2012 American Control Conference
Fairmont Queen Elizabeth, Montréal, Canada
June 27-June 29, 2012

Scalable, MDP-based Planning for Multiple, Cooperating Agents

Joshua D. Redding[+], N. Kemal Ure[*], Jonathan P. How[**]
Aerospace Controls Laboratory, MIT
Cambridge, MA, USA

Matthew A. Vavrina[†] and John Vian[††]
Boeing Research & Technology
Seattle, WA, USA

***Abstract*— This paper introduces an approximation algorithm for stochastic multi-agent planning based on Markov decision processes (MDPs). Specifically, we focus on a decentralized approach for planning the actions of a team of cooperating agents with uncertainties in fuel consumption and health-related models. The core idea behind the algorithm presented in this paper is to allow each agent to approximate the representation of its teammates. Each agent therefore maintains its own planner that fully enumerates its local states and actions while approximating those of its teammates. In prior work, the authors approximated each teammate individually, which resulted in a large reduction of the planning space, but remained exponential (in $n-1$ rather than in n, where n is the number of agents) in computational scalability. This paper extends the approach and presents a new approximation that aggregates all teammates into a single, abstracted entity. Under the persistent search & track mission scenario with 3 agents, we show that while resulting performance is decreased nearly 20% compared with the centralized optimal solution, the problem size becomes linear in n, a *very* attractive feature when planning online for large multi-agent teams.**

I. Introduction

Many interesting applications of autonomous robotic systems benefit from, or even require, the use of multiple agents. Enabling efficient, preferably near-optimal, cooperation between these agents is the objective of any cooperative planning algorithm. Cooperative planners based on Markov decision processes (MDPs) [1]–[3] have shown tremendous versatility in modeling such systems. Unfortunately, MDPs and related dynamic programming techniques have known scaling issues [4], and quickly become intractable as the number of participating agents increases. As a result, there exists many approximate dynamic programming (ADP) algorithms designed to compensate for this computational challenge [5,6]. While many of these approaches have shown success in classes of problems that cannot be solved exactly (e.g. [7]), most have fundamental limitations such as the difficulty of selecting an appropriate approximation architecture or the absence of proper guidance to tune algorithm parameters.

[+]Research Lead, Procerus Technologies joshr@procerus.com
[*]Ph.D. candidate in the ACL at MIT ure@mit.edu
[**]Richard C. Maclaurin Professor of Aeronautics and Astronautics, Aerospace Controls Laboratory (Director), MIT, jhow@mit.edu
[†]Research Eng. at Boeing R&T matthew.a.vavrina@boeing.com
[††]Technical fellow at Boeing R&T john.vian@boeing.com

Another concept directly associated with multi-agent systems is decentralized decision making, where agents need to decide their own actions while communicating observations etc. with other agents to maximize a global reward function. A thorough survey on the subject is presented [8]. One of the fundamental work in the area is [9], where each agent can only observe a portion of the environment and other agent's states, resulting in a decentralized partially observable Markov decision process (Dec-POMDP). The authors in [10] investigated the complexity of various decentralized formulations and showed that most of them are computationally intractable except some simple cases. Methods in [11] offer some guidance on how to relax this complexity for special cases. One potentially tractable approach to this problem is a transition independent MDP (TI-MDP), found in [12]. Although this approach scales to higher dimensions better than many others, it requires an extensive list of state transition histories per agent, referred to as *events*. Decentralized planning with communication constraints have been investigated in [13,14], where it is shown that the communication problem (namely, *if* communication is cost-effective) can be embedded into the decision making problem by separating communication actions from conventional actions and adding a communication cost to the reward function. In addition, decentralized sparse interaction MDPs (Dec-SIMDPs) have been introduced in [15] to deal with computation limitations by dividing by differentiating the parts of the state space where agents need to cooperate from the parts where they can act independently. Motivated by the limited scalability and implementation difficulties of the existing algorithms, this research aims to develop decentralized and scalable approximate planning models that are solvable in real-time without sacrificing too much optimality. The authors have previously introduced a decentralized problem formulation for a multi-agent persistent surveillance problem and experimentally verified it in [16]. This work extends these previous results by increasing the scalability of the formulation and thus allowing to solve larger, more complex problems. Specifically, we introduce a method for each agent to approximate the state-action space of all its teammates in cooperative, multi-agent domains.

