Multi-Agent Reinforcement Learning with Reward Delays

Yuyang Zhang YUYANGZHANG@G.HARVARD.EDU
Runyu Zhang RUNYUZHANG@FAS.HARVARD.EDU
Harvard University, School of Engineering and Applied Science

Yuantao Gu GYT@TSINGHUA.EDU.CN
Tsinghua University, Department of Electronic Engineering

Na Li NALI@SEAS.HARVARD.EDU
Harvard University, School of Engineering and Applied Science

Editors: N. Matni, M. Morari, G. J. Pappas

Abstract

This paper considers multi-agent reinforcement learning (MARL) where the rewards are received after delays and the delay time varies across agents and across time steps. Based on the V-learning framework, this paper proposes MARL algorithms that efficiently deal with reward delays. When the delays are finite, our algorithm reaches a coarse correlated equilibrium (CCE) with rate $\tilde{\mathcal{O}}(\frac{H^3\sqrt{S\mathcal{T}_K}}{K} + \frac{H^3\sqrt{SA}}{\sqrt{K}})$ where K is the number of episodes, H is the planning horizon, S is the size of the state space, A is the size of the largest action space, and $\mathcal{T}_K$ is the measure of total delay formally defined in the paper. Moreover, our algorithm is extended to cases with infinite delays through a reward skipping scheme. It achieves convergence rate similar to the finite delay case.

Keywords: Reward Delays, Markov Games, Multi-Agent Reinforcement Learning

1. Introduction

Multi-agent reinforcement learning (MARL) finds extensive applications such as recommendation systems (Zhao et al., 2020), medical treatments (Li et al., 2022; Martinho et al., 2021), multi-agent robotics systems (Brambilla et al., 2013; Malus et al., 2020; Choi and Ahn, 2010), autonomous driving (Kiran et al., 2021), etc. In these multi-agent problems, individuals aim to learn to interact with the environment under the influence of other agents.

Motivated by the empirical success of MARL, there is a recent surge of studies on MARL algorithms with theoretical convergence guarantees such as V-learning, V-learning OMD, SPoCMAR, etc (Jin et al., 2021; Song et al., 2021; Daskalakis et al., 2022; Mao and Başar, 2022). In these algorithms, agents rely heavily on real-time observations of reward values to update their policies or value functions. However, in real-life MARL applications, rewards generally come with delays. One example is the medical treatment process (Li et al., 2022), where the effectiveness of a treatment strategy cannot be observed immediately. It generally takes a long time for a patient to respond and recover. Similar reward delays also widely exist in recommendation systems (Aldhahri et al., 2015), autonomous driving (Chen et al., 2019), neuroscience (Kobayashi and Schultz, 2008), etc. Another example is the reward delays due to communication latency in all kinds of distributed systems (Duan et al., 2022; Joulani et al., 2013; Liu, 2007) where even infinite delays are common due to packet loss and network failure. Reward delays in these applications are typically time-varying, depending on factors including the patient's physiological state, the status of communication channels, etc. All the examples suggest that it is crucial to understand how reward delays affect the learning process and how to design MARL algorithms that could accommodate the delays efficiently.

In existing empirical work on MARL with reward delays, different approaches are proposed to handle delays, including but not restricted to learning temporal structures (Hauwere et al., 2011), predicting strategic interactions (Tang et al., 2018), evaluating curiosity (Shao et al., 2019), and predicting the environment (Firoiu et al., 2018) with neural networks. However, from the theoretical perspective, few results are known for MARL. We acknowledge the lines of work studying state or action delays in MARL (Agarwal and Aggarwal, 2021; Bouteiller et al., 2020b; Chen et al., 2020), but the settings are different and out of the scope of this paper. Other related settings include single-agent reinforcement learning (SARL) and multi-arm bandit (MAB). For SARL, recent work (Lancewicki et al., 2022; Jin et al., 2022) studies adversarial reward delays. Unfortunately, their methods suffer from the curse of dimensionality when directly extended to MARL. Other work (Walsh et al., 2009; Katsikopoulos and Engelbrecht, 2003) only focuses on constant reward delays. For MAB, Gyorgy and Joulani (2021); Zimmert and Seldin (2020); Gael et al. (2020) tackle adversarial reward delays while Cesa-Bianchi et al. (2016); Neu et al. (2010) focus on constant delays.

Our Contributions. In this paper, we focus on a specific MARL model, the general-sum Markov games (Shapley, 1953; Littman et al., 2001). We propose the delay-adaptive multi-agent V-learning (DA-MAVL) to learn coarse-correlated equilibria (CCEs) under time-varying reward delays, where the learning of the agents can be finished in a fully decentralized manner. Namely, every agent runs its own learning algorithm without communicating with others. Note that this is nontrivial because different agents may receive the same reward at different episodes due to the heterogeneous delays among them. Without careful design, fully distributed learning might lead to misalignment and divergent behavior. Our DA-MAVL algorithm circumvents this problem by carefully selecting proper reward information for learning and therefore aligning the behaviour of the agents.

For finite delays, our algorithm achieves the CCE-gap as small as $\tilde{\mathcal{O}}(\frac{H^3\sqrt{S\mathcal{T}_K}}{K} + \frac{H^3\sqrt{SA}}{\sqrt{K}})$ with samples from K episodes (Theorem 1). Here H is the planning horizon, S is the size of the state space, $A = \max_m |\mathcal{A}_m|$ is the largest size of one agent's action space, and $\mathcal{T}_K$ can be seen as a measure of the total delay. In the worst case, $\sqrt{\mathcal{T}_K}$ is the order of $\mathcal{O}(\sqrt{Kd_{max}})$, where d_{max} is the largest possible delay. This implies that the CCE-gap is as small as $\tilde{\mathcal{O}}(\frac{1}{\sqrt{K}})$. This dependence of K matches the original result of V-learning (Jin et al., 2021; Song et al., 2021), indicating that DA-MAVL successfully aligns the behaviour of the agents. Moreover, both terms are independent of the number of agents, meaning that DA-MAVL scales nicely with the system size. Our proposed DA-MAVL algorithm can be extended to settings with infinite delays. With a novel skipping metric inspired by Zimmert and Seldin (2020), our algorithm can skip the infinite delays without prior knowledge of the delay sequence and achieve the CCE-gap similarly to the finite delay case (Theorem 2). To the best of our knowledge, our results give the first convergence rate guarantee for general-sum MGs under time-varying reward delays.

Due to the space limit, we defer related work, some of the algorithms, proofs, and simulation settings to the appendix of our full paper (Zhang et al., 2022).

