Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents

Kaiqing Zhang [1] **Zhuoran Yang** [2] **Han Liu** [3] **Tong Zhang** [4] **Tamer Başar** [1]

Abstract

We consider the *fully decentralized* multi-agent reinforcement learning (MARL) problem, where the agents are connected via a time-varying and possibly sparse communication network. Specifically, we assume that the reward functions of the agents might correspond to different tasks, and are only known to the corresponding agent. Moreover, each agent makes individual decisions based on both the information observed locally and the messages received from its neighbors over the network. To maximize the globally averaged return over the network, we propose two fully decentralized actor-critic algorithms, which are applicable to large-scale MARL problems in an online fashion. Convergence guarantees are provided when the value functions are approximated within the class of linear functions. Our work appears to be the first theoretical study of fully decentralized MARL algorithms for networked agents that use function approximation.

1. Introduction

In reinforcement learning (Sutton & Barto, 1998), the agent aims to behave optimally in the presence of uncertainty by interacting with the environment, which is usually modeled as a Markov Decision Process (MDP). In this work, we study the problem of multi-agent reinforcement learning (MARL), where a common environment is influenced by the joint actions of multiple agents. Specifically, at each state, each agent takes an action, and these actions together determine the next state of the environment and the reward of each agent. In addition, the agents are allowed to observe only its own reward, which may vary for different agents. We are interested in the collaborative setting, where the agents have a common goal of jointly maximizing the globally averaged return of all agents in the environment.

For collaborative MARL problems, it is pivotal to specify the protocol of collaboration among the agents. One tempting choice is to have a central controller which receives the rewards of all agents, and determines the action for each agent. With information of all the agents available to the controller, the problem reduces to an MDP and can be readily solved by single-agent reinforcement learning algorithms. However, in many real-world scenarios, a central controller simply does not exist or may be costly to install. Moreover, the central controller needs to communicate with each agent to exchange the information, which incessantly increases the communication overhead at the single controller. This may degrade the *scalability* of the multi-agent system as well as its *robustness* to malicious attacks.

In contrast, we consider a decentralized protocol where the agents are connected by a possibly time-varying and sparse communication network, which serves as the channel for the agents to exchange information in absence of any central controller. At each time step, each agent executes an individual action based on both the local information and the message sent from its neighbors, with the joint goal of maximizing the average rewards of all agents over the network. We refer to this protocol as *fully decentralized* MARL with *networked agents*. By saying *fully* decentralized, we primarily emphasize that: i) no central controller exists; ii) the information available to each agent is local; iii) the control execution is local. Moreover, the networked architecture finds broad applications in practical multi-agent systems, such as unmanned vehicles (Fax & Murray, 2004), robotics (Corke et al., 2005), power grid (Dall'Anese et al., 2013), and mobile sensor networks (Cortes et al., 2004).

With only local reward and action, classical reinforcement learning algorithms can hardly maximize the networked-wide averaged return determined by the joint actions of all agents. Thus, we propose two decentralized actor-critic algorithms based on a novel policy gradient theorem for MARL. Specifically, the actor step is performed individually by each

[1]Department of Electrical and Computer Engineering & Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, USA [2]Department of Operations Research and Financial Engineering, Princeton University, USA [3]Department of Electrical Engineering and Computer Science and Statistics, Northwestern University, USA [4]Tencent AI Lab, China. Correspondence to: Kaiqing Zhang <kzhang66@illinois.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

agent, without the need to infer the policies of others. For the critic step, on the other hand, each agent shares its estimate of the value function with its neighbors on the network, so that a consensual estimate is achieved, which is further used in the subsequent actor step. In this regard, the local information at each agent diffuses across the network to help achieve the network-wide optimality. Our algorithms enjoy the advantages of scalability to large population of agents, robustness against malicious attacks, and communication efficiency, as standard distributed/decentralized algorithms over networked multi-agent systems (Nedic & Ozdaglar, 2009; Agarwal & Duchi, 2011; Chen & Sayed, 2012).

Furthermore, one long-standing challenge in RL is the huge capacity of the state-action spaces. This problem becomes more pronounced in MARL, since the number of joint actions grows exponentially with the number of agents in the system. To address this challenge, we apply function approximation to both the policy and value functions. Therefore, our algorithms can be readily applied to large-scale MARL problems where both the number of states and the number of agents are massive. More importantly, we show that our algorithms converge when linear function approximation is used, which provides theoretical support for the proposed fully decentralized MARL framework.

Main Contribution. Our contribution is three-fold. First, we formulate the fully decentralized MARL problem with networked agents, and prove a policy gradient theorem adapted to this setting. Second, we propose two decentralized actor-critic algorithms with function approximation, which are applicable to large-scale MARL problems. Third, when linear function approximation is applied, we establish convergence guarantees for the proposed algorithms. It appears that our work provides the first fully decentralized MARL algorithms, with provable convergence guarantees.

Related Work. Our algorithms belong to the class of actor-critic algorithms. There is rich literature on single-agent actor-critic algorithms, which are based on the policy gradient theorem (Sutton et al., 2000). The first actor-critic algorithm is proposed in Sutton et al. (2000); Konda & Tsitsiklis (2000), which also study the convergence of the algorithm with linear function approximation. Based on natural gradient descent, Peters & Schaal (2008); Bhatnagar et al. (2009) study the natural actor-critic algorithm. Convergence of natural actor-critic algorithms with linear function approximation are further studied in Bhatnagar et al. (2008; 2009). Recently, with deep neural networks as function approximators, various actor-critic algorithms have been proposed. See Lillicrap et al. (2016); Wang et al. (2016); Gruslys et al. (2017); Gu et al. (2017) and the references therein for details. A more related and popular work is Mnih et al. (2016), which proposes an asynchronous actor-critic algorithm, A3C. Unlike our MARL framework, however, the A3C algorithm essentially deals with single-agent RL but with multiple parallel workers/processors, where no interaction occurs among the workers. Moreover, a central controller is needed to coordinate the asynchronous update of the workers. In contrast, our actor-critic algorithms require no central controller in implementation.

A more relevant line of research is on MARL. Most existing work is based on the framework of Markov games, which is first studied in Littman (1994; 2001); Lauer & Riedmiller (2000); Hu & Wellman (2003). This general framework applies to the setting with both collaborative and competitive agents. However, these early algorithms are developed only for tabular cases, which are not tractable with large numbers of states and actions. Moreover, for the collaborative setting in this framework, notably team games, Wang & Sandholm (2003); Arslan & Yüksel (2017) take the reward of all agents to be identical, which greatly simplifies the problem since the value function can be estimated locally with no need of information exchange among agents. More recently, several MARL algorithms using deep neural networks have gained increasing attention. See Foerster et al. (2016); Gupta et al. (2017); Lowe et al. (2017); Omidshafiei et al. (2017) for details. However, most of them focus on showing empirical results, without convergence guarantees. Moreover, none of them exploit the communication among agents, since they assume either a central controller exists to coordination the agents (Foerster et al., 2016; Lowe et al., 2017), or the reward of all agents are common (Gupta et al., 2017; Omidshafiei et al., 2017). Thus, these algorithms do not apply to our fully decentralized framework directly. We defer a detailed comparison with existing MARL models and algorithms to §F in the appendix.

