Learning to Coordinate in Multi-Agent Systems: A Coordinated Actor-Critic Algorithm and Finite-Time Guarantees

Siliang Zeng ZENG0176@UMN.EDU
Department of Electrical and Computer Engineering, University of Minnesota, MN, USA
Tianyi Chen CHENT18@RPI.EDU
Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, NY, USA
Alfredo Garcia ALFREDO.GARCIA@TAMU.EDU
Department of Industrial and Systems Engineering, Texas A&M University, TX, USA
Mingyi Hong MHONG@UMN.EDU
Department of Electrical and Computer Engineering, University of Minnesota, MN, USA

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

[1]Multi-agent reinforcement learning (MARL) has attracted much research attention recently. However, unlike its single-agent counterpart, many theoretical and algorithmic aspects of MARL have not been well-understood. In this paper, we study the emergence of coordinated behavior by autonomous agents using an actor-critic (AC) algorithm. Specifically, we propose and analyze a class of coordinated actor-critic (CAC) algorithms in which individually parametrized policies have a *shared* part (which is jointly optimized among all agents) and a *personalized* part (which is only locally optimized). Such a kind of *partially personalized* policy allows agents to coordinate by leveraging peers' experience and adapt to individual tasks. The flexibility in our design allows the proposed CAC algorithm to be used in a *fully decentralized* setting, where the agents can only communicate with their neighbors, as well as in a *federated* setting, where the agents occasionally communicate with a server while optimizing their (partially personalized) local models. Theoretically, we show that under some standard regularity assumptions, the proposed CAC algorithm requires $\mathcal{O}(\epsilon^{-\frac{5}{2}})$ samples to achieve an ϵ-stationary solution (defined as the solution whose squared norm of the gradient of the objective function is less than ϵ). To the best of our knowledge, this work provides the first finite-sample guarantee for decentralized AC algorithm with partially personalized policies.

Keywords: Multi-Agent Reinforcement Learning, Actor-Critic, Parameter Sharing

1. Introduction

We consider the multi-agent reinforcement learning (MARL) problem, in which a common environment is influenced by the joint actions of multiple autonomous agents, each aiming to optimize their own individual objective. The MARL (Zhang et al., 2019; Lee et al., 2020) has received significant attention recently due to their outstanding performance in many practical applications including robotics (Stone and Veloso, 2000), autonomous driving (Shalev-Shwartz et al., 2016) and video games (Tampuu et al., 2017). Many efficient algorithms have been proposed (Lowe et al., 2017; Espeholt et al., 2018; Rashid et al., 2018), but unlike its single-agent counterpart, the theoretical

1. The appendix of this manuscript is included in our technical report (Zeng et al., 2021).

understanding of MARL is still very limited, especially in settings where there is no central controller to coordinate different agents, so that the information sharing is limited (Zhang et al., 2019).

An important subclass of MARL – the so-called *cooperative* MARL – has become popular recently due to its wide applications. In the cooperative MARL, the agents aim to collaborate with each other to learn and optimize a global objective. To this end, local information exchange and local communication may be used to jointly optimize a system-level performance measure (Zhang et al., 2018; Grosnit et al., 2021; Lu et al., 2021). Next, we provide a brief survey about related works in cooperative MARL, and discuss their settings as well as theoretical guarantees.

Related Works. The systematic study of the cooperative MARL can be traced back to (Claus and Boutilier, 1998; Wolpert et al., 1999), which extended Q-learning algorithm (Watkins and Dayan, 1992) or its variants to the multi-agent setting. More recently, there are a number of works that characterize the theoretical performance of cooperative MARL algorithms in a fully observable, decentralized setting (Kar et al., 2012; Zhang et al., 2018; Doan et al., 2019). In such a setting, the agents are connected by a time-varying graph, and they can only communicate with their immediate neighbors. The goal of the agents is to cooperatively maximize certain global reward, by communicating local information with their neighbors. Under the above cooperative MARL setting, there are several lines of works which studied different problem formulations, proposed new algorithms and analyzed their theoretical performance.

The first line of works about the coorperative and fully observable MARL has focused on developing and analyzing policy evaluation algorithms, where the agents jointly estimate the global value function for a given policy. In Wai et al. (2018), a decentralized double averaging primal-dual optimization algorithm was proposed to solve the mean squared projected Bellman error minimization problem. It is shown that the proposed algorithm converges to the optimal solution at a global geometric rate. In Doan et al. (2019), the authors obtained a finite-sample analysis for decentralized TD(0) method. Their analysis is closely related to the theoretical results of decentralized stochastic gradient descent method on convex optimization problems (Nedic et al., 2010).

However, the problem becomes much more challenging when the agents are allowed to optimize their policies. A recent line of works has focused on applying and analyzing various policy optimization methods in the MARL setting. In Zhang et al. (2018), the authors extended the actor-critic (AC) algorithm (Konda and Tsitsiklis, 2000) to the cooperative MARL setting. The algorithm allows each agent to perform its local policy improvement step while approximating the global value function. A few more recent works have extended Zhang et al. (2018) in different directions. For example in Grosnit et al. (2021), the authors considered the continuous action spaces and obtained the asymptotic convergence guarantee under both off-policy and on-policy settings. Moreover, Zhang et al. (2021) considered a new decentralized formulation where all agents cooperate to maximize general utilities in the cooperative MARL system, it developed AC-type algorithms to fit this setting but still suffering from high sampling cost in estimating the occupancy measure for all states and the nested loop of optimization steps. A concurrent work (Chen et al., 2021) adopts large-batch updates in decentralized (natural) AC methods to improve sample and communication efficiency, whose convergence rate matches the analysis results of the corresponding centralized versions (Xu et al., 2020). However, the proposed algorithms in Chen et al. (2021) needs to generate $\mathcal{O}(\epsilon^{-1}\ln\epsilon^{-1})$ samples to update critic parameter before performing each actor update. It is worth noting, that all the above mentioned works do not allow the agents to share their local policies.

Our Contributions. Although there have been a growing literature on analyzing theoretical aspects of cooperative MARL, many challenges still remain, even under the basic fully observed

setting. For example, most of the cooperative policy optimization algorithms, assume relatively simple collaboration mechanism, where the agents collaborate by jointly estimating the global value function, while *independently* optimizing their local policies. Such a form of collaboration decouples the agents' policy optimization process, and it is relatively easy to analyze. However, it fails to capture some intrinsic aspects of cooperative MARL, in the sense that when the agents' local tasks are similar (a.k.a. the *homogeneous* setting), the agent's policy should also be closely related to each other. Such an intuition has been verified in MARL systems (Gupta et al., 2017; Terry et al., 2020b), multi-task RL systems (Omidshafiei et al., 2017; Zeng et al., 2020; Yu et al., 2020), Markov games (Vadori et al., 2020) and mean-field multi-agent reinforcement learning (Liu et al., 2020; Li et al., 2021), where parameter sharing scheme results in more stable convergence due to the benefit of learning homogeneity among different agents.