2. Problem Setup & Preliminary

2.1. Markov Games with Reward Delays

We study general-sum Markov games (MGs, also called stochastic games in Shapley (1953)) with reward delays. In its episodic and tabular form, an MG can be defined by the following tuple:

$$\mathcal{MG}\Big(H, \mathcal{S}, \{\mathcal{A}_m\}_{m\in[M]}, \{\mathbb{P}_h\}_{h\in[H]}, \{r_{m,h}\}_{m\in[M],h\in[H]}, \{d^n_{m,h}(s)\}_{m\in[M],h\in[H],s\in\mathcal{S},n\in[K]}\Big). \quad (1)$$

Here we use $[i]$ to denote set $\{1, \ldots, i\}$ for any integer i. In the subscripts, $m \in [M]$ stands for the agents, $k \in [K]$ stands for the episode, $h \in [H]$ stands for the time step over the finite horizon. $\mathcal{S}$ is a global state space with cardinality $S = |\mathcal{S}|$. $\mathcal{A}_m$ is the action space of agent m. Define $A = \max_{m\in[M]} |\mathcal{A}_m|$. The joint action space is given by $\boldsymbol{\mathcal{A}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_M$, and the joint action is given as $\boldsymbol{a} = (a_1, \ldots, a_M)$. $\mathbb{P}_h(s'|s, \boldsymbol{a})$ with $s, s' \in \mathcal{S}, \boldsymbol{a} \in \boldsymbol{\mathcal{A}}$ is the transition function for step h. $r_{m,h}(s, \boldsymbol{a})$ is a deterministic reward for agent m at step h when the current state and joint action are s and $\boldsymbol{a}$ respectively. The sequence $\{d^n_{m,h}(s)\}_{m\in[M],h\in[H],s\in\mathcal{S},n\in\mathbb{N}}$ represents reward delays which will be detailed in later paragraphs. Without loss of generality, we assume every episode k starts from a fixed initial state s_1.[1] At every step h, every agent observes the current state s^k_h, takes action $a^k_{m,h}$. The environment transits to the next state s^k_{h+1} according to $\mathbb{P}_h$ until step $H + 1$ is reached.

Visits & happening order: When the agents visit state s at step h for the n-th time, we say that the n-th visit of (h, s) happens, and n is the happening order of this visit.

Reward delays: We allow the reward delays to be heterogeneous among different agents m, different visits (h, s) and different happening orders n. In specific, for the n-th visit of (h, s) which happens at episode k, agent m will receive its reward $r_{m,h}(s, \boldsymbol{a})$ by the end of episode $k + d^n_{m,h}(s)$. When $d^n_{m,h}(s) = 0$ for all $m \in [M], h \in [H], s \in \mathcal{S}, n \in \mathbb{N}$, our setting reduces to a classic MG.

(Un)received visits: When the reward of a visit has been received, we call the visit a received visit; otherwise, we call it an unreceived visit. It is worth mentioning that visits that happen early are not necessarily received early.

(Un)usable visits: We denote the episode when the n-th visit of (h, s) happens as $k^n_h(s)$. At the beginning of episode $k^n_h(s)$,[2] for agent m, some of the first $n-1$ visits of (h, s) may not be received because of the reward delays. In this case, we define index $e^n_{m,h}(s)$ as the earliest unreceived visit:

$$e^n_{m,h}(s) := \min\left\{j : d^j_{m,h}(s) + k^j_h(s) > k^n_h(s) - 1, j \in [n-1]\right\}. \tag{2}$$

If all of the first $n - 1$ visits have been received, we define

$$e^n_{m,h}(s) := n. \tag{3}$$

It means that all the visits of (h, s) that happen earlier than the $e^n_{m,h}(s)$-th visit have been received at the beginning of episode $k^n_h(s)$; but the $e^n_{m,h}(s)$-th visit has not been received yet. We call a received visit as *usable* if all visits happening earlier have all been received. The rest of the received visits are called *unusable*. At the beginning of episode $k^n_h(s)$, the usable visits of (h, s) have happening orders $1, 2, \ldots, e^n_{m,h}(s) - 1$. The unusable visit of (h, s), if $e^n_{m,h}(s) < n$, have happening orders $e^n_{m,h}(s), \ldots, n - 1$.

In our algorithms, to ensure that the agents are aligned, we only use the usable visits. Consequently, the performance of our algorithms strongly relates to the number of unusable and unreceived visits, for which we define a counting sequence $\{\mathcal{T}^n_{m,h}(s)\}_{m\in[M],h\in[H],s\in\mathcal{S},n\in[K]}$:

$$\mathcal{T}^n_{m,h}(s) := \sum_{i=1}^{n} \left(i - e^i_{m,h}(s)\right) = \sum_{i=1}^{n} \left(i - \min\left\{j : d^j_{m,h}(s) + k^j_h(s) > k^i_h(s) - 1\right\}\right). \tag{4}$$

$\mathcal{T}^n_{m,h}(s)$ counts the accumulated number of unusable and unreceived visits of (h, s) till the n-th visit. Note that in the classic MG setting without reward delays, we have $\mathcal{T}^n_{m,h}(s) = 0$ for all $m \in [M], h \in [H], s \in \mathcal{S}, n \in \mathbb{N}$.

1. For any MG with initial distribution μ, one can always add a step with only one state as the first time step and let the transition function be μ for all actions. This leads to an equivalent MG with a fixed initial state.
2. Without causing any confusion, we will use "at the beginning of episode k" and "by the end of episode $k - 1$" interchangeably.

2.2. Learning Objective - Coarse Correlated Equilibrium

Agent m's policy is denoted as $\pi_m = \{\pi_{m,h}\}_{h\in[H]}$. The policy at step h is $\pi_{m,h} : \Omega \times (\mathcal{S} \times \mathcal{A})^{h-1} \times \mathcal{S} \to \Delta_{\mathcal{A}_m}$, where $\pi_{m,h}$ maps a random sample ω_h from probability space Ω and a trajectory $(s_1, \boldsymbol{a}_1, \ldots, s_h)$ to a point in probability simplex $\Delta_{\mathcal{A}_m}$. An important subclass of policy is the *independent Markov policy*, with $\pi_{m,h} : \mathcal{S} \to \Delta_{\mathcal{A}_m}$ maps the current state to a point in probability simplex $\Delta_{\mathcal{A}_m}$.