978-1-4577-1096-4/12/$26.00 ©2012 AACC

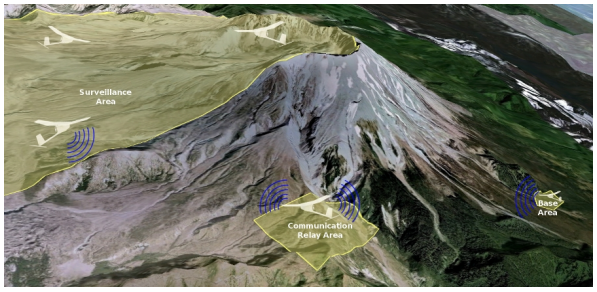

Fig. 1. Mission scenario: N autonomous agents cooperate to continuously survey a specified region and to track any objects of interest discovered there while maintaining constant communication with the base location. This behavior is to be persistently maintained even under sensor, actuator and battery health degradations.

II. Problem Description

Although the approximation techniques introduced in this paper are not tied to any specific multi-agent mission scenario, the remainder of this paper applies the approximation in the context of a persistent search and track (PST) mission scenario. The purpose of this section is to briefly describe this scenario, while Refs. [16,17] provide more details. As shown in Figure 1, the mission area is divided into three distinct, spatial regions. These regions are labeled as the *Surveillance*, *Communication Relay*, and *Base* areas. Several target vehicles are "hidden" among a number of civilian (e.g. neutral) vehicles in the surveillance area, and the overall mission objective is to continuously search for, and subsequently track, these target vehicles. Furthermore, there is an additional requirement that the communication area must be occupied by an agent whenever one or more agents are in the surveillance area, in order to relay communications to/from the base. Accomplishing these objectives is complicated by stochastic sensor, actuator and fuel dynamics. Thus, as fuel depletes and as failures or other health degradations occur, these agents must return to base for refueling and/or repair, potentially causing a "gap" in the coverage of the surveillance and/or communication areas. To prevent this gap, the agents must coordinate their actions to proactively anticipate failures and fuel-consumption.

Furthermore, each agent has a non-zero probability of experiencing sensor failure or actuator degradation, which may reduce their capabilities to below that which is required to perform certain aspects of the mission. For instance, an agent with a failed sensor can no longer perform search or track tasks in the surveillance area. However, it can still perform the duty of communication relay. Similarly, an agent with a damaged actuator, though still capable, becomes less effective at searching, tracking, or acting as a communication relay and therefore receives a slightly higher cost if assigned to perform such duties. All health infringements (fuel depletion, sensor failures and actuator degradations) are repaired once the agent returns to the base location. Further details regarding the stochastic health models employed are found in Section III.

III. Problem Formulation

In this section, we formulate the PST mission described above (see Section II) in a new fashion that seeks to tame this exponential explosion by allowing each agent to model all of its teammates in an aggregated, approximate sense. We first, however, offer a brief background covering MDPs, multi-agent MDPs (MMDPs), decentralized MDPs (Dec-MDPs) and decentralized multi-agent MDPs (Dec-MMDPs).

A. Markov Decision Process (MDP)

An infinite-horizon, discounted MDP is specified by the following tuple: $< \mathcal{S}, \mathcal{A}, P, g, \alpha >$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P_{ij}(u)$ gives the probability of transitioning into state $j \in \mathcal{S}$ from state $i \in \mathcal{S}$ having taken action $u \in \mathcal{A}$, and $g(i,u)$ gives the cost of taking action a in state s. We assume that the model, P, is known. Future costs are discounted by a factor $0 < \alpha < 1$. The outcome, e.g. solution, is a policy, denoted by $\mu : \mathcal{S} \rightarrow \mathcal{A}$, and is a mapping of states to actions. Given the MDP specification, the policy is found by minimizing the *cost-to-go* function J_μ over the set of admissible policies Π, as shown here:

$$\min_{\mu \in \Pi} J_\mu(i_0) = \min_{\mu \in \Pi} \mathbb{E}\left[\sum_{k=0}^{\infty} \alpha^k g(i_k, \mu(i_k))\right],$$

where i_0 is the initial state and i_k denotes the state at time k. The cost-to-go for a fixed policy μ satisfies the Bellman equation [5]

$$J_\mu(i) = g(i, \mu(i)) + \alpha \sum_{j \in \mathcal{S}} P_{ij}(\mu(i)) J_\mu(j) \quad \forall i \in \mathcal{S}, \qquad (1)$$

which can also be expressed compactly as $J_\mu = T_\mu J_\mu$, where T_μ is the (fixed-policy) dynamic programming operator [4].

