Reinforcement Learning for Mean-field Teams

Jayakumar Subramanian  
McGill University  
Montreal, Quebec, Canada  
jayakumar.subramanian@mail.mcgill.ca

Raihan Seraj  
McGill University  
Montreal, Quebec, Canada  
raihan.seraj@mail.mcgill.ca

Aditya Mahajan  
McGill University  
Montreal, Quebec, Canada  
aditya.mahajan@mcgill.ca

ABSTRACT

We develop reinforcement learning (RL) algorithms for a class of multi-agent systems called mean-field teams (MFT). Teams are multi-agent systems where agents have a common goal and receive a common reward at each time step. The team objective is to maximize the expected cumulative discounted reward over an infinite horizon. MFTs are teams with homogeneous, anonymous agents such that the agents are coupled in their dynamics and rewards through the mean-field (i.e., empirical distribution of the agents’ state). In our work, we consider MFTs with a mean-field sharing information structure, i.e., each agent knows its local state and the empirical mean-field at each time step. We obtain a dynamic programming (DP) decomposition for MFTs using a decomposition approach from literature called the common information approach, which splits the decision making process into a centralized coordination rule that yields prescriptions to be followed by each agent based on their local information. We develop an RL approach for MFTs under the assumption of parametrized prescriptions. We consider the parameters as actions and use conventional RL algorithms to solve the DP. We illustrate the use of these algorithms through two examples based on stylized models of the demand response problem in smart grids and malware spread in networks.

KEYWORDS

Reinforcement learning; mean-field teams; multi-agent reinforcement learning

1 INTRODUCTION

In this paper, we look at reinforcement learning in cooperative multi-agent systems. Several algorithms for multi-agent reinforcement learning have been proposed in the literature [2–4, 10–13, 21–24]. These algorithms perform well on certain benchmark domains but there is little theoretical analysis on whether these algorithms converge to a team optimal solution.

In this paper, we present a different view on multi-agent reinforcement learning. Our central thesis is that multi-agent systems for which the team optimal planning solution can be obtained by dynamic programming [14–16], it should be straightforward to translate these dynamic programs to reinforcement learning algorithms.

2 MODEL

Consider a multi-agent team with $n$ agents, indexed by the set $N = \{1, \ldots, n\}$. The team operates in discrete time for an infinite horizon. Let $X_i^t \in X$ and $U_i^t \in U$ denote the state and action of agent $i \in N$ at time $t$. Note that the state space $X$ and action space $U$ are the same for all agents. For ease of exposition, we assume that $X$ and $U$ are finite sets.

Given a vector $x = (x^1, \ldots, x^n) \in X^n$ of length $n$, let $\xi(x)$ denote the mean-field (or empirical distribution) of $x$, i.e.,

$$\xi(x) = \frac{1}{n} \sum_{i \in N} \delta_{x^i}.$$ 

Let $Z_t = \xi(X_t)$ denote the mean-field of the team at time $t$ and $Z$ denote the space of space of realizations of $Z_t$. Note that $Z$ has at most $(n + 1)^{|X|}$ elements.

Let $((X_{t|}, U_{t|})\geq0, (U_{t|})\geq0)$ denote a realization of $((X_{t|})\geq0, (U_{t|})\geq0)$ and $z_t = \xi(x_t)$. We assume that the initial states of all agents are independent, i.e.,

$$P_X(x_0) = \prod_{i \in N} P_X^i(x_{0^i}) =: \prod_{i \in N} P_0(x_{0^i}),$$

where $P_0$ denotes the initial state distribution of agents. We assume that the global state of the system evolves in a controlled Markovian manner, i.e.,

$$P(X_{t+1} = x_{t+1} | X_{t}, U_{t} = u_{t}) = P(X_{t+1} = x_{t+1} | X_{t} = x_{t}, U_{t} = u_{t}).$$

All agents are partially exchangeable, so the state evolution of a generic agent depends on the states and actions of other agents only through the mean-fields of the states, i.e., for agent $i$:

$$P(X_{t+1} = x_{t+1} | X_t = x_t, U_t = u_t) = \prod_{i \in N} P(X_{t+1}^i = x_{t+1}^i, U_{t} = u_{t}, Z_t = z_t)$$

$$=: \prod_{i \in N} P(x_{t+1}^i | x_{t}^i, u_{t}, z_t),$$

where $P$ denotes the control transition matrix. Combining all of the above, we have

$$P(X_{t+1} = x_{t+1} | X_0, U_0 = u_0) = \prod_{i \in N} P(x_{t+1}^i | x_{t}^i, u_{t}, z_t).$$

(1)

The system has mean-field sharing information-structure, i.e., the information available to agent $i$ is given by:

$$I_t^i = (X_t^i, Z_t).$$

(2)

We assume that all agents use identical (stochastic) control law: $\mu_t : X \times Z \rightarrow \Delta(U)$ to choose the control action at time $t$, i.e.,

$$U_t^i \sim \mu_t(X_t^i, Z_t).$$

(3)

Let $\mu = (\mu_1, \mu_2, \ldots)$ denote the team policy for all times. Note that, in general, restricting attention to identical policies may lead to a loss of optimality. See [1] for an example. Nonetheless, identical policies are attractive for reasons of fairness, simplicity, and robustness.
The team receives a per-step reward given by:
\[ R_t \sim r(X_t, U_t). \]  
(4)
Given strategy \( \mu = (\mu_1, \mu_2, \ldots) \) the expected total reward incurred by the team is given by:
\[ J(\mu) = \mathbb{E}^\mu \left[ \sum_{t=0}^\infty \gamma^t R_t \right]. \]  
(5)
where \( \gamma \in (0,1) \) is the discount factor. The objective is to choose a policy \( \mu \) to maximize the performance \( J(\mu) \) given by (5).

