Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Decision-Making Under Uncertainty in Multi-Agent and Multi-Robot Systems: Planning and Learning

Christopher Amato
CCIS, Northeastern University, Boston, MA, USA
camato@ccs.neu.edu

Abstract

Multi-agent planning and learning methods are becoming increasingly important in today's interconnected world. Methods for real-world domains, such as robotics, must consider uncertainty and limited communication in order to generate high-quality, robust solutions. This paper discusses our work on developing principled models to represent these problems and planning and learning methods that can scale to realistic multi-agent and multi-robot tasks.

1 Introduction

As hardware costs decrease, many more systems are being deployed with multiple agents (e.g., routers, sensors, people, robots), which must interact to produce a high-quality solution. However, agent interaction is often complicated in the real-world by communication limitations, latency or noise. Due to these communication limitations, agents that can make decisions on their own without perfect communication are critical. For example, consider a search and rescue scenario with a team of aerial and ground robots. The ground robots have a very limited view of the world, but are able to transport people. The aerial vehicles have a much wider view of the world, but are unable to carry passengers. With limited communication, the agents must reason about when and where to explore as well as when and how to share information in order to rescue people most efficiently. Communication limitations are the most extreme in disaster, underwater, space or military settings, but are common throughout robotics, networking and multi-agent systems.

Ideally, principled methods would be used for representing multi-agent problems with uncertainty and communication limitations. The decentralized partially observable Markov decision process (Dec-POMDP) is a general model for representing and solving these cooperative multi-agent systems [Bernstein *et al.*, 2002; Oliehoek and Amato, 2016]. In fact, Dec-POMDPs are so general that they can model any multi-agent coordination problem, including the problem above as well as more general problems with combined outcome, sensor and communication uncertainty. These types of uncertainty are ubiquitous in real-world scenarios. Therefore, the Dec-POMDP provides a principled framework to consider these common forms of uncertain outcomes, sensors and communication within a single framework.

So, a large class of real-world problems are Dec-POMDPs and we must decide how to solve them. If we could solve these Dec-POMDPs optimally, the best choice could be made for each agent while considering uncertainty and the decentralized nature of the problem. Because optimal Dec-POMDP solution methods are limited to small problems due to their complexity [Bernstein *et al.*, 2002], we are left with two options: approximate the model (by making additional domain assumptions that may or may not hold in practice) or approximate the solutions. We have focused on both options, incorporating more abstract (and asynchronous) actions in the form of macro-actions and approximate solutions in the form of sample-based planning and learning. These sample-based planning and learning methods do not require a full model of the domain, but instead generate solutions from sampled execution data. As a result, these approaches make only general and realistic assumptions, while solving large Dec-POMDPs.

This paper first discusses the Dec-POMDP model and how macro-actions (asynchronous, temporally extended actions) can be incorporated. It then describes our solution methods for these models, starting with planning methods (which require a simulator or full model of the domain) and then discusses reinforcement learning approaches. The paper also discusses our application of (and inspiration for) these planning and learning methods: multi-robot domains, which are a natural fit for macro-action-based Dec-POMDPs.

2 Dec-POMDPs

Dec-POMDPs generalize POMDPs to the multi-agent, decentralized setting [Bernstein *et al.*, 2002; Oliehoek and Amato, 2016]. In a Dec-POMDP, multiple agents operate under uncertainty based on partial views of the world, with execution unfolding over time. At each step, every agent chooses an action (in parallel) based purely on locally observable information, resulting in each agent obtaining an observation and the team obtaining a joint reward. The shared reward function makes the problem cooperative, but their local views mean that execution is decentralized.

Formally, a Dec-POMDP is defined by tuple $\langle I, S, \{A_i\}, T, R, \{\Omega_i\}, O, h\rangle$, where I is a finite set of agents; S is a finite set of states with designated initial

state distribution b_0; A_i is a finite set of actions for each agent i with $A = \times_i A_i$ the set of joint actions; T is a state transition probability function, $T : S \times A \times S \rightarrow [0, 1]$, that specifies the probability of transitioning from state $s \in S$ to $s' \in S$ when the actions $\vec{a} \in A$ are taken by the agents (i.e., $T(s, \vec{a}, s') = \Pr(s'|\vec{a}, s)$); 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 agent, i, with $\Omega = \times_i \Omega_i$ the set of joint observations; O is an observation probability function: $O : \Omega \times A \times S \rightarrow [0, 1]$, the probability of seeing observations $\vec{o} \in \Omega$ given actions $\vec{a} \in A$ were taken which results in state $s' \in S$ (i.e., $O(\vec{o}, \vec{a}, s') = \Pr(\vec{o}|\vec{a}, s')$); and h is the number of steps until termination, called the horizon.