B. Multi-Agent MDP (MMDP)

A multi-agent Markov decision process (MMDP) generalizes the above MDP to multi-agent, cooperative scenarios. An MMDP is represented by essentially the same tuple, but with the addition of n to indicate the number of participating agents. As $\mathcal{S}$ now represents the *joint* state space of all agents and $\mathcal{A}$ the *joint* action space, it follows that $s \in \mathcal{S}$ and $a \in \mathcal{A}$ now denote a *joint* state and a *joint* action. As a result, the policy becomes a mapping of joint states to joint actions. In an MMDP, each agent has *individual full observability* of the joint state. This means that each agent can itself observe the full, joint state and an observation model is not necessary. However, computational complexity scales exponentially in the number of agents, making it a poor choice for planning in large teams.

C. Decentralized MDP (Dec-MDP)

Relaxing individual full observability just a notch, we arrive at *joint full observability*, which means that the combined observations of all agents can represent the full joint state. A decentralized Markov decision process (Dec-MDP) is a generalization of the MMDP in the sense that local full observability is relaxed to joint full observability. Technically, a Dec-MDP is specified by the following tuple: $< n, \mathcal{S}, \mathcal{A}, P, g, \alpha, \mathcal{Z}, O >$, where n is the number of agents, $\mathcal{S}$ is again the joint state space, $\mathcal{A}$ is again the joint action space, $P(s'|s,a)$ again gives the probability of transitioning

to joint state s' when starting from joint state s and taking joint action a, and $g(s,a)$ again gives the cost of taking action a from state s. New to this formulation however, are the set $\mathcal{Z}$ and the model O. The set $\mathcal{Z}$ contains all joint observations and the model $O(z|s,a)$ gives the probability of receiving joint observation $z \in \mathcal{Z}$ given joint state s and action a. Under the assumption of joint full observability, for each joint state-action pair there exists a joint observation $z = \{z_1 \dots z_n\}$ such that $O(z|s,a) = 1$.

D. Decentralized MMDP (Dec-MMDP)

The MMDP and Dec-MDP inherently capture all inter-agent coupling, at the cost of exhaustively enumerating the joint state- and action-space, but do not scale well in the number of agents. Motivated by the need to construct and adapt planner output in as close to real-time as possible, previous work has considered alternate problem formulations where approximations were introduced in the formulation itself, rather than applied to the solution approach, e.g. decentralized sparse-interaction MDPs [15] and decentralized multi-agent MDPs (Dec-MMDPs) [16]. A Dec-MMDP requires individual full observability and in return provides an action for a single agent based on its local state and an abstraction/approximation of each of its teammate's local state-action spaces. By doing so, the Dec-MMDP provides the flexibility for the problem designer to control the inherent trade-off between problem size (i.e. solution speed) and the level of inter-agent coupling captured in the formulation (i.e. solution optimality), based on how this teammate abstraction/approximation is implemented (see also [16]).

IV. Group Aggregate Dec-MMDP (GA-Dec-MMDP)

This section provides the details of a novel decentralized approximate modelling approach for scalable cooperative planning. In the previous decentralized approximation Dec-MMDP (see [16]), each agent represented its teammates with a reduced-dimensional model, generated using state aggregation on the full model. The focus of this section is to extend this approach and allow each agent to approximate all of its teammates as a group with a single, reduced model. This model is generated using aggregation techniques on the joint state-action space of the teammates. The motivation for this approach comes from the decision-making perspective of an individual agent: that as long as someone satisfies mission goals/constraints, it does not need to know specifically who, or how. Thus, the aggregated state of all of an agent's teammates is reduced to a combination of features, such as the total number of agents in surveillance area, etc. The particular advantage of this formulation is that the growth of the size of the state space can be made linear in the number of agents, rather than exponential.

A GA-Dec-MMDP is a tuple $< n, \mathcal{S}, \mathcal{A}, P, g, \alpha, \mu^{n=1} >$ where n is the number of agents, $\mathcal{S}$ and $\mathcal{A}$ are again the state and action spaces and $P_{ij}(u)$ again gives the transition probability from state i to state j under action u. As before, $g(i,u)$ gives the cost of taking action u in state i and future costs are still discounted by a factor $0 < \alpha < 1$ and $\mu^{n=1}$ remains a fixed policy that results from a single-agent MDP. The differences are in how $\mathcal{S}$, $\mathcal{A}$ and P are constructed, which is outlined in the following sections where the components of the GA-Dec-MMDP are formulated for agent i, with its set of teammates denoted as $\Omega \equiv \{1 \dots n\} \setminus \{i\}$.

