Information Gathering in Decentralized POMDPs by Policy Graph Improvement

Mikko Lauri
University of Hamburg
Hamburg, Germany
lauri@informatik.uni-hamburg.de

Joni Pajarinen
TU Darmstadt
Darmstadt, Germany
pajarinen@ias.tu-darmstadt.de

Jan Peters
TU Darmstadt
Darmstadt, Germany
peters@ias.tu-darmstadt.de

ABSTRACT

Decentralized policies for information gathering are required when multiple autonomous agents are deployed to collect data about a phenomenon of interest without the ability to communicate. Decentralized partially observable Markov decision processes (Dec-POMDPs) are a general, principled model well-suited for such decentralized multiagent decision-making problems. In this paper, we investigate Dec-POMDPs for decentralized information gathering problems. An optimal solution of a Dec-POMDP maximizes the expected sum of rewards over time. To encourage information gathering, we set the reward as a function of the agents' state information, for example the negative Shannon entropy. We prove that if the reward is convex, then the finite-horizon value function of the corresponding Dec-POMDP is also convex. We propose the first heuristic algorithm for information gathering Dec-POMDPs, and empirically prove its effectiveness by solving problems an order of magnitude larger than previous state-of-the-art.

CCS CONCEPTS

• **Theory of computation → Markov decision processes**; Randomized local search; • **Computing methodologies → Planning under uncertainty**; **Multi-agent planning**;

KEYWORDS

decentralized POMDPs; multi-agent planning; planning under uncertainty; information theory

ACM Reference Format:
Mikko Lauri, Joni Pajarinen, and Jan Peters. 2019. Information Gathering in Decentralized POMDPs by Policy Graph Improvement. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

Autonomous agents and robots can be deployed in information gathering tasks in environments where human presence is either undesirable or infeasible. Examples include monitoring of deep ocean conditions, or space exploration. It may be desirable to deploy a team of agents, e.g., due to the large scope of the task at hand, resulting in a decentralized information gathering task.

Some recent works, e.g., [5, 21], tackle decentralized information gathering while assuming perfect, instantaneous communication between agents, while centrally planning how the agents should act. In terms of communication, we approach the problem from the other extreme as a decentralized partially observable Markov decision process (Dec-POMDP) [15]. In a Dec-POMDP, no explicit communication between the agents is assumed[1]. Each agent acts independently, without knowing what the other agents have perceived or how they have acted.

Informally, a Dec-POMDP model consists of a set of agents in an environment with a hidden state. Each agent has its own set of local actions, and a set of local observations it may observe. Markovian state transition and observation processes conditioned on the agents' actions and the state determine the relative likelihoods of subsequent states and observations. A reward function determines the utility of executing any action in any state. The objective is to centrally design optimal control policies for each agent that maximize the expected sum of rewards over a finite horizon of time. The control policy of each agent depends only on the past actions and observations of that agent, hence no communication during execution of the policies is required. However, as policies are planned centrally, it is possible to reasong about the joint information state of all the agents. It is thus possible to calculate probability distributions over the state, also known as joint beliefs.

A decentralized information gathering task differs from other multiagent control tasks by the lack of a goal state. It is not the purpose of the agents to execute actions that reach a particular state, but rather to observe the environment in a manner that provides the greatest amount of information while satisfying operational constraints. As the objective is information acquisition, the reward function depends on the joint belief of the agents. Convex functions of a probability mass function naturally model certainty [6], and have been proposed in the context of single-agent POMDPs [3] and Dec-POMDPs [10]. However, to the best of our knowledge no heuristic or approximate algorithms for convex reward Dec-POMDPs have been proposed, and no theoretical results on the properties of such Dec-POMDPs exist in the literature.

In this paper, we propose the first heuristic algorithm for Dec-POMDPs with a convex reward. We prove the value function of such Dec-POMDPs is convex, generalizing the similar result for single-agent POMDPs [3]. The Dec-POMDP generalizes other decision-making formalisms such as multi-agent POMDPs and Dec-MDPs [4]. Thus, our results also apply to these special cases removing parts required by the more general Dec-POMDP.

Our paper has three contributions. Firstly, we prove that in Dec-POMDPs where the reward is a convex function of the joint belief, the value function of any finite horizon policy is convex in the joint belief. Secondly, we propose the first heuristic algorithm for Dec-POMDPs with a reward that is a function of the agents' joint state information. The algorithm is based on iterative improvement

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada. 2019.

[1] If desired, communication may be included into the Dec-POMDP model [24, 27].

of the value of fixed-size policy graphs. We derive a lower bound that may be improved instead of the exact value, leading to computational speed-ups. Thirdly, we experimentally verify the feasibility and usefulness of our algorithm. For Dec-POMDPs with a state information dependent reward, we find policies for problems an order of magnitude larger than previously.

The paper is organized as follows. We review related work in Section 2. In Section 3, we define our Dec-POMDP problem and introduce notation and definitions. Section 4 derives the value of a policy graph node. In Section 5, we prove convexity of the value in a Dec-POMDP where the reward is a convex function of the state information. Section 6 introduces our heuristic policy improvement algorithm. Experimental results are presented in Section 7, and concluding remarks are provided in Section 8.

2 RELATED WORK

Computationally finding an optimal decentralized policy for a finite-horizon Dec-POMDP is NEXP-complete [4]. Exact algorithms for Dec-POMDPs are usually based either on backwards in time dynamic programming [9], forwards in time heuristic search [17, 26], or on exploiting the inherent connection of Dec-POMDPs to non-observable Markov decision processes [7, 11]. Approximate and heuristic methods have been proposed, e.g., based on finding locally optimal "best response" policies for each agent [13], memory-bounded dynamic programming [22], cross-entropy optimization over the space of policies [16], or monotone iterative improvement of fixed-size policies [19]. Algorithms for special cases such as goal-achievement Dec-POMDPs [2] and factored Dec-POMDPs, e.g., [18], have also been proposed. Structural properties, such as transition, observation, and reward independence between the agents, can also be leveraged and may even result in a problem with a lesser computational complexity [1]. Some Dec-POMDP algorithms [17] take advantage of plan-time sufficient statistics, which are joint distributions over the hidden state and the histories of the agents' actions and observations [14]. The sufficient statistics provide a means to reason about possible distributions over the hidden state, also called joint beliefs, reached under a given policy.

The expected value of a reward function that depends on the hidden state and action is a linear function of the joint belief. These types of rewards are standard in Dec-POMDPs. In the context of single-agent POMDPs, Araya-López et al. [3] argue that information gathering tasks are naturally formulated using a reward function that is a convex function of the state information and introduce the ρPOMDP model with such a reward. This enables application of, e.g., the negative Shannon entropy of the state information as a component of the reward function. Under certain conditions, an optimal value function of a ρPOMDP is Lipschitz-continuous [8] which may be exploited in a solution algorithm. An alternative formulation for information gathering in single-agent POMDPs is presented in [25], and its connection to ρPOMDPs is characterized in [20]. Recently, [10] proposes an extension of the ideas presented in [3] to the Dec-POMDP setting. Entropy is applied in the reward function to encourage information gathering. Problem domains with up to 25 states and 5 actions per agent are solved with an exact algorithm.

