2022 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, AUG. 22–25, 2022, XI'AN, CHINA

DATA-DRIVEN ROBUST MULTI-AGENT REINFORCEMENT LEARNING

Yudan Wang[†] *Yue Wang*[†] *Yi Zhou*[⋆] *Alvaro Velasquez*[‡] *Shaofeng Zou*[†]

[†]Department of Electrical Engineering, University at Buffalo
[⋆] Department of Electrical and Computer Engineering, University of Utah
[‡] Information Directorate, Air Force Research Laboratory
[†]{yudanwan, ywang294, szou3}@buffalo.edu, [⋆] yi.zhou@utah.edu, [‡]alvaro.velasquez.1@us.af.mil

2022 IEEE 32nd International Workshop on Machine Learning for Signal Processing (MLSP) | 978-1-6654-8547-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/MLSP55214.2022.9943500

ABSTRACT

Multi-agent reinforcement learning (MARL) in the collaborative setting aims to find a joint policy that maximizes the accumulated reward averaged over all the agents. In this paper, we focus on MARL under model uncertainty, where the transition kernel is assumed to be in an uncertainty set, and the goal is to optimize the worst-case performance over the uncertainty set. We investigate the model-free setting, where the uncertain set centers around an *unknown* Markov decision process from which a single sample trajectory can be obtained sequentially. We develop a robust multi-agent Q-learning algorithm, which is model-free and fully decentralized. We theoretically prove that the proposed algorithm converges to the minimax robust policy, and further characterize its sample complexity. Our algorithm, comparing to the vanilla multi-agent Q-learning, offers provable robustness under model uncertainty without incurring additional computational and memory cost.

Index Terms— Distributionally robust, model-free, sample complexity, finite-time analysis, robust MDP

1. INTRODUCTION

Multi-agent reinforcement learning (MARL) [1] finds a wide range of applications in modern artificial intelligence applications, where multiple autonomous agents interact with a common stochastic environment [2, 3, 4]. Multi-agent systems are usually distributed, and agents communicate through wireless channel, and therefore, they are vulnerable to external perturbations and adversarial attacks, which may result in a model deviation, and further lead to a significant performance degradation. However, existing results typically assume that the policy will be deployed in the same environment as the one where training samples are taken [5], and thus may not perform well when there is model deviation between the training and test environments. In this paper, we develop a robust MARL approach, where the Markov decision process (MDP) model is not fixed but lies in an uncertainty set, and the goal is to optimize the worst-case performance over the uncertainty set.

The framework of robust MDP was developed in [6, 7, 8] for the single-agent setting. A robust dynamic programming approach was developed, and was shown to be minimax optimal. This approach, however, requires *full* knowledge of the uncertainty set, and does not scale well to large or continuous problems. Following this framework, model-free approaches with function approximation are developed, e.g., [9, 10], but the convergence results require a stringent condition on the discount factor. There are also heuristic approaches on robust RL, e.g., [11, 12, 13, 14, 15], but they lack in provable performance guarantee. More importantly, the above studies are mostly focused on the single-agent case. Recently, the work [16] studied reward uncertainty in MARL, but did not take into consideration the Markov transition kernel uncertainty. There are also studies on MARL, e.g., [17, 18, 19, 20, 1], but they are limited to the non-robust case.

In this paper, we investigate the problem of robust MARL in the collaborative setting with uncertainty in the Markov transition kernel, where the agents aim to maximize the accumulative average reward over all the agents under the worst-case Markov transition kernel in the uncertainty set. We generalize the single-agent robust Q-learning algorithm in [21] to the decentralized multi-agent setting, where there is no fusion center, each agent's reward information is only locally observable, and each agent may only communicate with its neighbors in the network. Our contributions in this paper can be summarized in three-fold. First, we design an online model-free multi-agent robust Q-learning (MARQ) algorithm. Our MARQ algorithm can be updated in an online and incremental fashion, and at each time, the agent only needs to communicate with its neighbors. Moreover, its computational and memory complexity are the same as the vanilla Q-learning algorithm (within a constant factor). Second, we theoretically prove the convergence of MARQ, and derive its sample complexity in the tabular case, which matches with one of the centralized tabular Q-learning algorithm (within a constant factor). Our analysis is based on a novel combination of distributed optimization [22] and robust reinforcement

978-1-6654-8547-0/22//$31.00 ©2022 IEEE

learning, which requires an explicit characterization of the distributed optimization error and the stochastic error in robust Q-learning. Third, we numerically demonstrate the convergence and robustness of our algorithm. Our approach can be easily combined with the deep Q-learning algorithm [23] and the double Q-learning approach [24], and design robust deep Q-learning for large or continuous problems.