A *joint policy* π is a set of policies $\{\pi_m\}_{m\in[M]}$ of all agents. If the random samples $\{\omega_h \in \Omega\}_{h\in[H]}$ are shared among all agents, policies of all agents are correlated. In this case, we denote the joint policy π as $\pi = \pi_1 \odot \pi_2 \odot \ldots \odot \pi_M$, and call π as a correlated policy. We also use $\pi_{-m} = \pi_1 \odot \ldots \pi_{m-1} \odot \pi_{m+1} \ldots \odot \pi_M$ to denote the policy excluding agent m. If the randomness of π_m is independent of other policies π_{-m}, i.e., the random samples $\{\omega_h \in \Omega\}_{h\in[H]}$ are shared among agents except agent m, we denote the joint policy as $\pi = \pi_m \times \pi_{-m}$.

For a joint policy π, we define its value function for agent m as:

$$V_{m,h}^{\pi}(s_h) := \mathbb{E}_{\pi}\Big[\sum_{h'=h}^{H} r_{m,h'}(s_{h'}, \boldsymbol{a}_{h'})|s_h\Big], \ \ \forall m \in [M]. \tag{5}$$

Given policy π_{-m}, the best response for agent m is defined as the best policy that maximizes the value function for agent m, i.e., $\pi_m^{\dagger} = \arg\max_{\pi_m} V_{m,1}^{\pi_m \times \pi_{-m}}$. For notation simplicity, we denote the value function of the best response as $V_{m,h}^{\dagger,\pi_{-m}} = V_{m,h}^{\pi_m^{\dagger},\pi_{-m}}$. Our objective is to find a joint policy π that is an ϵ-coarse correlated equilibrium (CCE) defined as follows:

Definition 1 (Coarse Correlated Equilibrium (CCE (Young, 2004))) *We define the CCE-gap of a joint policy π as:*

$$\textit{CCE-gap}(\pi) := \max_{m\in[M]} (V_{m,1}^{\dagger,\pi_{-m}} - V_{m,1}^{\pi})(s_1). \tag{6}$$

A joint policy π is a CCE if the CCE-gap is zero:

$$\textit{CCE-gap}(\pi) = 0. \tag{7}$$

A joint policy π is an ϵ-CCE if the CCE-gap satisfies:

$$\textit{CCE-gap}(\pi) \leq \epsilon. \tag{8}$$

When the agents reach a CCE, they have no incentive to deviate to any independent policy.

3. Delay-Adaptive Multi-Agent V-Learning

In this section, we present our main algorithm: Delay-Adaptive Multi-Agent V-Learning (DA-MAVL). Similar to V-learning in Jin et al. (2021); Song et al. (2021), DA-MAVL contains two consecutive algorithms - i) the training algorithm (Algorithm 1), where the agents learn and store a set of independent Markov policies $\{\hat{\pi}_{m,h}^k\}_{m\in[M],h\in[H],k\in[K]}$, and ii) the output algorithm (Algorithm 2) that constructs the final output policies (which can be correlated and non-Markov) $\{\pi_m\}_{m\in[M]}$ from the set of independent Markov policies $\{\hat{\pi}_{m,h}^k\}_{m\in[M],h\in[H],k\in[K]}$.

The training algorithm is *fully decentralized*, i.e., the agents update their own policies with their own delayed reward information without communication with each other. The algorithm framework resembles the V-learning algorithm but comes with a mechanism that carefully chooses *usable* visits for learning. This mechanism enables agents to align their behaviour under the influence of heterogeneous reward delays and leads the algorithm toward convergence (see more discussions at the end of next subsection).

Recall that $k_h^n(s)$ is the episode when the agents visit (h, s) for the n-th time. For agent m, we also define $\overline{n}_{m,h}^k(s)$ as the count of happened visits of (h, s) and define $\underline{n}_{m,h}^k(s)$ as the count of usable visits of (h, s) at the beginning of episode k.

3.1. The Training Algorithm

We now present the training algorithm of DA-MAVL for agent m (Algorithm 1). The algorithm contains three major processes, which we name as 'Preparation', 'Learning' and 'Sampling'. At each episode k, for every time step h, the three processes are carried out iteratively:

- In the 'Preparation' process, we keep track of three important sets, namely the set of *visits to be used* $\mathcal{F}_{m,h}(s)$ (including all usable visits that have not been used previously), the set of *unusable visits* $\mathcal{M}_{m,h}^+(s)$ and the set of *unreceived visits* $\mathcal{M}_{m,h}^-(s)$. Usable Visits in $\mathcal{F}_{m,h}(s)$ will be fed into later processes and will no longer be used again in future episodes. Unusable and unreceived visits in $\mathcal{M}_{m,h}(s) = \mathcal{M}_{m,h}^+(s) \cup \mathcal{M}_{m,h}^-(s)$ are stored in memory until they become usable.

  For set $\mathcal{M} = \mathcal{M}_{m,h}(s)$ (or $\mathcal{M} = \mathcal{M}_{m,h}^-(s)$), whose entries are tuples $(i, a, \hat{\pi}, \overline{V}', \underline{V}', r)$ (or $(i, a, \hat{\pi}, \overline{V}', \underline{V}')$) indexed by the first element i, we define $\arg\{\mathcal{M}\} := \{i\}$ as the set of indices.

- In the 'Learning' process, visits in $\mathcal{F}_{m,h}(s)$ are fed into subroutines 'VALUE_UPDATE' and 'POLICY_OPT' (Algorithm 3 and Algorithm 4 in Appendix C.1 in Zhang et al. (2022)) *consecutively in their happening orders*. Subroutine 'VALUE_UPDATE' updates an "optimistic" value estimate $\overline{V}_{m,h}(s)$ by using all visits in $\mathcal{F}_{m,h}(s)$ with parameters $\{\alpha_i\}_i$ and $\{\overline{\beta}_{m,h}^i(s)\}_i$, where α_i can be viewed as the learning rate and $\overline{\beta}_{m,h}^i(s)$ is a bonus term. Subroutine 'POLICY_OPT' runs an adversarial-bandit-type algorithm (similar to the algorithm in Zimmert and Seldin (2020)) to update the policy, where the bandit loss is calculated using the optimistic value estimates $\overline{V}_{m,h}(s)$.

  Note that in Algorithm 1 and Subroutine 'VALUE_UPDATE', we also introduce a pessimistic value estimate $\underline{V}_{m,h}(s)$. This pessimistic estimate is an auxiliary variable that is not needed for running the algorithm but is used in the proof.

