On the Hardness of Decentralized Multi-Agent Policy Evaluation under Byzantine Attacks

Hairi[†] Minghong Fang[¶] Zifan Zhang[‡] Alvaro Velasquez[*] Jia Liu[§]
[†]Dept. of Computer Science, University of Wisconsin-Whitewater
[¶]Dept. of Computer Science and Engineering, University of Louisville
[‡]Dept. of Computer Science, North Carolina State University
[*]Dept. of Computer Science, University of Colorado, Boulder
[§]Dept. of Electrical and Computer Engineering, The Ohio State University

***Abstract*—In this paper, we study a fully-decentralized multi-agent policy evaluation problem, which is an important subproblem in cooperative multi-agent reinforcement learning, in the presence of up to f faulty agents. In particular, we focus on the so-called Byzantine faulty model with model poisoning setting. In general, policy evaluation is to evaluate the value function of any given policy. In cooperative multi-agent system, the system-wide rewards are usually modeled as the uniform average of rewards from all agents. We investigate the multi-agent policy evaluation problem in the presence of Byzantine agents, particularly in the setting of heterogeneous local rewards. Ideally, the goal of the agents is to evaluate the accumulated system-wide rewards, which are uniform average of rewards of the normal agents for a given policy. It means that all agents agree upon common values (the consensus part) and furthermore, the consensus values are the value functions (the convergence part). However, we prove that this goal is not achievable. Instead, we consider a relaxed version of the problem, where the goal of the agents is to evaluate accumulated system-wide reward, which is an appropriately weighted average reward of the normal agents. We further prove that there is no correct algorithm that can guarantee that the total number of positive weights exceeds $|\mathcal{N}| - f$, where $|\mathcal{N}|$ is the number of normal agents. Towards the end, we propose a Byzantine-tolerant decentralized temporal difference algorithm that can guarantee asymptotic consensus under scalar function approximation. We then empirically test the effective of the proposed algorithm.**

***Index Terms*—Multi-agent policy evaluation, Byzantine attack, Temporal difference learning**

I. Introduction

Reinforcement learning (RL) [34] is a powerful paradigm in learning sequential decision-making. The success of RL both in theory [1], [19], [29], [33], [35], [36], [43] and practice [14], [26], [28] has sparked the interest in the realm of multi-agent reinforcement learning (MARL) [20], [45], [46]. MARL [27], [46] is a multi-agent setting, a natural extension of single-agent RL, where agents interact within a common environment. The state dynamics and individual rewards are affected by both the global state and joint actions. Based on the system objective, there are in general two main categories of MARL problems, cooperative [46] and competitive [27] settings. Based on the assumption of the system infrastructure, there are also two categories, centralized setting and fully decentralized setting respectively. More specifically, in a fully-decentralized multi-agent setting, agents are only able to share information with each other through a communication network instead of a central server. In contrast, in a server-present centralized system, the server can collect and aggregate local information and disseminate appropriate information to agents (see an excellent survey [45] of MARL topics for further details). The focus of this paper is the cooperative and decentralized setting as in [46], where all agents work together to maximize a common goal.

Similar to the single-agent RL setting, a complete MARL algorithm searches for a certain optimal policy π^* that can maximize accumulated system-wide average reward, i.e.,

$$\pi^* = \arg\max_{\pi} \mathbb{E}\bigg[\sum_{t=0}^{\infty}\gamma^t \sum_{i=1}^{n}\frac{1}{n}r_{t+1}^i\bigg],$$

where n is the number of agents in the system, γ is a discount factor with $0 < \gamma < 1$ and the expectation is subject to the usual caveats about appropriate expectations existing in steady-state. We note that in a cooperative multi-agent system, the system-wide reward is typically modeled as the uniform average of all agents. An important subproblem is to study the multi-agent policy evaluation for a given policy π, as this can be incorporated into the actor-critic framework as the critic step. The goal of all agents, in this subproblem, is to learn the value functions defined as:

$$V(s) = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\gamma^t \sum_{i=1}^{n}\frac{1}{n}r^i(s_t,a_t)|s_0 = s\bigg],$$

for all states $s \in \mathcal{S}$. This implies that 1) all agents need to reach consensus and 2) the consensus values are the value functions defined above. The multi-agent policy evaluation problem has been studied extensively in fault-free setting [7]–[9], [17], [46].

We study a fully decentralized multi-agent policy evaluation problem in the presence of Byzantine agents. In addition, we consider a multi-agent system where up to $f > 0$ agents are Byzantine. Specifically, we explore the model poisoning faulty setting described in [6], [11], [32], where Byzantine agents could send arbitrary or carefully crafted information to their neighboring agents. In a fully decentralized system, it is typical

Hairi and Minghong Fang are co-primary authors.

ISBN 978-3-903176-65-2 © 2024 IFIP

for agents to share certain system parameters in order to achieve consensus as described above. However, Byzantine agents have the ability to modify these local parameters to arbitrary values, thereby disrupting the algorithm. Furthermore, it is important to highlight that in a fully decentralized system, a Byzantine agent can transmit inconsistent information to its neighbors. This means that a Byzantine agent can send different values to different neighbors. This presents a significant challenge compared to the centralized server-based setting, where a Byzantine agent can only send a single piece of information to the server. The existing literature in multi-agent reinforcement learning (MARL) lacks a comprehensive study on robust designs, particularly in heterogeneous settings that consider these challenges.