Notation. We denote the cardinality of a finite set $\mathcal{A}$ by $|\mathcal{A}|$. For simplicity, we use $\lim_t$, $\sup_t$, and $\sum_t$ to represent $\lim_{t\to\infty}$, $\sup_{t\to\infty}$, and $\sum_{t\geq 0}$, respectively. We denote by I and $\mathbb{1}$ the identity matrix and all-one vector of proper dimensions, respectively. We also use $[N]$ to denote the set of integers $\{1, \cdots, N\}$ and $\mathbb{N}$ to denote the set of non-negative integers.

2. Background

In this section, we introduce the background and formulation of the MARL problem with networked agents.

2.1. Networked Multi-Agent MDP

Consider a system of N agents operating in a common environment, denoted by $\mathcal{N} = [N]$. We focus on the decentralized setting where no central controller exists in the system that either collects rewards or makes the decisions for the agents. Instead, the agents are located on a possibly time-varying communication network, which is characterized by an undirected graph $\mathcal{G}_t = (\mathcal{N}, \mathcal{E}_t)$, where $\mathcal{E}_t$ is the

set of communication links connecting the agents at time $t \in \mathbb{N}$. We then define the following multi-agent Markov decision process with networked agents.

Definition 2.1 (Networked Multi-Agent MDP)**.** A networked multi-agent MDP is characterized by a tuple $(\mathcal{S}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, P, \{R^i\}_{i\in\mathcal{N}}, \{\mathcal{G}_t\}_{t\geq 0})$ where $\mathcal{S}$ is the global state space shared by all the agents in $\mathcal{N}$, $\mathcal{A}^i$ is the action space of agent i, and $\{\mathcal{G}_t\}_{t\geq 0}$ is a time-varying communication network. In addition, let $\mathcal{A} = \prod_{i=1}^N \mathcal{A}^i$ be the joint action space of all agents. Then, $R^i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the local reward function of agent i, and $P : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to [0, 1]$ is the state transition probability of the MDP. Moreover, we assume that the states are globally observable whereas the rewards are observed only locally.

At time t, each agent i chooses its own action a_t^i given state s_t, following a local policy $\pi^i : \mathcal{S} \times \mathcal{A}^i \to [0, 1]$, where $\pi^i(s, a^i)$ represents the probability of choosing action a^i at state s. Thus, the joint policy of all agents $\pi\colon \mathcal{S} \times \mathcal{A} \to [0, 1]$ satisfies $\pi(s, a) = \prod_{i\in\mathcal{N}} \pi^i(s, a^i)$. We note that our model is *fully decentralized* since both the reward is received locally and the action is executed individually by each agent.

For any agent i, we assume that the local policy is parameterized by $\pi^i_{\theta^i}$, where $\theta^i \in \Theta^i$ is the parameter, and $\Theta^i \subseteq \mathbb{R}^{m_i}$ is a compact set. We pack the parameters together as $\theta = [(\theta^1)^\top, \cdots, (\theta^N)^\top]^\top \in \Theta$, where $\Theta = \prod_{i=1}^N \Theta^i$. The joint policy is thus given by $\pi_\theta(s, a) = \prod_{i\in\mathcal{N}} \pi^i_{\theta^i}(s, a_i)$. We make a standard regularity assumption on the networked MDP and the policy function.

Assumption 2.2. We assume that for any $i \in \mathcal{N}$, $s \in \mathcal{S}$, and $a^i \in \mathcal{A}^i$, the policy function $\pi^i_{\theta^i}(s, a^i) > 0$ for any $\theta^i \in \Theta^i$. Also, $\pi^i_{\theta^i}(s, a^i)$ is continuously differentiable with respect to the parameter θ^i over Θ^i. In addition, for any $\theta \in \Theta$, let P^θ be the transition matrix of the Markov chain $\{s_t\}_{t\geq 0}$ induced by policy π_θ, that is, for any $s, s' \in \mathcal{S}$

$$P^\theta(s' \,|\, s) = \sum_{a\in\mathcal{A}} \pi_\theta(s, a) \cdot P(s' \,|\, s, a). \tag{2.1}$$

We assume that the Markov chain $\{s_t\}_{t\geq 0}$ is irreducible and aperiodic under any π_θ, with the stationary distribution denoted by d_θ.

Assumption 2.2 is standard in the work on actor-critic (AC) algorithms with function approximation (Konda & Tsitsiklis, 2000; Bhatnagar et al., 2009). It implies that the Markov chain of the state-action pair $\{(s_t, a_t)\}_{t\geq 0}$ has a stationary distribution $d_\theta(s) \cdot \pi_\theta(s, a)$ for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$.

In addition, the collective objective of the agents is to collaboratively find a policy π_θ that maximizes the *globally* averaged long-term return over the network based solely on *local* information. Let r^i_{t+1} denote the reward received by agent i at time t. Then the goal of all agents is as follows

$$\begin{aligned}\max_\theta \; J(\theta) &= \lim_T \frac{1}{T}\mathbb{E}\bigg(\sum_{t=0}^{T-1}\frac{1}{N}\sum_{i\in\mathcal{N}} r^i_{t+1}\bigg)\\ &= \sum_{s\in\mathcal{S},a\in\mathcal{A}} d_\theta(s)\pi_\theta(s, a) \cdot \overline{R}(s, a),\end{aligned} \tag{2.2}$$

where $\overline{R}(s, a) = N^{-1} \cdot \sum_{i\in\mathcal{N}} R^i(s, a)$ is the globally averaged reward function. Let $\overline{r}_t = N^{-1} \cdot \sum_{i\in\mathcal{N}} r^i_t$; then, we have $\overline{R}(s, a) = \mathbb{E}[\overline{r}_{t+1} \,|\, s_t = s, a_t = a]$. Thus, the global relative action-value function under policy π_θ becomes

$$Q_\theta(s, a) = \sum_t \mathbb{E}\big[\overline{r}_{t+1} - J(\theta) \,|\, s_0 = s, a_0 = a, \pi_\theta\big], \tag{2.3}$$

and the global relative state-value function $V_\theta(s)$ is defined as $V_\theta(s) = \sum_{a\in\mathcal{A}} \pi_\theta(s, a)Q_\theta(s, a)$. For simplicity, we will hereafter refer to V_θ and Q_θ as *state-value* function and *action-value* function only. Furthermore, the *advantage function* can be defined as $A_\theta(s, a) = Q_\theta(s, a) - V_\theta(s)$.

This architecture of multi-agent systems with networked agents finds broad applications in distributed cooperative control problems, including formation control of unmanned vehicles (Fax & Murray, 2004), cooperative navigation of robots (Corke et al., 2005), and load management in energy networks (Dall'Anese et al., 2013), etc. We extend these existing models to an explicit MDP model, where the collective objective is beyond reaching a stable state (Fax & Murray, 2004), or solving a static distributed optimization problem (Dall'Anese et al., 2013). Our framework also extends the existing framework on collaborative MARL, where the agents either are coordinated by a central controller (Gupta et al., 2017), or share a common team reward (Wang & Sandholm, 2003). See a detailed comparison of our model to the existing ones on multi-agent systems and collaborative MARL in §F in the appendix.

3. Actor-Critic with Networked Agents

In this section, we propose two actor-critic algorithms for the multi-agent RL with networked agents. We first establish a policy gradient theorem for MARL, which characterizes the gradient of $J(\theta)$ defined in (2.2) in closed form.