A solution to a Dec-POMDP is a *joint policy*—a set of policies, one for each agent. Because the state is not observed, it is typically beneficial for each agent to remember a history of its observations. A *local policy* for an agent is a mapping from local observation histories to actions, $H_i^o \rightarrow A_i$, where H_i^o is the set of local observation histories, $h_i^o = \{o_i^1, \ldots, o_i^t\}$, by agent i up to the current time step, t. Because the state depends on the behavior of all of the agents, it is not usually possible to estimate it from the history of a single agent (unlike in a POMDP [Kaelbling *et al.*, 1998]).

Policies are then typically represented explicitly. The common representations are policy trees, where the nodes indicate actions to execute and the edges indicate transitions conditioned on an observation, and finite-state controllers, which execute in a similar manner. Because one policy is generated for each agent and these policies depend only on local observations, they operate in a decentralized manner.

The value of a joint policy, π, from state s is $V^\pi(s) = \mathrm{E}\left[\sum_{t=0}^{h-1} \gamma^t R(\vec{a}^t, s^t)|s, \pi\right]$, which represents the expected value of the immediate reward for the set of agents summed for each step, given the the policy's actions. The discount factor, γ, is typically set to 1 in the finite-horizon case and $\gamma \in [0, 1)$ in the infinite-horizon case ($h = \infty$). An *optimal policy* beginning at state s is $\pi^*(s) = \mathrm{argmax}_\pi V^\pi(s)$. Planning and learning methods for Dec-POMDPs seek to generate optimal policies or other high-quality policies (see recent surveys for the many approaches [Amato *et al.*, 2013; Oliehoek, 2012; Oliehoek and Amato, 2016]).

3 Macro-Actions in Dec-POMDPs

While many of the Dec-POMDP solution methods can perform well, scalability has remained an issue. To combat this scalability issue, while minimizing additional assumptions we extended the Dec-POMDP framework to allow asynchronously chosen macro-actions (i.e., temporally extended actions which may require different amounts of time to complete) [Amato *et al.*, 2014]. Macro-actions enable decision-making to take place at a higher level—at the level of deciding which macro-actions to execute—and are executed to completion. This new formulation is a more realistic model of real-world systems, which allows asynchronous decision-making of higher-level actions (for e.g, waypoint navigation, grasping an object, waiting for a signal). Macro-actions also enable planning and learning for problems with significantly longer horizons.

Our approach extends the options framework [Sutton *et al.*, 1999] to Dec-POMDPs by adding macro-actions, m_i, that execute a policy in a low-level Dec-POMDP until some terminal condition is met. A Dec-POMDP with macro-actions is a Dec-POMDP where each agent i also has access to a finite set of macro actions, M_i, with $M = \times_i M_i$ being the set of joint macro-actions [Amato *et al.*, 2014]. We can then define policies for each agent, μ_i, for choosing macro-actions that depend on high-level (macro) observations. The multi-agent aspect of Dec-POMDPs introduces complications since all agents do not terminate their macro-actions at the same time, making decision-making asynchronous and evaluation more complicated. In cases when the macro-action policies and low-level Dec-POMDP are known, we can evaluate policies by 'unrolling' macro-actions (given a joint policy, the primitive action at each step is determined by the (high-level) policy, which chooses the macro-action, and the macro-action policy, which chooses the (primitive) action). When the macro-action models and low-level Dec-POMDP are not available, we can use a high-level model (as briefly described below), a simulator or the domain itself to execute the policies and provide the values.

We call this model a *MacDec-POMDP* [Amato *et al.*, 2014] when the low-level Dec-POMDP model and the policies of the macro-actions are known and a decentralized partially observable *semi-Markov* decision process (*Dec-POSMDP*) when a high-level model is defined which includes time to completion [Omidshafiei *et al.*, 2015; Amato *et al.*, 2015a] (but a simulator can be used in place of a model in each case). While these high-level models still include the states of the Dec-POMDP, they do not include the Dec-POMDP actions and observations. As a result, these low-level quantities can be continuous and the low-level dynamics and observation models may be very complicated, but we only consider the effects of the high-level macro-actions.