In this work, we investigate the multi-agent policy evaluation problem in the presence of Byzantine agents for any given policy π. Ideally, the goal of the agents is to evaluate the accumulated uniform average reward of the normal agents. Specifically, let $\mathcal{N}$ denotes the set of normal agents in the system, the decentralized multi-agent policy evaluation is to characterize the following value at any states s for the given policy π:

$$V(s) = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\gamma^t \sum_{i\in\mathcal{N}} \frac{1}{|\mathcal{N}|} r^i(s_t,a_t)|s_0 = s\bigg]. \tag{1}$$

However, we will prove later in Theorem 1 that evaluating Eq. (1) *cannot* be achieved. Thus, we consider a *relaxed* version of the multi-agent policy evaluation problem. In this relaxed problem, the goal of the agents is to evaluate accumulated weighted average reward, which can be written as:

$$V(s) = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\gamma^t \sum_{i\in\mathcal{N}} \alpha_i r^i(s_t,a_t)|s_0 = s\bigg], \tag{2}$$

where $\alpha_i \geq 0$ for all $i \in \mathcal{N}$ and $\sum_{i\in\mathcal{N}} \alpha_i = 1$. We further prove that for the case $f > 0$, there is *no* correct algorithm that can evaluate Eq. (2) with $\sum_{i\in\mathcal{N}} \mathbf{1}\{\alpha_i > 0\} > |\mathcal{N}| - f$, where $|\mathcal{N}|$ and f are number of normal and the maximum number of Byzantine agents. In other words, achieving more than $|\mathcal{N}| - f$ positive weights in the relaxed problem is impossible in general. In the end, we propose a Byzantine-tolerant decentralized temporal difference (BDTD) algorithm under linear scalar function approximation that can guarantee that all normal agents reach consensus.

The contributions of this paper are threefolds:

- First, we prove in Theorem 1 that evaluating the exact value functions defined by the uniform average reward of the agents in the presence of Byzantine is impossible. In other words, there is no correct algorithm that can achieve the value functions where system-wide rewards are modeled as the uniform average rewards of all normal agents in the presence of Byzantine agents. We further relax the problem to consider solving value function where system-wide rewards are modeled as appropriately weighted average rewards of the normal agents.
- Second, we further prove in Theorem 2 that there is no correct algorithm that can guarantee the number of positive weights exceeds $|\mathcal{N}| - f$ for the aforementioned relaxed problem.
- Last but not least, we propose a decentralized multi-agent policy evaluation algorithm with linear scalar function approximation, so that all normal agents can reach consensus.

II. Related work

A. Fault-free policy evaluation

Policy evaluation, which aims to evaluate how good a given policy is, is an important sub-problem in designing a complete RL algorithm, which can be incorporated into the actor-critic framework as the critic step. Temporal difference (TD) learning [33] is a simple yet effective learning algorithm first proposed in the single-agent setting to evaluate a given policy. The convergence theory in TD learning has been developed first in asymptotic regime [36], [37] and then in finite-time horizon [2], [29], [43].

The multi-agent policy evaluation, based on distributed TD learning, has been recently studied [8], [9], [41]. Various aspects of fully-decentralized MARL algorithms have been studied. Notably, the sample and communication efficiencies of actor-critic algorithms have been investigated in [7], [16], [17], [23].

B. Distributed Learning with Byzantine Agents

Byzantine agents with local model poisoning attack is a common modeling for robust design of distributed algorithm design. A large body of papers [3], [6], [11], [12], [18], [21], [22], [42], [44] in the literature have adopted it as a common failure model in federated learning problem, where a server is involved to facilitate the collaborative learning process within the supervised setting. In robust algorithm design, one feature that is different from fault-free counterpart is to design robust filtering mechanism. For instance, in the Krum aggregation rule, as described by [3], the server receives local models from agents and selects one received local model that has the smallest distance to its subset of neighbors as the output. In [4], a key system assumption is that the server holds a trusted dataset. The server maintains a server model based on the current global model and its trusted dataset. Upon receiving one local model from any agent, the server considers this received local model as benign if it is positive related to the server model. Recently work in [5], [10] studied the effect of Byzantine agents in the so-called federated reinforcement learning (FRL) framework, where a central server is assumed to be present. However, we note that FRL and MARL differ significantly in that FRL is a multiple independent identical learner and the action from one agent does not affect the outcomes of other agents. In contrast, the global state transition and local rewards are dependent upon joint actions in MARL. In [5], the results are further extended the results to the offline setting. The closest related work [41] studied the policy value evaluation in the presence of Byzantine agents for a given policy. However, the analysis implicitly assumes the setting of homogeneous rewards, i.e. the rewards for all agents are the same. In our

work, we consider a more general heterogeneous reward setting. The offline competitive MARL has been studied in [40], where the data poisoning fault model is considered. Specifically, the rewards in the offline data are adversarially changed so that the new Nash equilibrium learned from the poisoned data is significantly different from the Nash equilibrium learned from the original data.

There are a series of works [30]–[32] on decentralized optimization problems where the local objective functions are heterogeneous and convex. An important subproblem in both our work and work in decentralized optimization [30]–[32] is decentralized consensus, meaning all agents are required to agree with each other. Existing work in [38], [39] have focused on these fundamental problems and proposed f-trimmed-mean-based algorithms. A recent paper [13] has investigated on the topic of Byzantine-robust decentralized federated learning.

III. Byzantine Policy Evaluation in Multi-Agent Reinforcement Learning

Throughout this paper, $\|\cdot\|$ denotes the ℓ_2-norm for vectors and the ℓ_2-induced norm for matrices. $|\cdot|$ denotes cardinality of a set/multi-set or the absolute value of a scalar. $(\cdot)^T$ denotes the transpose for a matrix or a vector.

1) System model: Consider a multi-agent system with n agents, including up to f agents to be Byzantine agents. We denote the set of Byzantine agents as $\mathcal{F}$. Note that the actual number of Byzantine agents in the system can be smaller than f. We consider the scenario that all n agents are connected through a complete graph, where each edge serves as a communication channel that allows agents to send information to their neighbors. Later, we will show that our impossibility results hold even for this most ideal setting.