- In the 'Sampling' process, every agent chooses its action based on the updated policy, and the next state is sampled. Finally, every agent stores related information, receives delayed rewards and moves on to the next step $h + 1$.

Discussions - The role of usable visits. As previously mentioned, one key challenge for the decentralized learning algorithm is to avoid misalignment due to heterogeneous reward delays among different agents. Our algorithm addresses this challenge by only using usable visits for learning in subroutines 'VALUE_UPDATE' and 'POLICY_OPT'. The main intuition is to ensure that the *happening order* of the visits is also *the order in which they are used in the subroutines*. Consequently, although rewards of the visits might be received and used in different episodes for different agents, the order in which they are used remains the same among agents. This design leads to cooperative policies among the agents without any communication in the training algorithm.

To better understand the role of usable visits, we also compare our algorithm numerically with the naive algorithm, where visits are immediately fed into subroutines once they are received (see Appendix C.2 in Zhang et al. (2022) for details). In the naive algorithm, the reward of the same visit may be used in different orders among agents, which causes extra misalignment among the agents. The numerical results are discussed in Section 6, where we indeed observe that with the notion of usable visits, our algorithm outperforms the naive algorithm. However, it remains an open question to prove or to disapprove whether the naive method would converge to a CCE.

We also note that the notion of usable visits alone is not sufficient to fully align all agents, nor does it reduce the problem to MARL without reward delays. This is because different agents still have different amount of information in the episodes. This information mismatch is further addressed by a critical modification in Algorithm 2 in the following subsection.

Algorithm 1: DA-MAVL Training for Agent m

Init: $\forall(h,s), \overline{n}^0_{m,h}(s) \leftarrow 0, \underline{n}^0_{m,h}(s) \leftarrow 0, \mathcal{T}^0_{m,h}(s) \leftarrow 0, \mathcal{F}_{m,h}(s) \leftarrow \emptyset, \mathcal{M}_{m,h}(s) \leftarrow \emptyset$;

for *Episode* $k = 1, \ldots, K$ **do**
  Receive initial state s_1^k;
  for *Step* $h = 1, \ldots, H$ **do**
    // Preparation
    $s \leftarrow s_h^k$;
    for $(i, a, \hat{\pi}, \overline{V}', \underline{V}', r) \in \mathcal{M}^+_{m,h}(s)$ **do**
      if $\forall j < i, j \notin \arg\{\mathcal{M}^-_{m,h}(s)\}$ **then**
        Save $(i, a, \hat{\pi}, \overline{V}', \underline{V}', r)$ to $\mathcal{F}_{m,h}(s)$; Remove $(i, a, \hat{\pi}, \overline{V}', \underline{V}', r)$ from $\mathcal{M}^+_{m,h}(s)$;
    $\overline{n} \leftarrow \overline{n}^k_{m,h}(s) = \overline{n}^{k-1}_{m,h}(s) + 1$; $\underline{n} \leftarrow \underline{n}^k_{m,h}(s) = \underline{n}^{k-1}_{m,h}(s) + |\mathcal{F}_{m,h}(s)|$;
    $\mathcal{T}^{\overline{n}}_{m,h}(s) \leftarrow \mathcal{T}^{\overline{n}-1}_{m,h}(s) + |\mathcal{M}_{m,h}(s)|$;
    // Learning
    $\overline{V}^k_{m,h}(s), \underline{V}^k_{m,h}(s) \leftarrow \text{VALUE\_UPDATE}_{m,h,s}\big(\mathcal{F}_{m,h}(s), \underline{n}\big)$;
    $\hat{\pi}^k_{m,h}(\cdot|s) \leftarrow \text{POLICY\_OPT}_{m,h,s}\big(\mathcal{F}_{m,h}(s), \overline{n}\big)$;
    // Sampling
    Take action $a^k_{m,h} \sim \hat{\pi}^k_{m,h}(\cdot|s)$; Observe next state s^k_{h+1};
    for $s' \in \mathcal{S} \backslash s$ **do**
      $\overline{n}^k_{m,h}(s') \leftarrow \overline{n}^{k-1}_{m,h}(s')$; $\underline{n}^k_{m,h}(s') \leftarrow \underline{n}^{k-1}_{m,h}(s')$;
      $\overline{V}^k_{m,h}(s') \leftarrow \overline{V}^{k-1}_{m,h}(s')$; $\underline{V}^k_{m,h}(s') \leftarrow \underline{V}^{k-1}_{m,h}(s')$; $\hat{\pi}^k_{m,h}(\cdot|s') \leftarrow \hat{\pi}^{k-1}_{m,h}(\cdot|s')$;
  for *Step* $h = 1, \ldots, H$ **do**
    Save $\big(\overline{n}^k_{m,h}(s^k_h), a^k_{m,h}, \hat{\pi}^k_h(a^k_{m,h}|s^k_h), \overline{V}^k_{m,h+1}(s^k_{h+1}), \underline{V}^k_{m,h+1}(s^k_{h+1})\big)$ to $\mathcal{M}^-_{m,h}(s^k_h)$;
  Receive delayed rewards for all states s;
  for *Delayed Reward* (m, h, s, i, r) **do**
    Extract and remove $\big(i, a, \hat{\pi}, \overline{V}', \underline{V}'\big)$ from $\mathcal{M}^-_{m,h}(s)$;
    Save $\big(i, a, \hat{\pi}, \overline{V}', \underline{V}', r\big)$ to $\mathcal{M}^+_{m,h}(s)$;

3.2. Execution of the Output Policy

Algorithm 2: DA-MAVL Output for Policy π_m

Sample $k \sim \text{Uniform}([K])$;
for *step* $h = 1, \ldots, H$ **do**
  Observe current state s_h; $n \leftarrow \max_m \underline{n}^k_{m,h}(s_h)$;
  Sample i from $[n]$ with probability α^i_n; $k \leftarrow k^i_h(s_h)$;
  Take action $a_{m,h} \sim \hat{\pi}^k_{m,h}(\cdot|s_h)$;

Algorithm 1 outputs a set of independent Markov policies $\{\hat{\pi}^k_{m,h}\}_{m\in[M],h\in[H],k\in[K]}$. Based on this policy set, we now construct joint policy $\pi = \{\pi_m\}_{m\in[M]}$ as the output of DA-MAVL. The

policy is defined by its execution in Algorithm 2. Notice that all random samples (line 1 and line 4) are shared across all agents.