2. PROBLEM MODEL

A decentralized multi-agent MDP can be represented by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathsf{P}, \mathcal{N}, \mathcal{G}, r, \gamma \rangle$, where $\mathcal{S}$ denotes the state space and $|\mathcal{S}|$ is the number of the states; $\mathcal{A}$ denotes the joint action space which can be factorized as $\otimes_{i=1}^{N} \mathcal{A}^{(i)}$ and $\mathcal{A}^{(i)}$ is the action space of agent i; $\mathcal{N}$ is the set of all agents and $|\mathcal{N}| = N$ denotes the number of all agents; $\mathsf{P} = \{p^{s,a} \in \Delta(\mathcal{S}) | s, a \in \mathcal{S} \times \mathcal{A}\}$ is the transition kernel; $\mathcal{G}$ is an undirected graph with node set $\mathcal{N}$ and edge set $\mathcal{E}$; $r = \{r^{(i)}\}_{i \in \mathcal{N}}$ is the set of reward functions and $r^{(i)}$ is the reward function for agent i; and γ is the discount factor. The weight matrix is denoted by G, specifically, the weight for the edge connecting nodes i and j is denoted by $G_{i,j}$, and is non-negative. For agent i, denote by $\mathcal{N}(i) = \{j | G_{i,j} \neq 0\}$ the neighbors of i.

Let $r_t^{(i)}$ denote the reward received by agent i at time t, and $\bar{r}_t = \frac{1}{N} \sum_{i=1}^{N} r_t^{(i)}$ denote the average reward over all the agents at time t. At each time t, each agent i chooses its action $a_t^{(i)}$ given s_t according to a local policy $\pi^{(i)}(a_t^{(i)}|s_t)$, which is a distribution over $\mathcal{A}^{(i)}$. Denote by $a_t = \{a_t^{(i)}\}_{i \in \mathcal{N}}$ the joint action. We define the joint policy of all agents as $\pi(a_t|s_t) = \prod_{i \in \mathcal{N}} \pi^{(i)}(a_t^{(i)}|s_t)$. Here, we focus on the decentralized setting where there is no fusion center, and two agents communicate with each other only if there is an edge connecting them. We follow the standard multi-agent RL model, e.g., [17], and assume that the state and the joint action are fully observable to each agent, but the reward can only be observed locally, i.e., $r_t^{(i)}$ is only observable to agent i.

In this paper, we focus on robust MARL with uncertain transition kernel. Specifically, the transition kernel P is not fixed, but lies in an uncertainty set $\mathcal{P}$, i.e., $\mathsf{P} \in \mathcal{P}$. Denote the transition kernel at time t by $\mathsf{P}_t \in \mathcal{P}$. Let $\tau = \{\mathsf{P}_t\}_{t \geq 0}$, which is referred to as the nature's policy (as in [8]). The collection of all possible τ is denoted by $\mathcal{T}$. We focus on the (s,a)-rectangular uncertainty set [7, 8], i.e., $\mathcal{P} = \bigotimes_{s,a} \mathcal{P}^{s,a}$, where $\mathcal{P}^{s,a} \subseteq \Delta(\mathcal{S})$.

We define the robust value function for a given joint policy π as:

$$V_\pi(s) = \min_{\tau \in \mathcal{T}} \mathbb{E}_\tau \left[\sum_{t=0}^{\infty} \gamma^t \bar{r}_t(s_t, a_t) \Big| s_0 = s, \pi \right], \quad (1)$$

where $\mathbb{E}_\tau$ denotes the expectation when the state transition is according to τ. Similarly, we can define robust Q-value function of the policy π as:

$$Q_\pi(s,a) = \min_{\tau \in \mathcal{T}} \mathbb{E}_\tau \left[\sum_{t=0}^{\infty} \gamma^t \bar{r}_t(s_t, a_t) \Big| s_0 = s, a_0 = a, \pi \right], \quad (2)$$

The goal is to maximize $Q_\pi(s,a)$ for any $s \in \mathcal{S}$ and any $a \in \mathcal{A}$:

$$\max_\pi Q_\pi(s,a), \forall s \in \mathcal{S} \text{ and } a \in \mathcal{A}. \quad (3)$$

We denote the solution to (3) by π^*, V_{π^*} by V^*, and Q_{π^*} by Q^*. We also have that $V^*(s) = \max_{a \in \mathcal{A}} Q^*(s,a)$.