A. Single-agent Policy $\mu^{n=1}$

$\mu^{n=1}$ is a fixed policy that is the result of formulating and solving a single-agent MDP where $\mathcal{S}$ is $\mathcal{S}_i$ from IV-B, $\mathcal{A}$ is identical to IV-C, P is P_i from IV-D and g is modified to remove the communication relay requirement.

B. State Space $\mathcal{S}$

In the GA-Dec-MMDP formulation, the state space is factored as $\mathcal{S} = \mathcal{S}_i \times \mathcal{S}_\Omega$, where $\mathcal{S}_i$ denotes the local state of agent i and $\mathcal{S}_\Omega$ represents the collective state-action space for all of agent i's teammates, or the *group-aggregate* state.

For the PST mission, the local state of each agent is given by three scalar variables describing the agent's location, fuel remaining and health status. The location of agent i is denoted as y_i,

$$y_i \in \{Y_B, Y_C, Y_S\} \tag{2}$$

where Y_B is the *Base* area, Y_C is the *Communication Relay* area, and Y_S is the *Surveillance* area shown in Figure 1. Similarly, the fuel state of agent i, f_i, is described by a discrete set of possible fuel quantities,

$$f_i \in \{0, \Delta f, 2\Delta f, \dots, F_{max} - \Delta f, F_{max}\} \tag{3}$$

where Δf is an appropriate discrete fuel quantity. Agent i's health status, h_i, is described by a discrete set of possible health states, given by

$$h_i \in \{H_{nom},\ H_{sns},\ H_{act}\} \tag{4}$$

where H_{nom}, H_{sns} and H_{act} respectively represent nominal health, a failed sensor and a damaged actuator. Combining these parts, an agent's local state space, $\mathcal{S}_i$, is defined by the cross product of the states y_i, f_i and h_i, which yields

$$\mathcal{S}_i = [y_i \times f_i \times h_i]$$

The purpose of the group aggregate state, $\mathcal{S}_\Omega$, is to compactly represent all of agent i's teammates. Although the content of $\mathcal{S}_\Omega$ is problem-dependent, the objective function provides a guideline for its construction. For example, in the PST mission, the objective is to So, to avoid enumerating all possible combinations of y_i, f_i and h_i for each teammate, agent i aggregates the state-action spaces of its teammates into a set of features using ϕ

$$\mathcal{S}_\Omega = \phi(\mathcal{S}_{ij}),\ \forall j \in \Omega, \tag{5}$$

where $\mathcal{S}_{ij}$ is agent i's representation of agent j's local state-action space and ϕ is a function that extracts information relevant to the inter-agent coupling in the cost function. In this sense, ϕ is very similar to the features used in approximate dynamic programming [5] to quantize problems with large state spaces. In this paper, ϕ simply extracts the expected

location of each teammate at the next timestep. Forward propagation of the state of each teammate is accomplished using agent i's perception of their local state and agent i's non-cooperative policy $\mu^{n=1}$. In other words, agent i predicts each teammate's action based on what he would do if his teammate's local state were his own. Agent i then evaluates these predicted teammate states against elements of his local cost function to construct $\mathcal{S}_\Omega$. Hence, the aggregate state is written as

$$\mathcal{S}_\Omega = [\mathbf{I}_{[n_c>0]} \times n_s] = \begin{bmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & n-1 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & n-2 \end{bmatrix} \tag{6}$$

where n_c denotes the number of teammates in the communication area and $n_s \in \{0,\ n-1\}$ the number of teammates in the surveillance area. $\mathbf{I}_{[n_c>0]} \in \{0,\ 1\}$ is an indicator function on whether the communication requirement is satisfied. The system state vector $\mathbf{x}$ for this formulation is given by $\mathbf{x} = [y_i\ f_i\ h_i,\ \mathcal{S}_\Omega]$. The size of the state space is found by counting all possible realizations of the state vector $\mathbf{x}$, yielding $|\mathcal{S}| = (|Y| \times |F| \times |H|) \times |\mathcal{S}_\Omega|$, which scales linearly in n as $|Y|$, $|F|$, and $|H|$ are constants and $|\mathcal{S}_\Omega| = 2n$.