This algorithm follows V-learning in Jin et al. (2021); Song et al. (2021) except for the critical modification in line 3. Intuitively speaking, choosing $n = \max_m \underline{n}^k_{m,h}(s)$ ensures that agent m is aware of the extra information that the most informed agent possesses, and therefore guarantees that the output policy of agent m is compatible with that of the most informed agent. Technically speaking, it ensures the optimistic value estimates in Algorithm 1 upper bound the policy performance.

4. Performance Guarantee and Proof Sketch

Recall that the counting sequence $\{\mathcal{T}^n_{m,h}(s)\}_{m\in[M],h\in[H],s\in\mathcal{S},n\in[K]}$ (Equation (4)) is agent m's accumulated count of unusable and unreceived visits till the n-th visit of (h,s). Also, recall that $\underline{n}^k_{m,h}(s)$ is the count of usable visits of (h,s) at the beginning of episode k. Using the two notations, we define $\mathcal{T}_K := \max_{m,h}\sum_{s\in\mathcal{S}}\mathcal{T}^{\underline{n}^K_{m,h}(s)}_{m,h}(s)$ which will be used in bounding the CCE-gap after K episodes. We also assume that the reward delays of the MG are upper bounded by some constant.

Assumption 1 *The delays are bounded by d_{max}, that is,* $\max_{m\in[M],h\in[H],s\in\mathcal{S},n\in[K]} d^n_{m,h}(s) \le d_{max}$.

Now we are ready to present the performance guarantee for DA-MAVL:

Theorem 1 *Under Assumption 1, for any $\delta\in(0,1)$, $K \ge d^2_{max}S\iota^3$ where $\iota = \log(4MHSAK/\delta)$, suppose Algorithm 1 is run for K episodes, then the following equation holds for the output policy π of Algorithm 2 with probability at least $1-\delta$*

$$\textit{CCE-gap}(\pi) = \max_{m\in[M]}\left(V^{\dagger,\pi_{-m}}_{m,1} - V^{\pi}_{m,1}\right)(s_1) \lesssim H^3\sqrt{S\mathcal{T}_K/K^2}\iota^2 + H^3\sqrt{SA\iota/K}. \tag{9}$$

Under Assumption 1, it can be shown (with Lemma 7 in Appendix E.1 in Zhang et al. (2022)):

$$\mathcal{T}_K = \max_{m,h}\sum_{s\in\mathcal{S}}\mathcal{T}^{\underline{n}^K_{m,h}(s)}_{m,h}(s) = \max_{m,h}\sum_{s\in\mathcal{S}}\sum_{n=1}^{\underline{n}^K_{m,h}(s)}\left(n - e^n_{m,h}(s)\right) \le d_{max}\max_{m,h}\sum_{s\in\mathcal{S}}\overline{n}^K_{m,h}(s) \le Kd_{max}.$$

Substituting it into Theorem 1 gives the CCE-gap of order $\tilde{\mathcal{O}}(\frac{H^3\sqrt{Sd_{max}}+H^3\sqrt{SA}}{\sqrt{K}})$. In other words, in the worst case where the delays are always d_{max} and every (h,s) is visited for K times, at most $K = \tilde{\mathcal{O}}\big(\frac{H^6S(d_{max}+A)}{\epsilon^2}\big)$ episodes are needed for an ϵ-CCE. The influence of the reward delays is linearly bounded by term $\tilde{\mathcal{O}}(\frac{H^6Sd_{max}}{\epsilon^2})$ and tends to 0 when d_{max} goes to 0. Note that our result bears an extra factor H compared with V-learning (Jin et al., 2021; Song et al., 2021), even when all delays are zero. This is because we have to choose the parameters generously so that our algorithm is adaptive to potential delays.

4.1. Proof Sketch of Theorem 1

The proof can be broken down into the following three steps.

STEP 1: Bound the 'Policy Optimization Regret'. For every pair (m,h,s,n), we first define the policy optimization regret $R^n_{m,h}(s)$. For notational simplicity, we let k_n denote $k^n_h(s)$.

$$R^n_{m,h}(s) = \max_{a_m\in\mathcal{A}_m}\sum_{i=1}^{n}\alpha^i_n\Big[\mathbb{E}\Big(r_{m,h}(s,\boldsymbol{a}) + \overline{V}^{k_i}_{m,h+1}(s')\Big) - \Big(r^{k_i}_{m,h} + \overline{V}^{k_i}_{m,h+1}(s^{k_i}_{h+1})\Big)\Big], \tag{10}$$

where $\boldsymbol{a} = (a_m, \boldsymbol{a}_{-m})$, α_n^i is the weight which we define in Equation (22) in Appendix D in Zhang et al. (2022), and the expectation is taken over $\boldsymbol{a}_{-m} \sim \hat{\pi}_{-m,h}^{k_i}(\cdot|s)$ and $s' \sim \mathbb{P}_h(\cdot|s, \boldsymbol{a})$. Intuitively, it measures the performance of the first n outputs of subroutine 'POLICY_OPT$_{m,h,s}$' in Algorithm 1, i.e. Markov policies $\{\hat{\pi}_{m,h}^{k_i}(s)\}_{i\in[n]}$. Under Assumption 1, we give the following upper bound:

Lemma 1 *Let Assumption 1 holds. For $\forall(m, h, s, k) \in [M] \times [H] \times \mathcal{S} \times [K]$, the following inequality holds with probability at least $1 - \delta/2$*

$$R_{m,h}^n(s) \leq 12H^2\sqrt{\frac{nA + \mathcal{T}_{m,h}^n(s)}{n^2}\iota} + 2H^2\frac{d_{max}}{n}\iota. \tag{11}$$

In this lemma, the key difference from V-learning, and main technical difficulty, is that the subroutine needs to learn the n-th output, i.e. $\hat{\pi}_{m,h}^{k_n}(s)$, without access to all reward information of the first $n-1$ visits of (h, s) due to the reward delays. We have to measure the influence of the delays on outputs. By comparing it with the no-delay versions, we can show that the influence of the delays can be reflected by term $\sqrt{\mathcal{T}_{m,h}^n(s)/n^2}$ and d_{max}/n in Equation (11).

STEP 2: Optimism and Pessimism. Utilizing the regret defined above, we carefully design bonuses $\overline{\beta}_{m,h}^n(s)$ and $\underline{\beta}_n$ in subroutine 'VALUE_UPDATE' as follows:

$$\overline{\beta}_{m,h}^n(s) = R_{m,h}^n(s) + 2H^2\frac{d_{max}}{n}\iota, \quad \underline{\beta}_n = 2\sqrt{\frac{H^3}{n}\iota} + 2H^2\frac{d_{max}}{n}\iota. \tag{12}$$

With the bonuses, we can show that the value estimates $\overline{V}_{m,h}^k(s)$ and $\underline{V}_{m,h}^k(s)$ in Algorithm 1 upper and lower bound the performance of policy $\pi_{m,h}^k$.