Definition 1 (Networked Multi-Agent MDP)**.** Let the communication network be a complete graph. A networked multi-agent MDP is defined by following tuple $(\mathcal{S}, \{\mathcal{A}^i\}_{i=1}^n, P, \{r^i\}_{i=1}^n, \gamma)$, where $\mathcal{S}$ is the global state space observed by all agents, $\mathcal{A}^i$ is the action set for agent i, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is a global state transition kernel, $r^i : \mathcal{S} \times \mathcal{A}$ is the local reward function for agent i, and $\gamma \in (0, 1)$ is the discount factor. Let $\mathcal{A} = \prod_{i\in\mathcal{N}} \mathcal{A}^i$ be the joint action set of all agents.

In this paper, we assume that the global state space $\mathcal{S}$ is finite. We also assume that at any given time $t \geq 0$, all agents can observe the current global state s_t. $r^i(s, a)$ is individual agent i's reward given global state s and joint action a. For simplicity of the presentation, we assume that the rewards are deterministic. Even in this simple setting, we will show that our impossibility results hold, let alone for more general stochastic reward settings. We consider policies that are stationary. In our MARL system, each agent chooses its action following its local policy π^i that is conditioned on the current global state s, i.e., $\pi^i(a^i|s)$ is the probability for agent i to choose an action $a^i \in \mathcal{A}^i$. Then, the joint policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ can be written as $\pi(a|s) = \prod_{i\in\mathcal{N}} \pi^i(a^i|s)$. For any given policy π, the global value function for all $s \in \mathcal{S}$ is defined as follows: $V(s) = \mathbb{E}_{s\sim d_\pi, a\sim\pi(\cdot|s)}[\sum_{t=0}^{\infty} \frac{\gamma^t}{N} \sum_{i\in\mathcal{N}} r^i(s_t, a_t)|s_0 = s]$, where $d_\pi(\cdot)$ is the steady state distribution induced by π. The existence of such distribution is guaranteed by the Assumption 1.

Definition 2 (Byzantine Networked Multi-Agent MDP)**.** A Byzantine networked multi-agent MDP is a networked multi-agent MDP as defined in Definition 1 with up to f Byzantine agents, who may send arbitrary information when sharing to the neighboring agents.

We note that in the modelling of the Byzantine agents, the agents still strictly follow the sampling policies and receives true data from the environment. However, the Byzantine behavior appears in the communication process with neighboring agents when sending value function information. One can see such modelling in Algorithm 1.

2) Technical assumptions: We now state the following assumptions for the decentralized multi-agent MDP described above.

Assumption 1. For any policy π, the induced Markov chain $\{s_t\}_{t\geq 0}$ is irreducible and aperiodic.

Assumption 2. The reward r^i_{t+1} is uniformly bounded by a constant $r_{\max} > 0$, $\forall i \in [n]$ and $t \geq 0$.

Assumption 3. Each agent i's value function is parameterized by linear functions, i.e., $V(s; w) = \phi(s)w$, where $\phi(s) \in \mathbb{R}^d$ is a feature vector for state $s \in \mathcal{S}$. The feature matrix $\Phi \in \mathbb{R}^{|\mathcal{S}|\times d}$ is a full-rank matrix. The feature vectors $\phi(s)$ are uniformly bounded for any $s \in \mathcal{S}$. Without loss of generality, we assume that $\|\phi(s)\| \leq 1$.

Assumption 4. The total number of agents n and the maximum number of Byzantine agents f has the following inequality $n \geq 3f + 1$.

Assumption 1 guarantees that there exists a stationary distribution $d_\pi(\cdot)$ over $\mathcal{S}$ for the Markov chain induced by the given policy π. Assumption 2 is common in the RL literature (see, e.g., [8], [43], [46]) and easy to be satisfied in many practical MDP models with finite state and action spaces. Assumption 3 on features is standard and has been widely adopted in the literature, e.g., [29], [36], [46], for linear function approximations. Assumption 4 is a standard assumption in decentralized Byzantine consensus problem as in [38].

IV. General results in Byzantine faulty multi-agent policy evaluation

In this section, we start with the scope of the problems that we consider to facilitate the later discussions on our impossibility results. First, we introduce the Byzantine-free multi-agent policy evaluation problem [7], [8], [17], [46]. Note that the stochastic convergence we are referring to in this paper is *expected mean-squared convergence* as in Byzantine-free setting [7], [8], [17], [29]. **1) Byzantine-free multi-agent policy evaluation problem:**

Problem 1. *In decentralized TD learning, if all agents function normally, is there a correct distributed TD learn-*

ing algorithm converges to a TD fixed point that satisfies: $w^* = \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)\sum_{i=1}^n \frac{1}{n} r^i(s,a)]$*?*

Note that the above convergence requires that each normal agent i satisfy $\lim_{t\to\infty}\mathbb{E}[\|w_t^i - w^*\|^2] = 0$. This implies two important details. First, it signifies that all agents' parameters achieve consensus, meaning that they will all have the same values. Second, in addition to reaching a consensus, the agreed-upon value is w^*, which is referred to as the TD-fixed point. We further note that from the perspective of the actor-critic framework in decentralized MARL, consensus on certain global information like value function is essential in computing local policy gradients [7], [17], [46].

2) Byzantine faulty multi-agent policy evaluation problems: With the presence of Byzantine agents, since we consider model poisoning Byzantine attack, it is clearly impossible to guarantee Byzantine agents to converge to the aforementioned Byzantine-free TD-fixed point w^*, defined in Problem 1. A natural goal is to consider if there exist correct algorithms such that the parameters converge to the fixed point corresponding to normal agents, which is formally stated as follows.