In this paper, we present the first heuristic algorithm for Dec-POMDPs with rewards that depend non-linearly on the joint belief. Our algorithm is based on the combination of the idea of using a fixed-size policy represented as a graph [19] with plan-time sufficient statistics [14] to determine joint beliefs at the policy graph nodes. The local policy at each policy graph node is then iteratively improved, monotonically improving the value of the node. We show that if the reward function is convex in the joint belief, then the value function of any finite-horizon Dec-POMDP policy is convex as well. This is a generalization of a similar result known for single-agent POMDPs [3]. From this property, we obtain a lower bound for the value of a policy that we empirically show improves the efficiency of our algorithm. Compared to prior state-of-the-art in Dec-POMDPs with convex rewards [10], our algorithm is capable of handling problems an order of magnitude larger.

3 DECENTRALIZED POMDPS

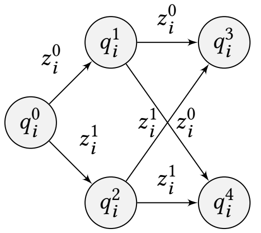
We next formally define the Dec-POMDP problem we consider. Contrary to most earlier works, we define the reward as a function of *state information* and action. This allows us to model information acquisition problems. We choose the finite-horizon formulation to reflect the fact that a decentralized information gathering task should have a clearly defined end after which the collected information is pooled and subsequent inference or decisions are made.

A finite-horizon Dec-POMDP is a tuple $\big(I, S, \{A_i\}, \{Z_i\}, P^s, P^z, b^0, T, \{\rho_t\}\big)$, where $I = \{1, \ldots, n\}$ is the set of agents, S is a finite set of hidden states, A_i and Z_i are the finite action and observation sets of agent $i \in I$, respectively, P^s is the state transition probability that gives the conditional probability $P^s(s^{t+1} \mid s^t, a^t)$ of the new state s^{t+1} given the current state s^t and joint action $a^t = (a_1^t, \ldots, a_n^t) \in A$, where A is the joint action space obtained as the Cartesian product of A_i for all $i \in I$, P^z is the observation probability that gives the conditional probability $P^z(z^{t+1} \mid s^{t+1}, a^t)$ of the joint observation $z^{t+1} = (z_1^{t+1}, \ldots z_n^{t+1}) \in Z$ given the state s^{t+1} and previous joint action a^t, with Z being the joint observation space defined as the Cartesian product of Z_i for $i \in I$, $b^0 \in \Delta(S)$ is the initial state distribution[2] at time $t = 0$, $T \in \mathbb{N}$ is the problem horizon, and $\rho_t : \Delta(S) \times A \to \mathbb{R}$ are the reward functions at times $t = 0, \ldots, T-1$, while $\rho_T : \Delta(S) \to \mathbb{R}$ determines a final reward obtained at the end of the problem horizon.

The Dec-POMDP starts from some state $s^0 \sim b^0$. Each agent $i \in I$ then selects an action $a_i^0 \in A_i$, and the joint action $a^0 = (a_1^0, \ldots, a_n^0) \in A$ is executed. The state then transitions according to P^s, and each agent perceives an observation $z_i^1 \in Z_i$, where the likelihood of the joint observation $z^1 = (z_1^1, \ldots, z_n^1) \in Z$ is determined according to P^z. The agents then select the next actions a_i^1, and the same steps are repeated until $t = T$ and the task ends.

Optimally solving a Dec-POMDP means to design a policy for each agent that encodes which action the agent should execute conditional on its past observations and actions; in a manner such that the expected sum of rewards collected is maximized. In the following, we make the notion of a policy exact, and determine the expected sum of rewards collected when executing a policy.

[2] We denote by $\Delta(S)$ the space of probability mass functions over S.



q_i	$\gamma_i(q_i)$
q_i^0	a_i^0
q_i^1	a_i^1
q_i^2	a_i^1
q_i^3	a_i^0
q_i^4	a_i^0

Figure 1: A local policy for agent i. The policy encodes the agent's behavior conditional on local observations. The shaded circles show the set of nodes Q_i. The starting node is q_0^i. The table on the right defines the output function γ_i, and the labels on the edges define the node transition function λ_i. First, the agent executes $\gamma_i(q_i^0)$. Conditional on the next observation perceived, the next node is q_i^1 or q_i^2. At the next node, the action to execute is again looked up from γ_i.

3.1 Histories and policies

Define the history set of agent i at time $t = 1, \dots, T$ as $H_i^t = \{(b^0, a_i^0, z_i^1, \dots, a_i^{t-1}, z_i^t) \mid a_i^k \in A_i, z_i^k \in Z_i\}$, and $H_i^0 = \{(b^0)\}$. A local history $h_i^t \in H_i^t$ contains all information available to agent i to decide its next action a_i^t. We define the joint history set H^t as the Cartesian product of H_i^t over $i \in I$. We write a joint history as $h^t = (h_1^t, \dots, h_n^t) \in H^t$, or equivalently as $h^t = (b_0, a^0, z^1, \dots, a^{t-1}, z^t) \in H^t$ where $a^k \in A$ and $z^k \in Z$. Both the local and joint histories satisfy the recursion $h^t = (h^{t-1}, a^{t-1}, z^t)$.

A solution of a finite-horizon Dec-POMDP is a local policy for each agent that determines which action an agent should take given a local history in H_i^t for any $t = 0, \dots, T-1$. We define a local policy similarly as [19] as a deterministic finite-horizon controller viewed as a directed acyclic graph.

Definition 3.1 (Local policy). For agent i, a local policy is $\pi_i = (Q_i, q_i^0, \gamma_i, \lambda_i)$, where Q_i is a finite set of nodes, $q_i^0 \in Q_i$ is a starting node, $\gamma_i : Q_i \to A_i$ is an output function, and $\lambda_i : Q_i \times Z_i \to Q_i$ is a node transition function.

Fig. 1 shows an example of a local policy. Note that a sufficiently large graph can represent any finite horizon local policy.

We constrain the structure of local policies by enforcing that each node can be identified with a unique time step. We call this the property of temporal consistency.

Definition 3.2 (Temporal consistency). A local policy $\pi_i = (Q_i, q_i^0, \gamma_i, \lambda_i)$ is temporally consistent if $Q_i = \bigcup_{t=0}^{T-1} Q_i^t$ where Q_i^t are pairwise disjoint and non-empty, and $Q_i^0 = \{q_i^0\}$, and for any $t = 0, \dots, T-2$, for $q_i^t \in Q_i^t$, for all $z_i \in Z_i$, $\lambda_i(q_i^t, z_i) \in Q_i^{t+1}$.

In a temporally consistent policy, at a node in Q_i^t the agent has $(T-t)$ decisions left until the end of the problem horizon. Temporal consistency guarantees that exactly one node in each set Q_i^t can be visited, and that after visiting a node in Q_i^t, the next node will belong to Q_i^{t+1}. In Fig. 1, $T = 3$, and $Q_i^0 = \{q_i^0\}$, $Q_i^1 = \{q_i^1, q_i^2\}$, $Q_i^2 = \{q_i^3, q_i^4\}$. Temporal consistency is assumed throughout the rest of the paper.

A joint policy describes the joint behaviour of all agents and is defined as the combination of the local policies π_i.