We then present the following strong duality results and robust analog of the Bellman recursion in [25].

Theorem 1. *[25, Thm 1 (Robust Dynamic Programming)] The following strong duality condition holds for all $s \in \mathcal{S}$:*

$$\begin{aligned} &\max_\pi \min_\tau \mathbb{E}_\tau \left[\sum_{t=0}^{n} \gamma^t \bar{r}_t(s_t, a_t) \Big| s_0 = s, \pi \right] \\ &= \min_\tau \max_\pi \mathbb{E}_\tau \left[\sum_{t=0}^{n} \gamma^t \bar{r}_t(s_t, a_t) \Big| s_0 = s, \pi \right]. \end{aligned} \quad (4)$$

The optimal robust value function satisfies the following Bellman equation: $V^*(s) = \max_a \{\bar{r}(s,a) + \gamma \sigma_{\mathcal{P}^{s,a}}(V^*)\}$, where $\sigma_{\mathcal{P}^{s,a}}(V^*) = \min_{p(\cdot|s,a) \in \mathcal{P}^{s,a}} \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^*(s')]$. The optimal robust action-value function satisfies $Q^*(s,a) = \bar{r}(s,a) + \gamma \sigma_{\mathcal{P}^{s,a}}(V^*)$.

In this paper, we focus on the R-contamination uncertainty set. Specifically, for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, define the uncertainty set $\mathcal{P}^{s,a}$:

$$\mathcal{P}^{s,a} := \{(1-R)\hat{p}^{s,a} + Rq | q \in \Delta(\mathcal{S})\}, \quad (5)$$

where $\hat{p}^{s,a}$ denotes the centroid of the uncertainty set. In this paper, $\hat{p}^{s,a}$ is unknown, but samples from $\hat{p}^{s,a}$ can be obtained sequentially.

The R-contamination model was firstly introduced in [26] (named as ϵ-contamination), and has been widely used to model distributional uncertainty. The R-contamination set models the scenario where the state transition could be arbitrarily perturbed with a small probability R, hence is more suitable for systems suffering from random perturbations, adversarial attacks, and outliers in sampling. The R-contamination set can also be connected to uncertainty sets defined by total variation, KL-divergence and Hellinger distance via inequalities, e.g., Pinsker's inequality.

3. MULTI-AGENT ROBUST Q-LEARNING

In this section, we present the design and finite-sample analysis for our multi-agent robust Q-learning (MARQ) algorithm.

From the robust Bellman equation (Theorem 1), we have

$$Q^*(s,a) = \bar{r}(s,a) + \gamma \sigma_{\mathcal{P}^{s,a}}(V^*(s)), \tag{6}$$

Consider the R-contamination set in (5), the support function in (6) can be further written as

$$\begin{aligned} &\sigma_{\mathcal{P}^{s,a}}(V^*) \\ &= \min_{p^{s,a} \in \mathcal{P}^{s,a}} \mathbb{E}_{s' \sim p^{s,a}}[V^*(s')] \\ &= (1-R)\mathbb{E}_{s' \sim \hat{p}^{s,a}}[V^*(s')] + R \min_{s' \in \mathcal{S}} V^*(s') \\ &= (1-R)\mathbb{E}_{s' \sim \hat{p}^{s,a}}[V^*(s')] + R \min_{s' \in \mathcal{S}} \max_{a' \in \mathcal{A}} Q^*(s',a'). \end{aligned} \tag{7}$$

We will then develop a stochastic and decentralized algorithm based on the robust Bellman equation.