Lemma 2 *Let Assumption 1 holds. For $\forall(m, h, s, k) \in [M] \times [H] \times \mathcal{S} \times [K]$, the following inequality holds with probability at least $1 - \delta$*

$$\overline{V}_{m,h}^k(s) \geq V_{m,h}^{\dagger,\pi_{-m,h}^k}(s), \quad \underline{V}_{m,h}^k(s) \leq V_{m,h}^{\pi_h^k}(s). \tag{13}$$

In this lemma, policy $\pi_h^k(s)$ can be seen as part of the output policy $\pi(s)$ in Algorithm 2, that is used from step h to H. It is formally defined in Algorithm 12 in Appendix D in Zhang et al. (2022).

We note that it is technically difficult to ensure optimism and pessimism under the influence of heterogeneous reward delays among agents. Notice that $\overline{V}_{m,h}^k(s)$ and $\underline{V}_{m,h}^k(s)$ are calculated only with information of agent m. However, the output policy π_h^k, as in $V_{m,h}^{\dagger,\pi_{-m,h}^k}(s)$ and $V_{m,h}^{\pi_h^k}(s)$, is a *correlated* policy that takes information of all agents into consideration. This information mismatch makes it technically challenging for $\overline{V}_{m,h}^k(s)$ and $\underline{V}_{m,h}^k(s)$ to upper or lower bound $V_{m,h}^{\dagger,\pi_{-m,h}^k}(s)$ and $V_{m,h}^{\pi_h^k}(s)$, and breaks the original optimism and pessimism results in V-learning (Jin et al., 2021; Song et al., 2021). Here we carefully design Algorithm 2 (especially line 3) to ensure that every agent is aware of the extra information of the most informed agent. Then with the carefully designed bonuses as in Equation 13, we are able tackle this difficulty and ensure optimism and pessimism.

STEP 3: Bound the CCE-gap. Finally, given Lemma 2, it suffices to bound the gap between the optimistic and pessimistic value estimates $\sum_{k=1}^{K}(\overline{V}_{m,1}^k - \underline{V}_{m,1}^k)(s_h^k)$.

As is mentioned in Step 2, the value estimates $\overline{V}_{m,h}^k(s)$ and $\underline{V}_{m,h}^k(s)$ are calculated without access to all information due to the reward delays. This fact increases the variance of the value estimates. In the proof of this theorem, we carefully analyze the number of unreceived and unusable visits for every episode and analyze its cumulative influence across all episodes.

5. Extension to Infinite Delays

5.1. The Skipping Scheme

The performance of the DA-MAVL algorithm in Section 3 heavily relies on the assumption that delays are finite. One single infinite delay could prevent the algorithm from convergence because all visits that happen later are unusable. In this case, it is worth skipping some of the rewards for better performance. Following the intuitions of Zimmert and Seldin (2020), we extend DA-MAVL and design a new skipping metric to deal with infinite delays in MARL. Details for the extended algorithm (DA-MAVL with Reward Skipping) are presented in Appendix C.3 in Zhang et al. (2022).

The critical part of the 'Skipping' process is to determine when to skip a visit. When the n-th visit of (h, s) happens, we maintain the skipping metric $\phi_{m,h}^{i,n}(s) = \sum_{j=i+1}^{n}(j-i)$ if the i-th visit of (h, s) is unreceived. Intuitively speaking, $\phi_{m,h}^{i,n}(s)$ upper bounds the contribution of the i-th visit to $\mathcal{T}_{m,h}^{n}(s)$. It is beneficial to skip the i-th visit if $\phi_{m,h}^{i,n}(s)$ becomes large enough. Following the intuition of previous reward skipping method in Zimmert and Seldin (2020) in the adversarial bandit setting, we skip the i-th visit if $\phi_{m,h}^{i,n}(s)$ exceeds threshold $\sqrt{\mathcal{T}_{m,h}^{n}(s)}$.

However, we would like to point out that our design of the skipping metric $\phi_{m,h}^{i,n}(s)$ is not a direct generalization of previous skipping method. Unlike the multi-agent setting considered in this paper, the adversarial bandit setting does not need to consider the heterogeneity of reward delays among agents, thus their algorithm update does not need to wait for visits to become usable. Correspondingly, the skipping metric $n-i$ in previous method would fail in our setting, because it no longer upper-bounds the contribution of the i-th visit to $\mathcal{T}_{m,h}^{n}(s)$.

5.2. Performance Guarantee for DA-MAVL with Reward Skipping

Recall the notation $k_h^n(s)$ stands for the episode when n-th visit of (h, s) happens. With the skipping scheme, we can also relax Assumption 1 to the following:

Assumption 2 *For $\forall(m, h, s, n) \in [M] \times [H] \times \mathcal{S} \times [K]$, there exists a constant C satisfying:*

$$|\{i \le n : d_{m,h}^{i}(s) + k_h^i(s) \ge k_h^n(s)\}| \le C. \tag{14}$$

Intuitively, Assumption 2 requires that for every pair (m, h, s, n), there are at most C unreceived visits before the n-th visit of $(h, s)$for agent m. This implies that either large delays do not appear too many times or delays are not large enough to influence performance. It is worth noting that the finite delay Assumption 1 implies Assumption 2 with $C = d_{max}$. But Assumption 2 is more general than Assumption 1 because Assumption 2 holds even if there are less than C infinite delays.

Given a subset of visit indices $\mathcal{L} \subset [K]$, at episode $k_h^n(s)$ when the n-th visit of (h, s) happens, we define variable $e_{m,h}^{n,\mathcal{L}}(s)$ as the earliest unreceived visit outside of $\mathcal{L}$:

$$e_{m,h}^{n,\mathcal{L}}(s) := \min\left\{j : d_{m,h}^{j}(s) + k_h^j(s) > k_h^n(s) - 1, j \in [n-1]\backslash\mathcal{L}\right\}. \tag{15}$$

If all of the first $n-1$ visits are received, we let $e_{m,h}^{n,\mathcal{L}}(s) = n$. Now we define $\mathcal{T}_{m,h}^{n,\mathcal{L}}(s)$ as follows:

$$\mathcal{T}_{m,h}^{n,\mathcal{L}}(s) := \sum_{i=1}^{n} i - e_{m,h}^{i,\mathcal{L}}(s). \tag{16}$$

It counts the accumulated number of unusable and unreceived visits outside of $\mathcal{L}$ for the first n visits of (h, s). Finally, we also define $\mathcal{T}_{m,h}^{K,\mathcal{L}} := \sum_{s\in\mathcal{S}} \mathcal{T}_{m,h}^{\overline{n}_{m,h}^{K}(s),\mathcal{L}}(s)$. Intuitively, it counts the accumulated number of unusable and unreceived visits outside of $\mathcal{L}$ during the K episodes.

