Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Informed Initial Policies for Learning in Dec-POMDPs

Landon Kraemer and **Bikramjit Banerjee**
118 College Dr. #5106
Hattiesburg, MS 39406-0001
landon.kraemer@eagles.usm.edu, bikramjit.banerjee@usm.edu
http://orca.st.usm.edu/%7emkraemer/aaai2012SA.html

Introduction

Decentralized partially observable Markov decision processes (Dec-POMDPs) offer a formal model for planning in cooperative multi-agent systems where agents operate with noisy sensors and actuators and local information. While many techniques have been developed for solving Dec-POMDPs exactly and approximately, they have been primarily centralized and reliant on knowledge of the model parameters. In real world scenarios, the model may not be known a priori, and a centralized computation method may not be feasible or desirable.

Simple distributed reinforcement learning, particularly Q-learning (Sutton and Barto 1998), can address both of the above limitations. Q-learning agents can learn mappings from their own action-observation histories (H_i) to their own actions (A_i), via a quality function ($Q_i : H_i \times A_i \mapsto \mathbb{R}$) that evaluates the long-term effects of selecting an action after observing a history.

We investigate two approaches to applying Q-learning to Dec-POMDPs: one in which agents learn concurrently (Q-Conc), and one in which agents take turns to learn the best responses to each other's policies (Q-Alt). In both Q-Conc and Q-Alt, an agent that is learning follows a dynamic policy (i.e. a policy that changes as the agent learns); however, in Q-Alt, each agent that is not currently learning follows a static policy. Thus, in Q-Alt an agent that has not yet learned a policy needs an initial policy to follow. There are simple ways to choose an arbitrary initial policy in constant time, e.g., choosing a pure policy at random, but such methods do not consider the transition, observation, and reward structure of the Dec-POMDP.

As the main contribution of this paper, we propose a simple and principled approach to building an initial joint policy that is based upon the transition and reward (but not the observation) functions of a Dec-POMDP. To lay the groundwork, we first discuss how to *compute* this initial joint policy in a centralized, model-based fashion. We then discuss how agents can *learn* such policies in a distributed, model-free manner, and then demonstrate for two benchmark problems that Q-Alt initialized with this *informed policy* (QIP-Alt) produces better joint policies than Q-Conc and Q-Alt initialized with uniform stochastic policies (QSto-Alt) and Q-Alt initialized with randomly-chosen pure policies (QRand-Alt).

Informed Initial Policies

The high-level idea is to map a Dec-POMDP problem into a constrained POMDP problem, solve that POMDP problem, and then map the resulting optimal POMDP policy back into the space of Dec-POMDP joint policies.

To ground the intuition on existing literature, consider the Q_{POMDP} heuristic used in (Szer, Charpillet, and Zilberstein 2005) which finds an upper-bound for a Dec-POMDP by mapping the problem into a POMDP where the action set is the set of all joint actions and the observation set is the set of all joint observations. In a policy found by Q_{POMDP}, an agent may have different actions associated with the same action-observation history, disambiguated by others' action-observation histories; however, agents do not have access to the observations of other agents during Dec-POMDP execution. Thus, the resulting POMDP policy cannot be (and was not intended to be) mapped to a Dec-POMDP policy in a meaningful way.

In this backdrop, we note that if there were only one possible observation that each agent could receive at each step, this ambiguity would disappear, and Q_{POMDP} policies could then be mapped into Dec-POMDP policies. Unfortunately, most Dec-POMDPs of interest will have more than one observation; however, if the agent policies *ignore* observations, then the same effect can be achieved.

Our proposed method is simple. Create and solve a POMDP where the action set is the set of all joint actions and the observation set is $\{\gamma\}$ (a dummy observation). Since γ is the only possible observation, the optimal POMDP policy can be thought of as a chain of joint actions $\pi_{opt} = \vec{a}^1, \vec{a}^2, \ldots, \vec{a}^T$, where T is the horizon. The informed initial Dec-POMDP policy can then be constructed for each agent i by setting all level t action nodes to a_i^t (agent i's contribution to a^t), i.e. agent i will always execute a_i^t at level t, regardless of what it has previously observed. Figure 1 depicts this process for $T = 2$.

Learning Initial Policies

If this initial policy is to be supplied to a distributed, model-free learning algorithm, a distributed, model-free algorithm

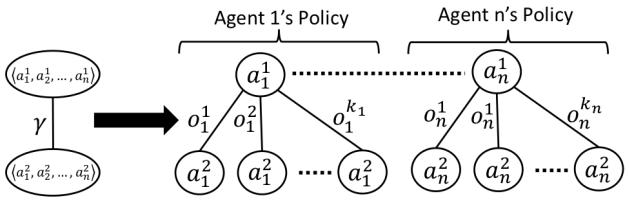

Figure 1: Conversion of the optimal policy of the constrained POMDP to a valid Dec-POMDP policy for T=2.

for learning the proposed initial policy is required. It is generally assumed in Dec-POMDPs that the agents cannot observe each others' actions, so without extra assumptions, the agents must learn independently. One simple approach would be concurrent Q-learning, where each agent i tries to learn $Q'_i : H_i^{A_i} \times A_i \mapsto \mathbb{R}$ for all individual action-only histories $h_i^{A_i} \in H_i^{A_i}$ and individual actions $a \in A_i$.