Note that in the decentralized setting, the reward is observable only locally, and each agent can only communicate with its neighbors since there is no fusion center. Moreover, in practice, agents may also want to keep their reward information private. To address this challenge, we generalize the idea of distributed optimization, and design our MARQ algorithm in Algorithm 1. Specifically, at each time t, agent i keeps its local copy of the Q-table $Q_t^{(i)}$ for $i \in \mathcal{N}$. The Q-table is firstly updated according to a stochastic version of the robust Bellman equation in (6) using only local reward information. Then, each agent collects local estimates of the Q-table from its neighbors, and compute the average, which is referred to as "average consensus".

Algorithm 1 Multi-Agent Robust Q-learning (MARQ)

Initialization: T, $Q_0 = \{Q_0^{(i)}\}_{i=1}^N$, π_b, s_0, $t = 0$, α_t

1: **for** $t \le T$ **do**
2: Each agent i takes action $a_t^{(i)} \sim \pi_b^{(i)}(\cdot|s_t)$, and receives reward $r_t^{(i)}$ for $i \in \mathcal{N}$
3: Each agent observes s_{t+1} and a_t,
4: **for** $i = 1, .., N$ **do**
5: $V_t^{(i)}(s) = \max_{a \in A} Q_t^{(i)}(s,a)$, for every $s \in S$
6: $\bar{Q}_{t+1}^{(i)}(s_t,a_t) = (1-\alpha_t)Q_t^{(i)}(s_t,a_t) + \alpha_t(r_t^{(i)} + \gamma R \min_{s \in S} V_t^{(i)}(s) + \gamma(1-R)V_t^{(i)}(s_{t+1}))$
7: **end for**
8: Each agent i sent $Q_t^{(i)}$ to its neighbors
9: $Q_{t+1}^{(i)} = \sum_{j \in \mathcal{N}(i)} G_{i,j} \bar{Q}_{t+1}^{(i)}$, for all $i \in \mathcal{N}$
10: **end for**

Output: Q_T

In the algorithm, Q_t is the collection of the Q-table estimates at all the agents, and thus is of the dimension $|\mathcal{S}||\mathcal{A}| \times N$, and π_b denotes the behavior policy.

In the following, we show that the estimate at each agent i converges almost surely to the optimal Q^*, i.e., $Q_T^{(i)} \to Q^*$ as $T \to \infty$. We will further characterize the finite-time error bound for the MARQ algorithm.

We first make three standard assumptions.

Assumption 1. *[17] The non-negative matrix G satisfies:*
a. G is a double stochastic matrix, i.e., $G\mathbf{1} = \mathbf{1}$ and $\mathbf{1}^\top G = \mathbf{1}^\top$, where $\mathbf{1}$ denotes an all-one vector with dimension N. Moreover, there exists constant $\eta \in (0,1)$ such that for any $G_{i,j} > 0$, $G_{i,j} \ge \eta$.
b. The weight $G_{i,j}$ of edge connecting nodes i, j is non-zero if and only if $(i,j) \in \mathcal{E}$.
c. Let $J = [\frac{1}{N}]^{N \times N}$. The largest eigenvalue of matrix $G - J$, denoted by $\lambda_{\max}$, is strictly less than 1.

Under Assumption 1, $\lim_{t \to \infty} G^t = J$ [22].

Assumption 2. *The learning rate α_t satisfies that:*
a. $\sum_t \alpha_t = \infty, \sum_t \alpha_t^2 < \infty$.
b. $\frac{\alpha_{t-t'}}{\alpha_t} \le K_c + K_a t'$, for any $t' < t$, where K_a and K_c are constants.

Assumption 3. *[27] [Bounded Reward] The reward $r^{(i)}(s,a)$ is bounded, $r^{(i)}(s,a) \le R_{\max}$, $\forall\, s \in \mathcal{S}, a \in \mathcal{A}$, $i \in \mathcal{N}$.*

We first show that our MARQ algorithm converges almost surely in the following theorem.

Theorem 2 (Asymptotic Convergence)**.** *Under Assumptions 1-3, $Q_T^{(i)} \to Q^*$ as $T \to \infty$ for any $i \in \mathcal{N}$ almost surely.*

The proof of this theorem follows from a novel generalization of the stochastic approximation convergence analysis to the decentralized setting.

In the following, we further characterize the finite-time error bound and sample complexity of our MARQ algorithm. We make the following assumption that is commonly used in reinforcement learning analysis [27, 28].

