Sampling-based policy graph improvement in decentralized POMDPs for information gathering

Mikko Lauri[1], Joni Pajarinen[2,3], Jan Peters[2]

I. Introduction

In this paper, we consider an information gathering task with a centralized planning stage, followed by a decentralized execution phase where no implicit communication between the robots is assumed. Such tasks are naturally modelled in the general framework of a decentralized partially observable Markov decision process (Dec-POMDP) [1]. This differs from other state-of-the-art works such as [2], who create decentrally executable plans online with an anytime algorithm that assumes each robot in turn plans its individual policy, then communicates it to the next robot which again plans its individual policy.

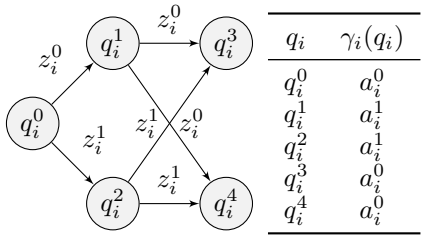
In our earlier work [3], [4], we introduced a variant of Dec-POMDPs which allows information gathering tasks by defining a reward function that depends on the joint belief state of the robots. This allows use of reward functions such as Shannon's information entropy directly in the general Dec-POMDP framework. Here, we propose an extension of the policy graph improvement (PGI) algorithm [4] based on an approximate particle-based evaluation of policy values. The main advantage of our proposed extension is that it allows solving tasks with continuous state, even when a parametric representation of belief states is not available. New applications can be tackled where continuous state spaces commonly appear, such as in robotics.

II. Dec-POMDPs for information gathering

We define the Dec-POMDP problem we consider, and desribe the policy graph improvement algorithm of [4].

A. Dec-POMDP for information gathering

We consider a finite-horizon Dec-POMDP $\big(I,\ S,\ \{A_i\},\ \{Z_i\},\ P^s,\ P^z,\ b^0,\ T,\ \{\rho_t\}\big)$, where $I = \{1, \dots, n\}$ is the set of agents, S is the finite or uncountable 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 $a^t = (a_1^t, \dots, a_n^t) \in A$, 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 $z^{t+1} = (z_1^{t+1}, \dots z_n^{t+1}) \in Z$ 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[1] 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 $t = 0, \dots, T-1$, while $\rho_T : \Delta(S) \to \mathbb{R}$ determines a final reward obtained at the end of the problem horizon.



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

Fig. 1. A policy graph for agent i encodes the agent's behavior conditional on its local observations z_i^t. The set of nodes Q_i is shown shaded in gray. 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 local observation, the next node is q_i^1 or q_i^2. Subsequent actions are again looked up from γ_i.

A solution of a Dec-POMDP is a joint policy π that describes the behaviour of all agents. The joint policy is a combination of decentralized local policies π_i, i.e., $\pi = (\pi_i)_{i \in I}$. Each local policy π_i can be viewed as a finite state machine which we represent as a directed acyclic *policy graph*. A policy graph of agent i consists of a set of nodes Q_i, a starting node q_i^0, an output function $\gamma_i : Q_i \to A_i$ and a node transition function $\lambda_i : Q_i \times Z_i \to Q_i$; see Fig. 1.

The objective in a Dec-POMDP is to find a sequence $(\pi_i)_{i \in I}$ of local policies such that the expected sum of rewards over the finite horizon T is maximized when agents act according to these policies. The belief states b^t can be tracked by Bayesian filtering given the histories of local actions and observations of all agents. Belief states can be used during the centralize planning stage, although based only on their local information available during execution the agents cannot estimate the belief. Rewards that are non-linear in the belief state are useful for information gathering, as they can model quantities such as information entropy.