3 SOLUTION APPROACH

The mean-field team model formulated above is a multi-agent team problem with non classical information structure. A planning solution of this model was presented in [1], which we summarize below for completeness. We then present a framework for using reinforcement learning in such models.

3.1 Planning solution for mean-field teams

Given any policy \( \mu = (\mu_1, \mu_2, \ldots) \) and any realization, \( z = (z_1, z_2, \ldots) \) of the mean-field, define prescriptions \( h_t: X \rightarrow \Delta(A) \) given by
\[ h_t(x) = \mu_t(x, z_t) \quad \forall x \in X. \]

Let \( \mathcal{H} \) denote the space of all such prescritions. When the mean field trajectory is a random process, the prescriptions \( h_t(x) \) is a random vector which we denote \( H_t \). The results of [1] relies on the following two key properties. Let \( (z_{t+1}, h_{t+1}) \) denote any realization of \( (Z_{t+1}, H_{t+1}) \). We have:

1. \( |Z_t|_{t \geq 1} \) is a controlled Markov process with control action \( h_t \), i.e.,
\[ \mathbb{P}(Z_{t+1} = z_{t+1} | Z_t = z_t, H_t = h_t) = \mathbb{P}(Z_{t+1} = z_{t+1} | Z_t = z_t, H_t = h_t). \]

Note that the right hand side does not depend on the choice of decision rule \( \mu \). Furthermore, the right hand side can be simplified as:
\[ \mathbb{P}(Z_{t+1} = z_{t+1} | Z_t = z_t, H_t = h_t) = \sum_{x_{t+1} \xi(x_{t+1}) = z_{t+1}} \prod_{i \in \mathbb{N}} P(x_{t+1}^i | x_t^i, h_t(x_t^i), z_t). \]

where \( x_t \) is any state such that \( \xi(x_t) = z_t \).

2. The expected per-step reward simplifies as follows.
\[ \mathbb{E}[r(X_t, U_t)|Z_t, H_t] = \mathbb{E}[r(X_t, U_t)|Z_t, H_{t+1}] = r(Z_t, H_t). \]  
(6)
It is shown in [1] that these two properties imply that the optimal policy \( \mu \) can be identified as follows.

Theorem 3.1. Let \( V: \mathcal{Z} \rightarrow \mathbb{R} \) be the unique bounded fixed point of the following equation:
\[ V(z) := \max_{h \in \mathcal{H}} \mathbb{E}[\tilde{r}(z, h) + \gamma V(Z_{t+1})|Z_t = z, H_t = h]. \]  
(7)
Let \( \psi(z) \) be an arg max of the right hand side of (7). Then the policy,
\[ \mu(x, z) = \psi(z)(x), \]  
(8)
is an optimal policy for Problem (5).

The action space \( \mathcal{H} \) of the above dynamic program is all functions from \( X \) to \( \Delta(\mathcal{U}) \). We assume that \( \mathcal{H} \) is approximated by some family of parametrized functions \( \mathcal{H}_\Phi = \{ h_\phi \}_{\phi \in \Phi} \) (where \( \Phi \) is a compact and convex set) such as Gibbs/Boltzmann functions or neural networks. With such a parametrization, the dynamic program of (7) may be approximated as:
\[ V(z) = \max_{\phi \in \Phi} \mathbb{E}[\tilde{r}(z, h_\phi) + \gamma V(Z_{t+1}) | Z_t = z, H_t = h_\phi] \]  
(9)
Let \( \hat{\psi}(z) \) be an arg max of the right hand side of (9). Then the policy,
\[ \mu(x, z) = h_{\hat{\psi}(z)}(x), \]  
(10)
is the best policy for Problem (5) when \( \mu_t(z_t) \) is restricted to belong to \( \mathcal{H}_\Phi \).

3.2 Reinforcement learning for mean-field teams (MFT-RL)

In this section, we present a reinforcement learning algorithm for the special case where the reward is a cumulative reward, i.e.,
\[ R_t = \frac{1}{n} \sum_{i \in \mathbb{N}} R_{t}^i, \]  
(11)
where \( R_{t}^i \sim r(X_{t}^i, U_{t}^i, Z_{i}). \) We assume that we have access to a simulator for \( P(\cdot | x_{t}^i, u_{t}^i, z_{t}) \) and \( \tilde{r}(x_{t}^i, u_{t}^i, z_{t}) \). This simulator is for a generic agent and takes the current local state, current local action and current mean-field as input and generates a sample of the local next state and the total reward as output. Using \( n \) copies of this simulator, we create a simulator for the mean-field dynamics. We start with \( n \) agents with initial local state sampled according to \( P_0 \). We assume that all these agents use a common stochastic policy \( \hat{\psi} : \mathcal{Z} \rightarrow \Phi \) to generate prescription parameters \( \phi_t \sim \hat{\psi}(z_t) \).