In this work, we aim at providing better theoretical and practical understandings about the cooperative MARL problem. We propose and analyze a Coordinated Actor-Critic (CAC) algorithm, which allows each agent to (partially) share its policy parameters with the neighbors for learning the homogeneity / common knowledge in the multi-agent system. To our knowledge, we provide the first non-asymptotic convergence result for two-timescale multi-agent AC methods. Moreover, we conduct extensive numerical experiments to demonstrate the effectiveness of the proposed algorithm.

2. Preliminaries

In this section, we introduce the background and formulation of the cooperative, fully observable MARL in a decentralized system. To model the communication pattern among the agents, let us define the time-varying graph $\mathcal{G}_t = (\mathcal{N}, \mathcal{E}_t)$ consisting of a set of $\mathcal{N}$ nodes and a set of $\mathcal{E}_t$ edges, with $|\mathcal{N}| = N$ and $|\mathcal{E}| = E$. Each node $i \in \mathcal{N}$ represents an agent and $\mathcal{E}_t$ represents the set of communication links at time t so that the agents are connected according to the links $\mathcal{E}_t$.

Consider the MARL problem, formulated as a discrete-time Markov Decision Process (MDP) $M := \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \eta, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the finite space for global state s and $\mathcal{A}$ is the finite space for joint action $\boldsymbol{a} = \{a_i\}_{i=1}^N$; $\eta(s) : \mathcal{S} \to [0,1]$ denotes the initial state distribution; $\mathcal{P}(s' \mid s, \boldsymbol{a}) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ denotes the transition probability; $r_i(s, \boldsymbol{a}) : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ denotes the local reward function of agent i; $\gamma \in (0,1)$ is the discounted factor. Furthermore, suppose the policy of each agent i is parameterized by θ_i, then $\boldsymbol{\theta} := \{\theta_i\}_{i=1}^N$ denotes the collections of all policy parameters in the multi-agent system. Then $\mu_{\boldsymbol{\theta}}(s)$ denotes the stationary distribution of each state s under joint policy $\pi_{\boldsymbol{\theta}}$, and $d_{\boldsymbol{\theta}}(\cdot)$ denotes the discounted visitation measure where $d_{\boldsymbol{\theta}}(s) := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \cdot \mathcal{P}^{\pi_{\boldsymbol{\theta}}}(s_t = s \mid s_0 \sim \eta)$. Under the joint policy $\pi_{\boldsymbol{\theta}}$, the probability for choosing any joint action $\boldsymbol{a} := \{a_i\}_{i=1}^N$ could be expressed as $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|s) := \Pi_{i=1}^N \pi_i(a_i|s, \theta_i)$.

Consider the discrete-time MDP under infinite horizon, the policy $\pi_{\boldsymbol{\theta}}$ can generate a trajectory $\tau := (s_0, \boldsymbol{a}_0, s_1, \boldsymbol{a}_1, \cdots)$ based on the initial state s_0 sampled from $\eta(\cdot)$. In this work, we consider the discounted cumulative reward setting and the global value function is defined as below:

$$V_{\pi_{\boldsymbol{\theta}}}(s) := \mathbb{E}\bigg[\sum_{t=0}^{\infty} \gamma^t \cdot \bar{r}(s_t, \boldsymbol{a}_t) \mid s_0 = s\bigg], \tag{1}$$

where we define $\bar{r}(s_t, \boldsymbol{a}_t) := \frac{1}{N}\sum_{i=1}^N r_i(s_t, \boldsymbol{a}_t)$ and the expectation is taken over the trajectory τ generated from joint policy $\pi_{\boldsymbol{\theta}}$. When $\pi_{\boldsymbol{\theta}}$ is fixed, the value function $V_{\pi_{\boldsymbol{\theta}}}(s)$ will satisfy the Bellman

equation (Bertsekas et al., 2000) for all states $s \in \mathcal{S}$:

$$V_{\pi_{\boldsymbol{\theta}}}(s) = \mathbb{E}_{\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s),s'\sim\mathcal{P}(\cdot|s,\boldsymbol{a})}\left[\bar{r}(s,\boldsymbol{a}) + \gamma \cdot V_{\pi_{\boldsymbol{\theta}}}(s')\right]. \tag{2}$$

The objective of RL is to find the optimal policy parameter $\boldsymbol{\theta}^*$ which maximizes the expected discounted cumulative reward as below:

$$\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) := \mathbb{E}_{s\sim\eta(\cdot)}[V_{\pi_{\boldsymbol{\theta}}}(s)] = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\gamma^t \cdot \bar{r}(s_t,\boldsymbol{a}_t)\bigg] = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\frac{\gamma^t}{N}\sum_{i=1}^{N} r_i(s_t,\boldsymbol{a}_t)\bigg]. \tag{3}$$

In order to optimize $J(\boldsymbol{\theta})$, the policy gradient (Sutton et al., 2000), could be expressed as

$$\nabla J(\boldsymbol{\theta}) := \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{\boldsymbol{\theta}}(\cdot),a\sim\pi_{\boldsymbol{\theta}}(\cdot|s),s'\sim\mathcal{P}(\cdot|s,\boldsymbol{a})}\left[\big(\bar{r}(s,\boldsymbol{a}) + \gamma V_{\pi_{\boldsymbol{\theta}}}(s')\big)\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|s)\right]. \tag{4}$$

3. The Proposed Coordinated Actor-Critic Algorithm

3.1. The Proposed Formulation

In this section, we describe our MARL formulation. Our proposed formulation is based upon (3), but with the key difference that we no longer require the agents to have independent policy parameters θ_i. Specifically, we assume that the agents can (partially) share their policy parameters with their neighbors. Hence, each agent will decompose its policy into $\theta_i := \{\theta_i^s, \theta_i^p\}$, where the shared part θ_i^s has the same dimension across all agents, and the personalized part θ_i^p will be kept locally.