Assumption 4. *The Markov chain induced by the behavior policy π_b and transition kernel $\hat{p}$ is uniformly ergodic.*

Denote by μ_{π_b} the stationary distribution induced by the behavior policy π_b and the transition kernel $\hat{P}^{s,a}$. Then define

$$\mu_{\min} := \min_{s \in \mathcal{S}, a \in \mathcal{A}} \mu_{\pi_b}(s,a),$$

$$t_{\text{mix}} := \min \left\{ t \,\middle|\, \max_{s_0 \in \mathcal{S}, a_0 \in \mathcal{A}} d_{TV}(P^t(\cdot|s_0,a_0), \mu_{\pi_b}) \le \frac{1}{4} \right\}$$

where $P^t(\cdot|s_0,a_0)$ denotes the distribution of (s_t,a_t) conditioned on the initial state and action (s_0,a_0), and $d_{TV}(\mu,\nu)$ denotes the total variation distance between two probability measures μ and ν. Based the definition of t_{mix}, for any $t > t_{\text{mix}}$, the distribution of (s_t,a_t) is close to the stationary distribution μ_{π_b} with total variation distance less than $\frac{1}{4}$.

Theorem 3 (Finite-time Error Bound)**.** *Under Assumptions 1- 4, consider the MARQ algorithm in Algorithm 1. For any $0 < \epsilon < \min\left\{ \frac{1}{1-\gamma}, \frac{(1-R)^2(1-\lambda_{\max})\gamma \log\left(\frac{|\mathcal{S}||\mathcal{A}|}{\delta}\right)}{4c_1\sqrt{N}R_{\max}(1-\gamma)^2} \right\}$,*

$$||Q_T - Q^*||_\infty < 5\epsilon \tag{8}$$

with probability at least $1-6\delta$*, when*

$$T \geq \frac{c_0}{\mu_{min}} \left\{ \frac{1}{(1-\gamma)^5\epsilon^2} + \frac{t_{mix}}{1-\gamma} \right\} \log\left(\frac{|S||A|T}{\delta}\right) \cdot \log\left(\frac{1}{(1-\gamma)^2\epsilon}\right) + \frac{1}{\log(\lambda_{\max})} \log\left(\frac{\epsilon(1-\gamma)^2}{4\sqrt{N}R_{\max}\gamma}\right), \tag{9}$$

$$\alpha_t = \frac{c_1}{\log(\frac{|S||A|T}{\delta})} \min\left\{ \frac{(1-\gamma)^4\epsilon^2}{(1-R)^2\gamma^2}, \frac{1}{t_{mix}} \right\}, \tag{10}$$

where c_0 *and* c_1 *are some positive constants.*

On the right hand side of (8), 2ϵ is from the error of average consensus, and 3ϵ is from the error of Q-learning. Note that there are two terms in (9), where the first term is due to the stochastic error, and the second term is due to the average consensus error in the decentralized setting. The overall sample size of our MARQ algorithm is $\mathcal{O}(\frac{1}{(1-\gamma)^5\epsilon^2} + \frac{t_{\text{mix}}}{1-\gamma} + \log\frac{\sqrt{N}}{\epsilon(1-\gamma)})$, which matches with the single agent and centralized settings in [21] and [27] (within a constant factor and for a large range of N). It can also be observed that as N increases, more samples will be needed in order to make the average consensus error small.

Here we provide a proof sketch of Theorem 3 that highlights the major technical steps.

Proof Sketch. Recall that $Q_t = \{Q_t^{(i)}\}_{i\in\mathcal{N}}$ is $|\mathcal{S}||\mathcal{A}| \times N$ dimensional matrix, and Q^* is an $|\mathcal{S}||\mathcal{A}| \times 1$ dimensional vector defined in (3). Let $\langle Q_t \rangle = Q_t J$.

For any $|\mathcal{S}||\mathcal{A}| \times N$ dimensional matrix Q, let $Q(s,a)$ denote its (s,a)-th row. We then define the D-norm of Q as $||Q||_D = \max_{a\in\mathcal{A},\, s\in\mathcal{S}} ||Q(s,a)||_2$. It can be shown that $||\cdot||_D$ is a norm, and is upper bounded by the infinity norm.