Suppose though that each agent can announce the random seed it will use for exploration before learning begins. Since rewards are shared and observations are ignored, this would allow agents to simulate each other's action selections and maintain every other agent's Q'-values. Thus, each agent can learn the joint Q'-Value function ($Q' : H^A \times A \mapsto \mathbb{R}$) directly, where A is the set of *joint* actions and H^A is the set of joint action-only histories. This eliminates miscoordination during initial policy learning and can improve the initial policy learned.

Experimental Results

In this section, we evaluate the performance of the four different Q-learning settings: QIP-Alt, QSto-Alt, QRand-Alt, and Q-Conc for the DecTiger and BroadcastChannel benchmark problems. For each domain, we chose eleven increasing values $k_1, \ldots, k_{11}$ and allowed each setting to learn for $k = k_1, \ldots, k_{11}$ episodes. The k values were in the ranges $\left[10^{4.4}, 10^{7.48}\right]$ and $\left[10^{5.34}, 10^{8.84}\right]$ for DecTiger and BroadcastChannel, respectively. The k values were rounded to integers, and derived from experiments with another algorithm (reported elsewhere). k represents the *total* number of episodes agents are allowed to execute. In Q-Conc both agents learn simultaneously for k episodes, in QSto-Alt and QRand-Alt each agent learns exclusively for $\frac{k}{n}$ episodes (where n is the number of agents), and in QIP-Alt each agent learns for $\hat{k} + \frac{k-\hat{k}}{n}$ episodes (where $\hat{k}$ is the number of episodes spent concurrently learning the initial policy). Note that since QIP-Alt must allocate $\hat{k}$ episodes for learning the initial policy, QIP-Alt is at a relative disadvantage if the initial policy is not useful because all other settings have $\hat{k}$ extra steps for learning. We used $\hat{k} = 20000$ for DecTiger and $\hat{k} = \max(200000, \frac{k}{20})$ for BroadcastChannel. In all runs, we used a learning rate $\alpha = .001$ and epsilon-greedy exploration with $\epsilon = .05$ (Sutton and Barto 1998).

Table 1 gives the relative error (based on known optimal values), averaged over 50 runs for each setting, domain, and horizon for the largest values of k investigated (i.e. k_{11}). The worst runtime (for $10^{8.84}$ and $T = 5$) was 16948 secs. From the table, we can see that QIP-Alt dominates all other settings at $k = k_{11}$ for both domains. Particularly of note is QIP-Alt's performance in DecTiger for horizons T=3,4, where QIP-Alt's error is zero and its the nearest competitor (Q-Conc) has error $\geq .15$. While, the results for BroadcastChannel are not nearly as striking, we note that the error produced by QIP-Alt is still almost an order of magnitude less than that of its nearest competitor (QSto-Alt).

Table 2 gives us an upper bound on the number of episodes required for QIP-Alt to dominate the other three settings. Such an upper bound represents the smallest value of k for which QIP-Alt dominates and beyond which it continues to dominate. We can see that QIP-Alt dominates all other settings well before k_{11} in both domains.

Method	DecTiger			BroadcastChannel		
	T=3	T=4	T=5	T=3	T=4	T=5
QIP-Alt	0	0	0.0294	0.00421	0.0042	0.0038
Q-Conc	0.1571	0.3561	0.3586	0.0615	0.0366	0.0303
QSto-Alt	1.0539	1.3203	1.2011	0.0322	0.0257	0.0231
QRand-Alt	5.8633	7.9897	5.8213	0.1074	0.1114	0.1538

Table 1: Average relative errors for all horizons and all settings found at $k = 10^{7.48}$ for DecTiger and $k = 10^{8.84}$ for BroadcastChannel.

Method	DecTiger			BroadcastChannel		
	T=3	T=4	T=5	T=3	T=4	T=5
Q-Conc	$\mathbf{10^{4.4}}$	$10^{4.6}$	$10^{6.3}$	$10^{5.73}$	$10^{6.27}$	$10^{6.00}$
QSto-Alt	$\mathbf{10^{4.4}}$	$\mathbf{10^{4.4}}$	$\mathbf{10^{4.4}}$	$10^{5.73}$	$10^{5.46}$	$\mathbf{10^{5.34}}$
QRand-Alt	$\mathbf{10^{4.4}}$	$\mathbf{10^{4.4}}$	$\mathbf{10^{4.4}}$	$10^{5.73}$	$10^{5.73}$	$\mathbf{10^{5.34}}$

Table 2: Upper bound on the number of episodes needed for QIP-Alt to dominate each setting. Bolded values are the smallest values for k used in our trials (i.e. k_1).

Conclusion

We have presented a simple, principled technique to compute a valid Dec-POMDP policy for use as an initial policy in conjunction with reinforcement learning. Furthermore, we have demonstrated how to learn such policies in a model-free manner, and we have shown for two benchmark problems that using these initial policies can improve the outcome of alternating Q-learning. This result is encouraging and suggests that this initial policy may be useful in other algorithms (both existing and future) that might require an initial policy.

Acknowledgment: This work was supported in part by the US Army under grant #W911NF-11-1-0124.

References

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. Cambridge, MA: The MIT Press.

Szer, D.; Charpillet, F.; and Zilberstein, S. 2005. MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, 576–583.