We finally note that as argued in [4], we can view the joint policy π as a larger graph composed of each local policy graph π_i. We shall denote a joint policy graph by $(Q, q^0, \gamma, \lambda)$, where the tuple elements are defined analogous to the local policy graphs. The nodes $q \in Q$ are now tuples

[1]Department of Informatics, University of Hamburg, Hamburg, Germany lauri@informatik.uni-hamburg.de
[2]IAS lab, TU Darmstadt, Darmstadt, Germany {pajarinen,peters}@ias.tu-darmstadt.de
[3]Tampere University, Tampere, Finland

[1]$\Delta(S)$ denotes the space of probability mass functions over S.

of nodes from the individual policy graphs, e.g., $q = (q_1, q_2)$. Further, γ and λ are built from individual γ_i and λ_i so that they agree with the transitions defined local policy graphs.

B. Policy graph improvement

Policy graph improvement consists of a forward phase and a backward phase, which are repeated one after the other. In the forward phase, given a joint policy, the algorithm first propagates belief states, i.e., probability mass functions over the hidden underlying state of the system, through the joint policy graph. In the backward phase, we loop over the nodes q_i in each agents' policy graph starting from the end, and update the local policy of the node, i.e., the action $\gamma_i(q_i)$ and the next node $\lambda_i(q_i, z_i)$, to maximize the expected sum of rewards from the node until the end of the horizon.

III. Sampling-based policy graph improvement

We outline the proposed approach to using a sampling-based evaluation in the forward and backward phases of PGI.

A. Forward pass

The PGI algorithm in [4] applies the forward pass using exact belief updates, and is thus only applicable for discrete beliefs or parametric beliefs with a closed form update rule. Here we propose to replace the exact belief udpates by approximate particle based updates.

We define an abstract state x^t to represent the forward pass using a given policy π, and derive a particle approximation for $P(x^t \mid \pi)$ for $t = 0, 1, \ldots, T$. In particular, we define $x^t = (s^t, q^t, r^t)$, where $s^t \in S$ is the true underlying hidden state, q^t is the current policy graph node, and r^t is the current cumulative reward obtained.

Consider first the initial time step $t = 0$. To get a particle approximation of $P(x^0 \mid \pi)$, we draw N samples $s^{0,k} \sim b^0$ from the initial belief state. As the initial node q^0 is fixed, and we start with zero reward, $q^{0,k} = q^0$ and $r^{0,k} = 0$ for $k = 1, \ldots, N$. Our particle approximation is $\{x^{0,k}, w^k\}_{k=1}^N$, where $x^{0,k} = (s^{0,i}, q^{0,k}, r^{0,k})$ and $w^k = 1/N$.

To propagate the particle approximation to time step $t \geq 1$, we apply standard particle filtering methods such as sequential importance resampling (SIR) [5]. The procedure is given in Algorithm 1. We use the known Dec-POMDP state transition and observation models to draw samples of the next state and policy node. To obtain a sample of the next reward, we must apply a Monte Carlo approximation since the reward function is dependent on the belief. To estimate the reward $r^{t+1,k}$, we apply the current particle approximation P^t to find the conditional marginal pdf $P(s^t \mid q^t = q^{t,k}, \pi)$. First, we find the subset of P^t that only contains the particles that have $q^{t,k}$ as their current node. Suppose there are N_q such particles, and let us denote them by $\hat{P}^t = \{\hat{s}^{t,k}, \hat{w}^k\}_{k=1}^{N_q}$. We then approximate $\rho_t(b^t, a^t) \approx \rho_t(\hat{P}^t, a^t)$, i.e., we use the particle approximation instead of the exact belief b^t. For rewards based on Shannon entropy, in the case of a finite state space this approximation is easily obtained by directly evaluating the entropy of the distribution defined by $\hat{P}^t$, or by alternative techniques such as [6]. In case of a uncountable state space, techniques such as kernel density estimation may be applied. The final reward ρ_T is approximated similarly.