Theorem 3.1 (Policy Gradient Theorem for MARL)**.** For any $\theta \in \Theta$, let π_θ be a policy and $J(\theta)$ be the globally averaged return defined in (2.2). Let Q_θ and A_θ be the corresponding action-value function and advantage function, respectively. Moreover, for any $i \in \mathcal{N}$, we define the local advantage function $A^i_\theta\colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as

$$A^i_\theta(s, a) = Q_\theta(s, a) - \widetilde{V}^i_\theta(s, a^{-i}), \tag{3.1}$$

where $\widetilde{V}^i_\theta(s, a^{-i}) = \sum_{a^i\in\mathcal{A}^i} \pi^i_{\theta^i}(s, a^i)\cdot Q_\theta(s, a^i, a^{-i})$. We denote by a^{-i} the actions of all agents except for i. Then the gradient of $J(\theta)$ with respect to θ^i is given by

$$\begin{aligned}\nabla_{\theta^i} J(\theta) &= \mathbb{E}_{s\sim d_\theta, a\sim\pi_\theta}\left[\nabla_{\theta^i} \log \pi^i_{\theta^i}(s, a^i) \cdot A_\theta(s, a)\right]\\ &= \mathbb{E}_{s\sim d_\theta, a\sim\pi_\theta}\left[\nabla_{\theta^i} \log \pi^i_{\theta^i}(s, a^i) \cdot A^i_\theta(s, a)\right].\end{aligned} \tag{3.2}$$

The proof for Theorem 3.1 is deferred to §B.1 in the appendix. The theorem shows that the policy gradient with respect to each θ^i can be obtained locally using the corresponding score function $\nabla_{\theta^i} \log \pi^i_{\theta^i}$, provided that agent i has an unbiased estimate of the advantage functions A^i_θ or A_θ. However, with only local information, these functions cannot be well estimated since the estimation requires the rewards $\{r^i_t\}_{i\in\mathcal{N}}$ of all agents. This motivates our consensus-based MARL algorithms that leverage the communication network to diffuse the local information, fostering collaboration among agents.

3.1. Algorithms

We first propose an algorithm based on the local advantage function A^i_θ defined in (3.1), which requires estimating the action-value function Q_θ of policy π_θ. To this end, we consider $Q(\cdot,\cdot;\omega)\colon \mathcal{S}\times\mathcal{A}\to\mathbb{R}$, a family of functions parametrized by $\omega\in\mathbb{R}^K$, where $K \ll |\mathcal{S}|\cdot|\mathcal{A}|$. We assume that each agent i maintains its own parameter ω^i and uses $Q(\cdot,\cdot;\omega^i)$ as a local estimate of Q_θ. Moreover, since Q_θ in (2.3) involves the globally averaged reward $\overline{r}_t$, to aggregate the local information, each agent i shares the local parameter ω^i with its neighbors on the network, in order to reach a consensual estimate of Q_θ. In this way, each agent is able to improve the current policy via the policy gradient theorem.

Accordingly, we propose an actor-critic algorithm that consists of two steps, the *actor* step and the *critic* step. In the critic step, an update based on *temporal difference (TD) learning* is performed at each agent to estimate $Q(\cdot,\cdot;\omega^i)$, followed by a linear combination of its neighbor's parameter estimates. Such a parameter sharing step is also known as the *consensus update*, which involves a weight matrix $\mathrm{C}_t = [c_t(i,j)]_{N\times N}$, where $c_t(i,j)$ is the weight on the message transmitted from agent j to agent i at time t. The construction of C_t depends on the topology of $\mathcal{G}_t$; see §4 for details. Thus, the critic step iterates as follows

$$\begin{cases} \mu^i_{t+1} = (1-\beta_{\omega,t})\cdot\mu^i_t + \beta_{\omega,t}\cdot r^i_{t+1}, \\ \widetilde{\omega}^i_t = \omega^i_t + \beta_{\omega,t}\cdot\delta^i_t\cdot\nabla_\omega Q_t(\omega^i_t), \\ \omega^i_{t+1} = \sum_{j\in\mathcal{N}} c_t(i,j)\cdot\widetilde{\omega}^j_t, \end{cases} \tag{3.3}$$

where μ^i_t tracks the long-term return of agent i, $\beta_{\omega,t}>0$ is the stepsize, and $Q_t(\omega)=Q(s_t,a_t;\omega)$ for any ω. Moreover, define the local *action-value TD-error* δ^i_t in (3.3) as

$$\delta^i_t = r^i_{t+1} - \mu^i_t + Q_{t+1}(\omega^i_t) - Q_t(\omega^i_t). \tag{3.4}$$

As for the actor step, motivated by (3.2) in Theorem 3.1, each agent i improves its policy via

$$\theta^i_{t+1} = \theta^i_t + \beta_{\theta,t}\cdot A^i_t\cdot\psi^i_t, \tag{3.5}$$

where $\beta_{\theta,t}>0$ is the stepsize. Moreover, A^i_t and ψ^i_t are defined as

$$A^i_t = Q_t(\omega^i_t) - \sum_{a^i\in\mathcal{A}^i}\pi^i_{\theta^i_t}(s_t,a^i)\cdot Q(s_t,a^i,a^{-i}_t;\omega^i_t),$$
$$\psi^i_t = \nabla_{\theta^i}\log\pi^i_{\theta^i_t}(s_t,a^i_t). \tag{3.6}$$

Note that both A^i_t and ψ^i_t can be computed at each agent i, in a fully decentralized fashion.

The update in (3.3) for ω^i_t resembles the so-termed *diffusion* update in Chen & Sayed (2012) for solving distributed optimization/estimation problems. However, it differs in two main aspects: i) the update direction $\delta^i_t\cdot\nabla_\omega Q_t(\omega^i_t)$ is not the stochastic gradient direction of any well-defined objective function, thus the update is not equivalent to solving any distributed optimization problem; ii) diminishing stepsizes are adopted for possibly almost sure convergence, whereas mean square convergence was established in Chen & Sayed (2012). Thus, the proof techniques there do not apply to the analysis of update (3.3). Moreover, we note that the updates (3.3)-(3.5) preserve the privacy of agents, in the sense that no information about the individual reward function or the policy is required for such network-wide collaboration. This inherits one of the advantages of fully decentralized algorithms. We refer to the steps (3.3)-(3.5) as Algorithm 1, whose pseudocode is provided in §A in the appendix.

For online implementation of Algorithm 1, the joint actions a_{t+1} is needed to evaluate the action-value TD-error δ^i_t. Moreover, since agent i also needs to store the estimates $\omega^i_t\in\mathbb{R}^K$ and $\theta^i_t\in\mathbb{R}^{m_i}$, the total memory complexity of agent i is $\mathcal{O}(N+m_i+K)$. On the contrary, in the tabular case, each agent i need to maintain a Q-table of dimension $|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|$ as in Kar et al. (2013), where $|\mathcal{A}|=\prod_{i\in\mathcal{N}}|A_i|$ grows exponentially with the number of agents N in the system.

Note that Algorithm 1 requires action a_{t+1} to compute the update at time t. In the following, we propose an AC algorithm which only uses the transition at time t, namely, the sample (s_t,a_t,s_{t+1}), based on estimating the global advantage function A_θ. In fact, one can estimate A_θ with the global state-value TD-error, since the latter is an unbiased estimate of the former, i.e., for any $s\in\mathcal{S},a\in\mathcal{A}$

$$\mathbb{E}\big[\overline{\delta}_t \,\big|\, s_t=s, a_t=a, \pi_\theta\big] = A_\theta(s,a), \tag{3.7}$$

where $\overline{\delta}_t = \overline{r}_{t+1} - J(\theta) + V_\theta(s_{t+1}) - V_\theta(s_t)$ is the global state-value TD-error. To this end, we first estimate $J(\theta)$ and V_θ with a scalar μ and a parametrized function $V(\cdot;v)\colon\mathcal{S}\to\mathbb{R}$, respectively, where parameter $v\in\mathbb{R}^L$ and $L\ll|\mathcal{S}|$. Similar to Algorithm 1, each agent i shares its own estimates v^i and μ^i with its neighbors, such that a consensual estimate of the global V_θ is obtained eventually.