Given this sampled value of \( \phi_t \), each agent independently samples a control action \( u_{t}^i \sim h_{\phi_t}(x_{t}^i) \). The actions \( u_{t}^i \) of agent \( i \) and the current mean-field \( z_t \) are given as input to the \( i \)th simulator and the sampled output \( (X_{t+1}^i, R_{t}^i) \) are averaged to obtain \( (Z_{t+1}, R_t) \). Thus, we have a simulator with internal state \( z_t \). This simulator takes \( \phi_t \) as an input and gives \( (Z_{t+1}, R_t) \) as sampled next-mean-field state and reward. Thus, this is a simulator for \( P(z_{t+1} | z_t, h_{\phi_t}) \) and \( \tilde{r}(z_t, h_{\phi_t}) \). We can use this simulator with any standard RL algorithm to find the optimal policy for the dynamic program (9). In our experiments below, we use TRPO [17], PPO [18] and NAIFDQN [5].

4 NUMERICAL EXPERIMENTS

4.1 Benchmark domains
We consider the following domains to illustrate different decentralized reinforcement learning algorithms.

4.1.1 Demand response in smart grids. This is a stylized model for demand response in smart grids [1]. The system consists of \( n \) agents, where \( X = \{0, 1, \} \), \( U = \{0, 1, \} \),
\[ P(\cdot | 0, z) = M \]  
(12)
\[ P(\cdot | 0, z) = (1 - \epsilon_1) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \epsilon_1 M \]  
(13)
\[ P(\cdot | 1, z) = (1 - \epsilon_2) \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + \epsilon_2 M, \]  
(14)
where \( M \) denotes the “natural” dynamics of the systems and \( \epsilon_1 \) and \( \epsilon_2 \) are small positive constants.
The per-step reward is given by:

\[ R_t = -\frac{1}{n} \sum_{i \in N} \left( c_0 \mathbb{1}_{\{U_t^i = 0\}} + c_1 \mathbb{1}_{\{U_t^i = 1\}} \right) + KL(Z_t, \xi), \quad (15) \]

where \( c_0 \) and \( c_1 \) are costs for taking actions 0 and 1 respectively, \( \xi \) is a given target distribution and \( KL(Z_t, \xi) \) denotes the Kullback-Leibler divergence between \( Z_t \) and \( \xi \).

In our experiments, we consider we consider a system with \( n = 100 \) agents, initial state distribution \( P_0 = [1/3, 2/3] \), \( M = [0.25, 0.75, 0.35, 0.625] \), \( c_0 = 0.1 \), \( c_1 = 0.2 \), \( \zeta = [0.7, 0.3] \), \( \epsilon_1 = \epsilon_2 = 0.2 \) and discount factor \( \gamma = 0.9 \).

4.1.2 Malware spread in networks. This is a stylized model for malware spread in networks [6–8]. The system consists of \( n \) agents where \( X = [0, 1], U = [0, 1] \). The dynamics are given by:

\[ X_{t+1}^{i} = \begin{cases} X_t^{i} + (1 - X_t^{i}) \omega_t, & \text{for } U_t = 0, \\ X_t^{i}, & \text{for } U_t = 1, \end{cases} \]

where \( \omega_t \sim \text{Uniform}[0, 1] \). The per-step reward is given by:

\[ R_t = -\frac{1}{n} \sum_{i \in N} \left( k + \langle Z_t \rangle \right) X_t^{i} + \lambda U_t^{i}, \]

where \( \langle Z_t \rangle \) denotes the average of \( Z_t \), and \( \lambda \) is the cost of taking action 1.

In our experiments, we consider \( k = 0.2 \), initial state distribution \( P_0 = \text{Uniform}(X) \), \( \lambda = 0.5 \) and discount factor \( \gamma = 0.9 \). For the simulation, we discretize the state space into 11 bins—0, 0.1, . . . , 1.

4.2 Simulation results

We consider three variants of MFT-RL algorithms, which use different RL algorithms for the mean-field system—TRPO, PPO and NADEDQN. Figure 1 shows the result for the demand response domain and Figure 2 shows the result for the malware spread domain. For each of the MFT-RL algorithms, the dark line shows the median performance and the shaded region shows the region between the first and third quartiles across multiple independent runs. For the demand response domain we also show the optimal performance obtained using the value iteration algorithm presented in [1].
5 CONCLUSION
There are many results in the Dec-POMDP/decentralized control literature where a team optimal solution can be obtained using dynamic programming. Our central thesis is that for such models one can easily translate the dynamic program to a reinforcement learning algorithm. We illustrate this point by using mean-field teams as an example. This allows us to use standard off-the-shelf RL algorithms to obtain solutions for some MARL setups.

REFERENCES