The above partially personalized policy structure leads to the following MARL formulation:

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & J(\boldsymbol{\theta}) := \mathbb{E}_{s\sim\eta(\cdot)}\bigg[V_{\pi_{\boldsymbol{\theta}}}(s)\bigg] = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\gamma^t \cdot \bar{r}(s_t,\boldsymbol{a}_t)\bigg] \\ \text{s.t.} \quad & \theta_i^s = \theta_j^s \ \text{ if } (i,j) \text{ are neighbors} \end{aligned} \tag{5}$$

where $\boldsymbol{\theta} := \{\theta_i^s, \theta_i^p\}_{i=1}^N$ is the collections of all local policy parameters $\theta_i := \{\theta_i^s, \theta_i^p\}$. To cast problem (5) into a more tractable form, we perform the following steps.

First, we approximate the global reward function for any $s \in \mathcal{S}$ and $\boldsymbol{a} \in \mathcal{A}$. Specifically, we use the following linear function $\widehat{r}(s,\boldsymbol{a};\lambda) := \varphi(s,\boldsymbol{a})^T\lambda$ to approximate the global reward $\bar{r}(s,\boldsymbol{a}) := \frac{1}{N}\sum_{i=1}^N r_i(s,\boldsymbol{a})$, where $\varphi(\cdot,\cdot) : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^L$ is the feature mapping. Then the optimal parameter $\lambda^*(\boldsymbol{\theta})$ can be found by solving the following problem:

$$\lambda^*(\boldsymbol{\theta}) \in \arg\min_{\lambda} \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\left[\Big(\frac{1}{N}\sum_{i=1}^N r_i(s,\boldsymbol{a}) - \varphi(s,\boldsymbol{a})^T\lambda\Big)^2\right] \tag{6a}$$

$$= \arg\min_{\lambda} \sum_{i=1}^N \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\left[\big(r_i(s,\boldsymbol{a}) - \varphi(s,\boldsymbol{a})^T\lambda\big)^2\right]. \tag{6b}$$

Second, we approximate the global value function $V_{\pi_{\boldsymbol{\theta}}}(s)$ for any $s \in \mathcal{S}$ under a fixed joint policy $\pi_{\boldsymbol{\theta}}$. Specifically, we use the following linear function $\widehat{V}(s;\omega) := \phi(s)^T\omega$ to approximate the global reward function $V_{\pi_{\boldsymbol{\theta}}}(s)$, where $\phi(\cdot) : \mathcal{S} \to \mathbb{R}^K$ is a given feature mapping. Towards achieving the

above approximation, we can solve the following mean squared Bellman error (MSBE) minimization problem (Tsitsiklis and Van Roy, 1997):

$$\omega^*(\boldsymbol{\theta}) \in \arg\min_{\omega} \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s),s'\sim\mathcal{P}(\cdot|s,\boldsymbol{a})}\left[\left(\frac{1}{N}\sum_{i=1}^{N} r_i(s,\boldsymbol{a}) + \gamma\widehat{V}(s';\omega) - \widehat{V}(s;\omega)\right)^2\right] \tag{7a}$$

$$= \arg\min_{\omega} \sum\nolimits_{i=1}^{N} \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s),s'\sim\mathcal{P}(\cdot|s,\boldsymbol{a})}\left[\left(r_i(s,\boldsymbol{a}) + \gamma\widehat{V}(s';\omega) - \widehat{V}(s;\omega)\right)^2\right]. \tag{7b}$$

To separate the objective into the sum of N terms (one for each agent), we introduce local copies of w and λ as $\{w_i\}_{i=1}^N$, $\{\lambda_i\}_{i=1}^N$, and define their vectorized versions $\boldsymbol{\omega} = [\omega_1, \cdots, \omega_N]^T$ and $\boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_N]^T$. Similarly, we also define $\boldsymbol{\omega}^*(\boldsymbol{\theta}) := [\omega_1^*(\boldsymbol{\theta}), \cdots, \omega_N^*(\boldsymbol{\theta})]^T$ and $\boldsymbol{\lambda}^*(\boldsymbol{\theta}) := [\lambda_1^*(\boldsymbol{\theta}), \cdots, \lambda_N^*(\boldsymbol{\theta})]^T$.

Summarizing the above discussion, problem (5) can be approximated using the following bi-level optimization problem:

$$\max_{\boldsymbol{\theta}} \quad \mathop{\mathbb{E}}_{\substack{s\sim\eta(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s)\\ s'\sim\mathcal{P}(\cdot|s,\boldsymbol{a})}}\left[\frac{1}{N}\sum_{i=1}^{N}\left(\widehat{r}(s,\boldsymbol{a};\lambda_i^*(\boldsymbol{\theta})) + \gamma\cdot\widehat{V}(s';\omega_i^*(\boldsymbol{\theta}))\right)\right] \tag{8a}$$

$$s.t. \quad \boldsymbol{\omega}^*(\boldsymbol{\theta}) \in \arg\min_{\omega} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\substack{s\sim\mu_{\boldsymbol{\theta}}(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s)\\ s'\sim\mathcal{P}(\cdot|s,\boldsymbol{a})}}\left[\left(r_i(s,\boldsymbol{a}) + \gamma\cdot\widehat{V}(s';\omega_i) - \widehat{V}(s;\omega_i)\right)^2\right], \tag{8b}$$

$$\boldsymbol{\lambda}^*(\boldsymbol{\theta}) \in \arg\min_{\lambda} \sum_{i=1}^{N} \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot),\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\left[\left(r_i(s,\boldsymbol{a}) - \widehat{r}(s,\boldsymbol{a};\lambda_i)\right)^2\right], \tag{8c}$$

$$\theta_i^s = \theta_j^s,\ \omega_i^*(\boldsymbol{\theta}) = \omega_j^*(\boldsymbol{\theta}),\ \lambda_i^*(\boldsymbol{\theta}) = \lambda_j^*(\boldsymbol{\theta}),\ \text{ if } (i,j) \text{ are neighbors.} \tag{8d}$$

In the subsequent discussion, we will refer to the problem of finding the optimal policy $\boldsymbol{\theta}$ as the *upper-level* problem, while referring to the problem of finding the optimal $\boldsymbol{\omega}^*(\boldsymbol{\theta})$ and $\boldsymbol{\lambda}^*(\boldsymbol{\theta})$ under a fixed policy parameters as the lower-level problem.

3.2. The Proposed Algorithm

In this subsection, we first present the assumptions related to network connectivity and communication protocols in the multi-agent systems. Then we describe the proposed Coordinated Actor-Critic (CAC) algorithm which is summarized in Algorithm 1.