Definition 3.3 (Joint policy). Given local policies $\pi_i = (Q_i, q_i^0, \gamma_i, \lambda_i)$ for all $i \in I$, a joint policy is $\pi = (Q, q^0, \gamma, \lambda)$, where Q is the Cartesian product of all Q_i, $q^0 = (q_1^0, \dots, q_n^0) \in Q$, and for $q = (q_1, \dots, q_n) \in Q$ and $z = (z_1, \dots, z_n) \in Z$, $\gamma : Q \to A$ is such that $\gamma(q) = (\gamma_1(q_1), \dots, \gamma_n(q_n))$, and $\lambda : Q \times Z \to Q$ is such that $\lambda(q, z) = (\lambda_1(q_1, z_1), \dots, \lambda_n(q_n, z_n))$.

Temporal consistency naturally extends to joint policies, such that there exists a partition of Q by pairwise disjoint sets Q^t.

3.2 Bayes filter

While planning policies for information gathering, it is useful to reason about the joint belief of the agents given some joint history. This can be done via Bayesian filtering as described in the following.

The initial state distribution b^0 is a function of the state at time $t = 0$, and for any state $s^0 \in S$, $b^0(s^0)$ is equal to the probability $P(s^0 \mid h^0)$. When action a^0 is executed and observation z^1 is perceived, we may find the posterior belief $P(s^1 \mid h^1)$ where $h^1 = (h^0, a^0, z^1)$ by applying a Bayes filter.

In general, given any current joint belief b^t corresponding to some joint history[3] h^t, and a joint action a^t and joint observation z^{t+1}, the posterior joint belief is calculated by

$$b^{t+1}(s^{t+1}) = \frac{P^z(z^{t+1} \mid s^{t+1}, a^t) \sum\limits_{s^t \in S} P^s(s^{t+1} \mid a^t, s^t) b^t(s^t)}{\eta(z^{t+1} \mid b^t, a^t)}, \quad (1)$$

where $\eta(z^{t+1} \mid b^t, a^t)$ is the normalization factor equal to the prior probability of observing z^{t+1}. Given b^0 and any joint history $h^t = (b^0, a^0, z^1, \dots, a^{t-1}, z^t)$, repeatedly applying Eq. (1) yields a sequence $b^0, b^1, \dots, b^t$ of joint beliefs. We shall denote the application of the Bayes filter by the shorthand notation $b^{t+1} = \zeta(b^t, a^t, z^{t+1})$. Furthermore, we shall denote the filter that recovers b^t given h^t by repeated application of ζ by a function $\tau : H^t \to \Delta(S)$.

3.3 Value of a policy

The value of a policy π is equal to the expected sum of rewards collected when acting according to the policy. We define value functions $V_t^\pi : \Delta(S) \times Q^t \to \mathbb{R}$ that give the expected sum of rewards when following policy π until the end of the horizon when t decisions have been taken so far, for any joint belief $b \in \Delta(S)$ and any policy node $q \in Q^t$.

The time step $t = T$ is a special case when all actions have already been taken, and the value function only depends on the joint belief and is equal to the final reward: $V_T(b) = \rho_T(b)$.

For $t = T-1$, one decision remains, and the remaining expected sum of rewards of executing policy π is equal to

$$V_{T-1}^\pi(b, q) = \rho_{T-1}(b, \gamma(q)) + \sum_{z \in Z} \eta(z \mid b, \gamma(q)) V_T(\zeta(b, \gamma(q), z)), \quad (2)$$

i.e., the sum of the immediate reward and the expected final reward at time T. From the above, we define V_t^π iterating backwards in time for $t = T-2, \dots, 0$ as

$$V_t^\pi(b, q) = \rho_t(b, \gamma(q)) + \mathbb{E}\left[V_{t+1}^\pi\left(\zeta(b, \gamma(q), z), \lambda(q, z)\right)\right], \quad (3)$$

[3] For notational convenience, we drop the explicit dependence of b^t on the joint history.

where the expectation is under $z \sim \eta(z \mid b, \gamma(q))$. The expected sum of rewards collected when following a policy π is equal to its value $V_0^{\pi}(b^0, q^0)$. The objective is to find an optimal policy π^* whose value is greater than or equal to the value of any other policy.

4 VALUE OF A POLICY NODE

Executing a policy corresponds to a stochastic traversal of the policy graphs (Fig. 1) conditional on the observations perceived. In this section, we first answer two questions related to this traversal process. First, given a history, when is it consistent with a policy, and which nodes in the policy graph will be traversed (Subsection 4.1)? Second, given an initial state distribution, what is the probability of reaching a given policy graph node, and what are the relative likelihoods of histories if we assume a given node is reached (Subsection 4.2)? With the above questions answered, we define the value of a policy graph node both in a joint and in a local policy (Subsection 4.3). These values will be useful in designing a policy improvement algorithm for Dec-POMDPs.

4.1 History consistency

As illustrated in Fig. 1, there can be multiple histories along which a node can be reached. We define when a history is consistent with a policy, i.e., when executing a policy could have resulted in the given history. As histories in H^T are reached after executing all actions, in the remainder of this subsection we consider $0 \leq t \leq T-1$.

Definition 4.1 (History consistency). We are given for all $i \in I$ $\pi_i = (Q_i, q_i^0, \gamma_i, \lambda_i)$, and the corresponding joint policy $\pi = (Q, q^0, \gamma, \lambda)$.

(1) A local history $h_i^t = (b_0, a_i^0, z_i^1, \ldots, a_i^{t-1}, z_i^t)$ is consistent with π if the sequence of nodes $(q_i^0, q_i^1, \ldots, q_i^t)$ where $q_i^k = \lambda_i(q_i^{k-1}, z_i^k)$ for $k = 1, \ldots, t$ satisfies: $a_i^k = \gamma_i(q_i^k)$ for every k. We say h_i^t ends at $q_i^t \in Q_i^t$ under π.

(2) A joint history $h^t = (h_1^t, \ldots, h_n^t)$ is consistent with π if for all $i \in I$, h_i^t is consistent with π and ends at q_i^t. We say h^t ends at $q^t = (q_1^t, \ldots, q_n^t) \in Q^t$ under π.

Due to temporal consistency, any $h_i^t \in H_i^t$ consistent with a policy will end at some $q_i^t \in Q_i^t$. Similarly, any $h^t \in H^t$ ends at some $q^t \in Q^t$.

4.2 Node reachability probabilities

Above, we have defined when a history ends at a particular node. Using this definition, we now derive the joint probability mass function (pmf) $P(q^t, h^t \mid \pi)$ of policy nodes and joint histories given that a particular policy π is executed.

We note that $P(q^t, h^t \mid \pi) = P(q^t \mid h^t, \pi) P(h^t \mid \pi)$ and first consider $P(h^t \mid \pi)$. The unconditional a priori probability of experiencing the joint history $h^0 = (b^0)$ is $P(h^0) = 1$. For $t \geq 1$, the unconditional probability of experiencing h^t is obtained recursively by $P(h^t) = \eta(z^t \mid \tau(h^{t-1}), a^{t-1}) P(h^{t-1})$. Conditioning $P(h^t)$ on a policy yields $P(h^t \mid \pi) = P(h^t)$ if h^t is consistent with π and 0 otherwise. Next, we have $P(q^t \mid h^t, \pi) = \prod_{i \in I} P(q_i^t \mid h_i^t, \pi)$, with $P(q_i^t \mid h_i^t, \pi) = 1$ if h_i^t ends at q_i^t under π and 0 otherwise.