Algorithm 1 Particle filter for forward pass of PGI

Input: Particles $P^t = \{s^{t,k}, q^{t,k}, r^{t,k}, w^i\}_{k=1}^N$
Output: $P^{t+1} = \{s^{t+1,k}, q^{t+1,k}, r^{t+1,k}, w^k\}_{k=1}^N$
  for $k = 1, \ldots, N$ **do**
    $a^t \leftarrow \gamma(q^{t,k})$ ▷ get action
    $s^{t+1,k} \sim P^s(s^{t+1} \mid s^{t,k}, a^t)$ ▷ sample state
    $\hat{z}^{t+1} \sim P^z(z^{t+1} \mid s^{t+1,k}, a^t)$ ▷ sample observation
    $q^{t+1,k} \leftarrow \lambda(q^{t,k}, \hat{z}^{t+1})$ ▷ get next node
    $r^{t+1,k} \leftarrow r^{t,k} + \textsc{EstimateReward}(P^t, a^t)$
    $w^k \leftarrow w^k \cdot P^z(\hat{z}^{t+1} \mid s^{t+1,k}, a^t)$
  end for
  normalize weights, resample if needed
  return $\{s^{t+1,k}, q^{t+1,k}, r^{t+1,k}, w^k\}_{k=1}^N$

B. Backward pass

For each agent i and each of their policy graph nodes q_i, the backward pass optimizes the policy parameters $\gamma_i(q_i)$ and $\lambda_i(q_i, \cdot)$ of the node to find policies with a greater expected utility. For node q_i the backward pass solves the problem

$$\max_{\substack{a_i^t \in A_i \\ \forall z_i \in Z_i : q_i^{z_i} \in Q_i}} \mathbb{E}_{q_{-i}} \left[\rho_t(b, a) + \mathbb{E}_z \left[V^\pi \left(\zeta(b, a, z), q^{t+1}(z) \right) \right] \right]$$

where the outer expectation is under $q_{-i}^t \sim P(q_{-i}^t \mid q_i^t, \pi)$, the distribution of nodes *other* than those of i, $a = (a_i, a_{-i})$ and $z = (z_{-i}, z_i)$ and the inner expectation is under the prior probability of perceiving an observation z, and $V^\pi(b, q)$ is the expected reward for policy π starting from belief b at node $q = (q_{-i}, q_i)$, and $\zeta(b, a, z)$ refers to the posterior belief after a Bayesian update of belief b given joint action a and joint observation z. To update the local policy, $\gamma_i(q_i)$ and $\lambda_i(q_i, \cdot)$ are assigned to the respective maximizing values.

Given a particle estimate $\{x^{t,k}, w^k\}_{i=1}^N$ of $P(x^t \mid \pi)$, we can easily estimate $P(q_{-i}^t \mid q_i^t, \pi)$ by marginalization. A particle approximation for the belief state b at any node is obtained by using the same strategy as in the previous subsection. Starting from this particle approximation, we estimate the expected value of a local policy by propagating it forward using Algorithm 1, and recording the expected cumulative reward. We obtain an approximation of the objective function above, and can assign the local policy configuration to the approximately optimal values.

IV. Conclusion

We outlined a sampling-based approach to modify the PGI algorithm for information gathering Dec-POMDPs [4] to be applicable to problems with an uncountable state space. Future work will implement and empirically evaluate the proposed method. Sampling-based evaluation can potentially be extended to the optimization in the backward pass, e.g., by maintaining a distribution over policy parameters with a high expected reward similar to [7].

REFERENCES

[1] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Springer, 2016.

[2] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime Planning for Decentralized Multirobot Active Information Gathering," *IEEE RA-L*, vol. 3, no. 2, pp. 1025–1032, 2018.

[3] M. Lauri, E. Heinänen, and S. Frintrop, "Multi-Robot Active Information Gathering with Periodic Communication," in *ICRA*, 2017, pp. 851–856.

[4] M. Lauri, J. Pajarinen, and J. Peters, "Information Gathering in Decentralized POMDPs by Policy Graph Improvement," in *AAMAS*, 2019, pp. 1143–1151.

[5] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[6] P. Valiant and G. Valiant, "Estimating the unseen: improved estimators for entropy and other properties," in *NIPS*, 2013, pp. 2157–2165.

[7] S. Omidshafiei, A.-A. Agha-Mohammadi, C. Amato, S.-Y. Liu, J. P. How, and J. Vian, "Graph-based cross entropy method for solving multi-robot decentralized POMDPs," in *ICRA*, 2016, pp. 5395–5402.