By extending Dec-POMDP algorithms to the macro-action case, realistic multi-agent coordination problems can be solved that are orders of magnitude larger than problems solved by previous methods and continuous state spaces can be considered, including for multiple mobile robots in warehouse [Amato *et al.*, 2015b], logistics [Amato *et al.*, 2015a; 2017] and aerial delivery [Omidshafiei *et al.*, 2015; 2017a; 2017b] scenarios. We discuss some of these methods and domains below.

4 Planning in Dec-POMDPs

We first discuss planning methods for the macro-action case and then discuss approaches for multi-robot planning.

4.1 Macro-Action-Based Methods

We first developed a tree-based policy representation using macro-actions. The key difference between the traditional Dec-POMDP case and the macro-action case is that nodes in a policy tree now select macro-actions (rather than primitive actions) and edges correspond to high-level observations. Evaluation and testing for termination becomes more complicated (e.g., Monte Carlo policy evaluations in the simulator or

Bellman equations that consider time), but we developed algorithms that search over these higher-level policies that perform well [Amato *et al.*, 2014]. The tree-based macro-action methods can produce similar values as previous Dec-POMDP approaches on smaller problems, while also solving problems that are orders of magnitude larger than those solved by previous methods [Amato *et al.*, 2014].

Because tree-based representations become intractable as the horizon grows, we also developed multiple methods for optimizing finite-state controllers in macro-action Dec-POMDPs using the MacDec-POMDP [Amato *et al.*, 2015a] and Dec-POSMDP [Omidshafiei *et al.*, 2015] models. Some of these approaches can provide solutions with only a high-level model of the macro-actions (i.e., distributions over time and outcomes) instead of a full model of the underlying Dec-POMDP [Amato *et al.*, 2015a; 2017; Omidshafiei *et al.*, 2015; 2017a]. Another approach automatically generates the macro-actions from low-level (continuous) dynamics models [Omidshafiei *et al.*, 2015], while another method generates (macro-)observations from low-level sensor (e.g., camera) information [Omidshafiei *et al.*, 2017b]. These approaches are the first Dec-POMDP-based methods to solve problems with continuous states by searching directly in the policy space and using Monte Carlo methods for estimating the values of different policies.

4.2 Multi-Robot Planning

Current multi-robot research is typically focused on subproblems or problems with little or no uncertainty (e.g., task allocation [Korsah *et al.*, 2013], formation control [Ren and Sorensen, 2008], exploration [Burgard *et al.*, 2005] or swarm control [Rubenstein *et al.*, 2014]). These common multi-robot problems address important situations, but by using a more general framework, we can solve more general problems with combined outcome, sensor and communication uncertainty (e.g., multi-robot search and rescue, decentralized exploration with an unknown map). The goal of our work has been to use ideas from Dec-POMDPs to allow us to solve these more complex multi-robot domains. In particular, the idea for macro-actions was inspired from trying to model and apply Dec-POMDPs to robotics domains. Using low-level actions would not be scalable and it was very difficult to ensure the robots were synchronized. Furthermore, high-quality controllers already existed for things like navigation and grasping, but outcomes of these controllers are uncertain both in terms of the outcomes and the time of completion.

We demonstrated our macro-action-based approaches on multiple multi-robot domains with limited sensing and communication. These domains included a warehousing problem, where robots have uncertainty about the location of boxes and limited communication, but must coordinate to collect the large boxes and individually push the small boxes to the shipping location [Amato *et al.*, 2015b]; a logistics (beer delivery) domain, where two robots must efficiently find out about and service beer orders in cooperation with a 'picker/bartender' robot, which can retrieve items [Amato *et al.*, 2015a; 2017]; and a package delivery domain, where a group of aerial robots must retrieve and deliver packages from base locations to delivery locations while dealing with limited battery life [Omidshafiei *et al.*, 2015; 2016; 2017b; 2017a].[1] Our methods outperformed simpler methods and 'expert' hand-coded solutions (e.g., delivering an additional three beers every 10 minutes). This is the first time that Dec-POMDP-based methods have been used to solve large multi-robot domains; other Dec-POMDP methods cannot solve problems of this size.