Combining the above, the joint pmf is defined as

$$P(q^t, h^t \mid \pi) = \begin{cases} P(h^t) & \text{if } h^t \text{ ends at } q^t \text{ under } \pi \\ 0 & \text{otherwise} \end{cases}.$$

Marginalizing over h^t, the probability of ending at node q^t under π is

$$P(q^t \mid \pi) = \sum_{h^t \in H^t} P(q^t, h^t \mid \pi), \tag{4}$$

and by definition of conditional probability,

$$P(h^t \mid q^t, \pi) = \frac{P(q^t, h^t \mid \pi)}{P(q^t \mid \pi)}. \tag{5}$$

We now find the probability of ending at q_i^t under π. Let Q_{-i}^t denote the Cartesian product of all Q_j^t *except* Q_i^t. Then $q_{-i}^t \in Q_{-i}^t$ denotes the nodes for all agents except i. We have $(q_{-i}^t, q_i^t) \in Q^t$. The probability of ending at q_i^t under π is

$$P(q_i^t \mid \pi) = \sum_{q_{-i}^t \in Q_{-i}^t} P\left((q_{-i}^t, q_i^t) \mid \pi\right), \tag{6}$$

where the sum terms are determined by Eq. (4). Again, by definition of conditional probability,

$$P(q_{-i}^t \mid q_i^t, \pi) = \frac{P\left((q_{-i}^t, q_i^t) \mid \pi\right)}{P(q_i^t \mid \pi)}, \tag{7}$$

where the term in the numerator is obtained from Eq. (4).

4.3 Value of policy nodes

We define the values of a node in a joint policy and an individual policy.

Definition 4.2 (Value of a joint policy node). Given a joint policy $\pi = (Q, q^0, \gamma, \lambda)$, the value of a node $q^t \in Q^t$ is defined as

$$V_t^{\pi}(q^t) = \mathbb{E}_{h_t \sim P(h^t \mid q^t, \pi)} \left[V_t^{\pi}(\tau(h^t), q^t) \right],$$

where $P(h^t \mid q^t, \pi)$ is defined in Eq. (5) and $\tau(h^t)$ is the joint belief corresponding to history h^t.

Definition 4.3 (Value of a local policy node). For $i \in I$, let $\pi_i = (Q_i, q_i^0, \gamma_i, \lambda_i)$ be the local policy and let $\pi = (Q, q^0, \gamma, \lambda)$ be the corresponding joint policy. For any $i \in I$, the value of a local node $q_i^t \in Q_i^t$ is

$$V_t^{\pi}(q_i^t) = \mathbb{E}_{q_{-i}^t \sim P(q_{-i}^t \mid q_i^t, \pi)} \left[V_t^{\pi}\left((q_{-i}^t, q_i^t)\right) \right],$$

where $P(q_{-i}^t \mid q_i^t, \pi)$ is defined in Eq. (7).

In other words, the value of a local node q_i^t is equal to the expected value of the value of the joint node (q_{-i}^t, q_i^t) under $q_{-i}^t \sim P(q_{-i}^t \mid q_i^t, \pi)$.

5 CONVEX-REWARD DEC-POMDPS

In this section, we prove several results for the value function of a Dec-POMDP whose reward function is convex in $\Delta(S)$. Convex rewards are of special interest in information gathering. This is because of their connection to so-called uncertainty functions [6], which are non-negative functions concave in $\Delta(S)$. Informally, an uncertainty function assigns large values to uncertain beliefs, and smaller values to less uncertain beliefs. Negative uncertainty functions are convex and assign high values to less uncertain beliefs, and are thus suitable as reward functions for information gathering. Examples of uncertainty functions include Shannon entropy, generalizations such as Rényi entropy, and types of value of information, e.g., the probability of error in hypothesis testing.

The following theorem shows that if the immediate reward functions are convex in the joint belief, then the finite horizon value function of any policy is convex in the joint belief.

Theorem 5.1. *If the reward functions $\rho_T : \Delta(S) \to \mathbb{R}$ and $\rho_t : \Delta(S) \times A \to \mathbb{R}$ are convex in $\Delta(S)$, then for any policy π, $V_T : \Delta(S) \to \mathbb{R}$ is convex and $V_t^\pi : \Delta(S) \times Q^t \to \mathbb{R}$ is convex in $\Delta(S)$ for any t.*

Proof. Let $\pi = (Q, q^0, \gamma, \lambda)$, and $b \in \Delta(S)$. We proceed by induction ($V_T(b) = \rho_T(b)$ is trivial). For $t = T-1$, let $q^{T-1} \in Q^{T-1}$, and denote $a := \gamma(q^{T-1})$. From Eq. (2), $V_{T-1}^\pi(b, q^{T-1}) = \rho_{T-1}(b, a) + \sum_{z \in Z} \eta(z \mid b, a) V_T(\zeta(b, a, z))$. We recall from above that V_T is convex, and by Eq. (1), the Bayes filter $\zeta(b, a, z)$ is a linear function of b. The composition of a linear and convex function is convex, so $V_T(\zeta(b, a, z))$ is a convex function of b. The non-negative weighted sum of convex functions is also convex, and by assumption ρ_{T-1} is convex in $\Delta(S)$, from which it follows that V_{T-1}^π is convex in $\Delta(S)$.

Now assume V_{t+1}^π is convex in $\Delta(S)$ for some $1 \le t \le T-1$. By the definition in Eq. (3) and the same argumentation as above, it follows that V_t^π is convex in $\Delta(S)$. □

Since a sufficiently large policy graph can represent any policy, we infer that the value function of an optimal policy is convex.

The following corollary gives a lower bound for the value of a policy graph node.

Corollary 5.2. *Let $g^t : H^t \to [0, 1]$ be a probability mass function over the joint histories at time t. If the reward functions $\rho_T : \Delta(S) \to \mathbb{R}$ and $\rho_t : \Delta(S) \times A \to \mathbb{R}$ are convex in $\Delta(S)$, then for any time step t and any policy π,*

$$\mathbb{E}_{h^t \sim g(h^t)}\left[V_t^\pi(\tau(h^t), q^t)\right] \ge V_t^\pi\left(\mathbb{E}_{h^t \sim g(h^t)}\left[\tau(h^t)\right], q^t\right).$$

Proof. By Theorem 5.1, $V_t^\pi : \Delta(S) \times Q^t \to \mathbb{R}$ is convex in $\Delta(S)$. The claim immediately follows applying Jensen's inequality. □

Applied to Definition 4.2, the corollary says the value of a joint policy node q^t is lower bounded by the value of the expected joint belief at q^t. Applied to Definition 4.3, we obtain a lower bound for the value of a local policy node q_i^t as

$$V_t^\pi(q_i^t) \ge \mathbb{E}_{q_{-i}^t \sim P(q_{-i}^t | q_i^t, \pi)}\left[V_t^\pi\left(\mathbb{E}_{h^t \sim P(h^t | q^t, \pi)}\left[\tau(h^t)\right], q^t\right)\right],$$

where inside the inner expectation we write $(q_{-i}^t, q_i^t) = q^t$. Thus, we can evaluate a lower bound for the value of any local node $q_i^t \in Q_i^t$ by finding the values $V_t^\pi(q^t)$ of all joint nodes $q^t \in Q^t$ and then taking the expectation of $V_t^\pi(q^t)$ where $q^t = (q_{-i}^t, q_i^t)$ under $P(q_{-i}^t \mid q_i^t, \pi)$.