Assumption 1 (Time-varying Network Connectivity) *There exists an integer B such that the union of the consecutive B graphs is connected for all positive integers ℓ. The following graph is connected:*

$$\left(\mathcal{N}, \mathcal{E}(\ell\cdot B)\cup\mathcal{E}(\ell\cdot B+1)\cdots\cup\mathcal{E}((\ell+1)B-1)\right),\ \forall\,\ell\geq 1$$

where $\mathcal{N}$ denotes the vertice set and $\mathcal{E}(t)$ denotes the set of active edges at time t.

Assumption 2 (Weight Matrices) *There exists a positive constant c such that $W_t = [W_t^{ij}] \in \mathcal{R}^{N\times N}$ is doubly stochastic and $W_t^{ii} \geq c$ for all $i \in \mathcal{N}$. Moreover, $W_t^{ij} \in [c, 1)$ if $(i,j) \in \mathcal{E}(t)$, otherwise $W_t^{ij} = 0$ for all $i, j \in \mathcal{N}$.*

Algorithm 1 *Coordinated Actor-Critic (CAC) Algorithm*

1: **Input:** Parameters $\{\alpha_t\}_{t=0}^{T-1}$, $\{\beta_t\}_{t=0}^{T-1}$, $\{\zeta_t\}_{t=0}^{T-1}$. Initialize $\theta_{i,0}, \omega_{i,0}, \lambda_{i,0}$ for all $i \in \mathcal{N}$
2: **for** $t = 0, 1, \ldots, T-1$ **do**
3: **Data Sampling**: $s_t \sim \mu_{\boldsymbol{\theta}_t}(\cdot)$, $\boldsymbol{a}_t := \{a_{i,t} \sim \pi_i(\cdot|s_t, \theta_{i,t})\}_{i=1}^N$, $s_{t+1} \sim \mathcal{P}(\cdot|s_t, \boldsymbol{a}_t)$
4: **Consensus Step**: $\widetilde{\boldsymbol{\omega}}_t = W_t \cdot \boldsymbol{\omega}_t$, $\widetilde{\boldsymbol{\lambda}}_t = W_t \cdot \boldsymbol{\lambda}_t$ and $\widetilde{\boldsymbol{\theta}}_t^s := W_t \cdot \boldsymbol{\theta}_t^s$
5: **for** $i \in \mathcal{N}$ **do**
6: Construct $\widetilde{\theta}_{i,t} = \{\widetilde{\theta}_{i,t}^s, \theta_{i,t}^p\}$ and update $\delta_{i,t} = r_{i,t} + \gamma \cdot \phi(s_{t+1})^T \omega_{i,t} - \phi(s_t)^T \omega_{i,t}$
7: $\omega_{i,t+1} = \Pi_{R_\omega}\big(\widetilde{\omega}_{i,t} + \beta_t \cdot \delta_{i,t} \cdot \phi(s_t)\big)$
8: $\lambda_{i,t+1} = \Pi_{R_\lambda}\Big(\widetilde{\lambda}_{i,t} + \zeta_t \cdot \big(r_{i,t} - \varphi(s_t, \boldsymbol{a}_t)^T \lambda_{i,t}\big) \cdot \varphi(s_t, \boldsymbol{a}_t)\Big)$
9: $\theta_{i,t+1} = \widetilde{\theta}_{i,t} + \alpha_t \Big(\varphi(s_t, \boldsymbol{a}_t)^T \lambda_{i,t} + \gamma \phi(s_{t+1})^T \omega_{i,t} - \phi(s_t)^T \omega_{i,t}\Big) \nabla_{\theta_i} \log \pi_i(a_{i,t}|s_t, \theta_{i,t})$
10: **end for**
11: **end for**

Assumption 1 ensures that the graph sequence is sufficiently connected for each agent to have repeated influence on other agents. Assumption 2 is standard in developing decentralized algorithms (Nedic et al., 2009), which could guarantee consensus results for shared parameter in each agent converging to a common vector.

After presenting the assumptions related to the network topology in the decentralized system, we are able to introduce the proposed CAC algorithm. The CAC algorithm takes two main steps, the policy optimization step (which optimizes $\boldsymbol{\theta}$), and policy evaluation step (which approximately solves the lower-level problem in (8)), as we describe below. For simplicity, we denote $\bar{r}(s_t, \boldsymbol{a}_t)$ as $\bar{r}_t$ and $r_i(s_t, \boldsymbol{a}_t)$ as $r_{i,t}$.

Policy Optimization. In this step, the agents optimize their local policy parameters, while trying to make sure that the shared parameters are not too far from their neighbors.

Towards this end, each agent i first produces a *locally averaged* shared parameter by linearly combining with its neighbors current shared parameters. Such an operation can be expressed as

$$\widetilde{\boldsymbol{\theta}}_t^s := W_t \cdot \boldsymbol{\theta}_t^s \tag{9}$$

where $\boldsymbol{\theta}_t^s := [\theta_{1,t}^s, \theta_{2,t}^s, \cdots; \theta_{N,t}^s]^T \in \mathbb{R}^{N \times H}$ is a matrix which stores all parameters $\{\theta_{i,t}^s\}_{i=1}^N$, and $\widetilde{\boldsymbol{\theta}}_t^s$ is defined similarly. In the decentralized setting, the global reward $\bar{r}_t$ and the global value function $V_{\pi_{\boldsymbol{\theta}_t}}(\cdot)$ are not available for each agent i. Instead, the agents can locally estimate the global reward and the global value function using some linear approximation, evaluated on their local variables, as described in the previous subsection. As shown in line 11 of Algorithm 1, in a decentralized system, we consider the policy optimization step for each agent as below:

$$\theta_{i,t+1} := \widetilde{\theta}_{i,t} + \alpha_t \cdot \widehat{\delta}_{i,t} \cdot \nabla_{\theta_i} \log \pi_i(a_{i,t}|s_t, \theta_{i,t}), \quad \forall i \in \mathcal{N} \tag{10}$$

$$\text{where} \quad \widehat{\delta}_{i,t} := \widehat{r}(s_t, \boldsymbol{a}_t; \lambda_{i,t}) + \gamma \cdot \widehat{V}(s_{t+1}; \omega_{i,t}) - \widehat{V}(s_t; \omega_{i,t}). \tag{11}$$