5 Learning in Dec-POMDPs

While using a high-level macro-action model makes it simpler to represent the problem domain (as fewer and higher-level parameters are used), the models or simulators may still be incorrect or unavailable. As a result, we have also focused on learning in Dec-POMDPs.

5.1 Macro-Action-Based Methods

We began by developing offline learning methods that could learn parameters for finite-state controllers from a set of trajectories of the agents acting in the environment (i.e., learning from demonstration) [Liu *et al.*, 2015]. That is, we assume a set of K trajectories resulting from N agents who choose actions for T_k steps: $\{(\vec{a}_0^k r_0^k \vec{o}_1^k \vec{a}_1^k r_1^k \cdots \vec{o}_{T_k}^k \vec{a}_{T_k}^k \ r_{T_k}^k)\}_{k=1,\cdots,K}$, where $\vec{a}_t^k$ is the vector of actions the agents chose on the t-th step of the k-th episode, after which they received the joint reward r_t^k and the vector of observations $\vec{o}_t^k$. The learning method combines Bayesian nonparametrics to learn the best controller size with an EM method for learning the parameters of the controllers (the action selection and node transition probabilities). Because Dec-POMDPs are cooperative problems and we assume the trajectories are known, we can learn the controllers in a centralized manner, but ensure that they can be executed in a decentralized way.

We also extended this method to learn in the macro-action case [Liu *et al.*, 2016]. This approach learns using high-level macro-action trajectories (macro-actions and macro-observations). Macro-action data is easier to obtain since it is less detailed and using macro-action data is more scalable as the low-level actions and observations may be numerous or continuous and the macro-action policies may be very complicated. On the contrary, the sets of macro-actions and high-level observations will often be discrete and the resulting policy will be much less complex.

For the macro-action case, let the trajectories be $\{(\vec{m}_0^k r_0^k \vec{\omega}_1^k \vec{m}_1^k r_1^k \cdots \vec{\omega}_{T_k}^k \vec{m}_{T_k}^k \ r_{T_k}^k)\}_{k=1,\cdots,K}$, where each m is a macro-action and $\omega \in \Omega^m \cup \omega^\emptyset$ is drawn from a set of augmented observations (at the macro-action level) which includes an observation that the macro-action has not terminated at the given time step, $\omega^\emptyset$. We developed methods that learn macro-action controllers for each agent (i.e., finite-state controllers that depend only on the macro-action-level information) from this macro-action data.

We developed EM algorithms for both the primitive and macro-action cases that are linear in the number of agents and at most square in the problem size, making them scalable to

[1] Domain videos can be seen at https://youtu.be/fGUHTHH-JNA, https://youtu.be/yloUp55DQ0c, https://youtu.be/34xHxXrnPHw, and https://youtu.be/XXYSAmdHn38.

large domains. Our experiments showed that the methods can also produce very high-quality solutions, even outperforming and improving upon hand-coded 'expert' solutions.

5.2 Multi-Robot Learning

We also extended the offline macro-action-based learning method above [Liu *et al.*, 2016] to the multi-robot case [Liu *et al.*, 2017]. Specifically, previous EM-based methods (including ours) suffer from local optimality and sensitivity to initial conditions. Therefore, we developed an iterative sampling-based EM method that is able to efficiently escape from local optima to generate higher-quality solutions. The resulting method still learns policies represented as finite-state controllers given only high-level macro-action-based trajectories, but can significantly outperform previous approaches.

We demonstrated our methods on two variants of multi-robot search and rescue domains (with and without obstacles) using both ground-based and aerial vehicles. The results showed that high-quality policies could be learned using only a small number of trajectories. The resulting policies coordinate the team of distributed robots in a partially observable stochastic environment. This approach fits well with multi-robot problems, as it may be expensive to generate many trajectories (demonstrations) with multiple complex robots in realistic domains.[2]

5.3 Deep Multi-Agent Reinforcement Learning

Deep reinforcement learning (RL) has been extremely successful in single agent (and some multi-agent) domains (e.g., [Mnih *et al.*, 2015; Silver *et al.*, 2016]), but methods typically assume the state is fully observable (even when it it is not, as in Atari) and only one agent is controlled. Recently, we [Omidshafiei *et al.*, 2017c] (and others such as [Foerster *et al.*, 2016; 2017; Mordatch and Abbeel, 2017]) have begun extending these ideas to the Dec-POMDP case. In general, these approaches use deep RL methods with recurrent architectures for learning some relevant history information.

