Decentralized Decision-Making Under Uncertainty for Multi-Robot Teams

Christopher Amato[1,2] and George Konidaris[2] and Jonathan P. How[2] and Leslie P. Kaelbling[1]

***Abstract*— Automatically generating solutions to general multi-robot coordination problems with communication limitations is challenging, but crucial in many domains. As one way to address this problem, we describe a probabilistic framework for synthesizing control policies for general multi-robot systems based on decentralized partially observable Markov decision processes with macro-actions (MacDec-POMDPs). MacDec-POMDPs are a general model of decision-making where a team of robots cooperates to optimize a shared objective in the presence of uncertainty. MacDec-POMDPs also consider communication limitations, so execution is decentralized. We describe how, in contrast to most existing methods that are specialized to a particular problem class, we can synthesize control policies that exploit whatever opportunities for coordination are present in the problem, while balancing uncertainty, sensor information, and information about other robots.**

I. Introduction

A wide range of approaches have been developed for solving specific classes of multi-robot problems, such as task allocation [1], navigation in a formation [2], cooperative transport of an object [3], and communication under various limitations [4]. Broadly speaking, the current state of the art is to hand-design special-purpose controllers that exploit some property of the environment or produce a specific desirable behavior. It would be preferable to instead specify a world model and a cost metric, and then have a general-purpose planner automatically derive controllers that find minimum cost solutions while remaining robust to uncertainty.

The decentralized partially observable Markov decision process (Dec-POMDP) is a general framework for representing multiagent coordination problems. Dec-POMDPs have been widely studied in artificial intelligence as a way to address the fundamental differences in decision-making in decentralized settings [5], [6], [7]. Like the POMDP [8] model that it extends, Dec-POMDPs consider general dynamics, cost and sensor models. Any problem where multiple robots share a single overall reward or cost function can be formalized as a Dec-POMDP. As such, Dec-POMDP solvers could automatically generate control policies (including policies over when and what to communicate) for very rich decentralized control problems, in the presence of uncertainty in outcomes, sensors and information about the other robots. Unfortunately, this generality comes at a cost: Dec-POMDPs are typically infeasible to solve except for very small problems [6], [9].

One reason for the intractability of solving large Dec-POMDPs is that current approaches model problems at a low level of granularity, where each robot's actions are primitive operations lasting exactly one time step. Recent research has addressed the more realistic *MacDec-POMDP* case where each robot has *macro-actions*: temporally extended actions which may require different amounts of time to execute [9]. This enables systems to be modeled so that coordination decisions only occur at the level of deciding which macro-actions to execute. Macro-actions are a natural model for the modular controllers (e.g., navigating to a waypoint or grasping an object) often sequenced to obtain robot behavior, bridging the gap between robotics research and Dec-POMDPs. This approach has the potential to produce high-quality general solutions for real-world heterogeneous multi-robot coordination problems by automatically generating control and communication policies, given a model.

[1]CSAIL, [2]LIDS, MIT, Cambridge, MA 02139

II. MacDec-POMDPs

The MacDec-POMDP formulation models a group of robots that must plan by sequencing an existing set of controllers. It extends the Dec-POMDP model to plan using *options*, or temporally extended actions [9]. In Dec-POMDPs (as depicted in Fig. 1), multiple robots operate based on partial and local views of the world. At each step, every robot chooses an action (in parallel) based purely on locally observable information, resulting in an an observation for each individual robot. The robots also share a single reward or cost function, making the problem cooperative, but their local views mean that execution is decentralized.

A MacDec-POMDP is defined by a tuple $\langle I, S, \{A_i\}, \{M_i\}, T, R, \{\Omega_i\}, O, h\rangle$, where I is a finite set of robots, S is a finite set of states, A_i is a finite set of low-level actions for each robot i with $A = \times_i A_i$ the set of joint actions; M_i is a finite set of options for each robot, i, with $M = \times_i M_i$ the set of joint options [9]; T is a state transition probability function, $T : S \times A \times S \rightarrow [0, 1]$ with $T(s, \vec{a}, s') = \Pr(s'|\vec{a}, s)$, R is a reward function: $R : S \times A \rightarrow \mathbb{R}$, the immediate reward for being in state $s \in S$ and taking the actions $\vec{a} \in A$, Ω_i is a finite set of observations for each robot, i, with $\Omega = \times_i \Omega_i$ the set of joint observations, O is an observation probability function: $O : \Omega \times A \times S \rightarrow [0, 1]$, with $O(\vec{o}, \vec{a}, s') = \Pr(\vec{o}|\vec{a}, s')$, and h is the horizon. Because the full state is not directly observed, optimal or approximately optimal behavior generally requires each agent to remember a history of its observations. We define an action-observation history for agent i (up to step t) as $H_i^A = (a_i^0, o_i^0, \ldots, a_i^t, o_i^t)$.

For simplicity, we consider only *local options* that only depend on a single robot's information: $M_i = (\beta_{m_i}, \mathcal{I}_{m_i}, \pi_{m_i})$,

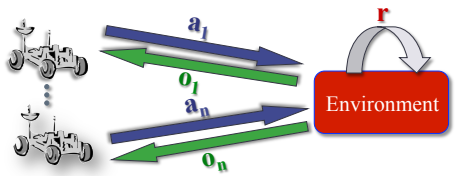


Fig. 1. Representation of an n-robot Dec-POMDP with actions a_i and observations o_i for each robot i along with a single reward r.