Let $V_t(v)=V(s_t;v)$ for any $v\in\mathbb{R}^L$, and $\beta_{v,t}>0$ be the stepsize. Also, with slight abuse of notation, we use

δ_t^i to denote the local *state-value TD-error* of agent i (in contrast to (3.4)). Then the estimates v_t^i and μ_t^i are updated as follows

$$\begin{cases} \widetilde{\mu}_t^i = (1-\beta_{v,t})\cdot\mu_t^i + \beta_{v,t}\cdot r_{t+1}^i, \\ \mu_{t+1}^i = \sum_{j\in\mathcal{N}} c_t(i,j)\cdot\widetilde{\mu}_t^j, \\ \delta_t^i = r_{t+1}^i - \mu_t^i + V_{t+1}(v_t^i) - V_t(v_t^i), \\ \widetilde{v}_t^i = v_t^i + \beta_{v,t}\cdot\delta_t^i\cdot\nabla_v V_t(v_t^i), \\ v_{t+1}^i = \sum_{j\in\mathcal{N}} c_t(i,j)\cdot\widetilde{v}_t^j. \end{cases} \tag{3.8}$$

Note that δ_t^i cannot be used in (3.7) to estimate the global advantage function, since it uses r_{t+1}^i rather than $\overline{r}_{t+1}$ in the update.

Therefore, we propose to estimate the globally averaged reward function $\overline{R}$ in the critic step as well. Let $\overline{R}(\cdot,\cdot;\lambda)\colon \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ be the class of parametrized functions, where $\lambda\in\mathbb{R}^M$ is the parameter with $M\ll|\mathcal{S}|\cdot|\mathcal{A}|$. To obtain the estimate $\overline{R}(\cdot,\cdot;\lambda)$, we seek to minimize the following weighted mean-square error

$$\min_{\lambda}\sum_{s\in\mathcal{S},a\in\mathcal{A}}\widetilde{d}_\theta(s,a)\big[\overline{R}(s,a;\lambda)-\overline{R}(s,a)\big]^2, \tag{3.9}$$

where recall that $\overline{R}(s,a)=\sum_{i\in\mathcal{N}}R^i(s,a)\cdot N^{-1}$ and we let $\widetilde{d}_\theta(s,a)=d_\theta(s)\cdot\pi_\theta(s,a)$. The optimization problem (3.9) can be equivalently characterized as

$$\min_{\lambda}\sum_{i\in\mathcal{N}}\sum_{s\in\mathcal{S},a\in\mathcal{A}}\widetilde{d}_\theta(s,a)\big[\overline{R}(s,a;\lambda)-R^i(s,a)\big]^2. \tag{3.10}$$

Note that the objective (3.10) has the same form with separable objectives over agents as in the distributed optimization literature (Nedic & Ozdaglar, 2009; Boyd et al., 2011; Chen & Sayed, 2012). This motivates the following updates for λ_t^i to minimize the objective (3.10)

$$\begin{cases} \widetilde{\lambda}_t^i = \lambda_t^i + \beta_{v,t}\cdot[r_{t+1}^i - \overline{R}_t(\lambda_t^i)]\cdot\nabla_\lambda\overline{R}_t(\lambda_t^i), \\ \lambda_{t+1}^i = \sum_{j\in\mathcal{N}} c_t(i,j)\cdot\widetilde{\lambda}_t^j, \end{cases} \tag{3.11}$$

where $\overline{R}_t(\lambda)=\overline{R}(s_t,a_t;\lambda)$ for any $\lambda\in\mathbb{R}^M$. The update in (3.11) forms the critic step of the AC algorithm together with (3.8). We note that the rewards of other agents are not transmitted directly to each agent, and the estimate $\overline{R}(\cdot,\cdot;\lambda)$ cannot recover the individual reward function of others. Hence, as in Algorithm 1, the consensual estimate of globally averaged reward function $\overline{R}$ does not harm the privacy of agents on their rewards and policies.

Unlike most existing work in distributed optimization, the samples obtained to estimate the gradient of (3.10) are correlated by the Markov chain $\{(s_t,a_t)\}_{t\geq0}$. We will instead resort to stochastic approximation (SA) to analyze the convergence of this update.

The estimate $\overline{R}(\cdot,\cdot;\lambda_t^i)$ is then used to evaluate the global state-value TD-error $\overline{\delta}_t$. Accordingly, the actor step becomes

$$\begin{cases} \widetilde{\delta}_t^i = \overline{R}_t(\lambda_t^i) - \mu_t^i + V_{t+1}(v_t^i) - V_t(v_t^i), \\ \theta_{t+1}^i = \theta_t^i + \beta_{\theta,t}\cdot\widetilde{\delta}_t^i\cdot\psi_t^i. \end{cases} \tag{3.12}$$

where $\beta_{\theta,t}>0$ is the stepsize and ψ_t^i is as defined in (3.6). We refer to the steps (3.8), (3.11), and (3.12) as Algorithm 2, and provide its pseudocode in §A in the appendix. Similar to Algorithm 1, the online implementation of Algorithm 2 requires the memory complexity of $\mathcal{O}(N+L+M+m_i)$ for each agent i, which results in a great reduction in contrast to the tabular case when N is large. Also note that both algorithms are applicable to general function approximators including neural networks. In addition, when linear function is applied, we provide convergence guarantees in §4.

4. Theoretical Results

In this section, we establish theoretical results for the proposed algorithms. The proofs for the results are deferred to §B in the appendix. We start with the following four assumptions which apply to both algorithms.

Assumption 4.1. The update of the policy parameter θ_t^i includes a local projection operator, $\Gamma^i:\mathbb{R}^{m_i}\to\Theta^i\subset\mathbb{R}^{m_i}$, that projects any θ_t^i onto the compact set Θ^i. Also, we assume that $\Theta=\prod_{i=1}^N\Theta^i$ is large enough to include at least one local minimum of $J(\theta)$.

This projection is a common technique for stabilizing the stochastic approximation algorithms (Kushner & Yin, 2003), and is a standard assumption in many analyses for actor-critic algorithms (Bhatnagar et al., 2009; Degris et al., 2012; Prasad et al., 2014). Note that the projection is merely for analysis purposes and may not be necessary in experiments.

Assumption 4.2. The instantaneous reward r_t^i is uniformly bounded for any $i\in\mathcal{N}$ and $t\geq0$.

We note that the boundedness assumption on the rewards is rather mild and is satisfied by the MDP model with finite state and action spaces. We then make the assumption that the stepsizes $\beta_{\omega,t}$, $\beta_{v,t}$, and $\beta_{\theta,t}$ satisfy the following standard condition, such that the critic step operates much faster than the actor step, creating a two-time-scale algorithm as standard single-agent AC algorithms.