Now we are ready to present the performance guarantee for DA-MAVL with Reward Skipping:

Theorem 2 *Under Assumption 2, for any $\delta \in (0,1)$, $K \geq C^6 S^3 \iota^3$ where $\iota = \log(4MHSAK/\delta)$, suppose Algorithm 9 is run for K episodes, then the following equation holds for the output policy π of Algorithm 2 with probability at least $1-\delta$*

$$\textit{CCE-gap}(\pi) \lesssim CH^3 \max_{m,h} \min_{\mathcal{L}} \left\{ \frac{S|\mathcal{L}|}{K} + \sqrt{\frac{S\mathcal{T}_{m,h}^{K,\mathcal{L}}}{K^2}} \right\} \iota^2 + H^3 \sqrt{\frac{SA}{K}} \iota. \tag{17}$$

Theorem 2 implies that DA-MAVL with Reward Skipping can still obtain convergence to CCE when there are infinite delays. Consider the case where all delays are upper bounded by constant d_{max}, except for C infinite delays for every (h,s). Let $\mathcal{L}_{m,h} = \{n : \exists s, d_{m,h}^n(s) = \infty\}$ denote all visit indices where the delay is infinite for some state s and fixed pair (m,h). We then have $|\mathcal{L}_{m,h}| \leq CS$ and $\mathcal{T}_{m,h}^{K,\mathcal{L}_{m,h}} \leq Kd_{max}$. Substituting into Theorem 2 gives CCE-gap of order $\tilde{\mathcal{O}}(\frac{H^3\sqrt{Sd_{max}}+H^3\sqrt{SA}}{\sqrt{K}})$, which is exactly the same as the result of Theorem 1.

6. Simulations

We simulate our algorithms in a simple MG with $M=3, S=3, A=2, H=2$. Due to the space limit, the simulation settings are deferred to Appendix B in Zhang et al. (2022). We only present the simulation results in Figure 1. We can see that our algorithm outperforms the naive algorithm (mentioned in Section 3) when delays are finite. Moreover, our novel skipping metric outperforms previous skipping method (mentioned in Section 5.1) when delays are infinitely large.

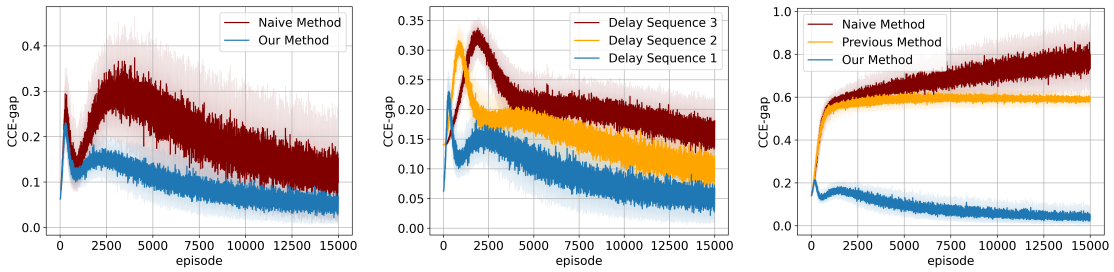

Figure 1: **Left**: CCE-gap for Output Policy of DA-MAVL (Our Method) and the Naive Algorithm (Naive Method); **Center**: CCE-gap for Output Policy of DA-MAVL under Different Delay Sequences; **Right**: CCE-gap for Skipping Metrics in DA-MAVL with Reward Skipping (Our Method), in Previous Work (Zimmert and Seldin, 2020) (Previous Method) and No Skipping (Naive Method).

7. Conclusion

This paper studies MARL with reward delays. For finite delays, we propose MARL algorithms with a novel mechanism to choose proper visits for learning, so that agents can reach a CCE even when facing heterogeneous delays. We also adapt our algorithm to cases with infinite delays using a novel reward skipping metric. High probability bounds are given on the CCE-gap of our algorithms. There are many interesting future directions, such as proving or disproving the convergence of the naive algorithm (Appendix C.2 in Zhang et al. (2022)), providing lower bounds on the CCE-gap for MARL with reward delays, relaxing Assumption 2 for infinite delays, extending current results to MGs with function approximation, etc.

Acknowledgments

This work is supported by the NSF grants CNS 2003111 and AI institute 2112085 and by the ONR YIP award N00014-19-1-2217.

References

Mridul Agarwal and Vaneet Aggarwal. Blind decision making: Reinforcement learning with delayed observations. *Pattern Recognition Letters*, 150:176–182, 2021.

Eman Aldhahri, Vivek Shandilya, and Sajjan Shiva. Towards an effective crowdsourcing recommendation system: A survey of the state-of-the-art. In *2015 IEEE Symposium on Service-Oriented System Engineering*, pages 372–377. IEEE, 2015.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

Yann Bouteiller, Simon Ramstedt, Giovanni Beltrame, Christopher Pal, and Jonathan Binas. Reinforcement learning with random delays. In *International conference on learning representations*, 2020a.

Yann Bouteiller, Simon Ramstedt, Giovanni Beltrame, Christopher Pal, and Jonathan Binas. Reinforcement learning with random delays. In *International conference on learning representations*, 2020b.

Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

Nicol'o Cesa-Bianchi, Claudio Gentile, Yishay Mansour, and Alberto Minora. Delay and cooperation in nonstochastic bandits. In *Conference on Learning Theory*, pages 605–622. PMLR, 2016.

Baiming Chen, Mengdi Xu, Zuxin Liu, Liang Li, and Ding Zhao. Delay-aware multi-agent reinforcement learning for cooperative and competitive environments. *arXiv preprint arXiv:2005.05441*, 2020.

Jianyu Chen, Bodi Yuan, and Masayoshi Tomizuka. Model-free deep reinforcement learning for urban autonomous driving. In *2019 IEEE intelligent transportation systems conference (ITSC)*, pages 2765–2771. IEEE, 2019.