C. Action Space $\mathcal{A}$

The control space differs from the centralized formulation in that here it is for a single agent. However, even if the control space were joint, since the construction of the state space allows for approximating teammate states, it cannot be guaranteed that the actions agent i chooses for his teammates will be the same actions they will choose for themselves. Otherwise, the actions available to each agent in general are $u \in \{-1,\ 0,\ +1\}$, which correspond to {"Toward *Base*", "Stay", "Away from *Base*"} respectively. However, the specific controls u_i available for the i^{th} agent depend on the agent's current location y_i and its remaining fuel f_i, according to the following rules:

$$u_i \in \begin{cases} \{-1,\ 0,\ +1\}, & \text{if } y_i = Y_C \\ \{-1,\ 0\}, & \text{if } y_i = Y_S \\ \{0,\ +1\}, & \text{if } y_i = Y_B \\ \{0\}, & \text{if } f_i = 0 \end{cases} \tag{7}$$

The total system action vector $\mathbf{u}$ for the decentralized formulation is simply $\mathbf{u} = u_i$, leaving the size of the action space $|\mathcal{A}| = (|u_i|)$.

D. State Transition Model P

The state transition model P captures the qualitative description of the dynamics of the state, given an action. As the state is divided into $\mathcal{S}_i$ and $\mathcal{S}_\Omega$, transitions for each, P_i and P_Ω, are given.

1) Local State Transitions, P_i: The model for agent location y_i is deterministic and is described by the rules:

$$y_i(k+1) = \begin{cases} y_i(k), & \text{if: } f_i = 0 \\ Y_B, & \text{if: } y_i(k) = Y_B,\ u_i(k) = 0 \\ Y_B, & \text{if: } y_i(k) = Y_C,\ u_i(k) = -1 \\ Y_C, & \text{if: } y_i(k) = Y_C,\ u_i(k) = 0 \\ Y_C, & \text{if: } y_i(k) = Y_B,\ u_i(k) = +1 \\ Y_C, & \text{if: } y_i(k) = Y_S,\ u_i(k) = -1 \\ Y_S, & \text{if: } y_i(k) = Y_S,\ u_i(k) = 0 \\ Y_S, & \text{if: } y_i(k) = Y_C,\ u_i(k) = +1 \end{cases} \tag{8}$$

The dynamics for the fuel state f_i are stochastic with parameter p_f representing the probability of burning fuel at the nominal rate of one Δf per timestep. Specifically, f_i evolves according to the following rules:

$$f_i(k+1) = \begin{cases} 0, & \text{if: } f_i(k) = 0 \\ F_{max}, & \text{if: } y_i(k) = Y_B \\ f_i(k) - \Delta f, & \text{w/ Pr}(p_f) \text{ if } y_i(k) \neq Y_B \\ f_i(k) - 2\Delta f, & \text{w/ Pr}(1-p_f) \text{ if } y_i(k) \neq Y_B \end{cases} \tag{9}$$

The health state of each agent is also a stochastic model with parameters p_s and p_a representing the probability of a sensor failure, and actuator damage respectively. This health model evolves according to the following rules:

$$h_i(k+1) = \begin{cases} h_i(k), & \text{if } f_i = 0 \\ H_{nom}, & \text{if } y_i(k) = Y_B \\ H_{nom}, & \text{w/ Pr}(1-p_s-p_a) \text{ if } h_i(k) = H_{nom} \\ H_{sns}, & \text{w/ Pr}(p_s) \text{ if } h_i(k) = H_{nom} \\ H_{act}, & \text{w/ Pr}(p_a) \text{ if } h_i(k) = H_{nom} \end{cases} \tag{10}$$

2) Feature Transitions, P_Ω: Since feature transitions strongly depend on the accuracy of $\mu^{n=1}$ and other agents actual actions, straightforward calculation of P_Ω is usually not possible. In order to circumvent this problem, a look-up table that quantitatively describes the transition probabilities between all realizations of $\mathcal{S}_\Omega$ was generated by evaluating state trajectories recorded while running the PST mission with $n = 5$ agents under a Dec-MMDP formulation [16]. Transferring this quantitative approximation to cases with more agents (cases intractable for MMDP and Dec-MMDP) is not immediately clear. One approach is gleaned from the image processing literature, as scaling the empirical look-up table is essentially the same problem as zooming in on an image. As visualized in Figure 2, bicubic interpolation was used to empirically estimate the transition probabilities for cases where $n = 5$ (left) to $n = 10$ (right). Darker areas indicate higher probability and each row sums to 1.