Corollary 5.2 has applications in policy improvement algorithms that iteratively improve the value of a policy by modifying the output and node transition functions at each local policy node. Instead of directly optimizing the value of a node, the lower bound can be optimized. We present one such algorithm in the next section.

As Corollary 5.2 holds for any pmf over joint histories, it could be applied also with pmfs other than $P(h^t \mid q^t, \pi)$. For example, if it is expensive to enumerate the possible histories and beliefs at a node, one could approximate the lower bound, e.g., through importance sampling [12, Ch. 23.4].

In standard Dec-POMDPs, the expected reward is a linear function of the joint belief. Then, the corollary above holds with equality.

Corollary 5.3. *Consider a Dec-POMDP where the reward functions are defined as $\rho_T(b) = \sum_{s \in S} b(s) R_T(s)$ and for $0 \le t \le T-1$, $\rho_t(b, a) = \sum_{s \in S} b(s) R_t(s, a)$, where $R_T : S \to \mathbb{R}$ is a state-dependent final reward function and $R_t : S \times A \to \mathbb{R}$ are the state-dependent reward functions. Then, the conclusion of Corollary 5.2 holds with equality.*

Proof. Let $\pi = (Q, q_0, \gamma, \lambda)$ and $b \in \Delta(S)$. First note that $V_T(b) = \rho_T(b) = \sum_{s \in S} b(s) R_T(s)$. Consider then $t = T-1$, and let $q^{T-1} \in Q^{T-1}$, and write $a := \gamma(q^{T-1})$. Then from the definition of V_{T-1}^π in Eq. (2), consider first the latter sum term which equals

$$\sum_{z \in Z} \eta(z \mid b, a) \sum_{s' \in S} \zeta(b, a, z)(s') R_T(s')$$
$$= \sum_{s' \in S}\left[\sum_{z \in Z} \sum_{s \in S} P^z(z \mid s', a) P^s(s' \mid a, s) b(s)\right] R_T(s')$$

which follows by replacing $\zeta(b, a, z)$ by Eq. (1), canceling out $\eta(z \mid b, a)$, and rearranging the sums. The above is clearly a linear function of b, and by definition, so is ρ_t, the first part of V_{T-1}^π. Thus, $V_{T-1}^\pi : \Delta(S) \times Q^{T-1} \to \mathbb{R}$ is linear in $\Delta(S)$. By an induction argument, it is now straightforward to show that V_t^π is linear in $\Delta(S)$ for all $0 \le t \le T-1$. Finally,

$$\mathbb{E}_{h^t \sim g(h^t)}\left[V_t^\pi(\tau(h^t), q^t)\right] = V_t^\pi\left(\mathbb{E}_{h^t \sim g(h^t)}\left[\tau(h^t)\right], q^t\right)$$

for any pmf g over joint histories by linearity of expectation. □

Corollary 5.3 shows that a solution algorithm for a Dec-POMDP with a reward convex in the joint belief that uses the lower bound from Corollary 5.2 will also work for standard Dec-POMDPs with a reward linear in the joint belief.

Since a linear function is both convex and concave, rewards that are state-dependent and rewards that are convex in the joint belief can be combined on different time steps in one Dec-POMDP and the lower bound still holds.

6 POLICY GRAPH IMPROVEMENT

The Policy Graph Improvement (PGI) algorithm [19] was originally introduced for standard Dec-POMDPs with reward function linear in the joint belief. PGI monotonically improves policies by locally modifying the output and node transition functions of the individual agents' policies. The policy size is fixed, such that the worst case computation time for an improvement iteration is known in advance. Moreover, due to the limited size of the policies the method produces compact, understandable policies.

We extend PGI to the non-linear reward case, and call the method non-linear PGI (NPGI). Contrary to tree based Dec-POMDP approaches the policy does not grow double-exponentially with the planning horizon as we use a fixed size policy. If the reward function is convex in $\Delta(S)$, NPGI may improve the lower bound from Corollary 5.2. The lower bound is tight when each policy graph node corresponds to only one history suggesting we can improve the quality of the lower bound by increasing policy graph size.

NPGI is shown in Algorithm 1. At each improvement step, NPGI repeats two steps: the forward pass and the backward pass. In the forward pass, we use the current best joint policy to find the set B

Algorithm 1 NPGI

```
Input: Policy $\pi = (Q, q^0, \gamma, \lambda)$, initial belief $b^0$
Output: Improved policy $\pi$
while not converged and time limit not exceeded do
    $B \leftarrow$ ForwardPass($\pi$, $b^0$)
    $\pi^+ \leftarrow$ BackwardPass($\pi$, $B$)
    if $V_0^{\pi^+}(b^0, q^0) \geq V_0^{\pi}(b^0, q^0)$ then $\pi \leftarrow \pi^+$
return $\pi$
```

of expected joint beliefs at every policy graph node. In practice, we do this by first enumerating for each agent the sets of local histories ending at all local nodes, then taking the appropriate combinations to create the joint histories for joint policy graph nodes. We then evaluate the expected joint beliefs at every joint policy graph node.

In the backward pass, we improve the current policy by modifying its output and node transition functions locally at each node. As output from the backward pass, we obtain an updated policy π^+ using the improved output and node transition functions γ^+ and λ^+, respectively. As NPGI may optimize a lower bound of the node values, we finally check if the value of the improved policy, $V_0^{\pi^+}(b^0, q^0)$, is greater than the value of the current best policy, and update the best policy if necessary.

Backward pass. The backward pass of NPGI is shown in Algorithm 2. At time step t for agent i, for each node $q_i^t \in Q_i^t$, we maximize either the value $V_t^{\pi^+}(q_i^t)$ or its lower bound with respect to the local policy parameters. In the following, we present the details for maximizing the lower bound, the algorithm for the exact value can be derived analogously.

For $t = T - 1$, we consider the last remaining action. Fix a local node $q_i^{T-1} \in Q_i^{T-1}$. Denote the expected belief at $q^{T-1} = (q_1^{T-1}, \ldots, q_n^{T-1}) \in Q^{T-1}$ as $b := \mathbb{E}_{h^{T-1} \sim P(h^{T-1} \mid q^{T-1}, \pi)} \left[\tau(h^{T-1})\right]$. We write $a = \left(\gamma_1^+(q_1^{T-1}), \ldots, a_i^{T-1}, \ldots, \gamma_n^+(q_n^{T-1})\right) \in A$ as the joint action where local actions of all other agents except i are fixed to those specified by the current output function. We solve

$$\max_{a_i^{T-1} \in A_i} \mathbb{E}\left[\rho_{T-1}(b, a) + \mathbb{E}\left[V_T(\zeta(b, a, z))\right]\right] \tag{8}$$

where the outer expectation is under $q_{-i}^{T-1} \sim P(q_{-i}^{T-1} \mid q_i^{T-1}, \pi)$, the distribution over the nodes of agents other than i, and the inner expectation is under $\eta(z \mid b, a)$. Note that in general, b is different for each q_{-i}^{T-1}, as $q^{T-1} = (q_{-i}^{T-1}, q_i^{T-1})$ will be different. We assign $\gamma_i^+(q_i^{T-1})$ equal to the local action that maximizes Eq. (8). Note that this modification of the policy does not invalidate any of the expected beliefs at the nodes in Q.