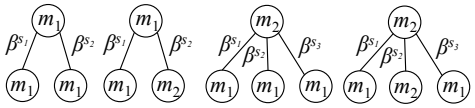


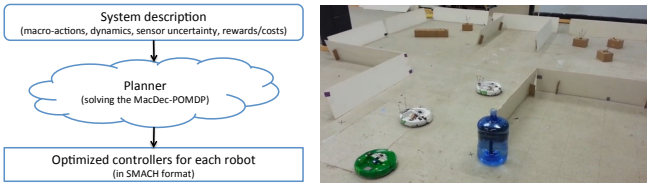
Fig. 2. Policies for four robots for two (macro-action) steps using options m_1 and m_2 and (deterministic) terminal states as β^s.

with stochastic termination condition $\beta_{m_i} : H_i^A \rightarrow [0,1]$, initiation set $\mathcal{I}_{m_i} \subset H_i^A$ and option policy $\pi_{m_i} : H_i^A \times A_i \rightarrow [0,1]$. Note that this representation uses action-observation histories in the termination and initiation conditions as well as the option policy. Simpler cases can consider reactive policies that map single observations to actions as well as termination and initiation sets that depend only on single observations.

Since it may be beneficial for robots to remember their histories when choosing which option to execute, we define an *option history*, which includes both the action-observation histories where an option was chosen and the selected options themselves, as $H_i^M = (h_i^0, m_i^1, \ldots, h_i^{t-1}, m_i^t)$. We then define a (stochastic) local policy, $\mu_i : H_i^M \times M_i \rightarrow [0,1]$ that depends on option histories and a joint policy for all robots as μ. The goal of planning to produce a local policy for each robot, that maps its observation history to a choice of option to execute, and maximizes reward. The selected option then executes a closed-loop policy (built out of primitive actions) to completion. Existing planners [9] output a deterministic policy tree (as shown in Figure 2) for each robot, which defines a policy based on local observations. The root node defines the option to execute in the known initial state, and another option is assigned to each of the legal terminal states of that option; this continues for the depth of the tree.

III. Solving Multi-robot problems with MacDec-POMDPs

The MacDec-POMDPs framework is a natural way to represent and generate behavior for general multi-robot systems. We assume an abstract model of the system is given in the form of macro-action representations, which include the associated policies as well as initiation and terminal conditions. These macro-actions are controllers operating in (possibly) continuous time with continuous actions and feedback, but their operation is discretized for use with the planner. Given the macro-actions and simulator, the planner then automatically generates a solution which optimizes the value function with respect to the uncertainty over outcomes, sensor information and other robots. This solution comes in the form of SMACH controllers [10] which are hierarchical



(a) (b)

Fig. 3. (a) High level system diagram and (b) The warehouse domain with three robots.

state machines for use in a ROS environment. A high-level description of this process is given in Figure 3(a).

We tested this approach in in a warehousing scenario using a set of iRobot Creates (Figure 3(b)), and demonstrate how the same general model and solution methods can be applied in versions of this domain with different communication capabilities.[1] The planner was able to produce controllers that leveraged the available communication capabilities to most efficiently solve the task—including determining how and when to communicate, and how to respond to communication signals. This is the first time that Dec-POMDP-based methods have been used to solve large multi-robot domains. The results demonstrate that our methods can automatically generate the appropriate motion and communication behavior while considering uncertainty over outcomes, sensor information and other robots. Additional details can be found in a preliminary paper [11].

References

[1] B. Gerkey and M. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. Journal of Robotics Research*, vol. 23, no. 9, 2004.

[2] T. Balch and R. C. Arkin, "Behavior-based formation control for multi-robot teams," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, 1998.

[3] C. Kube and E. Bonabeau, "Cooperative transport by ants and robots," *Robotics and Autonomous Systems*, vol. 30, no. 1-2, 2000.

[4] I. Rekleitis, V. Lee-Shue, A. P. New, and H. Choset, "Limited communication, multi-robot team based coverage," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 4. IEEE, 2004.

[5] C. Amato, G. Chowdhary, A. Geramifard, N. K. Ure, and M. J. Kochenderfer, "Decentralized control of partially observable Markov decision processes," in *Proc. of the 52nd IEEE Conf. on Decision and Control*, 2013.

[6] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Mathematics of Operations Research*, vol. 27, no. 4, 2002.

[7] F. A. Oliehoek, "Decentralized POMDPs," in *Reinforcement Learning: State of the Art*, ser. Adaptation, Learning, and Optimization, M. Wiering and M. van Otterlo, Eds. Springer, 2012, vol. 12.

[8] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, 1998.

[9] C. Amato, G. Konidaris, and L. P. Kaelbling., "Planning with macro-actions in decentralized POMDPs," in *Proc. 13th Int. Conf. on Autonomous Agents and Multiagent Systems*, 2014.

[10] J. Bohren, "SMACH," http://wiki.ros.org/smach/, 2010.

[11] C. Amato, G. Konidaris, G. Cruz, C. A. Maynor, J. P. How, and L. P. Kaelbling, "Planning for decentralized control of multiple robots under uncertainty," in *Proceedings of the Workshop on Planning and Robotics (PlanRob) at ICAPS*, 2014.

[1] Videos can be seen at `http://youtu.be/fGUHTHH-JNA`