In our case, we developed online decentralized learning methods for Dec-POMDPs [Omidshafiei *et al.*, 2017c]. Online learning requires decentralized learning, since the agents may all be learning at the same time. Decentralized learning is difficult because the problem becomes nonstationary (i.e., it changes over time due to the changing behavior of the other agents). We recently showed that deep RL methods could be extended to perform decentralized learning in Dec-POMDPs by introducing a decentralized extension of experience replay [Mnih *et al.*, 2015] for sample-efficient and stable multi-agent learning and incorporated hysteresis [Matignon *et al.*, 2007] for dealing with nonstationarity. The paper also extended the deep RL methods to the multi-task case, where solutions must generalize to a range of different tasks. Our methods were the first multi-task learning methods for Dec-POMDPs and we showed that we could learn high-quality solutions in a number of large domains.

[2]The domain video can be seen at https://youtu.be/B3b60VqWMIE.

6 Discussion and Conclusion

The methods that have been discussed have shown a lot of promise, but there are other methods that are also promising (e.g., [Dibangoye *et al.*, 2016; Claes *et al.*, 2017; Nguyen *et al.*, 2017]) and many open questions yet to solve. It is worth noting that the methods in this paper focused on the 'full' Dec-POMDP problem, where there was uncertainty about outcomes, sensing and communication and agents are dependent on all others, but some problems do not have all of these characteristics. In these cases, ideas from Dec-POMDP models and methods could still be used, but structure may be able to be exploited to allow higher-quality, more efficient solution methods. Some such structure has been explored [Oliehoek and Amato, 2016], but efficiently exploiting structure in other cases could lead to scalable methods that also consider uncertainty (e.g., probabilistic solutions to common multi-robot problems).

Traditionally, scalability of Dec-POMDPs solution methods has been an issue, but more recent methods (such as those discussed in this paper) can scale to large domains. One approach that is used to overcome this lack of scalability is deep reinforcement learning. Developing deep RL approaches for Dec-POMDP-based models is becoming a very active field (e.g., [Foerster *et al.*, 2016; 2017; Mordatch and Abbeel, 2017; Omidshafiei *et al.*, 2017c; Rashid *et al.*, 2018]) and the resulting methods can handle large state and observation spaces (and potentially large action spaces). These methods (including ours) require a lot of data and typically use a simulator, so they may fit better in situations akin to the offline sample-based planning scenarios described earlier.

Therefore, there has been a great deal of success in scaling planning and offline learning methods to large domains, but efficient online learning remains a challenge. As mentioned above, online learning in a Dec-POMDP must be decentralized (since fast and free communication is not available to centralize decision-making), leading to nonstationarity, which must (continue to) be tackled. Besides more effectively conquering nonstationarity in learning, other open research topics include developing planning and learning methods with additional efficiency and scalability (in terms of the number of agents as well as action and observation spaces) and how to properly represent and encode history information for planning and learning. Overall, there has been great progress in solving large, realistic Dec-POMDPs. It will be exciting to see what further developments and applications will arise in the future.

Acknowledgments

I thank all of my collaborators for making this work possible. This work was partially supported by NSF award #1463945 and Air Force Contract #FA8721-05-C-0002.

References

[Amato *et al.*, 2013] Christopher Amato, Girish Chowdhary, Alborz Geramifard, Nazim Kemal Ure, and Mykel J. Kochenderfer. Decentralized control of partially observable Markov decision processes. In *CDC*, 2013.

[Amato *et al.*, 2014] Christopher Amato, George D. Konidaris, and Leslie P. Kaelbling. Planning with macro-actions in decentralized POMDPs. In *AAMAS*, 2014.

[Amato *et al.*, 2015a] Christopher Amato, George D. Konidaris, Ariel Anders, Gabriel Cruz, Jonathan P. How, and Leslie P. Kaelbling. Policy search for multi-robot coordination under uncertainty. In *RSS*, 2015.

[Amato *et al.*, 2015b] Christopher Amato, George D. Konidaris, Gabriel Cruz, Christopher A. Maynor, Jonathan P. How, and Leslie P. Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *ICRA*, 2015.

[Amato *et al.*, 2017] Christopher Amato, George D. Konidaris, Ariel Anders, Gabriel Cruz, Jonathan P. How, and Leslie P. Kaelbling. Policy search for multi-robot coordination under uncertainty. *The International Journal of Robotics Research*, 2017.