Problem 2. *When $f > 0$, is there a correct TD learning algorithm that allows the agents to converge to $w_{\mathcal{N}}^* = \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)\sum_{i\in\mathcal{N}}\frac{1}{|\mathcal{N}|}r^i(s,a)]$, where $\mathcal{N}$ denotes the set of normal agents?*

The fixed point $w_{\mathcal{N}}^*$ proposed corresponds to modelling the system rewards as the uniform average of all normal agents. However, as we will prove in Theorem 1, it is impossible to reach the TD-fixed point defined in Problem 2. Thus, we further relax the problem to consider a TD fixed point that is an appropriately weighted average of all normal agents.

Problem 3. *When $f > 0$, is there a correct TD learning algorithm that allows the agents to converge to $w_\alpha^* = \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)\sum_{i\in\mathcal{N}}\alpha_i r^i(s,a)]$, where the weights α_i satisfies: $\sum_{i\in\mathcal{N}}\alpha_i = 1,\ \alpha_i \geq 0, \forall i \in \mathcal{N}$?*

The fixed point w_α^* proposed corresponds to modelling the system rewards as a non-uniform weighted average of all normal agents. In Theorem 2, we will answer this question formally. In general, there is no correct algorithm that can guarantee the number of positive weights exceeds $|\mathcal{N}| - f$. In other words, in some multi-agent policy evaluation problems, achieving $|\mathcal{N}| - f$ number of positive weights is the best an algorithm can do. Toward this end, we introduce a (ν,ξ)-admissible problem.

Problem 4. *((ν,ξ)-admissible problem) When $f > 0$, for given pair of $\nu \in \mathbb{N}^+$ and $\xi > 0$, is there a correct TD learning algorithm that allows the agents to converge to*

$$w_{\nu,\xi}^* = \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)\sum_{i\in\mathcal{N}}\alpha_i r^i(s,a)]. \tag{3}$$

where the weights α_i satisfies $\sum_{i\in\mathcal{N}}\alpha_i = 1,\ \alpha_i \geq 0,\ \forall i \in \mathcal{N}$, $\sum_{i\in\mathcal{N}}\mathbf{1}(\alpha_i \geq \xi) \geq \nu$?

Problem 4 is to learn the value functions with at least ν positive weights, which are bounded away from zero by at least ξ. It is easy to see that when $\xi = 0$ and $\nu = 1$, Problem 4 reduces to Problem 3.

3) Main theoretical results: The following theorems state that, in the presence of Byzantine agents, no algorithm ensures that the normal agents' parameters converge to a fixed point in Problem 2.

Theorem 1. *When $f > 0$, Problem 2 is not solvable.*

Theorem 2. *For any $\xi > 0$, Problem 4 is not solvable for any $\nu > |\mathcal{N}| - f$.*

Theorem 2 says that a $(|\mathcal{N}| - f, \xi)$ admissible solution is the best one can achieve for some $\xi > 0$.

We remark that even though the proofs for above two theorems are inspired by [32], there are two significant differences in the proofs and implications. First, our proof is convergence for stochastic terms whereas in [32], the proof is for deterministic terms. Secondly, the impossibility results hold for general multi-agent policy evaluation problem, including tabular case and linear approximations, whereas in [32], the impossibility result is just for scalar case.

We also remark that the impossibility results holds for general graph, not just limited to complete graph, in decentralized multi-agent settings as well where the proof will be the same. The reason that we assumed a complete graph in the beginning is to design the algorithm in Section V.

V. Byzantine-tolerant Decentralized Temporal Difference Learning

In this section, we provide a Byzantine-tolerant decentralized TD (BDTD) learning algorithm for normal agents to solve MARL policy evaluation in the sense of Theorem 2. In order to derive such an algorithm, we further assume in Assumption 3, the dimension $d = 1$, i.e. the features are reduced to scalar features.

1) Behavior of Byzantine agents: The behaviors of Byzantine agents are described in Algorithm 1. The parameters sent by the Byzantine agents can be arbitrary (denoted as $*$). We note that Byzantine agents can only poison the local models of their own, which are the information to be exchanged with their neighbors. This is referred to as local model poisoning [11]. We do not consider the data poisoning models, where Byzantine agents may change the data which may include local policies and local actions (global state as a result).

On the other hand, in a decentralized multi-agent setting, a Byzantine agent can send inconsistent parameters to its neighbors, which means that a Byzantine agent can send one parameter to one neighbor and a distinct parameter to another neighboring agent. There is a more restricted Byzantine model called Byzantine broadcast model [30], where a Byzantine agent sends the same parameter to neighboring agents. Here, in our work, we focus on the more general setting where Byzantine agents may send inconsistent parameters.

2) f-Trimmed mean subroutine: We will define f-trimmed mean, which is a subroutine we used for parameters.

[1]The arbitrary value * can be different to neighbors.

Algorithm 1: Byzantine Agent's Behavior.

Input : initial state s_0, given policy π, state features ϕ, step-size η_k, initial parameters $\{w_0^i\}_{i\in\mathcal{V}}$.

for $k = 0, 1, \cdots$ **do**
  for *all* $i \in \mathcal{F}$ **do**
    Execute action $a_k^i \sim \pi^i(\cdot|s_k)$;
    Observe the state s_{k+1} and reward r_{k+1}^i;
  end
  Send $*$ to neighbors[1] and receive values from neighbors;
end

Definition 3 (f-Trimmed Mean [44])**.** For any multi-set[2] $\{x^1, \cdots, x^n\}$, where $x^i \in \mathbb{R}$ for all i, sort the n values in ascending order (break the tie uniformly random), then remove the largest f and smallest f, respectively. For the remaining $n - 2f$ values, return the average value.