Young-Cheol Choi and Hyo-Sung Ahn. A survey on multi-agent reinforcement learning: Coordination problems. In *Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, pages 81–86. IEEE, 2010.

Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. The complexity of computing a nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009.

Constantinos Daskalakis, Noah Golowich, and Kaiqing Zhang. The complexity of markov equilibrium in stochastic games. *arXiv preprint arXiv:2204.03991*, 2022.

Peihu Duan, Lidong He, Zhisheng Duan, and Ling Shi. Distributed cooperative lqr design for multi-input linear systems. *IEEE Transactions on Control of Network Systems*, 2022.

Vlad Firoiu, Tina Ju, and Josh Tenenbaum. At human speed: Deep reinforcement learning with action delay. *arXiv preprint arXiv:1810.07286*, 2018.

Manegueu Anne Gael, Claire Vernade, Alexandra Carpentier, and Michal Valko. Stochastic bandits with arm-dependent delays. In *International Conference on Machine Learning*, pages 3348–3356. PMLR, 2020.

Andras Gyorgy and Pooria Joulani. Adapting to delays and data in adversarial multi-armed bandits. In *International Conference on Machine Learning*, pages 3988–3997. PMLR, 2021.

Yann-Michaël De Hauwere, Peter Vrancx, and Ann Nowé. Solving sparse delayed coordination problems in multi-agent reinforcement learning. In *International Workshop on Adaptive and Learning Agents*, pages 114–133. Springer, 2011.

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.

Tiancheng Jin, Tal Lancewicki, Haipeng Luo, Yishay Mansour, and Aviv Rosenberg. Near-optimal regret for adversarial mdp with delayed bandit feedback. *arXiv preprint arXiv:2201.13172*, 2022.

Pooria Joulani, Andras Gyorgy, and Csaba Szepesvári. Online learning under delayed feedback. In *International Conference on Machine Learning*, pages 1453–1461. PMLR, 2013.

Pooria Joulani, András György, and Csaba Szepesvári. A modular analysis of adaptive (non-) convex optimization: Optimism, composite objectives, and variational bounds. In *International Conference on Algorithmic Learning Theory*, pages 681–720. PMLR, 2017.

Konstantinos V Katsikopoulos and Sascha E Engelbrecht. Markov decision processes with delays and asynchronous cost collection. *IEEE transactions on automatic control*, 48(4):568–574, 2003.

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

Shunsuke Kobayashi and Wolfram Schultz. Influence of reward delays on responses of dopamine neurons. *Journal of neuroscience*, 28(31):7837–7846, 2008.

Tal Lancewicki, Aviv Rosenberg, and Yishay Mansour. Learning adversarial markov decision processes with delayed feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7281–7289, 2022.

Tianhao Li, Zhishun Wang, Wei Lu, Qian Zhang, and Dengfeng Li. Electronic health records based reinforcement learning for treatment optimizing. *Information Systems*, 104:101878, 2022.

Michael L Littman et al. Friend-or-foe q-learning in general-sum games. In *ICML*, volume 1, pages 322–328, 2001.

Qinghua Liu, Tiancheng Yu, Yu Bai, and Chi Jin. A sharp analysis of model-based reinforcement learning with self-play. In *International Conference on Machine Learning*, pages 7001–7010. PMLR, 2021.

Yong Liu. On the minimum delay peer-to-peer video streaming: how realtime can it be? In *Proceedings of the 15th ACM international conference on Multimedia*, pages 127–136, 2007.

Andreja Malus, Dominik Kozjek, et al. Real-time order dispatching for a fleet of autonomous mobile robots using multi-agent reinforcement learning. *CIRP annals*, 69(1):397–400, 2020.

Weichao Mao and Tamer Başar. Provably efficient reinforcement learning in decentralized general-sum markov games. *Dynamic Games and Applications*, pages 1–22, 2022.

Diogo Martinho, João Carneiro, José Neves, Paulo Novais, Juan Corchado, and Goreti Marreiros. A reinforcement learning approach to improve user achievement of health-related goals. In *EPIA Conference on Artificial Intelligence*, pages 266–277. Springer, 2021.

Gergely Neu, Andras Antos, András György, and Csaba Szepesvári. Online markov decision processes under bandit feedback. *Advances in Neural Information Processing Systems*, 23, 2010.

Kun Shao, Zhentao Tang, Yuanheng Zhu, Nannan Li, and Dongbin Zhao. A survey of deep reinforcement learning in video games. *arXiv preprint arXiv:1912.10944*, 2019.

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

Ziang Song, Song Mei, and Yu Bai. When can we learn general-sum markov games with a large number of players sample-efficiently? *arXiv preprint arXiv:2110.04184*, 2021.

Matthew Streeter and H Brendan McMahan. Less regret via online conditioning. *arXiv preprint arXiv:1002.4862*, 2010.

Jayakumar Subramanian, Amit Sinha, and Aditya Mahajan. Robustness and sample complexity of model-based marl for general-sum markov games. *arXiv preprint arXiv:2110.02355*, 2021.

Hongyao Tang, Jianye Hao, Tangjie Lv, Yingfeng Chen, Zongzhang Zhang, Hangtian Jia, Chunxu Ren, Yan Zheng, Zhaopeng Meng, Changjie Fan, et al. Hierarchical deep multiagent reinforcement learning with temporal abstraction. *arXiv preprint arXiv:1809.09332*, 2018.

Thomas J Walsh, Ali Nouri, Lihong Li, and Michael L Littman. Learning and planning in environments with delayed feedback. *Autonomous Agents and Multi-Agent Systems*, 18(1):83–105, 2009.

Hang Yin, Yu Wang, Xukai Zhang, and Peng Li. Feedback delay impaired reinforcement learning: Principal components analysis of reward positivity. *Neuroscience letters*, 685:179–184, 2018.

H Peyton Young. *Strategic learning and its limits*. OUP Oxford, 2004.

Yuyang Zhang, Runyu Zhang, Gen Li, Yuantao Gu, and Na Li. Multi-agent reinforcement learning with reward delays. *arXiv preprint arXiv:2212.01441*, 2022.

Xiangyu Zhao, Long Xia, Lixin Zou, Hui Liu, Dawei Yin, and Jiliang Tang. Whole-chain recommendations. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1883–1891, 2020.

Julian Zimmert and Yevgeny Seldin. An optimal algorithm for adversarial bandits with arbitrary delays. In *International Conference on Artificial Intelligence and Statistics*, pages 3285–3294. PMLR, 2020.