For $t \leq T - 1$, we consider both the current action and the next nodes via the node transition function. Fix a local node $q_i^t \in Q_i^t$, and define a and b similarly as above. Additionally, for any joint observation $z = (z_1, \ldots, z_n) \in Z$, define

$$q^{t+1}(z) = \left(\lambda_1^+(q_1^t, z_1), \ldots, q_i^{z_i}, \ldots, \lambda_n^+(q_n^t, z_n)\right)$$

as the next node in Q^{t+1} when transitions of all other agents except i are fixed to those specified by the current node transition function.

Algorithm 2 BackwardPass

```
Input: Policy $\pi = (Q, q^0, \gamma, \lambda)$, expected beliefs $B = \{b^q \mid q \in Q\}$
Output: Policy $\pi^+$ with improved output and node transition functions $\gamma^+$, $\lambda^+$
1:  $\gamma^+ \leftarrow \gamma, \lambda^+ \leftarrow \lambda$
2:  for $t = T - 1, \ldots, 0$ do
3:      for $i \in I$ do
4:          $W_i^t \leftarrow \emptyset$
5:          for $q_i^t \in Q_i^t$ do
6:              if $t = T - 1$ then
7:                  Solve Eq. (8), assign $\gamma_i^+(q_i^t)$
8:              else
9:                  Solve Eq. (9), assign $\gamma_i^+(q_i^t)$ and $\lambda_i^+(q_i^t, z_i) \forall z_i$
10:             if $\exists w_i^t \in W_i^t$ : SamePolicy($w_i^t$, $q_i^t$) then
11:                 Redirect($q_i^t$, $w_i^t$)
12:                 Randomize($q_i^t$)
13:             $W_i^t \leftarrow W_i^t \cup \{q_i^t\}$
14: return $(Q, q^0, \gamma^+, \lambda^+)$
15: procedure SamePolicy($q_i^t$, $w_i^t$)
16:     if $\gamma_i^+(q_i^t) == \gamma_i^+(w_i^t) \land \forall z_i : \lambda_i^+(q_i^t, z_i) == \lambda_i^+(w_i^t, z_i)$ then
17:         return True
18:     else
19:         return False
20: procedure Redirect($q_i^t$, $w_i^t$)
21:     for $(x, z_i) \in \{(x, z_i) \in Q_i^{t-1} \times Z_i \mid \lambda_i^+(x, z_i) = q_i^t\}$ do
22:         $\lambda_i^+(x, z_i) = w_i$
23: procedure Randomize($q_i^t$)
24:     $\gamma_i^+(q_i^t) \sim$ Uniform($A_i$)
25:     if $t \neq T - 1$ then
26:         $\forall z_i \in Z_i : \lambda_i^+(q_i^t, z_i) \sim$ Uniform($Q_i^{t+1}$)
```

We solve

$$\max_{\substack{a_i^t \in A_i \\ \forall z_i \in Z_i : q_i^{z_i} \in Q_i^{t+1}}} \mathbb{E}\left[\rho_t(b, a) + \mathbb{E}\left[V_{t+1}^{\pi^+}\left(\zeta(b, a, z), q^{t+1}(z)\right)\right]\right], \tag{9}$$

where the outer expectation is under $q_{-i}^t \sim P(q_{-i}^t \mid q_i^t, \pi)$, and the inner expectation is under $\eta(z \mid b, a)$. We assign $\gamma_i^+(q_i^t)$ and $\lambda_i^+(q_i^t, \cdot)$ to the respective maximizing values of Eq. (9). This assignment potentially invalidates the expected beliefs in B for any nodes in Q^k for $k \geq t + 1$. However, as in the subsequent optimization steps we only require the expected beliefs for Q^k, $k \leq t$, we do not need to repeat the forward pass.

Line 10 of Algorithm 2 checks if there exists a node w_i^t that we have already optimized that has the same local policy as the current node q_i^t. If such a node exists, we redirect all of the in-edges of q_i^t to w_i^t instead. This redirection is required to maintain correct estimates of the respective node probabilities in the algorithm. If we redirected the in-edges of q_i^t to w_i^t, on Line 12 we randomize the local policy of the now useless node q_i^t that has no in-edges[4], in

[4] To randomize the local policy of a node $q_i^t \in Q_i^t$, we sample new local policies until we find one that is not identical to the local policy of any other node in Q_i^t. Likewise, when randomly initializing a new policy in our experiments we avoid including in any Q_i^t nodes with identical local policies.

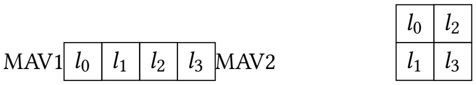


Figure 2: Arrangement of locations in the MAV domain (left) and rovers domain (right).

the hopes that it may be improved on subsequent backward passes. If a node q_i^t is to be improved that is unreachable, i.e., it has no in-edges or the probabilities of all histories ending in it are zero, we likewise randomize the local policy at that node.

Policy initialization. We initialize a random policy for each agent $i \in I$ with a given policy graph width $\left|Q_i^t\right|$ for each t as follows[5]. For example, for a problem with $T = 3$ and $\left|Q_i^t\right| = 2$, we create a policy similar to Fig. 1 for each agent, where there is one initial node q_i^0, and 2 nodes at each time step $t \geq 1$. The action determined by the output function $\gamma_i(q_i)$ is sampled uniformly at random from A_i. For each node $q_i^t \in Q_i^t$ for $0 \leq t \leq T-1$, we sample a next node from Q_i^{t+1} uniformly at random for each observation $z_i \in Z_i$ and assign the node transition function $\lambda_i(q_i, z_i)$ accordingly.

7 EXPERIMENTS

We evaluate the performance of NPGI on information gathering Dec-POMDPs. In the following, we introduce the problem domains, the experimental setup, and present the results.

7.1 Problem domains

We run experiments on the micro air vehicle (MAV) domain of [10] and propose an information gathering rovers domain inspired by the Mars rovers domain of [2]. In both tasks the objective of the agents is to maximize the expected sum of rewards collected minus the entropy of the joint belief at the end of the problem horizon.

MAV domain. A target moves between four possible locations, l_i in Figure 2. The target is either friendly or hostile; a hostile target moves more aggressively. Two MAVs, MAV1 and MAV2 in the figure, are tasked with tracking the target and inferring whether it is friendly or hostile. The MAVs can choose to use either a camera or a radar sensor to sense the location of the target. An observation from either sensor corresponds to a noisy measurement of the target's location. The camera is more accurate if the target is close, whereas the radar is more accurate when the target is further away. The Manhattan distance is applied, i.e., at l_0 the target is at distance 0 from MAV1 and at distance 3 from MAV2. If both MAVs apply their radars simultaneously, accuracy decreases due to interference.