We will show that $Q_t \to Q^*\mathbf{1}^\top$, i.e., $Q_t^{(i)} \to Q^*$ for any $i \in \mathcal{N}$ as $t \to \infty$. By the triangle inequality,

$$||Q_t - Q^*\mathbf{1}^\top||_D \leq \underbrace{||Q_t - \langle Q_t\rangle||_D}_{\text{part I}} + \underbrace{||\langle Q_t\rangle - \langle Q^*\mathbf{1}^\top\rangle||_D}_{\text{part II}},$$

where part II follows from the fact that $\langle Q^*\mathbf{1}^\top\rangle = Q^*\mathbf{1}^\top J = Q^*\mathbf{1}^\top$. Note that part I in (11) is the error of one-step average consensus, and part II in (11) is the error in Q-learning.

It can be shown that

$$\text{part I} \leq \lambda_{\max}^t ||Q_{\perp,0}||_D + \frac{2\lambda_{\max}\alpha_t\sqrt{N}\frac{R_{\max}}{1-\gamma}}{1-\lambda_{\max}}. \tag{11}$$

Note that $\alpha_t \sim \mathcal{O}(\epsilon^2)$, and thus the second term in (11) will is than $\frac{\epsilon(1-\gamma)}{2\gamma}$ when ϵ is small. To guarantee that first term in (11) is less than $\frac{\epsilon(1-\gamma)}{2\gamma}$, we need $t \geq \frac{1}{\log(\lambda_{\max})} \log\left(\frac{\epsilon(1-\gamma)^2}{4\sqrt{N}R_{\max}\gamma}\right) \triangleq t_I$.

We then bound part II. Let $\Delta_{t+1} := \langle Q_{t+1}\rangle - Q^*\mathbf{1}^\top$, then part II$= ||\Delta_{t+1}||_D$. Let $\Lambda_t \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|}$ be a diagonal matrix:

$$\Lambda_t((s,a),(s,a)) = \begin{cases} \alpha_t, & \text{if } (s,a) = (s_t,a_t), \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Denote by $\lambda_{\text{MAX}}(\Lambda)$ the largest eigenvalue of any matrix Λ. Set $P_t \in \{0,1\}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}|}$ such that $P_t(s_t,a_t,s_{t+1}) = 1$, and otherwise $P_t(s,a,s') = 0$. Then, we can show that

$$\begin{aligned} ||\Delta_{t+1}||_D \leq\ & \lambda_{\text{MAX}}(\Pi_{j=t_I}^t(I-\Lambda_j))||\Delta_{t_I}||_D \\ & + \gamma\lambda_{\text{MAX}}\left(\sum_{i=t_I}^t ||\Delta_i||_D \Pi_{j=i+1}^t (I-\Lambda_J)\Lambda_i\right) \\ & + \gamma(1-R)\left\|\sum_{i=t_I}^t \Pi_{j=i+1}^t (I-\Lambda_j)\Lambda_i(P_i-\hat{p})V^*\right\|_D \\ & + \gamma\epsilon\lambda_{\text{MAX}}\left(\sum_{i=t_I}^t \Pi_{j=i+1}^t (I-\Lambda_j)\Lambda_i\right), \end{aligned} \tag{13}$$

where the first three terms are from the Q-learning stochastic error, and the last term is due to the error of average consensus. If we choose

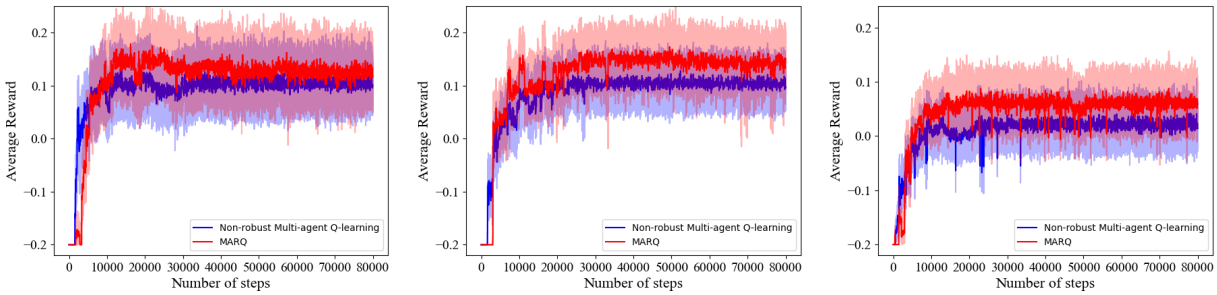
$$t - t_{\text{I}} \geq \frac{c_0}{\mu_{\min}} \left\{ \frac{1}{(1-\gamma)^5\epsilon^2} + \frac{t_{\text{mix}}}{1-\gamma} \right\} \log\left(\frac{|S||A|T}{\delta}\right) \cdot \log\left(\frac{1}{(1-\gamma)^2\epsilon}\right), \tag{14}$$