[Bernstein *et al.*, 2002] Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4), 2002.

[Burgard *et al.*, 2005] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3), 2005.

[Claes *et al.*, 2017] Daniel Claes, Frans Oliehoek, Hendrik Baier, and Karl Tuyls. Decentralised online planning for multi-robot warehouse commissioning. In *AAMAS*, 2017.

[Dibangoye *et al.*, 2016] Jilles S. Dibangoye, Christopher Amato, Olivier Buffet, and François Charpillet. Optimally solving Dec-POMDPs as continuous-state MDPs. *Journal of Artificial Intelligence Research*, 55, 2016.

[Foerster *et al.*, 2016] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate to solve riddles with deep distributed recurrent Q-networks. In *NIPS*, 2016.

[Foerster *et al.*, 2017] Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Philip Torr, Pushmeet Kohli, and Shimon Whiteson. Stabilising experience replay for deep multi-agent reinforcement learning. In *ICML*, 2017.

[Kaelbling *et al.*, 1998] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

[Korsah *et al.*, 2013] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12), 2013.

[Liu *et al.*, 2015] Miao Liu, Christopher Amato, Xuejun Liao, Lawrence Carin, and Jonathan P. How. Stick-breaking policy learning in Dec-POMDPs. In *IJCAI*, 2015.

[Liu *et al.*, 2016] Miao Liu, Christopher Amato, Emily Anesta, J. Daniel Griffith, and Jonathan P. How. Learning for decentralized control of multiagent systems in large partially observable stochastic environments. In *AAAI*, 2016.

[Liu *et al.*, 2017] Miao Liu, Kavinayan Sivakumar, Shayegan Omidshafiei, Christopher Amato, and Jonathan P. How. Learning for multi-robot cooperation in partially observable stochastic environments with macro-actions. In *IROS*, 2017.

[Matignon *et al.*, 2007] Laëtitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Hysteretic Q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *IROS*, 2007.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540), 2015.

[Mordatch and Abbeel, 2017] Igor Mordatch and Pieter Abbeel. Emergence of grounded compositional language in multi-agent populations. *arXiv preprint 1703.04908*, 2017.

[Nguyen *et al.*, 2017] Duc Thien Nguyen, Akshat Kumar, and Hoong Chuin Lau. Policy gradient with value function approximation for collective multiagent planning. In *NIPS*. 2017.

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

[Oliehoek, 2012] Frans A. Oliehoek. Decentralized POMDPs. In Marco Wiering and Martijn van Otterlo, editors, *Reinforcement Learning: State of the Art*, volume 12 of *Adaptation, Learning, and Optimization*. Springer Berlin Heidelberg, 2012.

[Omidshafiei *et al.*, 2015] Shayegan Omidshafiei, Ali-akbar Aghamohammadi, Christopher Amato, and Jonathan P. How. Decentralized control of partially observable Markov decision processes using belief space macro-actions. In *ICRA*, 2015.

[Omidshafiei *et al.*, 2016] Shayegan Omidshafiei, Ali-akbar Aghamohammadi, Christopher Amato, Shih-Yuan Liu, Jonathan P. How, and John Vian. Graph-based cross entropy method for solving multi-robot decentralized POMDPs. In *ICRA*, 2016.

[Omidshafiei *et al.*, 2017a] Shayegan Omidshafiei, Ali-akbar Agha-mohammadi, Christopher Amato, and Jonathan P. How. Decentralized control of multi-robot partially observable markov decision processes using belief space macro-actions. *The International Journal of Robotics Research*, 2017.

[Omidshafiei *et al.*, 2017b] Shayegan Omidshafiei, Shih-Yuan Liu, Michael Everett, Brett Lopez, Christopher Amato, Miao Liu, Jonathan P. How, and John Vian. Semantic-level decentralized multi-robot decision-making using probabilistic macro-observations. In *ICRA*, 2017.

[Omidshafiei *et al.*, 2017c] Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P. How, and John Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *ICML*, 2017.

[Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schröder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. *arXiv preprint 1803.11485*, 2018.

[Ren and Sorensen, 2008] Wei Ren and Nathan Sorensen. Distributed coordination architecture for multi-robot formation control. *Robotics and Autonomous Systems*, 56(4), 2008.

[Rubenstein *et al.*, 2014] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198), 2014.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 2016.

[Sutton *et al.*, 1999] Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1), 1999.