Assumption 4.3. The stepsizes $\beta_{\omega,t}$, $\beta_{v,t}$, and $\beta_{\theta,t}$ satisfy

$$\sum_t\beta_{\omega,t}=\sum_t\beta_{v,t}=\sum_t\beta_{\theta,t}=\infty,$$
$$\sum_t\beta_{\omega,t}^2+\beta_{v,t}^2+\beta_{\theta,t}^2<\infty.$$

In addition, $\beta_{\theta,t}=o(\beta_{\omega,t})=o(\beta_{v,t})$, and $\lim_t\beta_{\omega,t+1}\cdot\beta_{\omega,t}^{-1}=\lim_t\beta_{v,t+1}\cdot\beta_{v,t}^{-1}=1$.

Furthermore, we make the following assumption on the design of the weight matrix $\{\mathrm{C}_t\}$ for the consensus updates in both algorithms.

Assumption 4.4. The sequence of nonnegative random matrices $\{\mathrm{C}_t\}$ satisfies

(a.1) C_t is row stochastic and $\mathbb{E}(\mathrm{C}_t)$ is column stochastic, i.e., $\mathrm{C}_t\mathbb{1} = \mathbb{1}$ and $\mathbb{1}^\top\mathbb{E}(\mathrm{C}_t) = \mathbb{1}^\top$. Furthermore, there exists a constant $\eta \in (0,1)$ such that, for any $c_t(i,j) > 0$, we have $c_t(i,j) \geq \eta$.

(a.2) C_t respects the communication graph $\mathcal{G}_t$, i.e., $c_t(i,j) = 0$ if $(i,j) \notin \mathcal{E}_t$. Moreover, the spectral norm of $\mathbb{E}[\mathrm{C}_t^\top \cdot (\mathrm{I} - \mathbb{1}\mathbb{1}^\top/N) \cdot \mathrm{C}_t]$ is smaller than one.

(a.3) Given the σ-algebra generated by the random variables before time t, C_t is conditionally independent of r_{t+1}^i for any $i \in \mathcal{N}$.

We take the matrix C_t to be random for the sake of generality. The randomness can be attributed to either the randomness of the time-varying graph $\mathcal{G}_t$, e.g., random link failures in sensor networks (Kar & Moura, 2010), or the randomness of the consensus algorithms, e.g., randomized gossip schemes (Boyd et al., 2006). The condition (a.1) is standard in developing consensus algorithms, which guarantees convergence of the update for each agent to a common vector, and the limit $\lim_t \mathrm{C}_t\mathrm{C}_{t-1}\cdots\mathrm{C}_0$ exists (Nedic & Ozdaglar, 2009). Note that row stochasticity constraint $\mathrm{C}_t\mathbb{1} = \mathbb{1}$ is local, since it simply requires each agent to make the weights assigned to the updates coming from its neighbors summing up to one. The condition (a.2) is related to the connectivity of the communication network $\mathcal{G}_t$. It follows from Boyd et al. (2006); Aysal et al. (2009) that for the gossip type of communication schemes, (a.2) holds if and only if the underlying communication graph is connected. The condition (a.3) means that C_t and r_{t+1}^i are independent conditioned on the past. This is common for practical multi-agent systems, since the random communication link failures and the gossip schemes are usually independent of the past and irrelevant to the rewards received by the agents.

One particular choice of the weights in C_t that relies on only local information of the agents is known as Metropolis weights (Xiao et al., 2005)

$$c_t(i,j) = \big\{1 + \max[d_t(i), d_t(j)]\big\}^{-1}, \ \ \forall (i,j) \in \mathcal{E}_t,$$
$$c_t(i,i) = 1 - \sum_{j \in \mathcal{N}_t(i)} c_t(i,j), \ \ \forall i \in \mathcal{N},$$

where $\mathcal{N}_t(i) = \{j \in \mathcal{N} \colon (i,j) \in \mathcal{E}_t\}$ is the set of neighbors of agent i at time t, and $d_t(i) = |\mathcal{N}_t(i)|$ is the degree of agent i. Other common choices of C_t that satisfy the conditions (a.1)-(a.2) in Assumption 4.4 include pairwise gossip (Boyd et al., 2006), broadcast gossip (Aysal et al., 2009), and network dropouts (Bianchi et al., 2013).

4.1. Convergence of Algorithm 1

To show the convergence, we make the following additional assumption on the action-value functions.

Assumption 4.5. For each agent i, the function $Q(s,a;\omega)$ is parametrized as $Q(s,a;\omega) = \omega^\top\phi(s,a)$ where $\phi(s,a) = [\phi_1(s,a), \cdots, \phi_K(s,a)]^\top \in \mathbb{R}^K$ is the feature associated with (s,a). The feature vector $\phi(s,a)$ is uniformly bounded for any $s \in \mathcal{S}, a \in \mathcal{A}$. Furthermore, the feature matrix $\Phi \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times K}$ has full column rank, where the k-th column of Φ is $[\phi_k(s,a), s \in \mathcal{S}, a \in \mathcal{A}]^\top$ for any $k \in [K]$. Also, for any $u \in \mathbb{R}^K$, $\Phi u \neq \mathbb{1}$.

For theoretical analysis, we focus here on the algorithm with linear function approximation. Note that the TD-learning-based policy evaluation may fail to converge with nonlinear function approximation (Tsitsiklis & Van Roy, 1997). To the best of our knowledge, all existing online AC algorithms with theoretical convergence guarantees are built upon linear function approximation, e.g., Konda & Tsitsiklis (2000); Bhatnagar et al. (2009); Bhatnagar (2010). The assumption on the feature matrix Φ is standard and has also been made in Tsitsiklis & Van Roy (1997; 1999); Bhatnagar et al. (2009) to ensure that the critic step converges to a unique asymptotically stable point.

We use the two-time-scale stochastic approximation technique (Borkar, 2008) to analyze the convergence of Algorithm 1. In particular, we first analyze the convergence of the critic step at the faster time scale, assuming the joint policy π_θ is fixed. Then we analyze the convergence of the policy parameter θ_t upon the convergence of the critic step.

For notational simplicity, we define $P^\theta(s', a' \,|\, s, a) = P(s' \,|\, s, a)\pi_\theta(s', a')$[1], $\mathrm{D}_\theta^{s,a} = \mathrm{diag}[d_\theta(s) \cdot \pi_\theta(s,a), s \in \mathcal{S}, a \in \mathcal{A}]$, and $\overline{R} = [\overline{R}(s,a), s \in \mathcal{S}, a \in \mathcal{A}]^\top \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$. We then define the operator $T_\theta^Q : \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ for any action-value vector $Q \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ as

$$T_\theta^Q(Q) = \overline{R} - J(\theta)\cdot\mathbb{1} + P^\theta Q. \tag{4.1}$$

With these notations, we establish the convergence of the critic step (3.3) and (3.4) given policy π_θ as follows.

Theorem 4.6. Under Assumptions 2.2 and 4.2-4.5, for any given policy π_θ, with the sequences $\{\mu_t^i\}$ and $\{\omega_t^i\}$ generated from (3.3) and (3.4), we have $\lim_t \sum_{i\in\mathcal{N}} \mu_t^i \cdot N^{-1} = J(\theta)$ and $\lim_t \omega_t^i = \omega_\theta$ almost surely (a.s.) for any $i \in \mathcal{N}$, where $J(\theta)$ is the globally averaged return as defined in (2.2), and ω_θ is the unique solution to

$$\Phi^\top \mathrm{D}_\theta^{s,a}\big[T_\theta^Q(\Phi\omega_\theta) - \Phi\omega_\theta\big] = 0. \tag{4.2}$$

[1] With slight abuse of notation, the expression P^θ has the same form as the transition probability matrix of the Markov chain $\{s_t\}_{t\geq 0}$ under policy π_θ (see the definition in (2.1)). These two matrices can be easily differentiated by the context.