Policy Evaluation. Next, we update the local parameters $\lambda_{i,t}$ and $\omega_{i,t}$, which parameterize the global reward function and global value function. Towards this end, the parameters $\lambda_{i,t}$ and $\omega_{i,t}$ will be updated by first averaging over their neighbors, then performing one stochastic gradient descent step

to minimize the local objectives, which are defined as in (8b) - (8c) and under consensus constraints (8d). That is, we have the following updates for $\boldsymbol{\lambda}_t$ and $\boldsymbol{\omega}_t$:

$$\widetilde{\boldsymbol{\lambda}}_t = W_t \cdot \boldsymbol{\lambda}_t, \quad \lambda_{i,t+1} = \Pi_{R_\lambda}\Big(\widetilde{\lambda}_{i,t} + \zeta_t \cdot \big(r_{i,t} - \widehat{r}(s_t, \boldsymbol{a}_t; \lambda_{i,t})\big) \cdot \nabla_{\lambda_i} \widehat{r}(s_t, \boldsymbol{a}_t; \lambda_{i,t})\Big), \tag{12}$$

$$\widetilde{\boldsymbol{\omega}}_t = W_t \cdot \boldsymbol{\omega}_t, \quad \omega_{i,t+1} = \Pi_{R_\omega}\Big(\widetilde{\omega}_{i,t} + \beta_t \cdot \delta_{i,t} \cdot \nabla_{\omega_i} \widehat{V}(s_t; \omega_{i,t})\Big), \quad \forall i \in \mathcal{N} \tag{13}$$

where we define $\delta_{i,t} := r_{i,t} + \gamma \cdot \widehat{V}(s_{t+1}; \omega_{i,t}) - \widehat{V}(s_t; \omega_{i,t})$. Moreover, $\Pi_{R_\omega}(\cdot)$ and $\Pi_{R_\lambda}(\cdot)$ are the projection operators, with R_ω and R_λ being the predetermined projection radii which are used to stabilize the update process (Tsitsiklis and Van Roy, 1997). Please see lines 8-10 in Algorithm 1.

4. Theoretical Results

In this section, we first present Assumptions 3 - 4 about reward function and linear approximations for policy evaluation. Then we show our theoretical results for the proposed CAC algorithm.

Assumption 3 (Bounded Reward) *All the local rewards $r_i(s, a)$ are uniformly bounded, i.e., there exist constants $R_{\max}$ such that $|r_i(s, a)| \leq R_{\max}$ for all $i \in \mathcal{N}$, $s \in \mathcal{S}$ and $a \in \mathcal{A}$.*

Assumption 4 (Function Approximation) *For each agent i, the value function and the global reward function are both parameterized by the class of linear functions, i.e., $\widehat{V}(s; \omega_i) := \phi(s)^T \omega_i$ and $\widehat{r}(s, \boldsymbol{a}; \lambda_i) := \varphi(s, \boldsymbol{a})^T \lambda_i$ where we denote $\phi(s) := [\phi_1(s), \cdots, \phi_K(s)]^T \in \mathbb{R}^K$ and $\varphi(s, \boldsymbol{a}) = [\varphi_1(s, \boldsymbol{a}), \cdots, \varphi_L(s, \boldsymbol{a})]^T \in \mathbb{R}^L$ are the feature vector associated with s and $(s, \boldsymbol{a})$, respectively. The feature vectors $\phi(s)$ and $\varphi(s, \boldsymbol{a})$ are uniformly bounded for any $s \in \mathcal{S}, \boldsymbol{a} \in \mathcal{A}$, i.e., $\|\phi(s)\| \leq 1$ and $\|\varphi(s, \boldsymbol{a})\| \leq 1$. Furthermore, constructing the feature matrix $\Phi \in \mathbb{R}^{|\mathcal{S}| \times K}$ which has $[\phi_k(s), s \in \mathcal{S}]^T$ as its k-th column for any $k \in K$. Also constructing the feature matrix $\Psi \in \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}| \times L}$ which has $[\varphi_l(s), s \in \mathcal{S}]^T$ as its ℓ-th column for any $\ell \in L$. Then, we further assume both Φ and Ψ have full column ranks.*

Assumption 3 - 4 are common in analyzing TD with linear function approximation; see e.g., (Konda and Tsitsiklis, 2000; Bhandari et al., 2018; Wu et al., 2020). Observing the global state and actions, each agent could construct linear function approximations of the global value function and global reward function. Under these assumptions, it is guaranteed that there exist unique optimal solutions $\lambda^*(\boldsymbol{\theta})$ and $\omega^*(\boldsymbol{\theta})$ to approximate the global reward function in (6) and the global value function in (7) with linear functions. It is crucial to have the properties of unique optimal solutions in $\lambda^*(\boldsymbol{\theta})$ and $\omega^*(\boldsymbol{\theta})$ for constructing the convergence analysis of policy parameters $\boldsymbol{\theta}$.

Moreover, we have two more technical assumptions (i.e., Assumptions 5 - 6) to ensure the Lipschitz property of the policy and guarantee the Markov chain mixes at a geometric rate. Due to space limitation, we relegate Assumptions 5 - 6 to Appendix C and technical lemmas to Appendix D in our technical report (Zeng et al., 2021). We first present the convergence speed of the variables $\{\boldsymbol{\omega}_t\}$ and $\{\boldsymbol{\lambda}_t\}$ for the policy evaluation problem defined in (8b) - (8d). For the detailed proof, please see Appendix G in our technical report (Zeng et al., 2021).

Proposition 1 *Suppose Assumptions 1 - 6 hold. By selecting diminishing stepsizes*

$$\alpha_t = \frac{\alpha_0}{T^{\sigma_1}}, \ \beta_t = \frac{\beta_0}{T^{\sigma_2}}, \ \zeta_t = \frac{\zeta_0}{T^{\sigma_2}}$$

where $0 < \sigma_2 < \sigma_1 < 1$ *and* $\alpha_0, \beta_0, \zeta_0 > 0$ *are some fixed constants, the following holds:*

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}\left(\mathbb{E}\left[\|\omega_{i,t}-\omega^*(\boldsymbol{\theta}_t)\|^2\right]+\mathbb{E}\left[\|\lambda_{i,t}-\lambda^*(\boldsymbol{\theta}_t)\|^2\right]\right)\\ =&\mathcal{O}(T^{-1+\sigma_2})+\mathcal{O}(T^{-\sigma_2})+\mathcal{O}\left(T^{\sigma_2-2\sigma_1}\right)+\mathcal{O}(T^{-2\sigma_1+2\sigma_2})+\mathcal{O}(T^{-2+2\sigma_2})+\mathcal{O}(T^{-2\sigma_2})\end{aligned}$$

where the expectation is taken over the data sampling procedure as shown in line 3 of Algorithm 1.

