 2002. Nash equilibria in competitive societies, with applications to facility location, traffic routing and auctions. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, FOCS '02, 416–. Washington, DC, USA: IEEE Computer Society.

Yash Satsangi, S. W., and Oliehoek, F. 2015. Exploiting submodular value functions for faster dynamic sensor selection. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 3356–3363.

Yin, Z., and Tambe, M. 2011. Continuous time planning for multiagent teams with temporal constraints. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 465–471.