E. Cost Function g

The cost function $g(\mathbf{x}, \mathbf{u})$ in the decentralized case is set up to penalize any undesirable outcomes in the mission. However, the presence of approximations in forming $\mathcal{S}_\Omega$ remove some of the reward coupling between agents and cannot therefore "peak" into future possibilities the way the centralized problem can.

$$\begin{aligned} g(\mathbf{x}, \mathbf{u}) \quad = \quad & C_u n_u + C_{mot} n_{mot} + C_{sns} n_{sns} + \\ & C_s(n-1-n_s) + C_c\left(1 - \mathbf{I}_{[n_c>0]}\right) + C_x n_x \end{aligned} \tag{11}$$

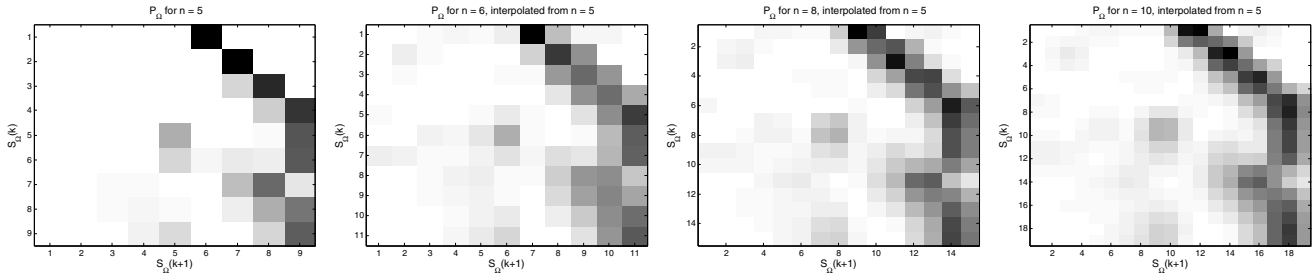

Fig. 2. Bicubic interpolation of the empirically-determined transition probabilities for P_Ω from $n = 5$ (left) to $n = 10$ (right). Darker areas indicate higher probability and each row sums to 1.

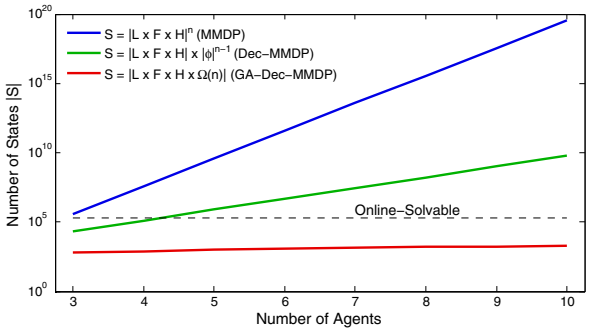

Fig. 3. As the number of agents n increases, the resulting state space, and therefore also the computational complexity associated with calculating a solution, can grow exponentially and the choice of planning algorithm becomes critical. Note, the y-axis is log scale.

where $C_\star$ are costs associated with the following numbers: $n_u \in \{0,\ 1\}$ denotes agent movement, $n_{mot} \in \{0,\ 1\}$ indicates if the agent is in the task area with a degraded motor, $n_{sns} \in \{0,\ 1\}$ indicates if the agent is in the task area with failed sensor, $n_s \in \{0,\ n\}$ denotes the number of capable agents in the surveillance area, $n_c \in \{0,\ n\}$ denotes the number of agents in the communication area, and n_X is the number of agents that have run out of fuel (crashed).

V. Simulation Results

The objective of the simulation is as described in Section II, to maintain communication between *Base* and *Surveillance* areas and to keep as many *capable* agents in the *Surveillance* area as possible.

Monte Carlo style simulations of the full, stochastic PST mission were run in this environment for several planning approaches, including: A non-cooperative approach; a state-dependent heuristic; a fully centralized MMDP (for the case of $n = 3$ only); a Dec-MMDP as given in Ref [16]; and a GA-Dec-MMDP, as formulated in IV.

First, the non-cooperative approach simulates each agent acting for itself only and is included as benchmark to allow a more complete understanding of the costs incurred by the cooperative planners. Second, the state-dependent heuristic represents the type of behaviour a human operator might encode for an unmanned agent in the PST mission. Heuristic policy simply sends $n - 1$ UAVs to surveillance area while reserving one UAV for communication purposes. Then it calls a UAV backs to base whenever its fuel level is below some constant threshold.