3) Policy evaluation for normal agents: Algorithm 2 describes the decentralized multi-agent policy evaluation algorithm for normal agents. For any given policy π, the algorithm learns the value function parameters using decentralized TD learning.[3] We note that for normal agents $i \in \mathcal{N}$, it is only required to know its own local policy π^i.

In Line 9, we have used projected TD learning, a variant of TD learning introduced in [2]. A choice for such radius R in our scalar case is $R = \frac{2r_{\max}}{\phi_{\min}(1-\gamma)^{3/2}}$, where $\phi_{\min} := \min_{s\in\mathcal{S}} |\phi(s)|$ (see [2, Lemma 7] for vector case). This projection step is mainly for theoretical analysis for bounding TD error terms. In practice, such a projection step may be dropped. The step sizes η_t used in Line 8 of Algorithm 2 are diminishing. The step sizes are known to all agents as priori and satisfy the standard conditions: $\sum_{t=1}^{\infty} \eta_t = \infty$ and $\sum_{t=1}^{\infty} \eta_t^2 < \infty$. A typical step size choice is $\eta_t = \frac{1}{t}$ for $t \geq 0$.

4) Main theoretical results for Algorithm 2 Let $\bar{w}_t = \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}} w_t^i$, i.e. the average of the parameters of normal agents at iteration $t \geq 0$. Then, we have the following consensus result that states parameters of all normal agents will converge to the average asymptotically.

Theorem 3. *The sequences generated in normal agents by Algorithm 2 will achieve consensus, i.e. for any $i \in \mathcal{N}$, we have* $\lim_{t\to\infty} |w_t^i - \bar{w}_t| = 0$.

Theorem 3 ensures that even starting with different initial parameters, in the heterogeneous reward setting and, more importantly, inconsistent Byzantine faulty model, the parameters among normal agents will eventually reach an agreement. However, the average parameter $\bar{w}_t$ itself may not have a limit depending on the heterogeneity of the problem.

[2]A multi-set allows the elements in it to be the same.

[3]For simplicity, we used TD(0) instead of TD(λ). The extension to TD(λ) where $\lambda \in (0, 1]$ is straightforward.

Algorithm 2: Byzantine-Tolerant Decentralized TD (BDTD) Learning for Normal Agents.

Input : initial state s_0, given policy π, state features ϕ, step-size η_k

1 **for** $k = 0, 1, \cdots$ **do**
2 Send parameter w_k^i to neighbors and receive values from neighbors
3 Consensus Update: $\tilde{w}_k^i \leftarrow f$-Trimmed Mean;
4 **for** *all* $i \in \mathcal{N}$ **do**
5 Execute action $a_k^i \sim \pi^i(\cdot|s_k)$;
6 Observe the state s_{k+1} and reward r_{k+1}^i;
7 Update $\delta_k^i \leftarrow r_{k+1}^i + \gamma\phi^T(s_{k+1})w_k^i - \phi^T(s_k)w_k^i$;
8 **end**
9 **Projected TD Step:** $w_{k+1}^i \leftarrow \Pi_{2,R}(\tilde{w}_k^i + \eta_k\delta_k^i \cdot \phi(s_k))$;
10 **end**

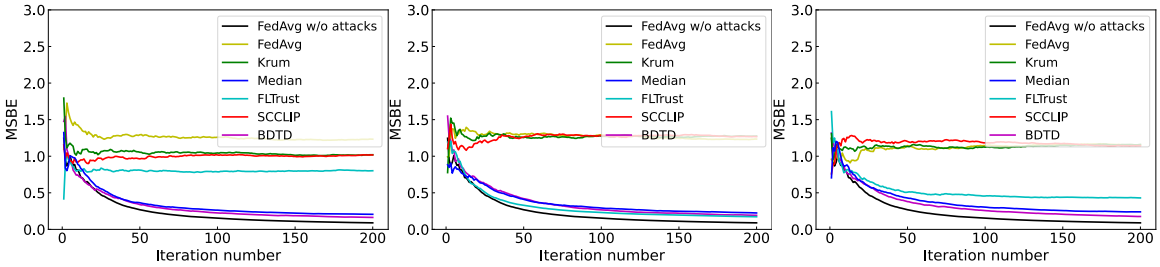
VI. Empirical Evaluation

A. Experimental Setup

1) Parameter Settings: We consider a cooperative navigation task known as Simple Spread, derived from the Multi-Particle Environment (MPE) [24]. The task involves 10 agents aiming to collectively cover all landmarks. There are two malicious agents among them. The agents receive rewards based on the proximity between the closest agent and each landmark. Collisions between agents result in negative rewards. Each agent selects actions from the action space $\mathcal{A}$ ={no action, move left, move right, move down, move up} using a uniformly random policy. The objective is to train all agents to identify and cover their respective landmarks while avoiding collisions. The malicious agents, on the other hand, attempt to deceive the other agents by providing arbitrary information. The feature dimension is 40, encompassing the agents' self-positions, relative positions of landmarks, and relative positions of other agents. The step-size is set to 0.1. We run our experiments on Intel(R) Core(TM) i9-12900K CPU. We repeat each experiment 10 times, and report the average results. Since the variances of results are small, we omit them here.

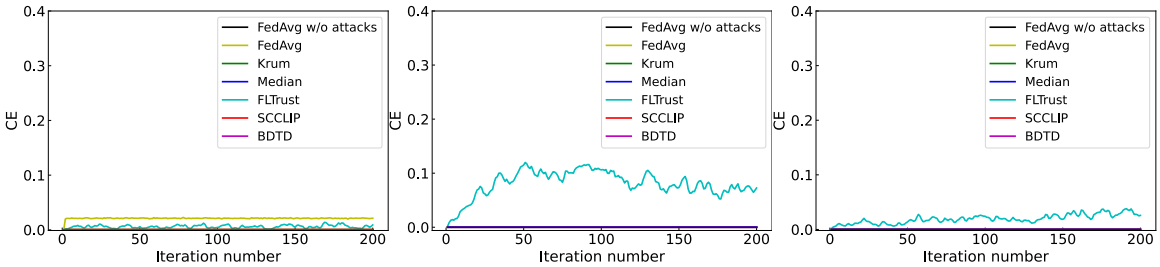
2) Compared Methods: We compare our BDTD algorithm with the following aggregation baselines.