We note that the solution to (4.2) is the limiting point of the TD(0) algorithm (Tsitsiklis & Van Roy, 1999), except that here we approximate the action-value function Q_θ rather than the state-value function V_θ. This point is also known as the minimizer of the Mean Square Projected Bellman Error (MSPBE) (Dann et al., 2014). Thus, Theorem 4.6 states that each agent is able to obtain a good approximation of the *globally* averaged action-value function, i.e., $\omega_t^i \to \omega_\theta$ for all $i \in \mathcal{N}$, even with merely local rewards and information received from neighbors. This approximation of global value function is then adopted in the actor step to estimate the policy gradient for each agent.

To show convergence of the actor step, we define the quantities $A_{t,\theta}^i$ and $\psi_{t,\theta}^i$ as

$$A_{t,\theta}^i = \phi_t^\top \omega_\theta - \sum_{a^i \in \mathcal{A}^i} \pi_{\theta^i}^i(s_t, a^i) \cdot \phi(s_t, a^i, a_t^{-i})^\top \omega_\theta,$$
$$\psi_{t,\theta}^i = \nabla_{\theta^i} \log \pi_{\theta^i}^i(s_t, a_t^i), \tag{4.3}$$

where we denote $\phi_t = \phi(s_t, a_t)$. Define a vector $\hat{\Gamma}^i(\cdot)$ as

$$\hat{\Gamma}^i[g(\theta)] = \lim_{0<\eta\to 0} \{\Gamma^i[\theta^i + \eta \cdot g(\theta)] - \theta^i\}/\eta \tag{4.4}$$

for any $\theta \in \Theta$ and $g : \Theta \to \mathbb{R}^{\sum_{i\in\mathcal{N}} m_i}$ a continuous function. In case the limit above is not unique, let $\hat{\Gamma}^i[g(\theta)]$ be the set of all possible limit points of (4.4). Then we state the convergence of Algorithm 1 with linear function approximation as follows.

Theorem 4.7. Under Assumptions 2.2 and 4.1-4.5, the sequence $\{\theta_t^i\}$ obtained from (3.5) converges a.s. to a point in the set of the asymptotically stable equilibria of

$$\dot{\theta}^i = \hat{\Gamma}^i\big[\mathbb{E}_{s_t\sim d_\theta, a_t\sim\pi_\theta}\big(A_{t,\theta}^i \cdot \psi_{t,\theta}^i\big)\big], \ \forall\ i \in \mathcal{N}. \tag{4.5}$$

Note that Theorem 4.7 is in the same spirit as the convergence results for single-agent AC algorithms with linear function approximation (Bhatnagar et al., 2009; Bhatnagar, 2010). Since the linear features here are not restricted to *compatible* features as in Tsitsiklis & Van Roy (1997); Sutton et al. (2000), convergence to the stationary point of $\mathbb{E}_{s_t\sim d_\theta, a_t\sim\pi_\theta}\big(A_{t,\theta}^i \cdot \psi_{t,\theta}^i\big) = 0$ in the set Θ^i is the best one can hope for any AC algorithms with general linear function approximators, even for the single-agent setting (Bhatnagar et al., 2009; Degris et al., 2012).

4.2. Convergence of Algorithm 2

Similar to Algorithm 1, we need an additional assumption for the convergence of Algorithm 2.

Assumption 4.8. For each agent i, the function $V(s; v)$ and $\overline{R}(s, a; \lambda)$ are parametrized as $V(s; v) = v^\top \varphi(s)$ and $\overline{R}(s, a; \lambda) = \lambda^\top f(s, a)$, respectively. Here $\varphi(s) = [\varphi_1(s), \cdots, \varphi_K(s)]^\top \in \mathbb{R}^L$ and $f(s, a) = [f_1(s, a), \cdots, f_M(s, a)]^\top \in \mathbb{R}^M$ are the features associated with s and (s, a). The feature vectors $\varphi(s)$ and $f(s, a)$ are uniformly bounded for any $s \in \mathcal{S}, a \in \mathcal{A}$. Furthermore, let the feature matrix $\Phi \in \mathbb{R}^{|\mathcal{S}|\times L}$ have $[\varphi_\ell(s), s \in \mathcal{S}]^\top$ as its ℓ-th column for any $\ell \in [L]$, and the feature matrix $\mathrm{F} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times M}$ have $[f_m(s, a), s \in \mathcal{S}, a \in \mathcal{A}]^\top$ as its m-th column for any $m \in [M]$. Then, both Φ and F have full column rank, and for any $u \in \mathbb{R}^L$, $\Phi u \neq \mathbb{1}$.

Recall $P^\theta(s' \mid s) = \sum_{a\in\mathcal{A}} P(s' \mid s, a)\pi_\theta(s, a)$ defined in (2.1) and let $\mathrm{D}_\theta^s = \mathrm{diag}[d_\theta(s), s \in \mathcal{S}]$. Also let $\overline{R}_\theta = [\overline{R}_\theta(s), s \in \mathcal{S}]^\top \in \mathbb{R}^{|\mathcal{S}|}$ with $\overline{R}_\theta(s) = \sum_a \pi_\theta(s, a)\overline{R}(s, a)$. We thus define the operator $T_\theta^V : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ for any state-value vector $V \in \mathbb{R}^{|\mathcal{S}|}$ as

$$T_\theta^V(V) = \overline{R}_\theta - J(\theta) \cdot \mathbb{1} + P^\theta V. \tag{4.6}$$

We now state the convergence of the critic step (3.8) and (3.11) as follows.

Theorem 4.9. Under Assumptions 2.2, 4.2-4.4, and 4.8, for any given policy π_θ, with sequences $\{\lambda_t^i\}$, $\{\mu_t^i\}$, and $\{v_t^i\}$ generated from (3.8) and (3.11), we have $\lim_t \mu_t^i = J(\theta)$, $\lim_t \lambda_t^i = \lambda_\theta$, and $\lim_t v_t^i = v_\theta$ a.s. for any $i \in \mathcal{N}$, where $J(\theta)$ is the globally averaged return under joint policy π_θ, λ_θ and v_θ are the unique solution to

$$\mathrm{F}^\top \mathrm{D}_\theta^{s,a}\big(\overline{R} - \mathrm{F}\lambda_\theta\big) = 0, \tag{4.7}$$
$$\Phi^\top \mathrm{D}_\theta^s\big[T_\theta^V(\Phi v_\theta) - \Phi v_\theta\big] = 0. \tag{4.8}$$

Similarly, the solution v_θ in (4.8) is exactly the limiting point of the TD(0) algorithm of policy evaluation for state-value functions, as if the rewards of all others are observable to each agent. Moreover, the solution λ_θ in (4.7) corresponds to the unique minimizer of the problem (3.9) under Assumption 4.8. Both v_θ and λ_θ are used to define the TD-error $\widetilde{\delta}_{t,\theta}^i$ upon the convergence of the critic step, notably,

$$\widetilde{\delta}_{t,\theta}^i = f_t^\top \lambda_\theta - J(\theta) + \varphi_{t+1}^\top v_\theta - \varphi_t^\top v_\theta, \tag{4.9}$$

where we define $f_t = f(s_t, a_t)$ and $\varphi_t = \varphi(s_t)$. Recalling that $\psi_{t,\theta}^i = \nabla_{\theta^i} \log \pi_{\theta^i}^i(s_t, a_t^i)$, we have the following theorem on the convergence of Algorithm 2 with linear function approximation.