The other three planners up for comparison are MDP-based, and as this research focuses on problem formulation rather than on particular solution approaches, all solutions were computed using exact value-iteration, which is known to be $\mathcal{O}(|S|^2|A|)$ [5]. Because of this, we can use the term "computationally complexity" in lieu of "state-space size". With this is mind, Figure 3 shows how the computational complexity of these three MDP-based planners grows as the number of agents increases. Note, the y-axis of the figure is log-scale and shows the exponential growth in n of MMDPs and Dec-MMDPs. The GA-Dec-MMDP formulation however, remains linear in n.

For the PST mission scenario simulations, the following parameters were used: $\Delta f = 1$, $F_{max} = 10$, $p_f = 0.50$, $p_a = 0.05$, $p_s = 0.10$. The simulations were run on a 64-bit quad-core Intel Xeon 3.33GHz CPU running Ubuntu 11.04 with GCC 4.5 and 12 Gb of RAM. For the cases where n is greater than three, the MMDP approach is simply not tractable, having more than 25 million states when $n = 4$. Therefore, comparisons beyond $n = 3$ do not include MMDP results. Similarly, the Dec-MMDP approach remains computationally tractable until $n = 5$ agents, beyond which it too becomes intractable. The GA-Dec-MMDP approach however, remains tractable through $n = 10$. We therefore compare GA-Dec-MMDP with the non-cooperative and state-dependent heuristic approaches for cases when $n > 5$.

After ensuring identical starting conditions, each planner was simulated for 500 steps, 50 times each, logging the joint state trajectories for each system. Using these state histories, the average cumulative return for each planner was calculated using an evaluation cost function described by

$$g(\mathbf{x}) = C_c\left(1 - \mathbf{I}_{[n_c>0]}\right) + C_s(n - 1 - n_s) \qquad (12)$$

where n_s is the number of capable agents in the surveillance area and n_c is the number of agents in the communication area. Figure 4 compares the scores of the different planners for the case of three agents ($n = 3$). As expected, the MMDP results in the minimum cost solution, while the Dec-MMDP and GA-Dec-MMDP approximations are within 10% and 20% respectively. As the non-cooperative and state-dependent heuristic approaches result in much higher cost, and Figure 5 removes them for a closer look at the MDP-based strategies alone.

Scaling up now to $n = 10$, Figure 6 shows how the cumulative cost scales with the number of participating agents.

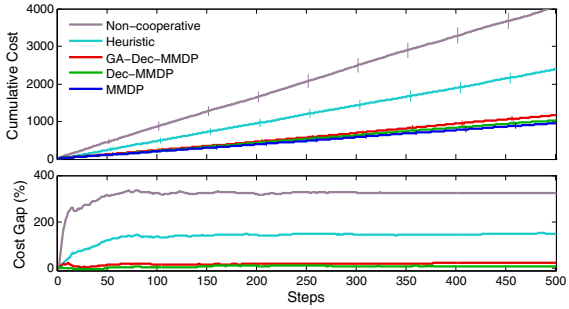

Fig. 4. Comparison of cumulative costs over a 500 step stochastic PST mission for the case of three agents ($n = 3$). As expected, the non-cooperative solution scores poorly while the MDP-based solutions provide the lowest cost solutions.

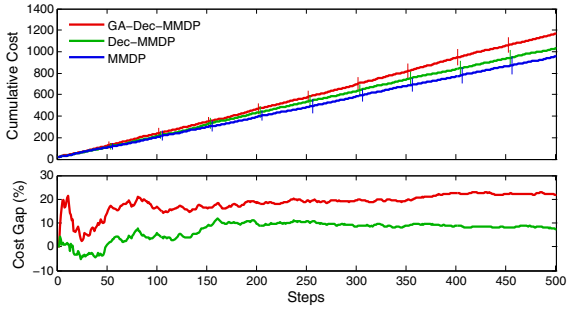

Fig. 5. Removing the non-cooperative and heuristic-based approaches allows a clearer comparison of the cumulative costs over a 500 step mission for the case of three agents between the MDP-based methods only. As seen, Dec-MMDP results in roughly 10% higher cost than the MMDP while GA-Dec-MMDP yields approximately 20% higher cost.

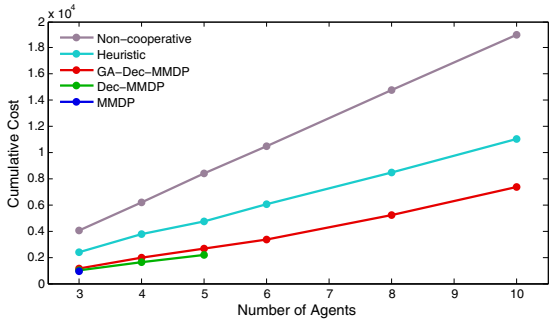

Fig. 6. Comparison of resulting cumulative cost after a 500 step stochastic PST mission as a function of the number of participating agents. GA-Dec-MMDP (red) remains the lowest cost tractable solution through $n = 10$ agents.