then (13) can be bounded by 4ϵ. □

4. NUMERICAL RESULT

In this section, we compare our robust multi-agent Q-learning algorithm with the non-robust (vanilla) multi-agent Q-learning algorithm. We consider a multi-agent MDP with $N = 5$ agents and $|\mathcal{S}| = 24$ states. The action space for each agent is $\mathcal{A}^{(i)} = \{0,1\}$, and thus the size of the joint action space is $|\mathcal{A}| = 32$. We simulate our algorithm under a 23-point game, where the state space is $\mathcal{S} = \{0,1,2,\ldots,23\}$. Then, we design an action mapping matrix, which maps each joint action a to a number $n(a)$. Then, given the current state s, the transition kernel $\hat{p}(\cdot|s,a)$ is that the next state is $s' = s + a$ if $s + n(a) \leq 23$ and $s' = 0$ if $s + n(a) > 23$. When the next state is 23, agents $2, 3, 4$ will get rewards 1, 4, 5, respectively. When the next state is larger than 15, and $n(a)$ is larger than 1, then agent 1 will get reward 0.8. At each step, each agent receives reward -0.2.

We compare our MARQ with the vanilla non-robust multi-agent Q-learning. Here, the vanilla non-robust multi-agent Q-learning algorithm is Algorithm 1 with $R = 0$. We train our MARQ and non-robust multi-agent Q-learning algorithm in the training environment specified above, and then



(a) R=0.2 p=0.2 (b) R=0.3 p=0.2 (c) R=0.3 p=0.3

Fig. 1: MARQ v.s. Non-robust Decentralized Multi-agent Q-learning.

evaluate the obtained policies in a perturbed environment. Here the perturbed environment is designed as follows. At state s if a joint action a is taken, then the system transits according to the transition kernel $\hat{p}(\cdot|s,a)$ with probability $1-p$, and transits to the worst-case state $\arg\min_s V^*(s)$ with probability p. The behavior policy π_b is a uniform distribution over the joint action space $\mathcal{A}$. Once the algorithm stops, each agent obtains its own policy by taking the greedy action with respect to its local estimate of the Q-function.

We evaluate the performance every 40 steps. In Figure 1, we plot the average over 100 test episodes per evaluation step. Moreover, we plot the upper and lower envelops of the shaded which correspond to 10 and 90 percentiles of the 100 test episodes. It can be seen that our MARQ algorithm achieves a higher reward than the vanilla one on the perturbed environment, and hence is robust to distributional uncertainty and adversarial perturbations. It can also be seen that when the perturbation parameters R, p are small(i.e., the model mismatch is small), our algorithm performs similarly to the non-robust one; and when the parameters are larger, our MARQ algorithm performs much better.

5. CONCLUSION

In this paper, we design an efficient MARQ algorithm for robust multi-agent decentralized RL with uncertainty transition kernel. We theoretically proved its convergence and provided its finite-time error bound. Our approach can be extended to make SARSA and other RL algorithms robust. Our future interest is to generalize our idea, and combine with the deep Q-learning approach and double Q-learning approach to solve robust RL problems with large or continuous state/action spaces. It is also of interest to generalize the robust policy gradient approach [29] to the decentralized multi-agents setting. Other type of uncertainty sets, e.g., KL-divergence and Wasserstein distance, are also of interest.

6. ACKNOWLEDGEMENT

The work of Yudan Wang, Yue Wang and Shaofeng Zou was supported in part by the National Science Foundation under Grants CCF-2106560 and CCF- 2007783. Yi Zhou's work was supported in part by U.S. National Science Foundation under the grants CCF-2106216 and DMS-2134223.