Using the camera has zero cost, and using the radar sensor has a cost of 0.1, and an additional cost of 1 or 0.1 if the target is at distance 0 or 1 to the MAV, respectively, to model the risk of revealing the MAVs own location to the (potentially hostile) target. To model information gathering, we set the final reward equal to the negative Shannon entropy of the joint belief, i.e., $\rho_T(b) = \sum_{s \in S} b(s) \log_2 b(s)$. The initial belief is a uniform over all states. This problem has 8 states; 4 target locations and a binary variable for friendly/hostile, and 2 actions and 4 observations per agent.

Information gathering rovers. Two rovers are collecting information on four sites l_i of interest arranged as shown in Figure 2. Each site is in one of two possible states which remains fixed throughout. The agents can move north, south, east, or west. Movement fails with probability 0.2, in which case the agent remains at its current location. The agents always fully observe their own location. Additionally, the agents can choose to conduct measurements of the site they are currently at. A binary measurement of the site status is recorded with false positive and false negative probabilities of 0.2 each. If the agents measure at the same location at the same time, the false positive and false negative probabilities are significantly lower, 0.05 and 0.01, respectively. Movement has zero cost, while measuring has a cost of 0.1. The final reward is equal to the negative entropy. The initial belief is such that one agent starts at l_0, the other at l_3, with a uniform belief over the site status. The problem has 256 states, and 5 local actions and 8 local observations per agent.

7.2 Experimental setup

We compare NPGI to one exact algorithm and two heuristic algorithms. The exact method we employ is the Generalized Multi-Agent A* with incremental expansion (GMAA*-ICE) [17] with the QPOMDP search heuristic. According to [17] a vector representation of the search heuristic, analogous to the representation of an optimal POMDP value function by a set of so-called α-vectors [23], can help scale up GMAA*-ICE to larger problems. However, since the vector representation only exists if the reward function is linear in the joint belief, we represent the search heuristic as a tree. The two heuristic methods are joint equilibrium based search for policies (JESP) [13] and direct cross-entropy policy search (DICEPS) [16].

All of the methods above are easily modified to our domains where the final reward is equal to the negative Shannon entropy. However, applicability of NPGI is wider as it allows the reward at *any* time step to be a convex function of the joint belief. We note that there are other algorithms such as FB-HSVI [7] and PBVI-BB [11] that have demonstrated good performance on many benchmarks. However, these algorithms rely on linearity of the reward to achieve compression of histories and joint beliefs, and non-trivial modifications beyond the scope of this work would be required to extend them to Dec-POMDPs with non-linear rewards.

As baselines, we report values of a greedy open loop policy that executes a sequence of joint actions that has the maximal expected sum of rewards under the initial belief, and the best blind policy that always executes the same joint action.

We run NPGI using both the exact value of nodes and the lower bound from Corollary 5.2. The number of policy graph nodes $\left|Q_i^t\right|$ at each time step t is 2, 3, or 4. For each run with NPGI we run 30 backward passes, starting from randomly sampled initial policies. For all methods, we report the averages over 100 runs. If a run does not finish in 2 hours, we terminate it.

7.3 Results

Tables 1 and 2 show the average policy values in the MAV and information gathering rovers problems, respectively. NPGI is indicated

[5] At the last time step, it is only meaningful to have $\left|Q_i^T\right| \leq |A_i|$. In our experiments if $\left|Q_i^T\right| > |A_i|$, we instead set $\left|Q_i^T\right| = |A_i|$.

Table 1: Average policy values in the MAV domain ($|S| = 8$, $|A_i| = 2$, $|Z_i| = 4$).

Method	$\left|Q_i^t\right|$	$T = 2$	$T = 3$	$T = 4$	$T = 5$
Ours	2	-1.919	-1.831	-1.768	-1.725
	3	-1.919	-1.831	-1.768	-1.725
	4	-1.919	-1.831	-1.768	-1.725
Ours (No LB)	2	-1.919	-1.831	-1.768	-1.726
	3	-1.919	-1.831	-1.768	-1.725
	4	-1.919	-1.831	-1.768	-1.726
DICEPS		-1.925	-1.937	-1.926	-1.940
JESP		-1.953	-1.859	-1.794	-1.750
GMAA*-ICE		-1.919	-1.831	-	-
Greedy		-2.156	-2.044	-1.978	-1.932
Blind		-1.945	-1.904	-1.909	-1.932

Table 2: Average policy values in the information gathering rovers domain ($|S| = 256$, $|A_i| = 5$, $|Z_i| = 8$).

Method	$\left|Q_i^t\right|$	$T = 2$	$T = 3$	$T = 4$	$T = 5$
Ours	2	-3.495	-3.189	-3.034	-2.989
	3	-3.498	-3.189	-3.034	-2.977
	4	-3.500	-3.189	-3.034	-3.004
Ours (No LB)	2	-3.495	-3.189	-3.035	-2.976
	3	-3.498	-3.189	-3.035	-3.085
	4	-3.500	-3.189	-3.035	-
DICEPS		-3.482	-3.535	-3.825	-4.792
JESP		-3.483	-3.536	-	-
GMAA*-ICE		-3.479	-3.189	-	-
Greedy		-3.844	-4.031	-3.877	-3.818
Blind		-3.479	-3.412	-3.418	-3.472

by "Ours" when the lower bound (LB) was used, and as "Ours (No LB)" when exact evaluation of node values was applied. Results are reported as function of the problem horizon T, and for NPGI also as function of the policy graph size $\left|Q_i^t\right|$. The symbol "-" indicates missing results due to exceeding the cut-off time.

GMAA*-ICE finds an optimal solution, but similarly to [10] we find that it does not scale beyond $T = 3$ in either problem. Considering $T = 2$ and $T = 3$, the average values of our method are very close to the optimal value in both problems. In these cases, we found that NPGI finds an optimal policy in all the MAV problem runs, and in about 60% of the MAV problem runs.

In the MAV problem (Table 1), performance of our method is consistent for varying policy graph size $\left|Q_i^t\right|$ and horizon T. This indicates that even a small policy suffices to reach a high value in this problem. We also note that applying the lower bound does not reduce the quality of the policy found by our approach.

In the rover problem (Table 2), we observe more variation in policy quality as function of the policy graph size. However, applying the Mann-Whitney U-test we do not find significant differences (significance level of 0.01) either for varying policy graph size, nor for exact computation versus applying the lower bound. A compact policy with as few as 2 nodes per time step in the policy graph can reach a high value in this problem as well.

Table 3: Average NPGI backward pass duration (in seconds) with or without lower bound (LB).

	MAV		Rovers	
T	With LB	No LB	With LB	No LB
2	0.002	0.002	0.04	0.04
3	0.04	0.05	0.26	0.34
4	1.20	2.74	1.43	4.40
5	31.02	55.34	32.23	158.7

Table 3 shows the average duration of one backward pass of Algorithm 1 as function of the problem horizon T with $\left|Q_i^t\right| = 2$, with or without using the lower bound (LB). The lower runtime requirement when applying the lower bound is seen clearly for $T \geq 4$. The runtime of NPGI is dominated by the backward pass and solving the local policy optimization problems, Eqns. (8) and (9), which applying the lower bound help reduce. As indicated by the results in Tables 1 and 2, applying the lower bound also does not degrade the quality of the policies found.