While we cannot compare with the centralized MMDP or Dec-MMDP in these cases, the main point of this figure is to show that the cumulative cost of the GA-Dec-MMDP approximation consistently results in a much lower cost than the heuristic approach.

VI. Conclusions

This paper introduced an approximation algorithm for stochastic multi-agent planning based on MDPs. The core idea behind the algorithm was to allow each agent to maintain its own planner that approximates the representation of its teammates, while fully enumerating its local state and action spaces. The focus of this paper is an extension to previous work that aggregates all teammates into a single, abstracted entity. Under the persistent search & track mission scenario, we showed that while performance is approximately 20% of that obtained by the centralized solution, the problem size is only linear in n, which is a very attractive feature when planning online for large multi-agent teams.

Acknowledgments

This research was generously supported by Boeing Research & Technology in Seattle, WA and in part by AFOSR grant FA9550-09-1-0522. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government.

References

[1] M. Valenti, B. Bethke, J. P. How, D. P. de Farias, and J. Vian, "Embedding Health Management into Mission Tasking for UAV Teams," in *American Control Conference (ACC)*, New York City, NY, 9-13 July 2007, pp. 5777–5783. [Online]. Available: http://acl.mit.edu/papers/acc_hlth_mgmt_2007_mv_v0.pdf

[2] B. Bethke, J. P. How, and J. Vian, "Group health management of UAV teams with applications to persistent surveillance," in *American Control Conference (ACC)*, Seattle, WA, 11-13 June 2008, pp. 3145–3150. [Online]. Available: http://acl.mit.edu/papers/acc-fuel-2008-bbethke.pdf

[3] ——, "Multi-UAV Persistent Surveillance With Communication Constraints and Health Management," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, August 2009, (AIAA-2009-5654).

[4] R. Bellman, *Dynamic Programming*. Dover Publications, March 2003.

[5] D. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 2007.

[6] L. Busoniu, R. Babuska, B. D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*, L. Busoniu, R. Babuska, B. DeSchutter, and D. Ernst, Eds. CRC Press, 2010.

[7] B. M. Bethke, "Kernel-based approximate dynamic programming using bellman residual elimination," Ph.D. dissertation, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge MA, February 2010. [Online]. Available: http://acl.mit.edu/papers/BethkePhD.pdf

[8] S. Seuken and S. Zilberstein, "Formal models and algorithms for decentralized decision making under uncertainty," *Autonomous Agents and Multiagent Systems*, vol. 17, no. 2, pp. 190–250, 2008.

[9] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Mathematics of Operation Research*, vol. 27, no. 4, pp. 819–840, 2002.

[10] C. V. Goldman and S. Zilberstein, "Decentralized control of cooperative systems: Categorization and complexity analysis," *Journal of Artificial Intelligence Research*, vol. 22, pp. 143–174, 2004.

[11] M. Allen and S. Zilberstein, "Complexity of decentralized control: Special cases," in *Annual Conf. on Neural Information Processing Systems*, 2009.

[12] R. Becker, S. Zilberstein, V. Lesser, and C. Goldman, "Transition-independent decentralized Markov decision processes," in *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM, 2003, pp. 41–48.

[13] V. L. Ping Xuan and S. Zilberstein, "Communication decisions in multi-agent cooperation: Model and experiments," in *Proceedings of the fifth international conference on Autonomous agents*, 2001.

[14] M. S. Spaan, G. J. Gordon, and S. Zilberstein, "Decentralized planning under uncertainty for teams of communicating agents," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006.

[15] F. S. Melo and M. Veloso, "Decentralized mdps with sparse interactions," *Artificial Intelligence*, vol. 175, pp. 1757–1789, 2011.

[16] J. D. Redding, T. Toksoz, N. K. Ure, A. Geramifard, J. P. How, M. Vavrina, and J. Vian, "Persistent distributed multi-agent missions with automated battery management," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, August 2011, (AIAA-2011-6480).

[17] B. Bethke, J. P. How, and J. Vian, "Multi-UAV Persistent Surveillance With Communication Constraints and Health Management," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, August 2009, (AIAA-2009-5654).