- *FedAvg [25]:* Every agent, upon receiving parameters from its neighboring agents, calculates the weighted mean of the received parameters.
- *Krum [3]:* In the Krum aggregation rule, each agent produces a single parameter that minimizes the sum of distances to its subset of neighbors, and the size of the subset is $n - f$, where n is the total number of agents and f is the maximum number of Byzantine agents.
- *Coordinate-wise median (Median) [44]:* In every dimension, each agent calculates the coordinate-wise median of all the parameters it receives.



(a) Gaussian attack (b) Krum attack (c) Trim attack

Fig. 1: Mean squared Bellman error (MSBE) of different methods under different attacks.



(a) Gaussian attack (b) Krum attack (c) Trim attack

Fig. 2: Consensus error (CE) of different methods under different attacks.

- *FLTrust [4]:* When an agent receives a parameter from its neighboring agent, it first calculates the cosine similarity between its own parameter and the received parameter. If the cosine similarity is positive, the agent then normalizes the received parameter to have the same magnitude as its own parameter. After that, the agent computes the weighted average of all the normalized parameters sent by its neighbors.
- *SCCLIP [18]:* The SCCLIP method mitigates the influence of Byzantine agents through the use of the clip operation. In this approach, when an agent receives parameters from its neighboring agents, it employs its own parameter as the reference point to limit or clip the received parameters.

3) Poisoning Attacks: We consider the following poisoning attack schemes in our experiments.

- *Gaussian attack [3]:* In a Gaussian attack, each Byzantine agent samples a vector from a Gaussian distribution with a mean of zero and a standard deviation of one, then sends it to its neighboring agent.
- *Krum attack [11]:* In the Krum attack, Byzantine agents manipulate their parameters to degrade the Krum method's performance.
- *Trim attack [11]:* In the Trim attack, the attacker carefully manipulates the parameters of Byzantine agents in a way that causes a significant deviation between the aggregated parameter before and after the attack.

4) Evaluation Metrics: We consider the following two evaluation metrics: i) mean squared Bellman error (MSBE) and ii) consensus error (CE). Given parameters $\{w_k^i\}_{i\in\mathcal{N}}$ and samples (s_k, s_{k+1}), the empirical squared Bellman error (SBE) of the κ-th sample is defined as $\text{SBE}(\{w_k^i\}_{i\in\mathcal{N}}, s_\kappa, s_{\kappa+1}) := \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}}\left(\bar{r}_\kappa + \gamma\phi(s_{\kappa+1})^T w_\kappa^i - \phi(s_\kappa)^T w_\kappa^i\right)^2$, where $\bar{r}_\kappa = \frac{1}{\mathcal{N}}\sum_{i\in\mathcal{N}} r_\kappa^i$. Then, MSBE up to the k-th sample is defined as the average of SBEs over the history, which is computed as $\text{MSBE} := \frac{1}{k}\sum_{\kappa=1}^{k}\text{SBE}(\{w_\kappa^i\}_{i\in\mathcal{N}}, s_\kappa, s_{\kappa+1})$. The consensus error is computed as $\text{CE} = \frac{1}{|\mathcal{N}|}\sum_{i\in\mathcal{N}}\|w_k^i - \bar{w}_k\|^2$. The smaller the MSBE and CE, the better the defense.

B. Experimental Results

Figures 1 and 2 show the MSBE and CE of different methods under different attacks. "FedAvg w/o attacks" means that there are no Byzantine agents in the system. We observe from Figures 1 and 2 that our proposed BDTD overall achieves the best performance across various attack scenarios. Even under the strong Trim attack, our proposed BDTD 's MSBE is comparable to that of FedAvg without any attacks. In contrast, existing Byzantine-robust aggregation rules, e.g., Krum and SCCLIP, are susceptible to poisoning attacks. For instance, FLTrust is vulnerable to both the Gaussian and Krum attacks. Under the Gaussian attack, the final MSBE of FLTrust is 0.801. Similarly, under the Krum attack, although MSBE of FLTrust is low, CE is large, indicating a lack of consensus among the normal agents when using the FLTrust aggregation rule. The Krum aggregation rule is susceptible to all three considered attacks. Specifically, under three poisoning attacks, the CE of Krum remains small, but the MSBE becomes large. This suggests that when normal agents employ the Krum aggregation rule, they tend to reach a poor consensus.

VII. Proofs for Theorems in Section IV

Let $[n] := \{1, \cdots, n\}$, i.e. the set of all agents. [4]

[4]For the remaining proofs, please see online companion [15].

A. Proof of Theorem 1

Proof. Assume that $f > 0$. Inspired by [32], the theorem is proved by contradiction.

Suppose that there exists a correct algorithm $\mathcal{A}$ that solves Problem 2. Define the rewards of the n agents as follows for all state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$ to be $r^i(s, a) = i$ for all $i \in [N]$.