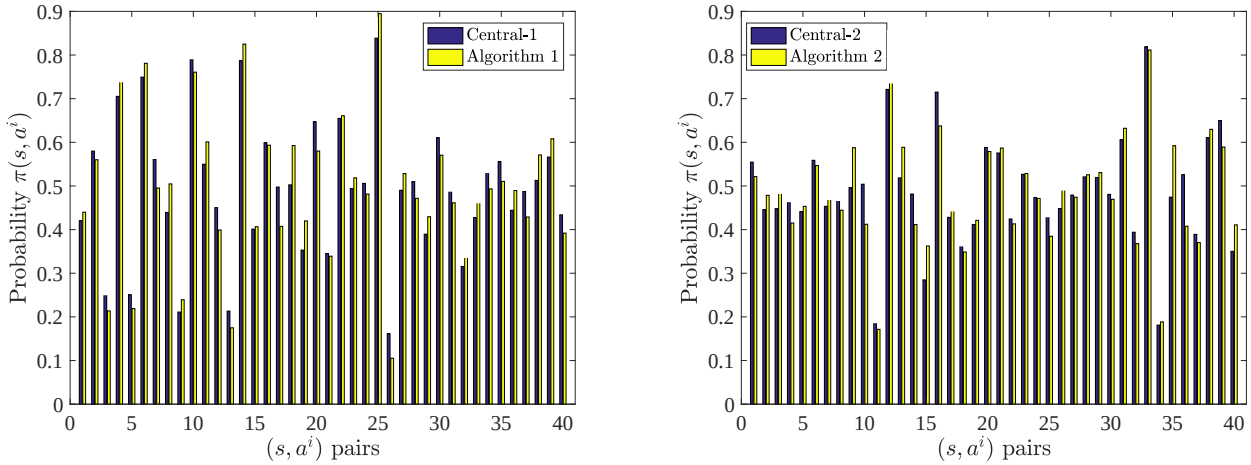
Theorem 4.10. Under Assumptions 2.2, 4.1-4.4, and 4.8, the sequence $\{\theta_t^i\}$ obtained from (3.12) converges a.s. to a point in the set of the asymptotically stable equilibria of

$$\dot{\theta}^i = \hat{\Gamma}^i\big[\mathbb{E}_{s_t\sim d_\theta, a_t\sim\pi_\theta}\big(\widetilde{\delta}_{t,\theta}^i \cdot \psi_{t,\theta}^i\big)\big], \ \forall\ i \in \mathcal{N}. \tag{4.10}$$

Similarly to Theorem 4.7, the convergent point of (4.10) is the best one can hope for, when linear function approximation for both $\overline{R}$ and V_θ is adopted.

5. Numerical Results

We first evaluate our algorithms with linear function approximation. We consider an environment with $|\mathcal{S}| = 20$ states



(a) $s = 2$ (b) $s = 18$

Figure 1. The policies, i.e., the probability distribution $\pi_\theta(s, a)$ at a randomly selected state s. In (a), we plot the policy obtained from Central-1 and Algorithm 1 at state 2. In (b), we plot the policy obtained from Central-2 and Algorithm 2 at state 18.

and $N = 20$ networked agents. The performances of the fully decentralized algorithms are compared with those of the centralized algorithms, in which the rewards r_t^i of all agents are available to a central controller and the joint policy π_θ is also updated there. The two centralized algorithms for comparison with Algorithms 1 and 2 are referred to as Central-1 and Central-2, respectively. Due to space limitation, we defer the details on the model and the algorithms to §E.1 in the appendix.

It is observed that the proposed fully decentralized AC algorithms successfully converge as we proved. In particular, Figure 1 shows that the resulting policies from Algorithms 1 and 2 resemble those achieved by the centralized counterparts. In other words, the joint policy obtained by agents using only local information is almost as good as the policy obtained by the centralized controller with full system information. More details on the desirable performance of the algorithms are also provided in §E.1.

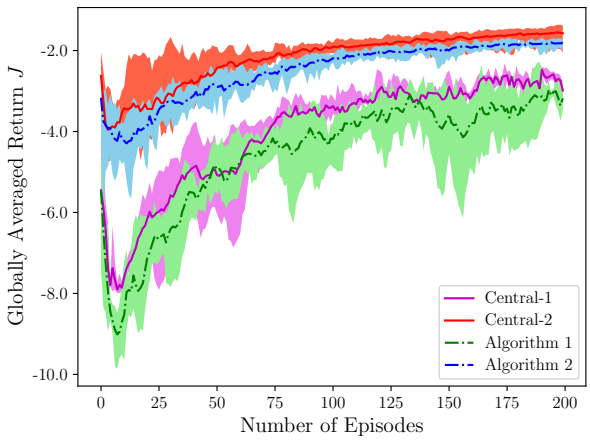


Figure 2. The globally averaged returns for Cooperative Navigation, when neural networks are used for function approximation.

We also empirically evaluate the proposed algorithms with nonlinear function approximators, namely, neural networks. We consider the MARL task of *Cooperative Navigation* from Lowe et al. (2017). To be compatible with our networked multi-agent MDP, we modify the environment and provide the details in §E.2 in the appendix. It is shown in Figure 2 that the proposed algorithms successfully converge even with such nonlinear function approximators, with Algorithm 2 converging to a return relatively higher than that Algorithm 1 converging to. Moreover, the decentralized algorithms can achieve globally averaged return close to the centralized ones, though at a slightly slower speed. This implies the potential of our algorithms to large-scale MARL problems with neural networks as function approximators. We also compare our algorithms with those where each agent performs single-agent RL with local information, with no communications with other agents. These non-cooperative algorithms are shown to be unstable, and achieve much worse return with greater variance. For better illustration, we plot our results in Figure 2 against those of the non-cooperative algorithms on a separate figure that is deferred to §E.2. This shows the necessity of communication among agents in decentralized cooperative MARL.

6. Conclusions

We consider the *fully decentralized* multi-agent reinforcement learning problem with networked agents. In this framework, the agents aim to optimize network-wide averaged return via communication with neighboring agents. We propose two decentralized online actor-critic algorithms with function approximation, which are applicable to large-scale MARL problems with numerous agents and massive state-action spaces. Moreover, we establish convergence results when linear function approximation is used, and provide empirical experiments to validate our theoretical results.

Acknowledgements

The work of K.Z. and T.B. was supported in part by the US Army Research Laboratory (ARL) Cooperative Agreement W911NF-17-2-0196, and in part by the US Army Research Office (ARO) Grant W911NF-16-1-0485. The research of H.L. was supported by NSF CAREER Award DMS1454377, NSF IIS1408910, NSF IIS1332109. This material is based upon work supported by the National Science Foundation under grant no. 1740762 "Collaborative Research: TRIPODS Institute for Optimization and Learning". We would also like to thank all the anonymous reviewers for their helpful suggestions and supportive comments.

References

Agarwal, A. and Duchi, J. C. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pp. 873–881, 2011.

Andrieu, C., Moulines, É., and Priouret, P. Stability of stochastic approximation under verifiable conditions. *SIAM Journal on Control and Optimization*, 44(1):283–312, 2005.

Arslan, G. and Yüksel, S. Decentralized Q-learning for stochastic teams and games. *IEEE Transactions on Automatic Control*, 62 (4):1545–1558, 2017.