Compared with existing works (Wai et al., 2018; Doan et al., 2019) which established finite-time convergence guarantees for decentralized policy evaluation problems under the fixed policy, our results in Proposition 1 are analyzed in a more challenging situation where both policies and critics are updated in an alternating manner. Here, we must set $\sigma_1 > \sigma_2$ to ensure the relation above is useful. This is reasonable since the optimal critic parameter $\omega^*(\boldsymbol{\theta}_t)$ is constantly drifting as the policy parameters $\boldsymbol{\theta}_t$ changes at each iteration, so the actor should update slowly compared with the critic.

Next, we study the convergence rate of policy parameters. We define $Q := I - \frac{1}{N}\mathbf{1}\mathbf{1}^T$ and define the average gradient of shared policy parameters as $\overline{\nabla_{\boldsymbol{\theta}^s} J(\boldsymbol{\theta})} := \frac{1}{N}\sum_{i=1}^{N}\nabla_{\theta_i^s} J(\boldsymbol{\theta})$. We will show that after averaging over the iterations, the expected stationarity condition violation for policy optimization problem defined in (8a) is small. Please see Appendix H in Zeng et al. (2021) for the proof.

Proposition 2 *Under the same setting as Proposition 1, there exist two constant error term* $\epsilon_{app} > 0$ *and* $\epsilon_{sp} > 0$*. Algorithm 1 generates a sequence of policies* $\{\boldsymbol{\theta}_t\}$*, which satisfies the following:*

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\left(\mathbb{E}\left[\|Q\cdot\boldsymbol{\theta}_t^s\|^2\right]+N\cdot\mathbb{E}\left[\|\overline{\nabla_{\theta^s} J(\boldsymbol{\theta}_t)}\|^2\right]+\sum_{i=1}^{N}\mathbb{E}\left[\|\nabla_{\theta_i^p} J(\boldsymbol{\theta}_t)\|^2\right]\right)\\ =&\mathcal{O}(T^{-1+\sigma_1})+\mathcal{O}(T^{-\sigma_1})+\mathcal{O}(T^{-1+\sigma_2})+\mathcal{O}(T^{-\sigma_2})+\mathcal{O}\left(T^{\sigma_2-2\sigma_1}\right)+\mathcal{O}(T^{-2\sigma_1+2\sigma_2})\\ &+\mathcal{O}(T^{-2+2\sigma_2})+\mathcal{O}(T^{-2\sigma_2})+\mathcal{O}\left(\epsilon_{app}+\epsilon_{sp}\right).\end{aligned}$$

The approximation error ϵ_{app} and sampling error ϵ_{sp} are defined in Appendix E of the technical report (Zeng et al., 2021). A few remarks about the above results follow. First, one challenge in analyzing the convergence of Actor-Critic algorithms is that the actor and critic updates are typically sampled from different distributions (i.e., the distribution mismatch problem). To see this, note that to obtain an unbiased estimator for the policy gradient in (4), one needs to sample from the discounted visitation measure $d_{\boldsymbol{\theta}}(\cdot)$, while to obtain an unbiased estimator for the gradient of the MSBE in (7) (which is utilized to update the critic parameters), one needs to sample from the stationary distribution $\mu_{\boldsymbol{\theta}}(\cdot)$. However, standard implementations for AC methods in practice only use one sampling procedure for both actor and critic updates (Mnih et al., 2016; Shen et al., 2020). Therefore, the mismatch between the two sampling distributions inevitably introduces constant biases, and this is where the error term ϵ_{sp} comes from.

Second, at each local agent i, the value function $V_{\pi_{\boldsymbol{\theta}}}(s)$ is approximated by $\phi(s)^T\omega_i$ and the global reward function is approximated by $\varphi(s,\boldsymbol{a})^T\lambda_i$. Due to the linear approximation, the approximation error is inevitable in the convergence analysis. Here, we use a constant term ϵ_{app} to quantify the approximation error due to utilizing linear function for policy evaluation.

By combining previous Propositions, and by properly selecting the stepsize parameters σ_1 and σ_2, we show the main result as below. In Appendix E, we will present more discussion about a special case where there is no policy parameter sharing.

Algorithm 2 *Double Sampling Procedures*

Input: Parameters $\{\omega_{i,t}\}_{i=1}^N$, $\{\lambda_{i,t}\}_{i=1}^N$, $\{\theta_{i,t}\}_{i=1}^N$.
Double i.i.d. Sampling:
1) Sample $s_t \sim \mu_{\boldsymbol{\theta}_t}(\cdot)$, $\boldsymbol{a}_t := \{a_{i,t} \sim \pi_i(\cdot|s_t, \theta_{i,t})\}_{i=1}^N$, $s_{t+1} \sim \mathcal{P}(\cdot|s_t, \boldsymbol{a}_t)$
2) Sample $\tilde{s}_t \sim d_{\boldsymbol{\theta}_t}(\cdot)$, $\tilde{\boldsymbol{a}}_t := \{\tilde{a}_{i,t} \sim \pi_i(\cdot|\tilde{s}_t, \theta_{i,t})\}_{i=1}^N$, $\tilde{s}_{t+1} \sim \mathcal{P}(\cdot|\tilde{s}_t, \tilde{\boldsymbol{a}}_t)$

Theorem 3 *(Convergence of the CAC Algorithm) Suppose Assumptions 1 - 6 hold. Consider Algorithm 1 with partially shared policy parameters* $\boldsymbol{\theta} := \cup_{i=1}^N \{\theta_i^s, \theta_i^p\}$. *Let* $\sigma_1 = \frac{3}{5}$ *and* $\sigma_2 = \frac{2}{5}$, *it holds that:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}\left(\mathbb{E}\left[\|\omega_{i,t} - \omega^*(\boldsymbol{\theta}_t)\|^2\right] + \mathbb{E}\left[\|\lambda_{i,t} - \lambda^*(\boldsymbol{\theta}_t)\|^2\right]\right) = \mathcal{O}(T^{-\frac{2}{5}}),$$

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\mathbb{E}\left[\|Q\cdot\boldsymbol{\theta}_t^s\|^2\right] + N\cdot\mathbb{E}\left[\|\overline{\nabla_{\boldsymbol{\theta}^s} J(\boldsymbol{\theta}_t)}\|^2\right] + \sum_{i=1}^{N}\mathbb{E}\left[\|\nabla_{\theta_i^p} J(\boldsymbol{\theta}_t)\|^2\right]\right) = \mathcal{O}(T^{-\frac{2}{5}}) + \mathcal{O}(\epsilon_{app} + \epsilon_{sp}).$$