7. REFERENCES

[1] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

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

[3] Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel, "Multi-agent reinforcement learning in sequential social dilemmas," in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, 2017, pp. 464–473.

[4] Shiyong Wang, Jiafu Wan, Daqiang Zhang, Di Li, and Chunhua Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," *Computer Networks*, vol. 101, pp. 158–168, 2016.

[5] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, Massachusetts, 2018.

[6] J Andrew Bagnell, Andrew Y Ng, and Jeff G Schneider, "Solving uncertain Markov decision processes," 09 2001.

[7] Arnab Nilim and Laurent El Ghaoui, "Robustness in Markov decision problems with uncertain transition matrices," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2004, pp. 839–846.

[8] Garud N Iyengar, "Robust dynamic programming," *Mathematics of Operations Research*, vol. 30, no. 2, pp. 257–280, 2005.

[9] Aurko Roy, Huan Xu, and Sebastian Pokutta, "Reinforcement learning under model mismatch," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 3046–3055.

[10] Kishan Panaganti Badrinath and Dileep Kalathil, "Robust reinforcement learning using least squares policy iteration with provable performance guarantees," in *Proc. International Conference on Machine Learning (ICML)*. PMLR, 2021, pp. 511–520.

[11] Eugene Vinitsky, Yuqing Du, Kanaad Parvate, Kathy Jang, Pieter Abbeel, and Alexandre Bayen, "Robust reinforcement learning using adversarial populations," *arXiv preprint arXiv:2008.01825*, 2020.

[12] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta, "Robust adversarial reinforcement learning," in *Proc. International Conference on Machine Learning (ICML)*. PMLR, 2017, pp. 2817–2826.

[13] Linfang Hou, Liang Pang, Xin Hong, Yanyan Lan, Zhiming Ma, and Dawei Yin, "Robust reinforcement learning with wasserstein constraint," *arXiv preprint arXiv:2006.00945*, 2020.

[14] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun, "Tactics of adversarial attack on deep reinforcement learning agents," in *Proc. International Joint Conferences on Artificial Intelligence (IJCAI)*, 2017, pp. 3756–3762.

[15] Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary, "Robust deep reinforcement learning with adversarial attacks," in *Proc. International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 2040–2042.

[16] Kaiqing Zhang, Tao Sun, Yunzhe Tao, Sahika Genc, Sunil Mallya, and Tamer Basar, "Robust multi-agent reinforcement learning with model uncertainty," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020, vol. 33.

[17] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar, "Fully decentralized multi-agent reinforcement learning with networked agents," in *Proc. International Conference on Machine Learning (ICML)*. PMLR, 2018, pp. 5872–5881.

[18] Qinghua Liu, Yuanhao Wang, and Chi Jin, "Learning markov games with adversarial opponents: Efficient algorithms and fundamental limits," *arXiv preprint arXiv:2203.06803*, 2022.

[19] 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.

[20] Michael L Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

[21] Yue Wang and Shaofeng Zou, "Online robust reinforcement learning with model uncertainty," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[22] Lin Xiao, Stephen Boyd, and Seung-Jean Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007.

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[24] Hado Van Hasselt, Arthur Guez, and David Silver, "Deep reinforcement learning with double Q-learning," in *Proc. the AAAI conference on artificial intelligence*, 2016, vol. 30.

[25] Arnab Nilim and Laurent Ghaoui, "Robustness in markov decision problems with uncertain transition matrices," *Advances in Neural Information Processing Systems (NIPS)*, vol. 16, 2003.

[26] P. J. Huber, "A robust version of the probability ratio test," *Ann. Math. Statist.*, vol. 36, pp. 1753–1758, 1965.

[27] Gen Li, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen, "Sample complexity of asynchronous Q-learning: Sharper analysis and variance reduction," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[28] Shaofeng Zou, Tengyu Xu, and Yingbin Liang, "Finite-sample analysis for SARSA with linear function approximation," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 8665–8675.

[29] Yue Wang and Shaofeng Zou, "Policy gradient method for robust reinforcement learning," *arXiv preprint arXiv:2205.07344*, 2022.