Aysal, T. C., Yildiz, M. E., Sarwate, A. D., and Scaglione, A. Broadcast gossip algorithms for consensus. *IEEE Transactions on Signal Processing*, 57(7):2748–2761, 2009.

Benaïm, M. Dynamics of stochastic approximation algorithms. *Séminaire de Probabilités, XXXIII*, 1709:1–68, 1999.

Bhatnagar, S. An actor–critic algorithm with function approximation for discounted cost constrained Markov Decision Processes. *Systems & Control Letters*, 59(12):760–766, 2010.

Bhatnagar, S., Ghavamzadeh, M., Lee, M., and Sutton, R. S. Incremental natural actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pp. 105–112, 2008.

Bhatnagar, S., Sutton, R., Ghavamzadeh, M., and Lee, M. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

Bianchi, P., Fort, G., and Hachem, W. Performance of a distributed stochastic approximation algorithm. *IEEE Transactions on Information Theory*, 59(11):7405–7418, 2013.

Borkar, V. S. *Stochastic approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2008.

Borkar, V. S. and Meyn, S. P. The ODE method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000.

Boutilier, C. Planning, learning and coordination in multi-agent decision processes. In *Conference on Theoretical Aspects of Rationality and Knowledge*, pp. 195–210, 1996.

Boyd, S., Ghosh, A., Prabhakar, B., and Shah, D. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking*, 14 (SI):2508–2530, 2006.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Chen, J. and Sayed, A. H. Diffusion adaptation strategies for distributed optimization and learning over networks. *IEEE Transactions on Signal Processing*, 60(8):4289–4305, 2012.

Corke, P., Peterson, R., and Rus, D. Networked robots: Flying robot navigation using a sensor net. *Robotics Research*, pp. 234–243, 2005.

Cortes, J., Martinez, S., Karatas, T., and Bullo, F. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

Dall'Anese, E., Zhu, H., and Giannakis, G. B. Distributed optimal power flow for smart microgrids. *IEEE Transactions on Smart Grid*, 4(3):1464–1475, 2013.

Dann, C., Neumann, G., Peters, J., et al. Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, 15:809–883, 2014.

Degris, T., White, M., and Sutton, R. Off-policy actor-critic. In *International Conference on Machine Learning*, 2012.

Fax, J. A. and Murray, R. M. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

Foerster, J., Assael, Y. M., de Freitas, N., and Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2137–2145, 2016.

Gruslys, A., Azar, M. G., Bellemare, M. G., and Munos, R. The reactor: A sample-efficient actor-critic architecture. *arXiv preprint arXiv:1704.04651*, 2017.

Gu, S., Lillicrap, T., Ghahramani, Z., Turner, R. E., and Levine, S. Q-prop: Sample-efficient policy gradient with an off-policy critic. In *International Conference on Learning Representations*, 2017.

Guestrin, C., Lagoudakis, M., and Parr, R. Coordinated reinforcement learning. In *International Conference on Machine Learning*, 2002.

Gupta, J. K., Egorov, M., and Kochenderfer, M. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multi-agent Systems*, pp. 66–83, 2017.

Hu, J. and Wellman, M. P. Nash Q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4 (Nov):1039–1069, 2003.

Kar, S. and Moura, J. M. Distributed consensus algorithms in sensor networks: Quantized data and random link failures. *IEEE Transactions on Signal Processing*, 58(3):1383–1400, 2010.

Kar, S., Moura, J. M., and Poor, H. V. QD-learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus + innovations. *IEEE Transactions on Signal Processing*, 61(7):1848–1862, 2013.

Kok, J. R. and Vlassis, N. Collaborative multi-agent reinforcement learning by payoff propagation. *Journal of Machine Learning Research*, 7(Sep):1789–1828, 2006.

Konda, V. R. and Tsitsiklis, J. N. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pp. 1008–1014, 2000.

Kushner, H. J. and Clark, D. S. *Stochastic approximation methods for constrained and unconstrained systems*. Springer Science & Business Media, 1978.

Kushner, H. J. and Yin, G. G. *Stochastic approximation and recursive algorithms and applications*. Springer, New York, NY, 2003.

Lauer, M. and Riedmiller, M. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *International Conference on Machine Learning*, 2000.

Lewis, F. L., Zhang, H., Hengster-Movric, K., and Das, A. *Cooperative control of multi-agent systems: Optimal and adaptive design approaches*. Springer Science & Business Media, 2013.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Littman, M. L. Markov games as a framework for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 157–163, 1994.

Littman, M. L. Value-function reinforcement learning in Markov games. *Cognitive Systems Research*, 2(1):55–66, 2001.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017.

Macua, S. V., Tukiainen, A., Hernández, D. G.-O., Baldazo, D., de Cote, E. M., and Zazo, S. Diff-dac: Distributed actor-critic for multitask deep reinforcement learning. *arXiv preprint arXiv:1710.10363*, 2017.

Mathkar, A. S. and Borkar, V. S. Nonlinear gossip. *SIAM Journal on Control and Optimization*, 54(3):1535–1557, 2016.

Metivier, M. and Priouret, P. Applications of a Kushner and Clark lemma to general classes of stochastic algorithms. *IEEE Transactions on Information Theory*, 30(2):140–151, 1984.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016.

Movric, K. H. and Lewis, F. L. Cooperative optimal control for multi-agent systems on directed graph topologies. *IEEE Transactions on Automatic Control*, 59(3):769–774, 2014.

Nedic, A. and Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

Nedich, A., Olshevsky, A., and Shi, W. Achieving geometric convergence for distributed optimization over time-varying graphs. *arXiv preprint arXiv:1607.03218*, 2016.

Neveu, J. *Discrete-parameter martingales*. Elsevier, 1975.

Omidshafiei, S., Pazis, J., Amato, C., How, J. P., and Vian, J. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *International Conference on Machine Learning*, pp. 2681–2690, 2017.

Peters, J. and Schaal, S. Natural actor-critic. *Neurocomputing*, 71 (7):1180–1190, 2008.

Prasad, H., Prashanth, L., and Bhatnagar, S. Actor-critic algorithms for learning Nash equilibria in n-player general-sum games. *arXiv preprint arXiv:1401.2086*, 2014.

Shoham, Y., Powers, R., and Grenager, T. Multi-agent reinforcement learning: A critical survey. *Technical Report*, 2003.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. Cambridge: MIT press, 1998.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1057–1063, 2000.

Teh, Y. W., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., and Pascanu, R. Distral: Robust multitask reinforcement learning. *arXiv preprint arXiv:1707.04175*, 2017.

Tsitsiklis, J. N. and Van Roy, B. Analysis of temporal-diffference learning with function approximation. In *Advances in Neural Information Processing Systems*, pp. 1075–1081, 1997.

Tsitsiklis, J. N. and Van Roy, B. Average cost temporal-difference learning. *Automatica*, 35(11):1799–1808, 1999.

Wang, X. and Sandholm, T. Reinforcement learning to play an optimal Nash equilibrium in team Markov games. In *Advances in Neural Information Processing Systems*, pp. 1603–1610, 2003.

Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and de Freitas, N. Sample efficient actor-critic with experience replay. In *International Conference on Learning Representations*, 2016.

Xiao, L., Boyd, S., and Lall, S. A scheme for robust distributed sensor fusion based on average consensus. In *International Symposium on Information Processing in Sensor Networks*, pp. 9, 2005.