As mentioned before, the sampling error ϵ_{sp} arises because there is a mismatch between the way that estimators of the actor's and the critics' updates are obtained. To remove the sampling error, one can implement separate sampling protocols for the critic and the actor. More specifically, we can use two different i.i.d. samples at each iteration step t: 1) $x_t := (s_t, \boldsymbol{a}_t, s_{t+1})$ where $s_t \sim \mu_{\boldsymbol{\theta}}(\cdot)$, $\boldsymbol{a}_t \sim \pi_{\boldsymbol{\theta}}(\cdot \mid s_t)$ and $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, \boldsymbol{a}_t)$; 2) $\tilde{x}_t := (\tilde{s}_t, \tilde{\boldsymbol{a}}_t, \tilde{s}_{t+1})$ where $\tilde{s}_t \sim d_{\boldsymbol{\theta}}(\cdot)$, $\tilde{\boldsymbol{a}}_t \sim \pi_{\boldsymbol{\theta}}(\cdot \mid s_t)$ and $\tilde{s}_{t+1} \sim \mathcal{P}(\cdot \mid \tilde{s}_t, \tilde{\boldsymbol{a}}_t)$; see Algorithm 2. Then x_t and $\tilde{x}_t$ will be utilized in policy evaluation and policy optimization, respectively. The corollary below shows the convergence result for the modified CAC algorithm. Please see Appendix I for the proof.

Corollary 1 *(Convergence under double sampling) Under the same setting as Theorem 3, consider CAC with the double sampling procedures in Algorithm 2. The following result holds:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{N}\left(\mathbb{E}\left[\|\omega_{i,t} - \omega^*(\boldsymbol{\theta}_t)\|^2\right] + \mathbb{E}\left[\|\lambda_{i,t} - \lambda^*(\boldsymbol{\theta}_t)\|^2\right]\right) = \mathcal{O}(T^{-\frac{2}{5}}),$$

$$\frac{1}{T}\sum_{t=0}^{T-1}\left(\mathbb{E}\left[\|Q\cdot\boldsymbol{\theta}_t^s\|^2\right] + N\cdot\mathbb{E}\left[\|\overline{\nabla_{\boldsymbol{\theta}^s} J(\boldsymbol{\theta}_t)}\|^2\right] + \sum_{i=1}^{N}\mathbb{E}\left[\|\nabla_{\theta_i^p} J(\boldsymbol{\theta}_t)\|^2\right]\right) = \mathcal{O}(T^{-\frac{2}{5}}) + \mathcal{O}(\epsilon_{app}).$$

5. Numerical Results

In this section, we present our simulation results on two environments: 1) the coordination game (Osborne and Rubinstein, 1994); 2) the pursuit-evasion game (Gupta et al., 2017), which is built on the PettingZoo platform (Terry et al., 2020a). Detailed experiment settings are present in Appendix A of our technical report (Zeng et al., 2021).

Coordination Game: In this setting, there are N agents staying at a static state and they choose their actions simultaneously at each time. After actions are executed at each time t, each agent i receives its reward as: $r_{i,t} = (a_{i,t} - 3.5)^2 + \sum_{j\neq i} I_{\{a_{j,t}=a_{i,t}\}} + \epsilon_{i,t}$ where the action space is $\{0, 1, 2, \cdots, 7\}$, $I_{\{a_{j,t}=a_{i,t}\}}$ is an indicator function and $\epsilon_{i,t}$ is a random payoff following standard Gumbel distribution. In this coordination game, there are multiple Nash equilibria where two optimal

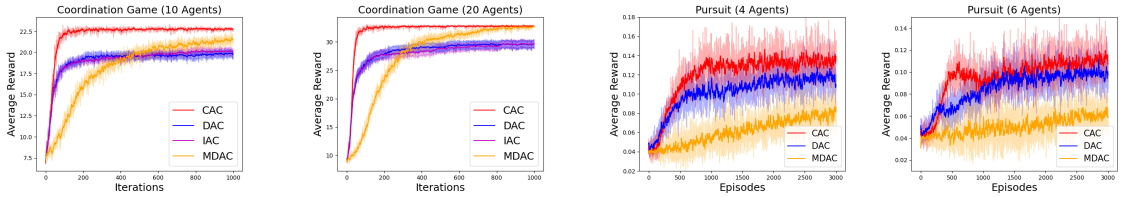

Figure 1: **Simulation Results.** The averaged reward versus the learning process. The performance is averaged over 10 Monte Carlo runs.

equilibria are that all agents select $a = \{0\}$ or $a = \{7\}$ simultaneously. In order to obtain high rewards and achieve efficient equilibria, it is crucial for agents to coordinate with others while only having limited communications. Here, the communication graph $\mathcal{G}_t$ between the agents is a complete graph every 5 iterations, and is not connected for the rest of time. We compare the performance of CAC with three benchmark algorithms: independent Actor-Critic (IAC); decentralized Actor-Critic (DAC) in Zhang et al. (2018); mini-batch decentralized Actor-Critic (MDAC) in Chen et al. (2021). For each algorithm, we set the actor stepsize and critic stepsize as 0.05 and 0.1. Theoretically, MDAC needs $\mathcal{O}(\epsilon^{-1} \ln \epsilon^{-1})$ batch size in its inner loop to update critic parameters before each update in policy parameters, which is inefficient in practice. Here, we set small batch $B = 5$ in the inner loop for MDAC to achieve fast convergence. The simulation results on this coordination game are present in Fig.1 (two left figures). According to the simulations, compared with the benchmarks, we see that the CAC algorithm converges faster and has higher probability to achieve efficient equilibria due to the use of policy sharing and coordination.

Pursuit-Evasion Game: there are two groups of nodes, pursuers (agents) and evaders. The pursuers aim to obtain reward through catching evaders. In a two-dimensional environment, an evader is considered caught if two pursuers simultaneously arrive at the evader's location. In order to catch an evader, each pursuer should learn to cooperate with other pursuers to catch the evaders. From this perspective, the pursuers share some similarities with each other since they need to follow similar strategies to achieve their local tasks: simultaneously catching a same evader with other pursuers. In Figure 1 (two right figures), we compare the numerical performance of the proposed CAC algorithm and two benchmarks: decentralized Actor-Critic (DAC) in Zhang et al. (2018); mini-batch decentralized Actor-Critic (MDAC) in Chen et al. (2021). Each agent maintains two convolutional neural networks (CNNs), one for the actor and one for the critic. Please see Fig.2 in Appendix A of Zeng et al. (2021) for the structure diagrams of actor network and critic network being used. In the CAC, two convolutional layers of actor network will be regarded as shared policy parameters, and the output layer is personalized (thus not shared).