Our method outperforms the baselines except for $T = 2$ in the rover problem where a blind policy of always measuring is optimal. In several cases, JESP and DICEPS return policies with a value lower than one of both of the baselines.

The size of the policy graph in NPGI must be specified before calculating the policy. As shown by our experiments, fixing the policy graph size effectively limits the space of policies to be explored and can produce compact and understandable policies. However, a potential weakness is that optimizing over fixed-size policies excludes the possibility to find a larger but potentially better policy.

8 CONCLUSION

We showed that if the reward function in a finite-horizon Dec-POMDP is convex in the joint belief, then the value function of any policy is then convex in the joint belief. Rewards that are convex in the joint belief are of importance in information gathering problems. We applied the result to derive a lower bound for the value, and empirically demonstrated that it improves the run-time of a heuristic planning algorithm without degrading solution quality.

We presented the first heuristic algorithm for Dec-POMDPs with rewards convex in the joint belief, and showed that it reaches good performance in large Dec-POMDPs. Future work includes developing an approximation algorithm with bounded suboptimality. Approximation of the reward function by a piecewise linear function similar to [3] is a potential first step towards this goal.

ACKNOWLEDGMENTS

J.P. and J.P. were supported by European Research Council under Grant No. 640554 (SKILLS4ROBOTS) and German Research Foundation project PA 3179/1-1 (ROBOLEAP).

REFERENCES

[1] Martin Allen and Shlomo Zilberstein. 2009. Complexity of decentralized control: Special cases. In *Advances in Neural Information Processing Systems (NIPS)*. 19–27.

[2] Christopher Amato and Shlomo Zilberstein. 2009. Achieving goals in decentralized POMDPs. In *Autonomous Agents and Multiagent Systems (AAMAS)*. 593–600.

[3] Mauricio Araya-López, Olivier Buffet, Vincent Thomas, and Francois Charpillet. 2010. A POMDP Extension with Belief-dependent Rewards. In *Advances in Neural Information Processing Systems (NIPS)*. 64–72.

[4] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. 2002. The complexity of decentralized control of Markov decision processes. *Mathematics of operations research* 27, 4 (2002), 819–840.

[5] Benjamin Charrow, Vijay Kumar, and Nathan Michael. 2014. Approximate representations for multi-robot control policies that maximize mutual information. *Autonomous Robots* 37, 4 (2014), 383–400.

[6] Morris H DeGroot. 2004. *Optimal Statistical Decisions*. John Wiley & Sons, Inc., Hoboken, NJ. Wiley Classics Library edition.

[7] Jilles Steeve Dibangoye, Christopher Amato, Olivier Buffet, and François Charpillet. 2016. Optimally solving Dec-POMDPs as continuous-state MDPs. *Journal of Artificial Intelligence Research* 55 (2016), 443–497.

[8] Mathieu Fehr, Olivier Buffet, Vincent Thomas, and Jilles Dibangoye. 2018. rho-POMDPs have Lipschitz-Continuous epsilon-Optimal Value Functions. In *Advances in Neural Information Processing Systems (NIPS)*. 6933–6943.

[9] Eric A Hansen, Daniel S Bernstein, and Shlomo Zilberstein. 2004. Dynamic programming for partially observable stochastic games. In *AAAI*. 709–715.

[10] Mikko Lauri, Eero Heinänen, and Simone Frintrop. 2017. Multi-robot active information gathering with periodic communication. In *IEEE Intl. Conference on Robotics and Automation (ICRA)*. 851–856.

[11] Liam C MacDermed and Charles L Isbell. 2013. Point Based Value Iteration with Optimal Belief Compression for Dec-POMDPs. In *Advances in Neural Information Processing Systems (NIPS)*. 100–108.

[12] Kevin Murphy. 2012. *Machine Learning: A probabilistic perspective*. MIT Press.

[13] Ranjit Nair, Milind Tambe, Makoto Yokoo, David Pynadath, and Stacy Marsella. 2003. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Intl. Joint Conference on Artificial Intelligence (IJCAI)*. 705–711.

[14] Frans A Oliehoek. 2013. Sufficient Plan-Time Statistics for Decentralized POMDPs. In *Intl. Joint Conference on Artificial Intelligence (IJCAI)*. 302–308.

[15] Frans A Oliehoek and Christopher Amato. 2016. *A Concise Introduction to Decentralized POMDPs*. Springer.

[16] Frans A Oliehoek, Julian FP Kooij, and Nikos Vlassis. 2008. The cross-entropy method for policy search in decentralized POMDPs. *Informatica* 32, 4 (2008), 341–357.

[17] Frans A Oliehoek, Matthijs TJ Spaan, Christopher Amato, and Shimon Whiteson. 2013. Incremental clustering and expansion for faster optimal planning in Dec-POMDPs. *Journal of Artificial Intelligence Research* 46 (2013), 449–509.

[18] Frans A Oliehoek, Matthijs TJ Spaan, Shimon Whiteson, and Nikos Vlassis. 2008. Exploiting locality of interaction in factored Dec-POMDPs. In *Autonomous Agents and Multiagent Systems (AAMAS)*. 517–524.

[19] Joni K Pajarinen and Jaakko Peltonen. 2011. Periodic Finite State Controllers for Efficient POMDP and DEC-POMDP Planning. In *Advances in Neural Information Processing Systems (NIPS)*. 2636–2644.

[20] Yash Satsangi, Shimon Whiteson, Frans A Oliehoek, and Matthijs TJ Spaan. 2018. Exploiting submodular value functions for scaling up active perception. *Autonomous Robots* 42, 2 (2018), 209–233.

[21] Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J Pappas. 2018. Anytime Planning for Decentralized Multirobot Active Information Gathering. *IEEE Robotics and Automation Letters* 3, 2 (2018), 1025–1032.

[22] Sven Seuken and Shlomo Zilberstein. 2007. Memory-Bounded Dynamic Programming for DEC-POMDPs.. In *Intl. Joint Conference on Artificial Intelligence (IJCAI)*. 2009–2015.

[23] Richard D Smallwood and Edward J Sondik. 1973. The optimal control of partially observable Markov processes over a finite horizon. *Operations research* 21, 5 (1973), 1071–1088.

[24] Matthijs TJ Spaan, Geoffrey J Gordon, and Nikos Vlassis. 2006. Decentralized planning under uncertainty for teams of communicating agents. In *Autonomous Agents and Multiagent Systems (AAMAS)*. 249–256.

[25] Matthijs TJ Spaan, Tiago S Veiga, and Pedro U Lima. 2015. Decision-theoretic planning under uncertainty with information rewards for active cooperative perception. *Autonomous Agents and Multi-Agent Systems* 29, 6 (2015), 1157–1185.

[26] Daniel Szer, François Charpillet, and Shlomo Zilberstein. 2005. MAA*: a heuristic search algorithm for solving decentralized POMDPs. In *Uncertainty in Artificial Intelligence (UAI)*. 576–583.

[27] Feng Wu, Shlomo Zilberstein, and Xiaoping Chen. 2011. Online planning for multi-agent systems with bounded communication. *Artificial Intelligence* 175, 2 (2011), 487–511.