Consider the following two executions that in the first one, agent 1 is the Byzantine agent and the rest are normal agents whereas in the second one, agent n is the Byzantine agent and the rest are normal agents. In both executions, Byzantine agent behaves correctly as the correct algorithm, this is reasonable as Byzantine agents can behave arbitrarily. As a result, for execution 1, algorithm A outputs the result $w_t^{i,1}$ for each agent i and given t such that, we have

$$\begin{aligned} w_t^{i,1} &\xrightarrow{L^2} \frac{1}{n-1} \sum_{i\in\{2,\cdots,n\}} \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[i\phi(s)] \\ &= \frac{\sum_{i\in\{2,\cdots,n\}} i}{n-1} \mathbb{E}_{s\sim d_\pi(\cdot)}[\phi(s)] \\ &= \frac{n(n+1)-2}{2(n-1)} \mathbb{E}_{s\sim d_\pi(\cdot)}[\phi(s)] \triangleq w^{*,1} \end{aligned} \tag{4}$$

where L^2 denote expected mean-square convergence. More specifically,

$$\lim_{t\to\infty} \mathbb{E}\|w_t^{i,1} - w^{*,1}\|^2 = 0.$$

Similarly, for execution 2, we have

$$\begin{aligned} w_t^{i,2} &\xrightarrow{L^2} \frac{1}{n-1} \sum_{i\in\{1,\cdots,n-1\}} \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[i\phi(s)] \\ &= \frac{\sum_{i\in\{1,\cdots,n-1\}} i}{n-1} \mathbb{E}_{s\sim d_\pi(\cdot)}[\phi(s)] \\ &= \frac{n}{2} \mathbb{E}_{s\sim d_\pi(\cdot)}[\phi(s)] \triangleq w^{*,2}. \end{aligned} \tag{5}$$

Note that

$$w^{*,1} - w^{*,2} = \mathbb{E}_{s\sim d_\pi(\cdot)}[\phi(s)].$$

By the assumption of linear independence of feature vectors $\phi(\cdot)$, by Assumption 3, and the fact that $d_\pi(\cdot)$ is a distribution, we know that there exists an entry in vector $\mathbb{E}_{s\sim d_\pi(\cdot)}[\phi(s)]$ is on-zero. As a result, $w^{*,1} \neq w^{*,2}$.

However, for any agent $i \in \{2, \cdots, n-1\}$ perspective, they can't distinguish the above 2 executions, as a result, they must output the same results for both executions. However, this contradicts with the assumption that both executions would converge to distinct fixed points shown in (4) and (5) respectively. Therefore, there's no correct algorithm exists for ***Problem*** 2 and the proof is complete. □

B. Proof of Theorem 2

Proof. Recall that we assume $n > 3f+1$ and denote the actual number of Byzantine agents in the system as q, i.e. $q = |\mathcal{F}|$. Let the rewards for any state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$ and agent $i \in [n]$ to be

$$r^i(s, a) = i, \quad \text{for } 1 \le i \le f \text{ and } n-q+1 \le i \le n$$

$$r^i(s, a) = f+1, \quad \text{for } f+1 \le i \le n-q.$$

For any correct algorithm, consider the following two cases, where in both cases, Byzantine agents follow the correct algorithm.

- Case 1: In this case, agents $n-q+1 \le i \le n$ are Byzantine agents. The output of the correct algorithm converges to
$$w_\alpha^* \in [1, f+1]\mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)].$$
- Case 2: In this case, agents $1 \le i \le f$ are Byzantine agents. The output of the correct algorithm converges to
$$w_\alpha^* \in [f+1, n]\mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)].$$

As for any normal agent $i \in \{f+1, \cdots, n-q\}$ can't distinguish the above two cases, they must converge to an identical value in both cases. So, w_α^* must be $(f+1)\mathbb{E}_{s\sim d_\pi(\cdot),a\sim(\cdot|s)}[\phi(s)]$. In other words,

$$\begin{aligned} w_\alpha^* &= (f+1)\mathbb{E}_{s\sim d_\pi(\cdot),a\sim(\cdot|s)}[\phi(s)] \\ &= \sum_{i=1}^{n-q} \alpha_i r^i \mathbb{E}_{s\sim d_\pi(\cdot),a\sim\pi(\cdot|s)}[\phi(s)], \end{aligned}$$

where the second equality is due to the definition of Case 1, which is equivalent to

$$w_\alpha^* = \sum_{i=1}^{n-q} \alpha_i r^i = f+1. \tag{6}$$

The above equivalency is again because feature vectors are linearly independent. By the reward setting given above, we further have, for (6),

$$\sum_{i=1}^{f} \alpha_i i + (f+1) \sum_{i=f+1}^{n-q} \alpha_i = f+1,$$

which is equivalent to

$$\sum_{i=1}^{f} \alpha_i i = (f+1)(1 - \sum_{i=f+1}^{n-q} \alpha_i) = (f+1) \sum_{i=1}^{f} \alpha_i$$

which is only possible when $\alpha_i = 0$ for all $1 \le i \le f$. As a result, there could be at most $|\mathcal{N}| - f$ can be positive in Case 1 regardless of ξ. And ν can't be larger than $|\mathcal{N}| - f$ and the proof is complete. □

VIII. Conclusion

In this paper, we studied fully decentralized multi-agent policy evaluation problem in the presence of Byzantine agents. We first established the impossibility of designing a correct algorithm that obtains value functions where the system-wide rewards are modelled as the uniform average rewards of all normal agents. We then proceeded to relax the problem by considering the system-wide rewards being appropriately weighted average rewards of the normal agents. Subsequently, we demonstrated that there is no correct algorithm capable of ensuring that the number of positive weights surpasses $|\mathcal{N}| - f$ for the aforementioned relaxed problem. Lastly, we proposed a decentralized multi-agent policy evaluation algorithm, which guarantees consensus among all normal agents.

ACKNOWLEDGMENTS

This work has been supported in part by NSF grants CAREER CNS-2110259, CNS-2112471, IIS-2324052, ECCS-2331104, a DARPA YFA Award D24AP00265, and an ONR grant N00014-24-1-2729.