6. Conclusion

This paper develops a novel collaboration mechanism for designing robust MARL systems. Further, it develops and analyzes a novel multi-agent AC method, where agents are allowed to (partially) share their policy parameters with the neighbors to learn from different agents. To our knowledge, this is the first non-asymptotic convergence result for two-timescale multi-agent AC methods.

Acknowledgments

We thank the anonymous reviewers for their valuable comments and suggestions. M. Hong and S. Zeng are partially supported by AFOSR Grant No. 19RT0424, and NSF Grant No. CMMI-1727757. A. Garcia is partially supported by AFOSR Grant No. 19RT0424.

References

Dimitri P Bertsekas et al. *Dynamic programming and optimal control: Vol. 1*. Athena scientific Belmont, 2000.

Jalaj Bhandari, Daniel Russo, and Raghav Singal. A finite time analysis of temporal difference learning with linear function approximation. In *Conference On Learning Theory*, pages 1691–1692. PMLR, 2018.

Ziyi Chen, Yi Zhou, Rongrong Chen, and Shaofeng Zou. Sample and communication-efficient decentralized actor-critic algorithms with finite-time analysis. *arXiv preprint arXiv:2109.03699*, 2021.

Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998(746-752):2, 1998.

Thinh Doan, Siva Maguluri, and Justin Romberg. Finite-time analysis of distributed td (0) with linear function approximation on multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 1626–1635. PMLR, 2019.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

Antoine Grosnit, Desmond Cai, and Laura Wynter. Decentralized deterministic multi-agent reinforcement learning. *arXiv preprint arXiv:2102.09745*, 2021.

Jayesh K Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 66–83. Springer, 2017.

Soummya Kar, José MF Moura, and H Vincent Poor. Qd-learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus. *arXiv preprint arXiv:1205.0047*, 2012.

Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014. Citeseer, 2000.

Donghwan Lee, Niao He, Parameswaran Kamalaruban, and Volkan Cevher. Optimization for reinforcement learning: From a single agent to cooperative agents. *IEEE Signal Processing Magazine*, 37(3):123–135, 2020.

Yan Li, Lingxiao Wang, Jiachen Yang, Ethan Wang, Zhaoran Wang, Tuo Zhao, and Hongyuan Zha. Permutation invariant policy optimization for mean-field multi-agent reinforcement learning: A principled approach. *arXiv preprint arXiv:2105.08268*, 2021.

Lewis Liu, Zhuoran Yang, Yuchen Lu, and Zhaoran Wang. Decentralized policy gradient method for mean-field linear quadratic regulator with global convergence. 2020.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017.

Songtao Lu, Kaiqing Zhang, Tianyi Chen, Tamer Basar, and Lior Horesh. Decentralized policy gradient descent ascent for safe multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8767–8775, 2021.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

Angelia Nedic, Alex Olshevsky, Asuman Ozdaglar, and John N Tsitsiklis. On distributed averaging algorithms and quantization effects. *IEEE Transactions on automatic control*, 54(11):2506–2517, 2009.

Angelia Nedic, Asuman Ozdaglar, and Pablo A Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938, 2010.

Shayegan Omidshafiei, Jason Pazis, Christopher Amato, Jonathan P How, and John Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *International Conference on Machine Learning*, pages 2681–2690. PMLR, 2017.

Martin J Osborne and Ariel Rubinstein. *A course in game theory*. MIT press, 1994.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4295–4304. PMLR, 2018.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

Han Shen, Kaiqing Zhang, Mingyi Hong, and Tianyi Chen. Asynchronous advantage actor critic: Non-asymptotic analysis and linear speedup. *arXiv preprint arXiv:2012.15511*, 2020.

Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, 2000.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395, 2017.

Justin K Terry, Benjamin Black, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis Santos, Clemens Dieffendahl, Niall L Williams, Yashas Lokesh, Caroline Horsch, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *arXiv preprint arXiv:2009.14471*, 2020a.

Justin K Terry, Nathaniel Grammel, Ananth Hari, Luis Santos, and Benjamin Black. Revisiting parameter sharing in multi-agent deep reinforcement learning. *arXiv preprint arXiv:2005.13625*, 2020b.

John N Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE transactions on automatic control*, 42(5):674–690, 1997.

Nelson Vadori, Sumitra Ganesh, Prashant Reddy, and Manuela Veloso. Calibration of shared equilibria in general sum partially observable markov games. *arXiv preprint arXiv:2006.13085*, 2020.

Hoi-To Wai, Zhuoran Yang, Zhaoran Wang, and Mingyi Hong. Multi-agent reinforcement learning via double averaging primal-dual optimization. *arXiv preprint arXiv:1806.00877*, 2018.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

David H Wolpert, Kevin R Wheeler, and Kagan Tumer. General principles of learning-based multi-agent systems. In *Proceedings of the third annual conference on Autonomous Agents*, pages 77–83, 1999.

Yue Wu, Weitong Zhang, Pan Xu, and Quanquan Gu. A finite time analysis of two time-scale actor critic methods. *arXiv preprint arXiv:2005.01350*, 2020.

Tengyu Xu, Zhe Wang, and Yingbin Liang. Improving sample complexity bounds for (natural) actor-critic algorithms. *Advances in Neural Information Processing Systems*, 33, 2020.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.

Sihan Zeng, Aqeel Anwar, Thinh Doan, Justin Romberg, and Arijit Raychowdhury. A decentralized policy gradient approach to multi-task reinforcement learning. *arXiv preprint arXiv:2006.04338*, 2020.

Siliang Zeng, Tianyi Chen, Alfredo Garcia, and Mingyi Hong. Learning to coordinate in multi-agent systems: A coordinated actor-critic algorithm and finite-time guarantees. *arXiv preprint arXiv:2110.05597*, 2021.

Junyu Zhang, Amrit Singh Bedi, Mengdi Wang, and Alec Koppel. Marl with general utilities via decentralized shadow reward actor-critic. *arXiv preprint arXiv:2106.00543*, 2021.

Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *International Conference on Machine Learning*, pages 5872–5881. PMLR, 2018.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*, 2019.