REFERENCES

[1] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *J. Mach. Learn. Res.*, 22(98):1–76, 2021.

[2] 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.

[3] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *NeurIPS*, 2017.

[4] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. In *NDSS*, 2021.

[5] Yiding Chen, Xuezhou Zhang, Kaiqing Zhang, Mengdi Wang, and Xiaojin Zhu. Byzantine-robust online and offline distributed reinforcement learning. In *AISTATS*, 2023.

[6] Yudong Chen, Lili Su, and Jiaming Xu. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. In *POMACS*, 2017.

[7] 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.

[8] Thinh Doan, Siva Maguluri, and Justin Romberg. Finite-time analysis of distributed td (0) with linear function approximation on multi-agent reinforcement learning. In *ICML*, 2019.

[9] Thinh T Doan, Siva Theja Maguluri, and Justin Romberg. Finite-time performance of distributed temporal-difference learning with linear function approximation. *SIAM Journal on Mathematics of Data Science*, 3(1):298–320, 2021.

[10] Xiaofeng Fan, Yining Ma, Zhongxiang Dai, Wei Jing, Cheston Tan, and Bryan Kian Hsiang Low. Fault-tolerant federated reinforcement learning with theoretical guarantee. In *NeurIPS*, 2021.

[11] Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Gong. Local model poisoning attacks to byzantine-robust federated learning. In *USENIX Security Symposium*, 2020.

[12] Minghong Fang, Jia Liu, Neil Zhenqiang Gong, and Elizabeth S Bentley. Aflguard: Byzantine-robust asynchronous federated learning. In *ACSAC*, 2022.

[13] Minghong Fang, Zifan Zhang, Hairi, Prashant Khanduri, Jia Liu, Songtao Lu, Yuchen Liu, and Neil Gong. Byzantine-robust decentralized federated learning. In *CCS*, 2024.

[14] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *AAAI*, 2018.

[15] Hairi, Minghong Fang, Zifan Zhang, Alvaro Velasquez, and Jia Liu. On the hardness of decentralized multi-agent policy evaluation under byzantine attacks. *arXiv:2409.12882*.

[16] Hairi, Zifan Zhang, and Jia Liu. Sample and communication efficient fully decentralized marl policy evaluation via a new approach: Local td update. In *AAMAS*, 2024.

[17] FNU Hairi, Jia Liu, and Songtao Lu. Finite-time convergence and scounterfactual multi-agent policy graic reinforcement learning with average reward. In *ICLR*, 2022.

[18] Lie He, Sai Praneeth Karimireddy, and Martin Jaggi. Byzantine-robust decentralized learning via self-centered clipping. In *arXiv preprint arXiv:2202.01545*, 2022.

[19] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *NeurIPS*, 2018.

[20] Chi Jin, Qinghua Liu, Yuanhao Wang, and Tiancheng Yu. V-learning–a simple, efficient, decentralized algorithm for multiagent rl. *arXiv preprint arXiv:2110.14555*, 2021.

[21] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Learning from history for byzantine robust optimization. In *ICML*, 2021.

[22] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via bucketing. In *ICLR*, 2022.

[23] Xin Liu, Honghao Wei, and Lei Ying. Scalable and sample efficient distributed policy gradient algorithms in multi-agent networked systems. *arXiv preprint arXiv:2212.06357*, 2022.

[24] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NeurIPS*, 2017.

[25] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

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

[27] Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100, 1953.

[28] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[29] Rayadurgam Srikant and Lei Ying. Finite-time error bounds for linear stochastic approximation andtd learning. In *COLT*, 2019.

[30] Lili Su and Nitin Vaidya. Byzantine multi-agent optimization: Part i. *arXiv preprint arXiv:1506.04681*, 2015.

[31] Lili Su and Nitin Vaidya. Fault-tolerant multi-agent optimization: Part iii. *arXiv preprint arXiv:1509.01864*, 2015.

[32] Lili Su and Nitin H Vaidya. Fault-tolerant multi-agent optimization: optimal iterative distributed algorithms. In *PODC*, 2016.

[33] Richard S Sutton. Learning to predict by the methods of temporal differences. In *Machine learning*, 1988.

[34] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[35] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. -leith linear function approximation ong with function approximation. In *NeurIPS*, 1999.

[36] John N Tsitsiklis and Benjamin Van Roy. Average cost temporal-difference learning. *Automatica*, 35(11):1799–1808, 1999.

[37] John N Tsitsiklis and Benjamin Van Roy. On average versus discounted reward temporal-difference learning. In *Machine Learning*, 2002.

[38] Nitin Vaidya. Matrix representation of iterative approximate byzantine consensus in directed graphs. *arXiv preprint arXiv:1203.1888*, 2012.

[39] Nitin H Vaidya, Lewis Tseng, and Guanfeng Liang. Iterative approximate byzantine consensus in arbitrary directed graphs. In *PODC*, 2012.

[40] Young Wu, Jermey McMahan, Xiaojin Zhu, and Qiaomin Xie. Reward poisoning attacks on offline multi-agent reinforcement learning. *arXiv preprint arXiv:2206.01888*, 2022.

[41] Zhaoxian Wu, Han Shen, Tianyi Chen, and Qing Ling. Byzantine-resilient decentralized policy evaluation with linear function approximation. In *IEEE Transactions on Signal Processing*, 2021.

[42] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. Generalized byzantine-tolerant sgd. *arXiv preprint arXiv:1802.10116*, 2018.

[43] Tengyu Xu, Zhe Wang, and Yingbin Liang. Improving sample complexity bounds for (natural) actor-critic algorithms. *arXiv preprint arXiv:2004.12956*, 2020.

[44] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, 2018.

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

[46] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. Fully decentralized multi-agent reinforcement learning with networked agents. In *ICML*, 2018.